**Version 1.1:
new & improved**

Rich Statistical Parsing
and Literary Language

Andreas van Cranenburgh

Rich Statistical Parsing and Literary Language

INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

Rich Statistical Parsing
and Literary Language

Contents

Acknowledgments

T HIS THESIS was supervised by Rens Bod and Remko Scha, joined in the final stages by Ivan Titov. Rens played a crucial role in convincing me to accept this position when I had some initial misgivings. All turned out well in the end since I was given ample freedom to pursue my own projects.

Remko Scha supervised the work reported in the first part of this thesis. This reflected the continuation of an academic mentorship that started in my bachelor's in artificial intelligence and proved to be a pivotal influence. We discussed plans for the second part of this thesis extensively. But alas, it was not to be. Remko passed away in November 2015.

Remko was a crucial source of inspiration and always ready to indulge in a speculative train of thought or an overly ambitious plan, and generously viewing it in the best possible light to distill the practical or promising bits. I will remember our many meetings in café Luxembourg fondly. I hope to carry on with the spirit of conducting research pursuing a broader agenda and vision, rather than incremental improvements of scores and other existing results.

The work of Federico Sangati on exploiting recurring patterns in parse trees provided the groundwork for this thesis. His approach proved to be an exceedingly fruitful foundation for improvements and new applications. Similarly, my work on discontinuous parsing, started in my master's thesis, benefited from Wolfgang Maier generously making the code of his parser available.

The research in the second part aims to fulfill the goals of the project the Riddle of Literary Quality, which was conceived by Karina van Dalen-Oskam. Several sections report on joint work with my fellow PhD candidates in this project, Corina Koolen and Kim Jautze, who always brought a lively atmosphere to our collaborations.

The advisory committee of the project provided valuable feedback and reflections on the two occasions that it was assembled; especially David Hoover and Patrick Juola were helpful. The project was part of the computational humanities program, facilitated by the eHumanities group at the Meertens institute which organized research meetings and activities. The Alpino parser of Gertjan van Noord was crucial to the second part, and I thank him for responding to practical queries. Karina van Dalen-Oskam provided comments on Chapter 5.

I am grateful to Albert Meroño Peñuela for commenting on Chapter 3, and to Berit Janssen & Allen Riddell for commenting on Chapter 7. I have not heeded all their advice; therefore I stress that any remaining shortcomings are strictly my own.

I want to thank all the students whose various projects I got to supervise: Rémi, Anouk, Alexa, Frank, Benno, Sara, Maartje, Minh, Peter, and the students of the digital humanities course I taught with Corina. Thanks to Wolfgang Maier and Laura Kallmeyer for inviting me to hold a lecture in Düsseldorf, to Federico Sangati for inviting me to visit FBK in Trento, and to Krasimir Angelov for inviting me to visit the University of Gothenburg.

The research in this thesis was conducted almost exclusively with Free Software. This thesis would not have been possible without the code generously made available by projects such as Linux, Python, Cython, scikit-learn, matplotlib, and LaTeX.

Beyond academia I want to express my gratitude to my parents for all their support. I also thank my friends for all the wine, pub quizzes, table top games, and other essential distractions. I am grateful for the support of my *paranimfen* Jessica & Tjaard in particular. I thank my rowing mates for an indispensable and reliable regimen of physical exercise. Enrico Perotti shared timely academic advice and wisdom.

On a lighter note, I should acknowledge here the tireless support of Jasmine, Earl Grey, and Lapsang Souchon, without which this thesis could not have been written.

Düsseldorf   Andreas van Cranenburgh
August, 2016.

> Meanwhile, let us have a sip of tea. The afternoon glow is brightening the bamboos, the fountains are bubbling with delight, the soughing of the pines is heard in our kettle. Let us dream of evanescence, and linger in the beautiful foolishness of things.
> — Okakura (1906, ch. 1), *The Book of Tea*

Publication statement

Part of the results presented in this thesis have been published before.

The work in Chapter 2 has appeared in the CLIN journal (van Cranenburgh, 2014).

The work in Chapter 3 was presented at the EACL (van Cranenburgh, 2012a) and IWPT (van Cranenburgh and Bod, 2013) conferences, and has appeared in the Journal of Language Modelling (van Cranenburgh et al., 2016).

Section 6.2 is based on joint work with Kim Jautze not previously published.

Section 6.3 is based on Jautze et al. (2016a), an abstract presented at Digital Humanities 2016 that is joint work with Kim Jautze and Corina Koolen.

The work in Section 7.1 was presented at the CLFL workshop (van Cranenburgh, 2012c).

The work in Section 7.2 was presented at the CLFL workshop (van Cranenburgh and Koolen, 2015).

The work in Section 7.3 and 7.4 has been presented at EACL (van Cranenburgh and Bod, 2017) after the defense of this thesis.

Several additional publications that I co-authored in the context of my doctoral research are not included in this thesis.

Aloni et al. (2012, LREC) presents a corpus of indefinites annotated with fine-grained semantic functions.

Jautze et al. (2013, CLFL workshop) is a pilot study of chick lit and literature using simple syntactic measures.

Roorda et al. (2014, CLIN journal) includes an experiment with Data-Oriented Parsing for parsing the Hebrew bible.

Sangati and van Cranenburgh (2015, MWE workshop) presents results on Data-Oriented Parsing of the French treebank and Multi-Word Expression detection in several languages.

The source code of the parser, fragment extractor, and syntactic search engine developed in the context of this research is available at:
https://github.com/andreasvc/disco-dop

x

Overview

In which we introduce the topics & contributions of this thesis: syntax and literature, analyzed with computational models.

THERE IS A traditional contrast in the study of language between linguistics and philology; this thesis presents computational work in both areas. Computational linguistics, and its applied variant Natural Language Processing (NLP), takes language as its object of study and uses computers as an instrument to develop and evaluate models. Evaluation of predictive models provides an important methodological heuristic that sets computational linguistics apart from other fields of linguistics and many branches of science. New tasks or improved models for existing tasks are benchmarked with quantitative metrics (this has been referred to as the *common task framework*, Liberman 2015b), giving an immediate indication of how much has been achieved and what remains to be done. One such task which is considered in the first part of this thesis is that of syntactic parsing, where the goal is to analyze the syntactic phrases or relations between words in a sentences.

Philology means the love of words, learning, interpretation, and literature; often from a historical perspective. The second part of this thesis deals with a topic in what could be called computational philology, more specifically, computational literary stylistics. The increasing availability of digitized texts and effective computational methods to study them offer new opportunities. These methods not only allow more data to be processed, they also suggest different questions. Machine learning and the broader field of data science offer the possibility of extracting knowledge from data in an automated, reproducible manner.

RESEARCH QUESTIONS. Since this thesis covers two main topics, we state the research question for each topic, and another connecting the two.

PARSING LANGUAGE: To what extent is it possible to create linguistically rich parsing models without resorting to handwritten rules by exploiting statistics from annotated data?

Probabilistic algorithms for parsing and disambiguation select the most probable analysis for a given sentence in accordance with a certain probability distribution. A fundamental property of such algorithms is thus the definition of the

space of *possible sentence structures* that constitutes the domain of the probability distribution. Modern statistical parsers are often automatically derived from corpora of syntactically annotated sentences ("treebanks"). In this case, the "linguistic backbone" of the probabilistic grammar naturally depends on the convention for encoding syntactic structure that was used in annotating the corpus.

Statistical parsers are effective but are typically limited to producing projective dependencies or constituents. On the other hand, linguistically rich parsers recognize non-local relations, and analyze both form and function phenomena but rely on extensive manual grammar engineering. We combine advantages of the two by building a statistical parser that produces richer analyses.

Markers of literariness: What sorts of syntactic and lexical patterns may correlate with and explain the concept of literature?

In contrast to genre fiction, literary novels do not deal with specific themes and topics. However, they may still share stylistic and other implicit characteristics that may be uncovered using text analysis and machine learning.

These two research questions are connected by the following question:

Syntax in literature: For what sorts of stylistic and stylometric tasks and under which conditions can morphosyntactic information be exploited fruitfully?

Previous work has shown that simple textual features that are easy to extract, in particular Bag-of-Words features, typically outperform structural features such as syntax, which are comparatively expensive to extract. Our aim is to see to what degree this holds in the case of (literary) fiction, and whether there are specific aspects for which syntax is important.

Outline. The common themes in the two parts of this thesis are (a) the use of tree fragments as building blocks and predictive features, and (b) non-local and functional relations.

Part I deals with parsing and is concerned with general language use as made available in annotated data sets of several languages.

Part II deals with literature and focuses on contemporary Dutch novels and in particular on what differentiates literary language from the language of genre fiction. This work is done in the context of the project "The Riddle of Literary Quality," which aims to investigate the concept of literature empirically by searching for textual features of literary conventions in contemporary novels.

The two parts of this thesis can be read independently. One exception is that the algorithm for extracting recurring tree fragments defined in part I is used in part II, and should be referred to for specifics on that method.

Contributions. The contributions of this thesis can be summarized as follows:

- Efficient extraction of recurring patterns in parse trees, which can be used to build grammars, as features in machine learning tasks, and in linguistic research in general. The method that is presented provides a significant improvement in efficiency and makes it possible to handle much larger corpora.

- A statistical parser automatically learned from treebanks, reproducing rich linguistic information from the treebanks, such as discontinuous constituency & function tags. The parsers are induced from data with minimal manual intervention and evaluated on several languages.

- An investigation of what makes texts literary, use ratings from a large online survey, and machine learning models of texts to predict those ratings.

- These models, based on lexical, topical, and syntactic features, demonstrate that the concept of literature is non-arbitrary, and predictable from textual characteristics to a large extent.

Part I
Parsing

1 Syntax & Parsing Background

In which we survey previous work on syntactic analysis by computer, with particular attention to Data-Oriented Parsing (DOP).

> La plus part des occasions des troubles du monde sont Grammariens. Noz procez ne naissent que du debat de l'interpretation des loix; et la plus part des guerres, de cette impuissance de n'avoir sçeu clairement exprimer les conventions et traictez d'accord des Princes.
> *Most of the occasions for the troubles of the world are grammatical. Our lawsuits spring only from debate over the interpretation of the laws, and most of our wars from the inability to express clearly the conventions and treaties of agreement of princes.*
> — de Montaigne (*Essais*, II, 12; trans. D. Frame, 1958)

Since this thesis is intended for a multi-disciplinary readership, this chapter introduces the background for statistical parsing, including basic linguistic notions. What is syntax, and why do we need it? It turns out that the answer is not obvious, and ultimately boils down to the question why natural languages are as complex as they are.

The rest of this chapter reviews syntactic representations and parsing technology for automatically assigning syntactic analyses to sentences. We end with an exposition of Data-Oriented Parsing, the parsing framework that will be used in this thesis, followed by a reflection on the competence-performance distinction.

1.1 Syntax

Grammar determines the set of well-formed sentences that are part of a language. Grammar consists of syntax and morphology. The syntax of a language governs

which combinations of words form sentences[1] while morphology governs which combinations of morphemes form words and how they function in the sentence. A trivial example:[2]

(1)   a.   the book
       b.   *book the

Note that there is nothing natural about the fact that a determiner should precede its noun. There are languages in which determiners follow nouns, as well as languages without any determiners.

Another role of syntax is to assign structure to a sentence. Syntactic relations answer questions such as *who did what to whom*, while structure may also reveal the building blocks of the sentence. For example:[3]

(2)   a.   a book [about walking [in the woods]]
       b.   #a book [about walking] [in the woods]

Here we also see an example of an *attachment ambiguity*, which is a major source of syntactic ambiguity. The first bracketing expresses the normal interpretation of book that tells you about the kind of walking in woods. The second interpretation is of a book that is specified to be both about walking and in the woods (i.e., it is the book which is in the woods). Although the latter interpretation is highly improbable, note that there is no structural reason why this is so.

The division of labor between syntax and morphology depends on the definition of what is a word. This is not an unproblematic notion, because there is no definition that works across all languages. Two ways of defining words are the prosodic definition (words are identified by stressed syllables) and the grammatical definition (words are the units that can occur in isolation according to morphology and syntax). For a given language with a clear definition of what a word is, the division between morphology and syntax is clear.

On the other side syntax interfaces with semantics, viz. the meaning of words and sentences. Although syntax is sometimes viewed as completely separate from semantics and other linguistics divisions ("the autonomy of syntax"), there is overlap in many cases, such as in the aforementioned *who did what to whom*, in which semantic information is realized by syntactic means, and, conversely, in syntactic ambiguities that can be resolved by semantics, such as the attachment ambiguity above.

---

1 Word order is often invoked when defining syntax. However, this reliance is not an inherent feature of syntax. In less- and non-configurational languages morphology plays a larger role.

2 This thesis follows the common convention of prefixing ungrammatical examples with an asterisk.

3 The # symbol indicates a sentence or interpretation that is not felicitous, although not ungrammatical.

1.1.1 WHY DO LANGUAGES HAVE SYNTAX?

It is instructive to wonder whether syntax is actually necessary for a language, or at least, why it is as complex as it is. We will distinguish two aspects of syntax (Koenig and Michelson, 2014). Compositional syntax specifies how the meanings of expressions may be combined. Formal syntax determines which combinations of words are accepted as part of the language. It is especially the latter whose complexity seems incidental and non-essential.

Consider the language of simple arithmetic expressions. These are ordinarily written in infix notation, with the operator between its operands:

$$2 + 2$$
$$(3 + 4) * 5$$

The following are all ungrammatical:

$$2\ 2 +$$
$$3\ 4 + 5 *$$

The latter examples use postfix notation,[4] in which operators come after their operands. In postfix notation, every permutation of numerals and operators arguably forms a syntactically valid expression.[5] Additionally, any ordering of operations can be achieved without parentheses. This does mean that tokens must be delimited in some way, because there is no alternation "operand–operator" as in infix notation. Clearly, in the case of arithmetic expressions, syntax (beyond the lexical level) is optional. Infix notation introduces a context-free grammar along with attachment ambiguities that need to be resolved with parentheses and precedence order, but this complexity is not intrinsic to the task.[6]

What this example shows is that it is perfectly possible to define a language in which every possible sequence of tokens is a valid expression. So what is the trade-off that is being made in a language that rejects a large part of the space of possible expressions? Or is it simply an aspect of what makes natural languages *natural*?

We can speculate about various reasons:

PROCESSING: The encoding and decoding of a message with minimal syntax may be more expensive. In postfix notation of arithmetic, interpretation relies on keeping track of one or more stacks. The stack may be modified at each step, and the effect of a given token depends on the current state of the stack.

---

4 Postfix notation is also known as reverse Polish notation, in honor of logician Jan Łukasiewicz.
5 Caveat: there may not be enough operands on the stack, or operands may be left on the stack at the end of the expression. For the sake of the argument this should be considered a semantic issue. After all, the same expression could be used in another context where it does fit.
6 The similarities between infix notation and natural language do not end here. Experimental research has shown that arithmetic attachment ambiguities influence natural language ambiguities, and *vice versa* (Scheepers et al., 2011), suggesting that they rely on the same cognitive representations.

Reliability: There is an advantage to being able to recognize a malformed message, e.g., when communicating through an unreliable channel and a sentence comes through only partially, this may be detected by its ungrammaticality.

Language change: Since natural languages are generally not planned but change for a variety of reasons, it may not be possible to maintain a language where every combination of tokens is interpreted consistently; i.e., given that other aspects of language are subject to change, there is no reason why syntax would stay perfectly fixed.

Inherent complexity: What natural languages are used for is inherently more complex than arithmetic expressions or computer instructions. Specifically, it could be that language users have hierarchical internal representations (Kirby, 2002), and it would therefore be natural that language evolves to express them.

Instead of looking for *a priori* reasons for syntax, we can also consider empirical evidence from linguistic typology, which maps the common and distinguishing features of the world's languages.

There exist languages that are much simpler than most. The so-called pidgin languages emerge when different groups of people come into contact that do not share a language. The language combines vocabulary of the languages of each of the group, but is syntactically simple.

Pidgin languages may creolize, which means that the language evolves into a creole as the next generation learns the pidgin as a first language. This tends to make the language more complex, including syntactically.

Another example is the case of Pirahã, a language spoken by a tribe of the same name from the Amazonian jungle (Everett, 2005, 2009). The language is claimed not to employ recursion—although this is a contentious claim (e.g., Nevins et al., 2009), which is difficult to resolve since the only outsiders who have learned the language are Everett and two other missionaries. The language makes it impossible to refer to something not directly witnessed, or indirectly to the first degree (you know the person who witnessed it).[7] Furthermore, Pirahã counting is restricted to 1, 2, and many, noun phrases may only contain one or two words, there are no words for colors, nor future or past tenses. In fact, both the language and its syntax, as well as their culture, appear to be simpler than most extant languages or cultures,[8] and Everett (2005) concludes that these are connected.

Jackendoff and Wittenberg (2014) argue for a new hierarchy to describe grammatical complexity. Unlike the Chomsky hierarchy which is strictly concerned with the formal power of types of rewrite rules, their hierarchy is designed to

---

7 This fact played a role in his failure to convert the tribe to Christianity; he gave up his ambitions as a missionary and proceeded with an anthropological and linguistic study. He also abandoned Chomskyan linguistics.

8 The language does make elaborate use of suffixing and compounding, but this does not affect the argument that its syntax is exceptionally simple.

accommodate empirical evidence on languages of differing grammatical complexity. The hierarchy is as follows:

1. (a) One-word grammar
   (b) Two-word grammar
   (c) Linear grammar: sentences consist of an arbitrary number of words
2. Simple phrase grammar: words may be grouped into phrases
3. Recursive phrase grammar: possibility of recursive phrases is introduced.
4. Morphology: structure in words, independent of syntax.
5. Fully complex grammars may include: syntactic categories, grammatical functions, long-distance dependencies, etc.

Almost all languages are at least in the recursive category. Languages with a linear grammar or a simple phrase grammar may still fulfill all the semantic and pragmatic needs of language users; they achieve this by relying more on pragmatics and contextual factors. Jackendoff and Wittenberg (2014) cite Pirahã, described above, and Riau Indonesian, as languages with a simple phrase grammar. On the other hand, more complex grammars entail the possibility for more precise sentences (generally less ambiguous).[9]

In conclusion, it seems that there is no clear and precise *a priori* reason why languages need to rely on syntax and accept only particular sentences. Other kinds of languages are possible. However, looking at different languages of the world suggests that cognitive and cultural factors are most important in explaining the role of syntax. There is a trade-off where complex syntax increases precision and reduces the difficulty of interpretation and reliance on contextual information.

1.1.2 Representations of syntax

Two major representations of syntax are constituency and dependency structures. Both representations are hierarchical. This is evident for constituency structures, which are often rendered as tree structures; dependency structures can be rendered similarly, except that nodes consist solely of terminals.

Constituency structures, also known as phrase-structure trees, are built around the notion of a constituent: a grouping of words or constituents that functions as a unit (constituents with two or more words are called phrases). Constituents may be labeled by a syntactic category. Edges between constituents may be labeled by a (grammatical) function tag, although this is typically not done for historical reasons.[10] Figure 1.1 shows an example of a constituency tree.

---

9 An important distinction here is that a complex grammar enables more precisely specified sentences that are less ambiguous to the hearer; however, the complexity of the grammar would make the language harder to parse for a computer. The simpler grammar is underspecified.

10 Within generative linguistics work on English, functional relations were assumed to be derivable from phrase structure. This works well for English, since it is a highly configurational language in which for example the subject typically precedes the verb.
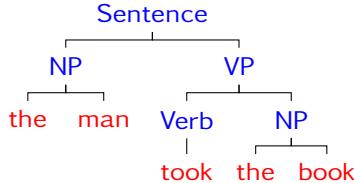
Figure 1.1: The first context-free grammar parse tree (Chomsky, 1956).
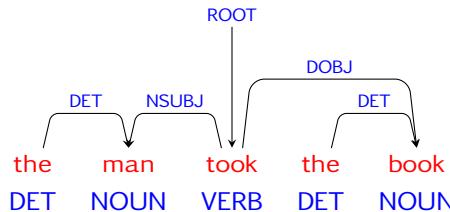
Figure 1.2: A dependency structure for the same sentence as in Figure 1.1.

When words can be said to form a unit is not always self-evident. There exist a range a constituency tests that can be applied as heuristics. We briefly discuss the most commonly used tests (Santorini and Kroch, 2007, ch. 2). The substitution test is the most basic. If a group of words in a given context can be exchanged for a "pro-form," then this provides evidence for constituent-hood. A pro-form is a function word that stands in for a word or phrase (e.g., *he* for a person, *it* for a noun, *how* for an adverb, *so* for an adjective). The movement test moves a given phrase to another position in the sentence, to see if the sentence is still grammatical. This may require the right intonation, for example when a direct object is topicalized and moved to the sentence-initial position. The question test attempts to use the phrase as the answer to a question, which needs to be a constituent.

Dependency structures consist of word-to-word relations (with specific constraints). To adequately represent a sentence, these relations should be directed and labeled. Each relation is from a head word to one of its dependents (an argument or modifier). They do not directly encode higher-level units such as constituents,[11] nor do they carry syntactic category labels. The labels express functional relations. This means that dependency structures are simpler and more minimal than constituency structures; the same sentence may contain half as much nodes in a dependency analysis as in a constituency analysis. Figure 1.2 shows an example of a dependency structure.

---

[11] Some phrases can be derived by starting from the head word and recursively following its dependents. On the other hand, dependency structures deliberately avoid encoding finite verb phrases, so these cannot be inferred without additional heuristic rules. This is an instance of an *exocentric* constituent, where two or more elements are combined without a clearly identifiable head.

It is generally not possible to mechanically convert a constituency structure into a dependency structure, or *vice versa*. This is because the constituency structure will typically not encode the head word of each constituent, nor its functional relations, while the dependency structure does not encode phrases and their categories. Going from dependency to constituency is harder. For particular languages and particular annotation schemes, it may be possible to design a conversion procedure, although this will typically need to rely on heuristics (e.g., Daum et al., 2004; Choi and Palmer, 2010).

However, if we consider dependency and constituency structures purely as data structures, the labels can be used to encode anything we want, including additional structural information. This strategy has been used to encode dependency structures in constituency trees (Eisner, 1996), and *vice versa* (Hall and Nivre, 2008; Fernández-González and Martins, 2015), in a way that is reversible without information loss. Structures may be transformed for processing, and back-transformed to the final output used in evaluation. In other words, the linguistic and theoretical considerations about the right representations are completely separate from the practical, technical aspects of which is the most expedient structure to use.

The labels in syntactic analyses represent discrete distributionally-defined categories based on intuitions of linguists.[12] For purposes of evaluation, these categories are taken as-is. However, it is unlikely that a given set of categories is ideal, since it must lie on an arbitrary point in the spectrum of granularity. Furthermore, certain choices are bound to be arbitrary due to different possible perspectives. For example, consider verbs that are used as adjectives: the walking man. 'Walking' is a verb in the form of a gerund, but it is modifying a noun so it behaves like an adjective. Clearly, it has properties of both. Finally, it can be argued that the categoricity itself is artificial and that the categories may be replaced altogether by a continuous vector representation. For practical purposes, the original syntactic categories will be used as gold standard in this thesis.

Until now we considered the basic syntactic structure of sentences, which is typically the focus in statistical parsing. Syntactic representations can also encode more elaborate structure such as morphology, long-distance dependencies, as well as preserving both form and function (syntactic categories and functional relations). In fact it is possible to encode all of this information and synthesize the information in dependency and constituency structures (Skut et al., 1997). This is the kind of representation we will work with in Chapter 3.

More fine-grained representations are common in linguistically-oriented work. Examples are attribute-value matrices and feature structures, which may encode more detailed information, from morphological features to semantic de-

---

12 Note that ultimately most evidence and data in (computational) linguistics rests on intuitions of language users. Worse, most annotations are based not on field work and sufficient samples of speakers, but on the introspective judgments of linguists themselves, with manifest limits in terms of inter-annotator agreement.

pendencies. These representations are more closely tied to particular linguistic theories.

1.2 Parsing technology

Parsing consists of assigning the correct syntactic analysis to a given sentence, chosen from among a set of candidates licensed by the grammar. Two main challenges are to ensure that the correct analysis is among the possible candidates (coverage), and that the correct analysis picked out from among these candidates (accuracy, which requires resolving ambiguity).

It is perhaps surprising that the number of candidates is huge, even when no ambiguity is apparent. Computational linguistics has been instrumental to this realization. Church (1988, p. 138) gives an example of the trivial sentence "I see a bird" and shows that the part of speech of each word is ambiguous according to the dictionary (e.g., *I* may be a numeral, *see* occurs as noun in "the holy see", etc). Most combinations of part-of-speech tags are grammatical, so syntax cannot rule the ambiguities out either. The way to cope with such ambiguity is to take probabilities into account; the alternate part-of-speech tags for these words are vanishingly rare.

We can distinguish several axes on which parsing technologies vary:

Manual versus automatic: manual grammars are handwritten by linguists; other grammars are automatically induced from corpora. However, there are several other degrees of manual intervention in between. Tree transformations and syntactic refinements can help bring out linguistic generalizations. Statistical and machine learning methods have tunable parameters, which can have a large effect on the final outcome.

Statistical/probabilistic versus knowledge-intensive: statistics from language use can be used to make parsing decisions. Grammars that do not use statistics are sometimes referred to as rule-based or symbolic—but this terminology is not helpful, since statistical parsers use rules and symbols as well; a better terminology might be data-intensive versus knowledge-intensive models.

Linguistically rich versus basic analyses: A basic analysis consists of phrase-structures or dependencies. A linguistically rich analysis contains more detailed information and aims for a consistent cross-linguistic account of linguistic phenomena. Basic analyses enable effective statistical learning methods; generalizing from richer analyses gets increasingly difficult due to the complexity and sparsity of the data. Despite this, it should be stressed that it is a pragmatic consideration.

Computational efficiency: On one end there are dependency and shift-reduce parsers, which have become exceedingly fast in recent years (measured

in hundreds of sentences per second). On the other end there are parsing formalisms that are computationally intractable, which can only be used with heuristics and approximations.

Historically, parsers have been based on handwritten grammars. This tradition continues to this day under the names grammar engineering, high-precision grammars, and broad- or wide-coverage parsing; e.g., Grammatical Framework (Ranta, 2011), Head-Driven Phrase-Structure Grammar (HPSG; Bender et al., 2002; Bouma et al., 2001), and Lexical-Functional Grammar (LFG; Kaplan and Bresnan, 1982). This work is characterized by detailed linguistic analyses, often represented in attribute-value matrices. Analyses include grammatical functions, long-distance relations, morphology, and co-indexation. However, writing a grammar by hand is time consuming, and it is a task that is difficult to manage: correcting an error may introduce more errors, all grammatical sentences must be accepted (coverage), and only correct analyses should be produced (precision). Another common problem is that rich grammar formalisms are often computationally intractable (Trautwein, 1995).

The availability of large amounts of data and computing power makes it possible to exploit statistics of actual language use. Such data can be used for estimating the probabilities of handwritten rules, or disambiguating the analyses licensed by them. However, the logical next step is derive the grammar from data as well. This has led to a line of work known as statistical parsing.

Most work in statistical constituency parsing is based on Probabilistic Context-Free Grammars (PCFG) and extensions thereof.[13] A context-free grammar (CFG) consists of productions that rewrite a constituent into its direct descendants; the latter may either be further constituents, or terminals (words):

$$\text{NP} \rightarrow \text{DET NOUN} \qquad\qquad \text{NOUN} \rightarrow \textit{book}$$

The above productions state that one valid noun phrase consist of a determiner followed by a noun, and that 'book' may be used as a noun. A set of productions can be induced automatically from syntactic trees. When relative frequencies of productions are included as well, a probabilistic model obtains, a PCFG. This provides a simple, efficient and reasonably accurate model, given simple refinements to overcome the strong independence assumptions made by the PCFG model. These assumptions can be broken down as follows (taken from Manning and Schütze, 1999, ch. 11):

PLACE INVARIANCE: The probability of a subtree does not depend on where in the string the words it dominates are [...]

CONTEXT-FREE: The probability of a subtree does not depend on words not dominated by the subtree. [...]

ANCESTOR-FREE: The probability of a subtree does not depend on nodes in the derivation outside the subtree. [...]

---

13 Another important line of work is dependency parsing (e.g., Nivre et al., 2007); however, this thesis is focused on constituency analyses.

Examples of refinements to weaken these assumptions are including parent categories in the labels and introducing linguistic distinctions not present in the original annotation (Klein and Manning, 2003). Higher accuracy can be obtained with further, automatic refinements, as well as different estimation methods that go beyond the generative model. Examples are coarse-to-fine parsing (Charniak and Johnson, 2005), discriminative re-ranking (Collins and Koo, 2005), latent variable models (Matsuzaki et al. 2005; later picked up by Petrov et al. 2006, etc.), and neural parsers (Henderson, 2004; Socher et al., 2013). These models have established a steep curve of increasing accuracy and efficiency over the years, but there is a single-minded focus on attaining the highest score, without much attention to whether the analyses adequately describe the linguistic phenomena, or whether the parser could serve as a plausible cognitive model.

An alternative is to strike a balance between linguistic adequacy and efficiency, and find a grammar formalism that is just powerful enough to describe the syntax of natural language. Joshi (1985) proposes Mildly Context-Sensitive grammars, which are beyond context-free, but avoid the computational complexity that comes with the full class of context-sensitive grammars. The first formalism developed in this framework was Tree-Adjoining Grammar (TAG; Joshi, 1985). There has been work on automatic extraction of tree-adjoining grammars from corpora (Chiang, 2000; Xia et al., 2001; Kaeshammer and Demberg, 2012), and formal extensions such as multi-component TAG (Weir, 1988; Schuler et al., 2000; Kallmeyer, 2009). Another successful formalism in this line of research is Combinatory Categorial Grammar (CCG; Steedman, 2000), a lexicalized grammar formalism based on combinatory logic.

The (somewhat informal) notion of mild context-sensitivity was introduced by Joshi (1985) to capture precisely the amount of generative capacity needed to describe natural languages—as opposed to employing richer frameworks which require ad-hoc constraints to be tractable. Mildly context-sensitive languages are characterized by the following properties:

1. limited crossed dependencies
2. constant growth
3. polynomial time parsing

For a formal description of these properties, refer to e.g., Groenink (1997). A diverse set of formalisms with these properties has since developed. However, while their structures and operations differ wildly, it has been observed that they share two common properties (Vijay-Shanker et al., 1987; Weir, 1988):

LINEAR: only a bounded amount of structure can be added or removed by applying productions, i.e., operations are size preserving

CONTEXT-FREE: choices during a derivation are independent of the context in the derivation (where context is anything which is not being rewritten)

Furthermore, it does not matter whether the formalism rewrites strings, tuples, or trees. This led to the introduction of Linear Context-Free Rewriting

Systems (LCFRS), which subsumes all formalisms with these properties. Groenink (1997) states that "[t]he class of mildly context-sensitive languages seems to be most adequately approached by LCFRS."

This thesis will use LCFRS to model discontinuous constituents.

1.3 Data-Oriented Parsing (DOP)

Data-Oriented Parsing (DOP; Scha, 1990; Bod, 1992) departs from the assumption that language users process sentences based on fragments from previous language experience. This experience can help in two ways:

A MEMORY BIAS: "[T]he number of constructions that is used to re-construct the sentence in order to recognize it must be as small as possible." (Scha, 1990).
A PROBABILISTIC BIAS: "More frequent constructions are to be preferred above less frequent ones." (Scha, 1990).

In Data-Oriented Parsing the grammar is implicit in the treebank itself, and in principle all possible fragments from its trees can be used to derive new sentences. Grammar induction is therefore conceptually simple (even though the grammar may be very large), as there is no training or learning involved. This maximizes re-use of previous experience.

The use of all possible fragments allows for multiple derivations of the same tree; this spurious ambiguity is seen as a virtue in DOP, because it combines the specificity of larger fragments and the smoothing of smaller fragments. For example, consider two large fragments from sentence A and B that provide evidence for a given analysis of a sentence, but they cannot form a derivation together because they overlap or a piece is missing. By using them in two separate derivations, they both contribute towards preferring this analysis.

This is in contrast to parsimonious approaches which decompose each tree in the training corpus into a sequence of fragments representing a single derivation, such as in Bayesian Tree-Substitution Grammars (TSG; Post and Gildea, 2009; Cohn et al., 2010). In Bayesian approaches for TSG induction the treebank is used to induce a corpus of derivations. This leads to a gain in efficiency and is based on sound statistical theory, but the performance of Bayesian TSG models has been lower than heuristically trained DOP models. This may be simply due to the fact that the former models are parsimonious, which leads to smaller models with less fragments.

The definition of a DOP model can be broken down into four parts (Bod, 1995a):

FRAGMENTS: what are the units on which the model operates?
OPERATIONS: what operations can be performed to combine or alter fragments?
ESTIMATION: how will the probability of performing operations on particular fragments be determined?
DISAMBIGUATION: how will the most appropriate parse tree be selected among candidates?

1.3.1 DOP OPERATIONALIZED WITH TREE-SUBSTITUTION GRAMMAR

The first instantiation of DOP is DOP1 (Bod, 1992), which is a Probabilistic Tree-Substitution Grammar (PTSG).[14] A tree-substitution grammar can be seen as a context-free grammar which rewrites trees instead of strings. It is defined by a set of elementary trees and a substitution operation which combines these trees until they form a derivation of a complete sentence.

A derivation is defined as a sequence of elementary trees combined through left-most substitution. Left-most substitution is defined for any two trees t_1 and t_2, such that t_1 has a frontier node labeled X and root$(t_2) = X$; the result of $t_1 \circ t_2$ is a new tree where t_2 is substituted for the first frontier node labeled X in t_1. The probability of a derivation is the product of the weights of its elementary trees.

In general, a tree-substitution grammar is not equivalent to a context-free grammar. However, in the case of DOP1, the set of elementary trees is such that their generative powers are in fact identical. Specifically, all fragments are built up out of CFG rules, and all CFG rules are themselves fragments of depth 1, so the generative power must coincide.

Although the generative power of the underlying grammar is identical to a context-free grammar, probabilities are estimated not just on the basis of the frequencies of CFG rules, but by considering all connected fragments of the trees in the training[15] data. More specifically, a fragment of a tree is a tree of depth ≥ 1, such that every node has a corresponding node in the original tree, and has either no children, or the same children as in the original tree. When a node in a fragment has zero children, it is called a frontier node. Frontier nodes are the substitution sites of fragments, and correspond to open slots in constructions. Figure 1.3 shows the bag of fragments extracted from a sentence; Figure 1.4 shows a DOP1 derivation with these fragments.

Since these fragments can cover an arbitrary number of terminals & non-terminals, the independence assumptions made in parsing are much weaker, and much more information from the training data is exploited during parsing. It is tempting to conclude that DOP models *all* possible statistical dependencies, because DOP uses *all* fragments. This is not true, however, for several reasons. For one, there could be more general definitions of what constitutes a fragment; e.g., relaxing the assumption that a 'fragment' must be a connected subset. Furthermore, certain statistical regularities cannot be captured using frequencies of fragments, such as Markov processes or phenomena that violate the place-invariance assumption. Lastly, and most importantly, while DOP1 is strong on modeling structural relations, it is not sensitive to lexical dependencies (Sima'an, 2000). The DOP1 model does weaken both the context-free and

---

14 Sometimes the name Stochastic Tree-Substitution Grammar (STSG) is used, but 'probabilistic' avoids confusion with synchronous grammars.

15 Technically, a DOP model is not trained, because its probabilities are directly derived from data. We will, however, maintain the terminology of training and testing to distinguish the part of the data which the parser has at its disposal and the strictly separated part which the model is evaluated on.
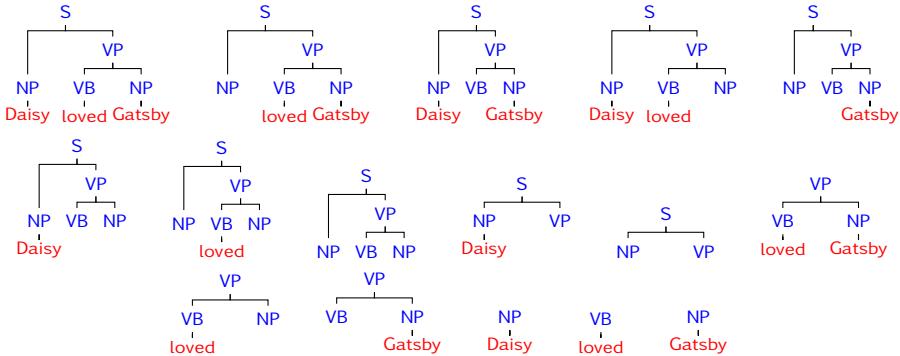
Figure 1.3: The fragments as extracted from "Daisy loved Gatsby."
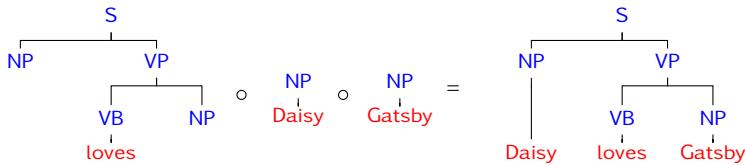
Figure 1.4: A DOP1 derivation. Note that "Daisy" becomes the subject, because fragments are combined with left-most substitution.

the ancestor-free assumptions made by PCFG models, through its probabilities of larger fragments. Additionally, there can be multiple sequences of fragments which cover the sentence, because there will be overlap. This so-called spurious ambiguity should be exploited because it allows a more fine-grained comparison of possible analyses for a sentence.

This suggests two fundamental methods of disambiguation based on frequencies: the most probable derivation (MPD), and the most probable parse (MPP). The former maximizes the probability of an individual derivation (one sequence of fragments leading to a complete analysis). The latter maximizes the sum of derivations leading to the same analysis, i.e., we choose t to maximize

$$P(t) = \sum_{d \in D(t)} \prod_{f \in d} p(f)$$

where $D(t)$ is the set of possible derivations of t. Bod (1995b) cites a score of 65% when using the MPD, and 96% with the MPP. Unfortunately, this step of calculating not just the MPD but the MPP is often neglected in DOP-inspired tree-substitution grammars (O'Donnell et al., 2009; Cohn et al., 2009; Post and Gildea, 2009), in pursuit of a more economical or efficient model. However, there are arguments for keeping track of all possible frequencies, namely the importance of frequency in grammaticalization and the formation of idioms (Bybee, 2007).

Since it appears that arbitrary units can partake in this process, all fragments & frequencies must be available. This leaves the door open to topics such as language change & acquisition, instead of modeling a parsimonious but synchronic snapshot provided by the sample that is the training corpus.

It has been shown that the problem of finding the most probable parse is NP-hard (Sima'an, 2002). Consider that there is an exponential number of fragments for each tree, hence a potentially exponential number of derivations, and it follows that the exact best parse cannot be identified in polynomial time, in the general case. However, this is not a problem in practice, as there are methods to approximate the best parse effectively, using any number of random or best derivations.

There is also a non-probabilistic method of disambiguation, the shortest derivation (Bod, 2000). The objective is to minimize the length of the derivation. In order to break ties of multiple shortest derivations, some additional criterion is necessary. An example of this is the most probable shortest derivation (MPSD), which breaks ties by looking at derivation probabilities.

1.3.2 ESTIMATORS

In DOP1 the probability of a fragment f from the set of all fragments $\mathcal{F}$ being substituted for a frontier node with label $\text{root}(f)$ in a derivation is given by its relative frequency:

$$\frac{\text{freq}(f)}{\sum_{f' \in \mathcal{F}'} \text{freq}(f')} \quad \text{where } \mathcal{F}' = \{ f' \in \mathcal{F} \mid \text{root}(f') = \text{root}(f) \}$$

Johnson (2002a) has shown that this estimator is *biased* and *inconsistent*. The bias of an estimator is the difference between the estimator's expected value[16] and the true value of the parameter being estimated. For a DOP estimator, a sample consists of a sequence of parse trees (a training corpus) sampled from the true distribution; the parameter being estimated is a distribution over parse trees. A DOP estimator is biased *iff* there is a distribution such that the estimator's expected probability distribution given all training corpora of a certain size has a non-zero difference with the true distribution. Bias can be a good thing: it allows the estimator to make systematic generalizations not licensed by the data, e.g., a preference for re-use. It has been shown that an unbiased DOP estimator is useless: it must assign a weight of zero to any parse tree not part of the training corpus (Prescher et al., 2003). Still, the kind of bias is crucial. An issue with DOP1 is that it has a bias for larger trees Bonnema et al. (1999). There are many more large trees than small trees; analogously, a large tree has many more fragments than a small tree. This is reflected in DOP1's probabilities since these directly reflect frequencies. However, it is rather easy to remedy this particular deficiency by scaling the probabilities appropriately, as suggested

---

16 The expected value of an estimator is the average estimate given all possible samples.

by Bonnema et al. (1999) and Goodman (2003). Goodman's strategy, the equal weights estimate (EWE), is employed by Bod (2003) and yields good results.

A more serious challenge is inconsistency. An estimator is consistent *iff* the predictions of the estimator get arbitrarily close to the real distribution as the amount of training data grows to infinity. Note that this property only guarantees convergence in the limit; an estimator can be consistent without performing well on real-world corpora, and *vice versa*. A reason for DOP1's inconsistency is the fact that, by design, it reserves probability mass for all fragments, even those for which productivity has not been attested. For example, in the Wall Street Journal, the contraction 'won't' is annotated as two words, but 'wo' does not combine with any other word, so the true distribution simply assigns a probability of zero for any fragment containing 'wo' but not 'n't', while DOP1 will always reserve probability mass for such combinations which may never materialize (Zuidema, 2007).

Observe why bias and consistency are orthogonal: bias is a property of all the estimates taken together (does the mean of the estimates equal the true distribution?), whereas consistency is a property of a sequence of estimators (will it progress towards and reach the true distribution?). If we take as an example throwing darts at a dartboard while aiming for the bulls-eye, then hitting a circular pattern around the bulls-eye is unbiased, no matter the error, while consistency refers to approaching the bulls-eye after practice.

Zuidema (2006) argues convincingly that bias and consistency are simply not useful criteria for judging PTSG estimators: the problem of estimating fragment weights from corpora is *underdetermined*. We can also consider the frequencies of trees produced by a PTSG compared to the expected frequencies of trees in the true distribution. In this case it is possible to achieve consistency, as DOP* does, but only with an estimator that, in the limit, assigns all its weight to full parse trees. Different criteria such as speed of convergence (Zollmann, 2004) or robustness in the face of noise might yield more immediate benefits.

Despite these theoretical issues, the best results in DOP on the Penn tree-bank have been attained with inconsistent estimators based on relative frequencies (Bod, 2003; Sangati and Zuidema, 2011).

1.3.3   The importance of fragment selection

Until now we have considered the statistical properties of estimators and whether they assign the right probabilities. A more fundamental question is which productions the grammar should be composed of. In the case of DOP, this amounts to which fragments to assign a non-zero weight. The minimal model which corresponds to a treebank PCFG includes all productions of the training set which guarantees that those trees can be generated. Between this minimalist model and the maximalist all-fragments model, there is a whole spectrum of possible choices of fragments. It is important to note that while the fragments are extracted from the treebank, they are not *observed* as such, since

there is no evidence of fragment boundaries; another way of stating this is that the treebank provides a corpus of complete parse trees, but not of derivations with derivation steps. This choice of fragments can be considered from a statistical point of view, but this is not necessary. It is also possible to turn to heuristics, as we will see in the next chapter.

In my experience getting the right productions is the more decisive aspect in practice: without the right productions or fragments, no amount of statistical sophistication will produce the right analyses. In terms of the DOP biases, this means that the memory bias trumps the probabilistic bias. If a given sentence fragment has been memorized, there is no need to determine its internal structure using probabilistic reckoning. Concretely, this means that, all else being equal, a larger fragment grammar will typically perform better, even with simplistic relative-frequency weights.

Relatedly, all manner of preprocessing steps are more crucial and contribute more to the final performance of the model than its statistical modeling or clever optimization techniques. A case in point is the Collins parser, a parser that exploits bilexical dependencies and incorporates an impressive amount of linguistic knowledge (Collins, 1999). However, the efforts of Bikel (2004) at reimplementing the Collins parser show

> that bilexical dependencies are barely used by the model and that head choice is not nearly as important to overall parsing performance as once thought.

Various details that were previously unpublished accounted for an 11 % relative error reduction, while a host of published preprocessing steps amount to an even more dramatic improvement over the treebank PCFG baseline.

1.4 REVISITING COMPETENCE AND PERFORMANCE

The notions of competence and performance (Chomsky, 1965) have been influential in linguistics, including computational linguistics.[17] Linguistic competence comprises a language user's "knowledge of language," usually described as a system of rules, while linguistic performance includes the details of the user's production and comprehension behavior. For a computational model, its syntactic competence defines the set of possible sentences that it can process in principle, and the structures it may assign to them, while its performance includes such aspects as disambiguation using occurrence frequencies of grammatical constructions. Thus, the choice of a formalism to describe the system's

---

17 The notions of competence and performance go further back. De Saussure introduced the related but not equivalent concepts *langue* and *parole*. Both have been influenced by Wilhelm von Humboldt's notions of *energeia* (activity) and *ergon* (the product of such activity), which are again not equivalent to the previous distinctions. However, the details shall not concern us since in this section we are specifically concerned with the influence of the notion of competence on computational linguistics.

competence grammar depends on one's decisions on how syntax should be formalized.

Regular and context-free grammars have been argued to be too limited (Chomsky, 1956; Shieber, 1985), while richer alternatives—context-sensitive and beyond—are considered too powerful to allow for an efficient computational implementation; this applies to Transformational Grammar (Peters and Ritchie, 1973), Lexical-Functional Grammar, and Head-Driven Phrase Structure Grammar (Trautwein, 1995). A third option is to seek a careful comprise, such as Mildly Context-Sensitive grammar (Joshi, 1985), which are beyond context-free, but avoid the computational complexity that comes with the full class of context-sensitive grammars. This has been relatively successful, although there always seem to be recalcitrant phenomena that fall outside the generative capacity of a particular formalism, rendering it unlikely that the search for the 'true' formalism will converge on describing all of natural language.

Irrespective of whether one accepts the competence-performance dichotomy, a practical natural language system needs to deal with phenomena that depend on world knowledge reflected in language use (e.g., the fact that in "*eat pizza with a fork*", *with a fork* is prototypically related to *eat* rather than to *pizza*). This has led to a statistical turn in computational linguistics, in which models are directly induced from treebanks (Scha, 1990; Charniak, 1996; Bod et al., 2003; Geman and Johnson, 2004). If the end goal is to make an adequate model of language *performance*, there is actually no need to have a competence grammar which is 'just right.' Instead, we might reduce some of the formal complexity by encoding it in statistical patterns. Concretely, we can opt for a grammar formalism that deliberately overgenerates, and count on grammatical analyses having a higher probability of being selected during disambiguation. This operationalizes the idea of there being a spectrum between ungrammaticality, markedness, and felicity. In a later chapter (Section 3.3.1) we introduce an approximation of LCFRS that makes it possible to produce discontinuous constituents in cubic time using a context-free grammar, by encoding information in non-terminal labels. A probabilistic variant of the resulting grammar makes stronger independence assumptions than the equivalent LCFRS, but as a component in a larger statistical system this does not have to pose a problem.

In the debate about the context-freeness of language, cross-serial dependencies have played an important role (Huybregts, 1976; Bresnan et al., 1982; Shieber, 1985). Consider the following example in Dutch:

(3)   Jan zag dat Karel hem haar laat leren zwemmen.
         Jan saw that Karel him her   lets teach swim.
         'Jan saw that Karel lets him teach her to swim.'

Ojeda (1988) gives an account using discontinuous constituents; cf. Figure 1.5. In Section 3.3.1 we show how such analyses may be produced by an overgenerating context-free grammar.

This is an instance of the more general idea of approximating rich formal
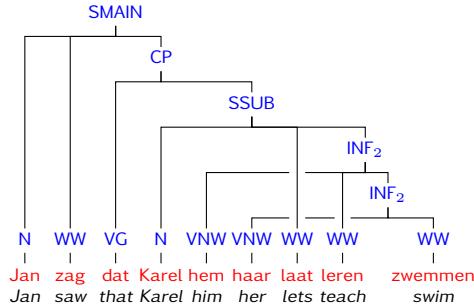
Figure 1.5: Cross-serial dependencies in Dutch expressed with discontinuous constituents.

models in formally weaker but statistically richer models, i.e., descriptive aspects of language that can be handled as a performance rather than a competence problem. Another instance of this is constituted by the various restricted versions of TAG, whose string languages form a proper subset of those of LCFRS. Restricted variants of TAG that generate context-free string languages are Tree-Insertion Grammar (Schabes and Waters, 1995; Hoogweg, 2003; Yamangil and Shieber, 2012), and off-spine TAG (Swanson et al., 2013); TSG is an even more restricted variant of TAG in which the adjunction operation is removed altogether. These results suggest that there is a trade-off to be made in the choice of formalism. While on the one hand Mild Context-Sensitivity already aims to limit formal complexity to precisely what is needed for adequate linguistic description, a practical, statistical implementation presents further opportunities for constraining complexity.

The idea that non-local relations can be expressed in context-free grammar is not new. For example, Schmid (2006) uses slash features in non-terminal labels, which in turn was inspired by generalized phrase structure grammar (GPSG; Gazdar et al., 1985). GPSG was a project which aimed to offer both a practical and formal description of natural language in a context-free framework using innovations such as generated grammar rules, feature instantiations, and separation of immediate dominance and linear precedence. However, as the evidence against the context-freeness of language surfaced, and GPSG was displaced by its successor Head-Driven Phrase Structure Grammar (HPSG), which as a unification-based formalism lacks the computational properties to ensure efficient analysis, the aim of restricting the complexity of grammar formalisms lost appeal, and the notion that the unbounded possibilities of competence should dictate the capabilities of the grammar formalism was, tacitly or not, reinstated.

Another performance aspect of language relevant for computational linguistics is pruning. While normally considered an implementation aspect made necessary by practical computational limitations, finding linguistically and psychologically plausible shortcuts in language processing forms an interest-

ing research question. Schuler et al. (2010) present a parser with human-like memory constraints based on a finite-state model. Although Roark et al. (2012) are not concerned with cognitive plausibility, they also work with finite-state methods and show that CFG parsing can be done in quadratic or even linear time with finite-state pruning methods.

As a specific example of a cognitive limitation relevant to parsing algorithms, consider center embedding. Karlsson (2007) reports from a corpus study that center embeddings only occur up to depth 3 in written language, and up to depth 2 in spoken language. If a statistical parser would take such cognitive limitations into account, many implausible analyses could be ruled out from the outset. More generally, it is worthwhile to strive for an explicit performance model that incorporates such cognitive and computational limitations as first class citizens.

In this work we do not go all the way to a finite-state model, but we do show that the non-local relations expressed in discontinuous constituents can be expressed in a context-free grammar model. We start with a mildly context-sensitive grammar formalism to parse discontinuous constituents, augmented with tree substitution. We then show that an approximation with context-free grammar is possible and effective. We find that the reduced independence assumptions and larger contexts taken into account as a result of tree substitution make it possible to capture non-local relations without going beyond context-free. Tree substitution thus increases the capabilities of the performance side without increasing the complexity of the competence side. A performance phenomenon that is modeled by this is that non-local relations are only faithfully produced as far as observed in the data.

2 Extracting recurring tree fragments

In which we present an efficient method for finding recurring patterns in treebanks, which we will use to build grammars and analyze texts.

> Einmal ist keinmal.
> *Once doesn't count*
> — German proverb

WE PRESENT an algorithm and implementation for extracting recurring fragments from treebanks. Using a tree-kernel method the largest common fragments are extracted from each pair of trees. The algorithm presented achieves a thirty-fold speedup over the previously available method on the Wall Street Journal data set. It is also more general, in that it supports trees with discontinuous constituents. The resulting fragments can be used as a tree-substitution grammar or in classification problems such as authorship attribution and other stylometry tasks.

Treebanks are a rich source of lexical and structural patterns. A simple and common approach is to consider the frequencies of individual grammar productions; the main example being treebank grammars for parsing (Charniak, 1996), but also stylometry (cf. Baayen et al., 1996; Raghavan et al., 2010; Ashok et al., 2013). Richer patterns involve multiple lexical items or constituents; i.e., they may consist of the co-occurrence of a sequence of productions that make up a specific phrase or a grammatical construction. Kernel methods, which quantify similarity by decomposing a signal into components, and specifically tree-kernel methods (Collins and Duffy, 2001, 2002), consider such patterns but obtain only a numeric value about the degree of similarity of structures,[1] without making explicit what the structures have in common. The usefulness of extracting explicit fragments, however, is underscored by one of the conclusions of Moschitti et al. (2008, p. 222):

> The use of fast tree kernels (Moschitti, 2006a) along with the proposed tree representations makes the learning and classification

---

[1] In fact, the practice of using a kernel to quantify similarity without making it explicit is referred to as the 'kernel trick' in the machine learning literature.

> much faster, so that the overall running time is comparable with polynomial kernels. However, when used with [Support Vector Machines] their running time on very large data sets (e.g., millions of instances) becomes prohibitive. Exploiting tree kernel-derived features in a more efficient way (e.g., by selecting the most relevant fragments and using them in an explicit space) is thus an interesting line of future research.

Post and Bergsma (2013) report experiments demonstrating this difference in efficiency between implicit and explicit features. svm-like, adaptive algorithms now exist (Crammer et al., 2006; Shalev-Shwartz et al., 2011), which may overcome the efficiency challenge. but that still leaves that fact that the enormous feature space may be inherently unwieldy, and being able to reason about an explicit list of features remains useful.

Aside from their use as features in machine learning tasks, tree fragments also have applications in computational linguistics for statistical parsing and corpus linguistics.

Since we will focus on the problem of finding the largest common fragments in tree pairs, there is an intuitive relation to the problem of finding all longest common subsequences of a string pair. However, in the case of tree structures the problem is more constrained than with sequences, since any matching nodes must be connected through phrase-structure.

An algorithm for extracting recurring phrase-structure fragments was first presented by Sangati et al. (2010). Their algorithm is based on a Quadratic Tree Kernel that compares each node in the input to all others, giving a quadratic time complexity with respect to the number of nodes in the treebank. Moschitti (2006b) presents the Fast Tree Kernel, which operates in linear average time. However, his algorithm only returns a list of matching nodes. This chapter presents an algorithm that exploits the Fast Tree Kernel of Moschitti (2006b) to extract recurring fragments, providing a significant speedup over the quadratic approach.

2.1 Applications, related work

The two main applications of tree fragment extraction so far are in parsing and classification problems.

Tree fragments can be used as grammar productions in Tree-Substitution Grammars (tsg). tsgs are used in the Data-Oriented Parsing framework; dop (Scha, 1990; Bod, 1992). In Data-Oriented Parsing the treebank is considered as the grammar, from which all possible fragments can in principle be used to derive new sentences through tree-substitution. Grammar induction is therefore conceptually straightforward (although the grammar is very large), as there is no training or learning involved. This maximizes re-use of previous experience.

Since representing all possible fragments of a treebank is not feasible (their number is exponential in the number of nodes), one can resort to using a subset, but sampling or arbitrary restrictions are likely to lead to a suboptimal set of fragments, since the vast majority of fragments occur only once in the treebank (Sangati et al., 2010). Double-DOP (2DOP; Sangati and Zuidema, 2011) avoids this by restricting the set to fragments that occur at least twice. The heuristic of this model is to construct the grammar by extracting the largest common fragments for every pair of trees, just as the tool presented in this chapter. This model has been generalized to handle trees with discontinuous constituents (cf. Chapter 3).

An alternative to the all-fragments assumption of DOP takes a Bayesian approach to selecting fragments and assigning probabilities (O'Donnell et al., 2009; Post and Gildea, 2009; Cohn et al., 2010; Shindo et al., 2012) Such Bayesian TSGs are induced by sampling fragments using Markov Chain Monte Carlo (MCMC) following a Zipfian long tail distribution.

Aside from the generative use of fragments in TSGs, tree fragments are also used for discriminative re-ranking. Implicit fragment features (counted but not extracted) are used in Collins and Duffy (2001, 2002) through a tree kernel, and explicit fragment features are used in Charniak and Johnson (2005, p. 178: HeadTree and NGramTree features). Another discriminative application of recurring fragments is in text classification tasks. Common tree fragments can be used to define a similarity measure between texts, which can be applied to the task of authorship attribution (cf. Section 7.1). In the latter case the efficiency of extracting fragments has been exploited by extracting fragments at classification time, i.e., a memory-based approach, without defining features in advance. Other classification tasks have been modeled with fragments induced by Bayesian TSGs: e.g., native language detection (Swanson and Charniak, 2012), stylometry of scientific papers (Bergsma et al., 2012), and corpus analysis of source code (Allamanis and Sutton, 2014).

Finally, there is a tradition in the data mining literature called frequent tree mining or frequent structure mining (Jiménez et al., 2010). This tradition does not apply the maximal fragment constraint and explores different kinds of fragments and trees. The approach is based on the Apriori algorithm and works by generating candidates that occur with a specified minimum frequency. However, the relaxation of the maximality constraint results in an exponential number of fragments, while for linguistic applications the focus on frequent fragments is arguably in conflict with the importance of the long tail in language, specifically, the importance of low frequency events. See Martens (2010) though, which applies a frequent tree mining approach to unordered dependency trees.

2.2 Definitions

The notion of a tree fragment is defined as follows:

Definition 1. A *fragment* f of a tree T is a connected subgraph of T, such that

f contains at least 2 nodes, and each node in f either is an empty node (a substitution site), or immediately dominates the same immediate children as the corresponding node in T.

For the purposes of fragment extraction, we identify a non-terminal node of a tree with its (grammar) production. The production at a non-terminal node is a tuple of labels of that node and its immediate children. A label is either the phrase label of a non-terminal node, or the lexical item of a terminal node.

Note that in Moschitti (2006a), this fragment definition corresponds to the subset tree (SST) kernel. The SST kernel can be contrasted with the subtree (ST) kernel where all descendants of a subtree are retained, and the partial tree (PT) kernel where nodes may include a subsequence of the children of the corresponding node in the original tree. We only address the SST kernel. The ST kernel is simple but not able to recognize fragments with any lexical or syntactic variation, while the PT kernel is more complex, allowing for fragments with partially matching constituents, which makes it possible to factor out optional elements (e.g., adjuncts, modifiers) from fragments. However, this capability of finding partial matches among constituents can be simulated to an extent in the SST kernel with the right kind of binarization.

Concretely, a binarization scheme may relax the requirement that an n-ary constituent is generated in an all-or-nothing manner, by factoring it into a sequence of binary nodes that may be applied separately, and recombined into new configurations. Klein and Manning (2003) present such a scheme, which binarizes parse trees head-outward with two markovization parameters: h and v; with $h = \infty$ and $v = 1$ being the binarization that behaves equivalently to the original n-ary trees in terms of parsing or fragment extraction. By tweaking these parameters, generalizations can be both increased and constrained. Reducing h limits the required horizontal context to a specified number of siblings, while v increases the vertical context by specifying parent labels that need to match. In terms of fragment extraction, this means that only a prefix of children of a node needs to match, with h determining the minimum length. Additionally, the binarization method offers more control than the partial tree kernel, since the latter has no principled definition of arguments and adjuncts, it may also leave out essential elements from fragments (e.g., consider verbs that are both transitive and intransitive). This distinction could in principle be added in the binarization scheme.

The subset of possible recurring fragments we consider is defined by the largest overlapping fragments for any pair of trees in the input:

DEFINITION 2. Given a pair of trees $\langle a, b \rangle$, and a pair of nodes $\langle p, q \rangle$ such that p is a node in a, q is a node in b, with p and q having the same production, the *maximal common fragment* defined by $\langle p, q \rangle$ is the largest fragment f that occurs in a incorporating node p and occurs in b incorporating node q.

Note that our definition of maximal common fragment differs from the one in Sangati et al. (2010, sec. 2):

"A (partial) fragment τ shared between [two] trees is maximal if there is no other shared (partial) fragment starting with the same node as in τ and including all its nodes."
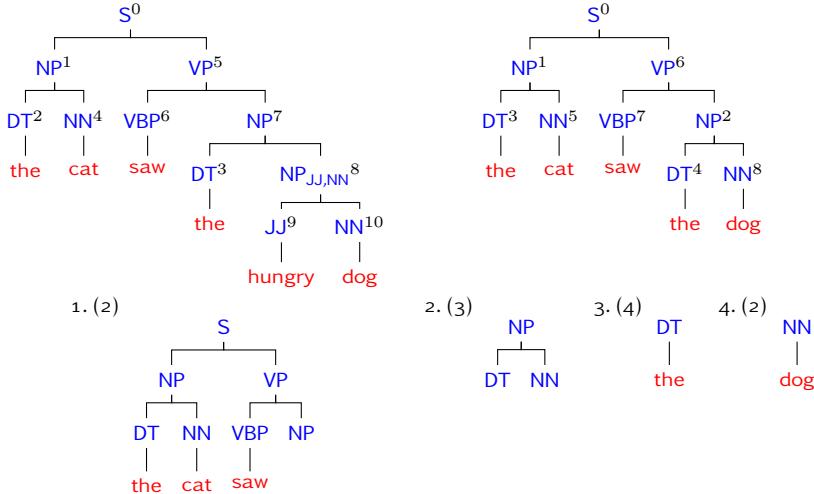
Figure 2.1: Top: two example trees containing recurring patterns; the superscript denotes an identifier for each node. Bottom: all maximal fragments in the pair of trees on the left (frequencies in parentheses). Note that the second fragment is a subgraph of the first fragment, but is still a maximal fragment when considering $\langle NP^1, NP^2 \rangle$ of the first and second tree, respectively.

Furthermore, in their formulation, maximality of fragments is determined with respect to the first tree in the comparison (cf. Sangati et al., 2010, Algorithm 2, second statement). Our formulation determines maximality with respect to node pairs, i.e., involving both trees. A consequence of this is that for the algorithm of Sangati et al. (2010), the order of the trees in the input can have an effect on the result; i.e., the extraction of fragments from a node pair is not a commutative operation. Consider a pair of trees and the fragments that FragmentSeeker (Sangati et al., 2010) extracts from them:

When the order of the input is reversed, the second fragment is not extracted, because it is a subset of the first. With our algorithm following Definition 2, the two fragments are extracted regardless of the order of the input.

We are also interested in the frequencies of fragments. The frequency is not defined with respect to occurrences as maximal common fragment, but with respect to all occurrences:

Definition 3. The *occurrence count* of a fragment in a treebank is the total number of occurrences of a fragment in a collection of trees.

As an example of the preceding definitions, see Figure 2.1, which shows two trees and their maximal common fragments with occurrence counts.

2.3 Fragment extraction with tree kernels

In this section we first discuss the Fast Tree Kernel, followed by the algorithm to extract fragments from the matching nodes it finds. We then discuss two extensions that find occurrence counts and handle discontinuous constituents.

Algorithm 1 lists the pseudocode for the Fast Tree Kernel (FTK) by Moschitti (2006b). The pseudocode for extracting fragments is shown in Algorithm 2; the main entry point is FUNCTION recurring-fragments. This formulation is restricted to binary trees, without loss of generality, since trees can be binarized into a normal form and debinarized without loss of information.

Sangati et al. (2010) define the extraction of fragments as considering all pairs of trees from a single treebank. Since it may also be interesting to investigate the commonalities of two different treebanks, we generalize the task of finding recurring fragments in a treebank to the task of finding the common fragments of two, possibly equal, treebanks. In case the treebanks are equal, only half of the possible tree pairs have to be considered: the fragments extracted from $\langle t_n, t_m \rangle$, with $n < m$, are equal to those of $\langle t_m, t_n \rangle$; we ignore the trivial case $n = m$ since it produces full parse trees rather than fragments.

2.3.1 The Fast Tree Kernel

See Algorithm 1 for the pseudocode of the Fast Tree Kernel. The insight that makes this kernel fast on average is that the problem of finding common productions can be viewed as the problem of finding the intersection of two multisets that have been sorted in advance. The input of the function fast-tree-kernel is a pair of trees $\langle a, b \rangle$, that are represented as a list of nodes sorted by their productions.[2] The requirement for sorted input depends on an ordering defined over the grammar productions, but note that the nature of this ordering is irrelevant, as long as it is consistently applied (e.g., productions may be sorted lexicographically by the labels of the left and right hand sides of productions;

---

2 The requirement for sorted input does not affect the asymptotic complexity of the algorithm because each tree is sorted separately, so is not affected by the total number of nodes in the treebank. Furthermore, the sorting is done offline and only once.

INPUT: A pair of trees $\langle a, b \rangle$, with the nodes of each tree sorted by a common ordering on their productions. $a[i]$ returns the i-th production in a.

OUTPUT: A boolean matrix $\mathcal{M}$ with $\mathcal{M}[i, j]$ true *iff* the production at $a[i]$ equals the one at $b[j]$.

1: FUNCTION fast-tree-kernel(a, b)
2:     $i \leftarrow 0, j \leftarrow 0$
3:     $\mathcal{M} \leftarrow |a| \times |b|$ boolean matrix, values initialized as false.
4:     WHILE $i < |a| \wedge j < |b|$
5:         IF $a[i] < b[j]$
6:             $j \leftarrow j + 1$
7:         ELSE IF $a[i] > b[j]$
8:             $i \leftarrow i + 1$
9:         ELSE
10:            WHILE $a[i] = b[j]$
11:                $j\prime \leftarrow j$
12:                WHILE $a[i] = b[j\prime]$
13:                    $\mathcal{M}[i, j\prime] \leftarrow$ *true*
14:                    $j\prime \leftarrow j\prime + 1$
15:                $i \leftarrow i + 1$

Algorithm 1: The Fast Tree Kernel, adapted from Moschitti (2006b).

or productions can be assigned a sequence number when they are first encountered). The output is a boolean matrix where the bit at (n, m) is set *iff* the nodes at those indices in the respective trees have the same production. From this table the bitvectors corresponding to fragments are collected and stored in the results table.

Given the trees from the introduction as input, the resulting set of matching node pairs can be seen as a matrix; cf. Table 2.1. The matrix visualizes what makes the algorithm efficient: there is a diagonal path along which comparisons have to be made, but most node pairs (i.e., the ones with a different label) do not have to be considered. The larger the number of production types, the higher the efficiency. In case there is only a single non-terminal label, and hence only one phrasal, binary production, the efficiency of the algorithm degenerates and the worst-case quadratic complexity obtains.

2.3.2 EXTRACTING MAXIMAL CONNECTED SUBSETS

After the matrix with matching nodes has been filled, we identify the maximal fragments that matching nodes are part of. We traverse the second tree in depth-first order, in search for possible root nodes of fragments; cf. Algorithm 2, line 6.

The fast tree kernel in Algorithm 1 returns matching node pairs as output. The resulting adjacency matrix $\mathcal{M}$ is iterated over in line 8. This may appear to

	S^0	NP^1	NP^2	DT^3	DT^4	NN^5	VP^6	VBP^7	NN^8
S^0	1								
NP^1		1	1,2						
DT^2				1,3	3				
DT^3				3	3				
NN^4						4			
VP^5							1		
VBP^6								1	
NP^7									
$NP^8_{JJ,NN}$									
JJ^9									
NN^{10}									1

Table 2.1: Matrix when trees in Figure 2.1 are compared. Highlighted cells are nodes with a common production. The numbers indicate the fragments extracted, corresponding with those in Figure 2.1. Note that for expository purposes, the nodes are presented in depth-first order.

be quadratic in the number of nodes of the trees, but only the cells for matching node pairs need to be visited, and each pair is visited once. Since it is possible to scan for cells with 1-bits efficiently (cf. Section 2.5), the linear average time complexity is maintained.

Whenever a 1-bit corresponding to a matching node pair is encountered, a fragment with that node pair is extracted. Both trees are traversed in parallel, top-down from that node pair onwards, to collect the subset of nodes for the fragment; cf. Algorithm 2, line 9. Both trees need to be considered to ensure that extracted subsets are connected in both trees. Every node pair that is visited contributes a subtree corresponding to the production at the node pair, which is added to the fragment that is being constructed. The node pair is marked such that it will not be used in another fragment.

Note that the algorithm of Sangati et al. (2010) combines the tree kernel and extraction of maximal connected subsets in a single pass, and is thus a dynamic programming approach. That is, as the algorithm iterates over possibly matching pairs of nodes, a new fragment may result which subsumes a fragment extracted at an earlier stage. Our approach is able to use a greedy algorithm for finding maximal subgraphs by using a two pass approach consisting of first finding all matching nodes, and then extracting fragments.

INPUT: A treebank *TB*, with the nodes of each tree sorted by their productions.
OUTPUT: A set F with maximal recurring fragments from *TB*.
  1: FUNCTION recurring-fragments(*TB*)
  2:     $F \leftarrow \varnothing$
  3:     FOR ALL $\langle a, b \rangle \in TB \times TB$ with $a \neq b$
  4:        $\mathcal{M} \leftarrow$ fast-tree-kernel(a, b)
  5:        $F \leftarrow F \cup$ fragments$(a, b, \text{root}(b), \mathcal{M})$

INPUT: a, b are trees with $a[i]_l, a[i]_r$ the index of
    the left and right child of node i of a (*mutatis mutandis* for $b[j]_l$ and $b[j]_r$).
    $\mathcal{M}$ is a boolean matrix with $\mathcal{M}[i, j]$ true *iff* the production at $a[i]$ equals the
    one at $b[j]$.
OUTPUT: The set F of maximal fragments in a and b.
  6: FUNCTION fragments$(a, b, j, \mathcal{M})$ {Traverse tree b top-down starting from j.}
  7:     $F \leftarrow \varnothing$
  8:     FOR ALL i such that $\mathcal{M}[i, j]$
  9:        $F \leftarrow F \cup$ extract-at$(a, b, i, j, \mathcal{M})$
10:     IF has-left-child$(b[j])$
11:        $F \leftarrow F \cup$ fragments$(a, b, b[j]_l, \mathcal{M})$
12:     IF has-right-child$(b[j])$
13:        $F \leftarrow F \cup$ fragments$(a, b, b[j]_r, \mathcal{M})$

INPUT: Indices i, j denoting start of a fragment in trees a, b.
OUTPUT: The fragment f rooted at $a[i]$ and $b[j]$.
14: FUNCTION extract-at$(a, b, i, j, \mathcal{M})$ {Traverse a and b top-down, in parallel.}
15:     $f \leftarrow$ tree-from-production$(a[i])$ {Create a depth 1 subtree from a
        grammar production.}
16:     $\mathcal{M}[i, j] \leftarrow$ *false* {do not extract a fragment with this node pair again.}
17:     IF has-left-child$(a[i]) \wedge \mathcal{M}[a[i]_l, b[j]_l]$
18:        $f_l \leftarrow$ extract-at$(f, a, b, a[i]_l, b[j]_l, \mathcal{M})$ {add as left subtree}
19:     IF has-right-child$(a[i]) \wedge \mathcal{M}[a[i]_r, b[j]_r]$
20:        $f_r \leftarrow$ extract-at$(f, a, b, a[i]_r, b[j]_r, \mathcal{M})$ {add as right subtree}

Algorithm 2: Extract maximal recurring fragments given common nodes for
         each pair of trees in a treebank.

2.3.3 Occurrence counts

It is possible to keep track of the number of times a fragment is extracted. Because of the maximality constraint, this count is a lower bound on the true occurrence count of a fragment. The true occurrence count may be useful for a probabilistic model or for corpus analysis. To obtain such a count, a second pass needs to be made over the treebank, in which for each fragment, all its occurrences are counted—including occurrences that are not maximal for any tree pair.

Counting exploits an inverted index listing the set of trees which contain a given production. The inverted index is implemented with a compressed bitmap (Chambi et al., 2016) that computes intersections efficiently. By taking the intersection of the sets for all productions in a fragment, a set of candidate trees can be composed efficiently. These candidates will contain all of the productions in the fragment, but it is necessary to traverse the candidates exhaustively to confirm that the productions are present in the right configuration corresponding to the fragment.

Aside from obtaining counts, it is also possible to obtain for each fragment a vector of indices of all trees that contain the fragment. This makes it possible to inspect the contexts in which a fragment occurs—and also to address diachronic questions, if the trees in the input are presented in chronological order.

2.4 Discontinuous fragments

The aforementioned fragment extraction algorithms can be adapted to support trees with discontinuous constituents. A discontinuous constituent is a constituent whose yield does not consist of a single contiguous string, but rather a tuple of strings. As a simple example in English, consider:

(1)   a.   Wake your friend up
          b.   [$_{VP}$ Wake . . . up]

Where the phrasal verb in (1-a) may be said to constitute the discontinuous constituent (1-b).

The first treebank that introduced discontinuous constituents as a major component of its annotation is the German Negra treebank (Skut et al., 1997). Syntax trees in this style of annotation are defined as unordered trees, with the caveat that there is a total ordering of the words in the original sentence. This caveat is important because for example in the case of the problem of calculating the tree-edit distance between trees, the problem is tractable for ordered trees, but not for unordered trees (Zhang et al., 1992).

We use a transform-backtransform approach which makes it possible to use the fragment extraction algorithm without further modification. In order to use the unordered trees from treebanks with discontinuous constituents, we use the ordering of the words in the sentences to induce a canonical order for the
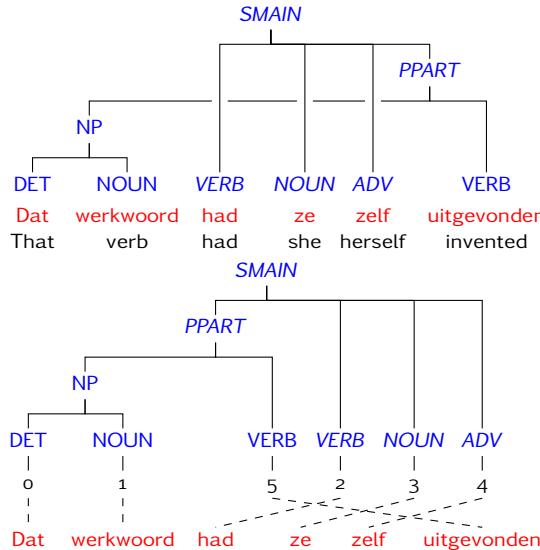
Figure 2.2: An illustration of the representation for trees with discontinuous constituents. Shown on the left is the original discontinuous tree from the Alpino treebank (van der Beek et al., 2002). On the right is a version in which the phrase structure has been decoupled from its surface form using indices.
Translation: *That verb she had invented herself.*

internal nodes of the tree. This makes it possible to use the same data structures as for continuous trees.

We use a representation where leaf nodes are decorated with indices indicating their position in the sentence (cf. Figure 2.2). Using the indices a canonical order for internal nodes is induced based on the lowest index dominated by each node.

Indices are used not only to keep track of the order of lexical nodes, but also to store where contributions of substitution sites end up relative to the contributions of other non-terminals. This is necessary in order to preserve the configuration of the yield in the original sentence. When leaf nodes are compared, the indices stand in for the token at the sentence position referred to. After a fragment is extracted, any indices need to be canonicalized. The indices originate from the original sentence, but need to be decoupled from this original context. This process of canonicalization is analogous to how a production of a Linear Context-Free Rewriting System (LCFRS) can be read off from a tree with discontinuous constituents (Maier and Søgaard, 2008), where intervals of indices are replaced by variables.

The canonicalization of fragments is achieved in three steps, as defined

in the pseudocode of Algorithm 3; Figure 2.3 illustrates the process. In the examples, substitution sites have spans denoted with inclusive *start:end* intervals, as extracted from the original parse tree, which are reduced to variables denoting contiguous spans whose relation to the other spans is reflected by their indices.

As an example of this procedure, consider two variants of a famous quotation attributed to Churchill, mocking the notion that sentences should not end with a preposition:[3]

(2)   a.   Ending a sentence with a preposition is the sort of English up with which I will not put.

        b.   This is the sort of tedious nonsense up with which Winston Churchill did not put.

INPUT: A tree fragment t with indexed terminals w_i or intervals $\langle i : j, \ldots \rangle$ as leaves
    ($0 \leqslant i < j < n$)
OUTPUT: A tree fragment with modified indices.
1: $k \leftarrow$ the smallest index in t
2: subtract k from each index in t
3: FOR ALL intervals I $= \langle i : j, \ldots \rangle$ of the frontier non-terminals in t
4:      FOR ALL $i : j \in I$
5:         replace $i : j$ with i
6:         subtract $j - i$ from all indices k s.t. $k > j$
7: FOR ALL indices i in t
8:      IF the indices $i + 1$ and $i + 2$ are not in t
9:         $k \leftarrow$ the smallest index in t s.t. $k > i$
10:       subtract $k - i$ from all indices y s.t. $y > i$

Algorithm 3: Canonicalizing discontinuous fragments.

If we apply an analysis with discontinuous constituents, these sentences contain a common verb phrase as an argument to the verb phrase "I will not" or "Winston Churchill did not", namely (3-a) and (3-b). If these are canonicalized, they map to the same fragment (3-c):

(3)   a.   [VP up$_{11}$ with$_{12}$ which$_{13}$ ... put$_{17}$]

        b.   [VP up$_7$ with$_8$ which$_9$ ... put$_{14}$]

        c.   [VP up$_0$ with$_1$ which$_2$ ... put$_4$]

However, the unmarked word order for this sentence would form a different fragment, because a fragment represents a fixed configuration:

(4)   This is the sort of tedious nonsense ...

        a.   up with which I will not put
           [VP up with which ... put]

        b.   with which I will not put up
           [VP with which ... put up]

---
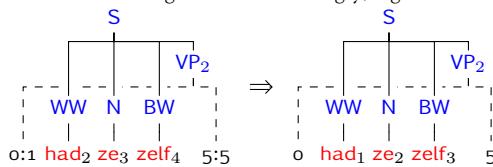
3 The (arguably unwarranted) proscription against preposition stranding is due to the influence of Latin grammar and the notion that it signals an informal style (Yáñez-Bouza, 2006).

1. Translate indices so that they start at 0; e.g.:

   WW   WW
   | $\Rightarrow$ |
   uitgevonden_5 uitgevonden_0

2. Reduce spans of frontier non-terminals to length 1;
   move surrounding indices accordingly; e.g.:

3. Compress gaps to length 1; e.g.:
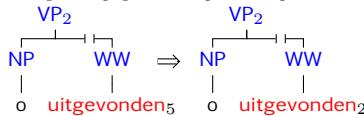
Figure 2.3: Canonicalization of fragments extracted from parse trees. These sample fragments have been extracted from the tree in Figure 3.1. The fragments are visualized here as discontinuous tree structures, but since the discontinuities are encoded in the indices of the yield, they can be represented in a standard bracketing format as used by the fragment extractor.

    c.   which I will not put up with
        [$_{VP}$ which … put up with]

When recurring fragments are extracted from the Tiger treebank (cf. Section 3.5.1), we find that 10.4 % of fragment types contain a discontinuous node (root, internal, or substitution site). This can be contrasted with the observation that 30 % of sentences in the Tiger treebank contain one or more discontinuous constituents, and that 20.9 % of production types in the PLCFRS treebank grammar of Tiger contain a discontinuous non-terminal. On the other hand, when occurrence frequencies are taken into account, both the fragments and productions with discontinuities account for around 6.5 % of the total frequency mass.

2.5 Implementation

A number of strategies have been employed to implement the fragment extraction algorithm as efficiently as possible. We use the Cython programming language (Behnel et al., 2011) which is a superset of the Python language that translates to C code and compiles to Python extension modules that seamlessly integrate with larger Python applications. We use manual memory management for the treebank and temporary arrays of the algorithm. This avoids the overhead of garbage collection and the requirement that all values must reside in an object ('boxed') as in Python and other managed languages.

    A tree is represented as an array of tightly packed structs for each node, with the grammar production represented as an integer ID, and child nodes using array indices. Mapping productions to integer IDs ensures that comparisons between productions are cheap. In contrast to a pointer-based tree, traversing this array representation does not require indirection and has good memory locality.

    When a pair of trees is compared, the results are stored in a bit matrix. The operations that need to iterate over set bits exploit CPU instructions to do this efficiently on one machine word (typically 64 bits) at a time (e.g., the 'find first set' instruction that finds the index of the first set bit of an integer). Fragments are first stored as a bitvector of a given tree, where each bit refers to the presence or absence of a particular production in the tree. These bitvectors are later converted to a string with the fragment in bracket notation. This ensures that fragments as extracted from different trees are recognized as equivalent and it is the format in which the results are returned.

    The problem of extracting fragments from a treebank is a so-called *embarrassingly parallel* problem. This means that the work on a single tree pair is not affected by any other part of the treebank, and computations can be trivially distributed over any available computing power. In order to exploit multiple processing cores, we use the Python multiprocessing library. After reading the treebank the tree pairs are distributed evenly over a pool of processes, after

which the results are collected.

Method	Corpus	Number of		Time (hr:min)	
		Trees	Fragments	Wall	CPU
Sangati et al. (2010):					
QTK	WSJ 2–21	39,832	990,156	8:23	124:04
This work:					
FTK	WSJ 2–21	39,832	990,890	0:18	4:01
FTK	Negra, train set	18,602	176,060	0:04	0:32
FTK	Gigaword, NYT 1999–11	502,424	9.7 million	9:54	∼ 160

Table 2.2: Performance comparison of fragment extraction with the Quadratic Tree Kernel (QTK) and the Fast Tree Kernel (FTK) based algorithm. Wall clock time is when using 16 cores.

2.6 Benchmark

As a benchmark we use the training sections of the WSJ and Negra treebanks (Marcus et al., 1993; Skut et al., 1997), and a section of the Annotated Gigaword (Napoles et al., 2012, nyt_eng_199911), to validate that the more efficient algorithm makes it possible to handle larger treebanks. The WSJ treebank is binarized with $h = 1$, $v = 2$ markovization (Klein and Manning, 2003) and stripped of traces and function tags. To demonstrate the capability of working with discontinuous treebanks, we use the German Negra treebank, also binarized with $h = 1$, $v = 2$ markovization, and punctuation re-attached as described in Section 3.5.1. The Gigaword section is binarized without markovization. Work is divided over 16 CPU cores to demonstrate that the algorithm lends itself to parallelization.

See Table 2.2 for a performance comparison. For the Sangati et al. (2010) implementation we use the implementation published as part of Sangati and Zuidema (2011). Because we use the markovization setting of $h = 1$, $v = 2$ described above, we get a larger number of fragments and longer running time than the results for the non-binarized WSJ treebank in Sangati et al. (2010). The times reported include the work for obtaining occurrence counts of fragments.[4] Note that there is a slight difference in the number of fragments found, but both implementations agree on 99.93 % of fragments found in the WSJ treebank. This difference is due to the difference in definition of what constitutes a maximal fragment discussed in Section 2.2. The figure of 124 hours for the WSJ treebank with the implementation of Sangati et al. (2010) might seem large, since it might be expected that the running time is dominated by the number of trees (e.g.,

---

[4] Erratum: an earlier report on this work (van Cranenburgh, 2012b) reported run times without occurrence counts, while the results of Sangati et al. (2010) include them, which exaggerated the speedup.

40K in the case of the WSJ training set), giving rise to a large number of tree pairs need to be compared, which remains constant across both implementations. However, this large number of comparisons multiplies the quadratic to average linear time speedup in terms of the number of nodes per tree.

The Fast Tree Kernel delivers a substantial asymptotic improvement: we obtain a 30-fold speedup over Sangati et al. (2010). This speedup opens up the possibility of applying fragment extraction to much larger corpora. In addition to the improvement in run time, the memory usage is eight times lower (less than 1 GB per process).

Part of this speedup is attributable to the more low-level style of programming using bitvectors and tightly packed data structures. However, in earlier experiments with a re-implementation of the algorithms specified by Sangati et al. (2010) with these optimizations, only a two-fold speedup was achieved. Therefore, most of the speedup comes from the Fast Tree Kernel and the greedy depth-first fragment extraction introduced in this chapter.

When the fragment extraction is applied to a larger number of trees from Gigaword, the extraction of fragments is still feasible, as long as occurrence counts are not requested, which does not scale as well as the other parts in terms of run time. This is because obtaining the counts requires iterating over all fragments and trees, both of which grow proportionately with treebank size. Future work should focus on optimizing this aspect of the algorithm by improving the indexing of the treebank. Perhaps this can be done using a variant of a suffix array. A suffix array of a corpus allows for arbitrary substring search in time linear to the length of the substring (Abouelhoda et al., 2004). Suffix arrays have been used for machine translation (Lopez, 2007). However, suffix arrays only deal with strings. They would either need to be generalized to tree structures, or adapted as a filter to speed up finding trees with matching terminals. Alternatively, it may be the case that for the application of Probabilistic Tree-Substitution Grammars, approximate frequencies suffice.

Figure 2.4 plots the number of fragments and run time as the number of trees is increased. It is clear that the number of fragments extracted grows linearly with the number of trees. The plot of the running time is not linear, but has a slight curvature (similar to the graphs in Moschitti 2006a). While the tree kernel is linear average time in the number of nodes, the number of tree pairs that are compared is quadratic in the number of trees, when all tree pairs are considered. The complexity can be limited further by only considering tree pairs with a certain number of overlapping lexical items, or looking only at a sample of tree pairs. A further optimization, suggested in Collins and Duffy (2001, sec. 5) and implemented in Aiolli et al. (2007), is to exploit common subtrees in the input by representing the set of trees as a single directed acyclic graph. Rieck et al. (2010) present an Approximate Tree Kernel which attains a large speedup by significantly reducing the space of fragments that are considered.
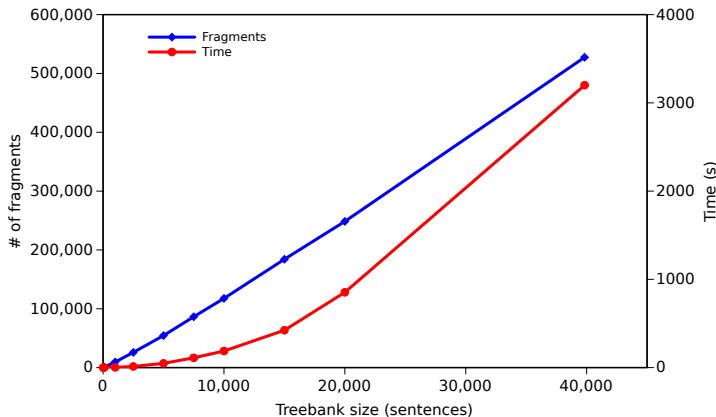
Figure 2.4: A plot of the running time and the number of fragments as a function of the number of trees in the input. The input is the Penn treebank, Wall Street Journal section 2–21, binarized $h = \infty, v = 1$; without extracting occurrence counts.

2.7 SUMMARY

We have presented a method and implementation for fragment extraction using an average case linear time tree kernel. We obtain a substantial speedup over the previously presented quadratic-time algorithm, and the resulting fragments and frequencies have been validated against the output of the latter. Additionally, we introduced support for discontinuous constituents.

3 Richer Data-Oriented Parsing models

In which we extend DOP to handle discontinuous constituents and function tags, and evaluate on multiple languages.

I N THIS chapter we investigate new techniques to implement treebank-based parsers that allow for discontinuous constituents. We present two systems. One system is based on a string-rewriting Linear Context-Free Rewriting System (LCFRS), while using a Probabilistic Discontinuous Tree-Substitution Grammar (PDTSG) to improve disambiguation performance. Another system encodes the discontinuities in the labels of phrase-structure trees, allowing for efficient context-free grammar parsing.

3.1 TOWARDS DISCONTINUOUS PARSING

When different parsing and disambiguation algorithms are applied to the same treebank, their relative accuracy scores can be objectively assessed if the treebank is split into a training set (that is used to induce a grammar and its probabilities) and a test set (that provides a "gold standard" to assess the performance of the system). This is common practice now. In many cases, however, the linguistic significance of these evaluations may be questioned, since the test sets consist of phrase-structure trees, i.e., part-whole structures where all parts are contiguous chunks. Non-local syntactic relations are not represented in these trees; utterances in which such relations occur are therefore skipped or incorrectly annotated.

For certain practical applications this restriction may be harmless, but from a linguistic (and cognitive) viewpoint it cannot be defended. Since Chomsky's transformational-generative grammar, there have been many proposals for formal grammars with a less narrow scope. Some of these formalisms have been employed to annotate large corpora; in principle, they can thus be used in treebank grammars extracted from these corpora.

The Penn treebank, for instance, enriches its phrase-structure representations with "empty constituents" that share an index with the constituent that, from a transformational perspective, would be analyzed as originating in that position. Most grammars based on the Penn treebank ignore this information, but it was
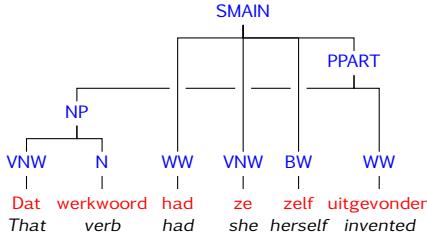
Figure 3.1: A tree from the Dutch Alpino treebank (van der Beek et al., 2002). PPART is a discontinuous constituent (indicated with crossing branches) due to its extraposed NP object. Key to part-of-speech tags: VNW=pronoun, N=noun, WW=verb, BW=adverb. The tags also contain additional morphological features not shown here, which distinguish personal pronouns from others, and auxiliary verbs from main verbs, etc. Translation: *That verb she had invented herself.*
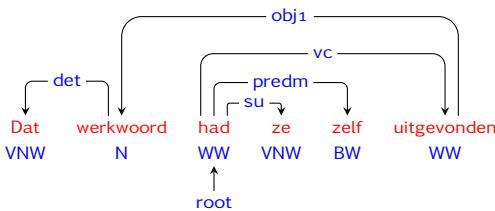
Figure 3.2: A dependency structure derived from the tree in Figure 3.1. The *obj1* arc makes this structure non-projective.

used by, e.g., Johnson (2002b), Dienes and Dubey (2003), and Gabbard et al. (2006).

Another perspective on non-local syntactic dependencies generalizes the notion of a "syntactic constituent," in that it allows "discontinuous constituent structures," where a non-terminal node dominates a lexical yield that consists of different non-contiguous parts (McCawley, 1982). Several German and Dutch treebanks have been annotated in terms of discontinuous constituency, and some statistical parsers have been developed that use these treebanks. Also, phrase structures with co-indexed traces can be converted into discontinuous constituent structures; the Penn treebank can therefore be transformed and used in the discontinuous constituency approach (Evang and Kallmeyer, 2011). Figure 3.1 shows an example of a tree with discontinuous constituents.

It is an annotation choice to employ discontinuous constituents; some treebanks elect not to model non-local phenomena, while others may choose different mechanisms. For example, two German treebanks employ discontinuous constituents (Skut et al., 1997; Brants et al., 2002), while another German treebank does not (Telljohann et al., 2004, 2012). The annotation scheme of the

latter treebank lacks information expressed in the former two. For instance, it cannot encode the heads of non-local modifiers; with discontinuous constituents, a modifier is a sibling of its head, regardless of their configuration. On the other hand, the co-indexed traces of the Penn treebank provide more information than discontinuous constituents, because they assume that constituents have been moved from somewhere else in the tree and encode the original position. Discontinuous constituents describe surface structure without making such assumptions. Some phenomena that can be analyzed with discontinuous constituents are extraposition, topicalization, scrambling, and parentheticals; cf. Maier et al. (2014) for an overview of such phenomena in German.

The notion of discontinuous constituents in annotation is useful to bridge the gap between the information represented in constituency and dependency structures. Constituency structures capture the hierarchical structure of phrases— which is useful for identifying re-usable elements; discontinuous constituents extend this to allow for arbitrary non-local relations that may arise due to such phenomena as extraposition and free word order. There is a close relation of discontinuous constituency to non-projectivity in dependency structures (Maier and Lichte, 2011). Compare Figure 3.2, which shows a dependency structure for the constituency tree in Figure 3.1. Note that in this dependency structure, the edge labels are grammatical functions present in the original treebank, while the constituent labels in Figure 3.1 are syntactic categories. The dependency structure encodes the non-local relations within the discontinuous constituent. On the other hand, it does not represent the hierarchical grouping given by the NP and PPART constituents. By encoding both hierarchical and non-local information, trees with discontinuous constituents combine the advantages of constituency and dependency structures. We will also come back to grammatical function labels.

The present chapter is concerned with treebank-based parsing algorithms that accept discontinuous constituents. It takes its point of departure in work by Kallmeyer and Maier (2010, 2013) that represents discontinuous structures in terms of a string-rewriting version of Linear Context-Free Rewriting Systems (Section 3.2.1). In addition, we employ Tree-Substitution Grammar (TSG). We make the following contributions:

1. We show that Tree-Substitution Grammar can be applied to discontinuous constituents (Section 3.2.2) and that it is possible, using a transformation, to parse with a Tree-Substitution Grammar without having to write a separate parser for this formalism (Section 3.3.2).

2. We induce a tree-substitution grammar from a treebank (Section 3.4) using a method called Double-DOP (Sangati and Zuidema, 2011). This method extracts a set of recurring tree fragments. We show that compared to another method which implicitly works with all possible fragments, this explicit method offers an accuracy and efficiency advantage (Section 3.3.2, Section 3.6).
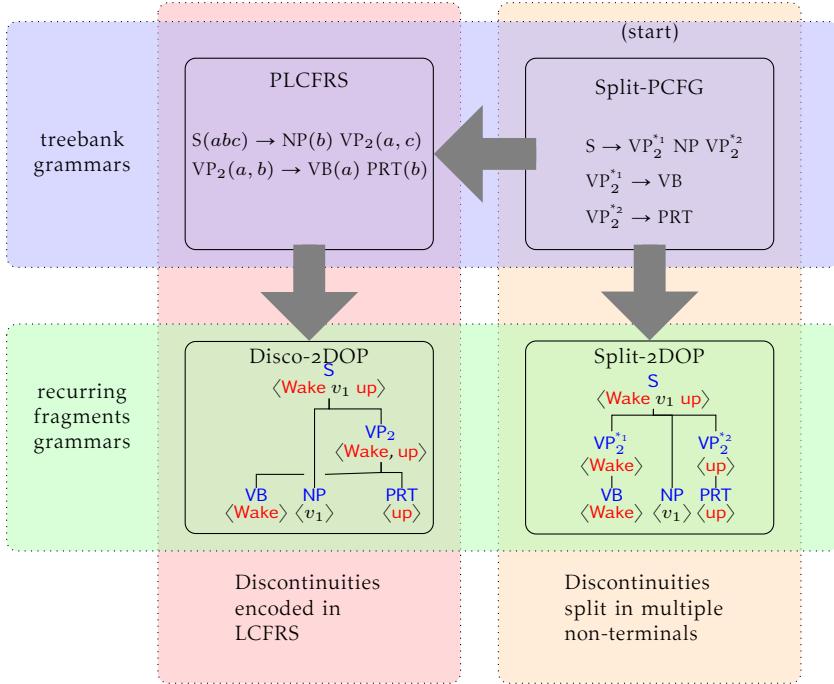
Figure 3.3: Diagram of the systems explored in this chapter.

3. Fragments make it possible to treat discontinuous constituency as a statistical phenomenon within an encompassing context-free framework (Section 3.3.1, Section 3.4.5); this yields a considerable efficiency improvement without hurting accuracy (Section 3.6).

4. Finally, we present an evaluation on three languages. We employ manual state splits from previous work for improved performance (Section 3.5) and discuss methods and results for grammars that produce function tags in addition to phrasal labels (Section 3.5.3).

This work explores parsing discontinuous constituents with Linear Context-Free Rewriting Systems and Context-Free Grammar, as well as with and without the use of tree fragments through tree substitution. Figure 3.3 gives an overview of these systems and how they are combined in a coarse-to-fine pipeline (cf. Section 3.4.4).

3.2 Grammar formalisms

In this section we describe two formalisms related to discontinuous constituents: (string-rewriting) Linear Context-Free Rewriting Systems and Discontinuous

Tree-Substitution Grammar.

(String-rewriting) Linear Context-Free Rewriting Systems (LCFRS; Vijay-Shanker et al., 1987) can produce such structures. An LCFRS generalizes CFG by allowing non-terminals to rewrite tuples of strings instead of just single, contiguous strings. This property makes LCFRS suitable for directly parsing discontinuous constituents (Kallmeyer and Maier, 2010, 2013), as well as non-projective dependencies (Kuhlmann and Satta, 2009; Kuhlmann, 2013).

A tree-substitution grammar (TSG) provides a generalization of context-free grammar (CFG) that operates with larger chunks than just single grammar productions. A probabilistic TSG can be seen as a PCFG in which several productions may be applied at once, capturing structural relations between those productions.

Before defining these formalisms, we first define the tree structures they operate on. The notion of a "discontinuous tree" stems from a long linguistic tradition (Pike 1943, §§4.12–14; Wells 1947, §§55–62; McCawley 1982). It generalizes the usual notion of a phrase-structure tree in that it allows a non-terminal node to dominate a lexical span that consists of non-contiguous chunks. In our interpretation of this idea, it results in three formal differences:

1. A non-terminal with non-contiguous daughters does not have a non-arbitrary place in the left-to-right order with respect to its sibling nodes. Therefore, it is not obvious anymore that the left-to-right order of the terminals is to be described in terms of their occurrence in a tree with totally ordered branches. Instead, we employ trees with *unordered* branches, while every node is augmented with an explicit representation of its (ordered) yield.

2. An "ordinary" (totally ordered) tree has a contiguous string of leaf nodes as its yield. When we allow discontinuities, this property still applies to the (totally lexicalized) complete trees of complete sentences. But for tree fragments, it fails; their yields may contain gaps. In the general case, the yield of a discontinuous tree is thus a tuple of strings.

3. Extracting a fragment from a tree now consists of two steps:

   (a) Extracting a connected subset of nodes, and
   (b) Updating the yield tuples of the nodes. In the yield tuple of every non-terminal leaf node, every element (a contiguous chunk of words) is replaced by a *terminal variable*. This replacement is percolated up the tree, to the yield tuples of all nodes. Different occurrences of the same word carry a unique index, to allow for the percolation to proceed correctly.

We now proceed to give a more formal definition of our notion of a discontinuous tree.

DEFINITION 4. A *discontinuous syntactic tree* is a rooted, unordered tree. Each node consists of a label and a yield. A yield is a tuple of strings composed of lexical items; the tuple of strings denotes a subsequence of the yield at the root of the tree. We write ⟨a b⟩ to denote a yield consisting of the contiguous sequence of lexical items 'a' and 'b', while ⟨a b, c⟩ denotes a yield containing 'a b' followed by 'c' with an intervening gap. Given a node X,

- the yield of X is composed of the terminals in the yields of the children of X;
- conversely, the yield of each child of X is a subsequence of the yield of X;
- the yields of siblings do not overlap.
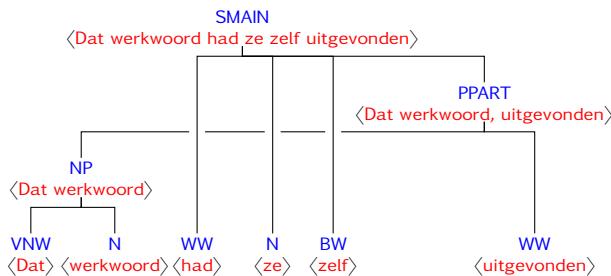
Figure 3.4: A discontinuous tree with yield tuples.
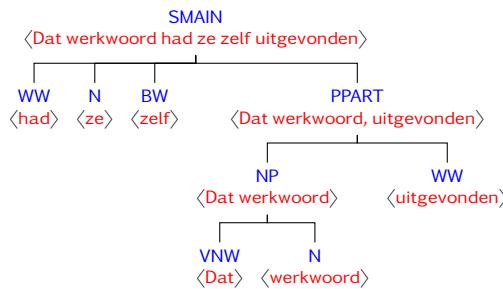
Figure 3.5: An equivalent representation of the tree in Figure 3.4, without crossing branches.

Figure 3.4 shows a tree according to this definition in which discontinuities are visualized with crossing branches as before. The same tree is rendered in Figure 3.5, without crossing branches, to highlight the fact that the information about discontinuities is encoded in the yields of the tree nodes.

DEFINITION 5. An *incomplete tree* is a discontinuous tree in which the yields may contain variables v_n with $n \in \mathbb{N}$ in addition to lexical items. Variables stand in for any contiguous string of lexical items. An incomplete tree contains 2 or

SMAIN
$\langle v_1 v_2 v_3 v_4 v_5 v_6 \rangle$ $\implies$ SMAIN
$\langle v_1 v_2 v_3 v_4 v_5 \rangle$

| WW | N | BW | PPART$_2$ | WW | N | BW | PPART$_2$ |
| $\langle v_3 \rangle$ | $\langle v_4 \rangle$ | $\langle v_5 \rangle$ | $\langle v_1 v_2, v_6 \rangle$ | $\langle v_2 \rangle$ | $\langle v_3 \rangle$ | $\langle v_4 \rangle$ | $\langle v_1, v_5 \rangle$ |
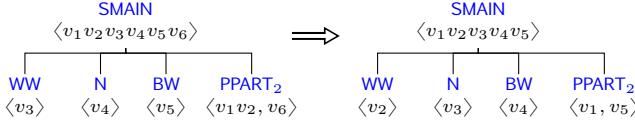
Figure 3.6: Reducing variables in a fragment extracted from the tree in Figure 3.4.

more nodes, or a single node with only lexical items in its yield. A node without children whose yield consists solely of variables is called a *substitution site*.

An incomplete tree may be derived from an extracted tree fragment. The tree fragment may contain variables for substrings which need to be distinguished in other parts of the tree, but only occur contiguously in the fragment. We reduce these strings of contiguous variables to single variables; i.e., we abstract fragments from their original context by reducing strings of variables that appear contiguously across the fragment into single variables. Figure 3.6 shows an example.

The *fan-out* of a non-terminal node equals the number of terminals in its yield that are not directly preceded by another terminal in the same yield; i.e., the number of contiguous substrings (components) of which the yield consists.[1] From here on we denote the fan-out of a discontinuous non-terminal with a subscript that is part of its label.

3.2.1 Linear Context-Free Rewriting Systems

String-rewriting LCFRS can be seen as the discontinuous counterpart of CFG, and its probabilistic variant can be used to articulate a discontinuous treebank grammar. LCFRS productions differ from CFG productions in that they generate for a given non-terminal one or more strings at a time in potentially non-adjacent positions in the sentence. The number of these positions, the measure of discontinuity in a constituent, is called the fan-out. A CFG is an LCFRS with a maximum fan-out of 1. Together with the number of non-terminals on the right-hand side, the fan-out defines a hierarchy of grammars with increasing complexity, of which CFG is the simplest case. We use the simple RCG notation (Boullier, 1998) for LCFRS. We focus on string-rewriting LCFRS and use the tree produced as a side-effect of a string's derivation as its syntactic analysis. It is possible to define an LCFRS that rewrites trees or graphs; however, the formalisms used in this thesis are all expressible as string-rewriting LCFRSs.

DEFINITION 6. A string-rewriting LCFRS is a tuple $G = \langle N, T, V, P, S \rangle$. N and T are disjoint finite sets of non-terminals and terminals, respectively. A function $\varphi : N \rightarrow \{1, 2, \ldots, \}$ specifies the unique fan-out for every non-terminal symbol.

---

[1] Note that a distinction is often made between the *fan-out* of non-terminals in grammar productions, and the *block degree* of nodes of a syntactic tree (Maier and Lichte, 2011; Kuhlmann, 2013). Due to the fact that the productions of a TSG are trees, these notions coincide for our purposes.

V is a finite set of variables; we refer to the variables as x_j^i with $i, j \in \mathbb{N}$. S is the distinguished start symbol with $S \in N$ and $\varphi(S) = 1$. P is a finite set of productions, of the form:

$$A(\alpha_1, \ldots \alpha_{\varphi(A)}) \rightarrow B_1(x_1^1, \ldots, x_{\varphi(B_1)}^1) \ldots B_r(x_1^r, \ldots, x_{\varphi(B_r)}^r)$$

for $r \geqslant 0$, where $A, B_1, \ldots, B_r \in N$, each $x_j^i \in V$ for $1 \leqslant i \leqslant r$, $1 \leqslant j \leqslant \varphi(B_i)$, and $\alpha_j \in (T \cup V)^+$ for $1 \leqslant j \leqslant \varphi(A)$. Observe that a component α_j is a concatenation of one or more terminals and variables.

The rank r refers to the number of non-terminals on the right-hand side of a production, while the fan-out φ of a non-terminal refers to the number of components it covers. A rank of zero implies a lexical production; in that case the right hand side (RHS) is notated as ε implying no new non-terminals are produced (not to be confused with generating the empty string), and the hand side (LHS) argument is composed only of terminals.

Productions must be *linear* and *non-erasing*: if a variable occurs in a production, it occurs exactly once on the LHS, and exactly once on the RHS. A production is *monotone*[2] if for any two variables x_1 and x_2 occurring in a non-terminal on the RHS, x_1 precedes x_2 on the LHS *iff* x_1 precedes x_2 on the RHS. Due to our method of grammar extraction from treebanks, (cf. Section 3.2.1 below) all productions in this work are monotone and, except in some examples, at most binary ($r \leqslant 2$); lexical productions ($r = 0$) have fan-out 1 and introduce only a single terminal.

A production is *instantiated* when its variables are bound to spans such that for each component α_j of the LHS, the concatenation of the strings that its terminals and bound variables point to forms a contiguous, non-overlapping span in the input. In the remainder we will notate discontinuous non-terminals with a subscript indicating their fan-out.

When a sentence is parsed by an LCFRS, its derivation tree (Boullier 1998, § 3.3; Kallmeyer 2010, p. 115–117) is a discontinuous tree. Conversely, given a set of discontinuous trees, a set of productions can be extracted that generate those trees.

In a probabilistic LCFRS (PLCFRS), each production is associated with a probability and the probability of derivation is the product of the probabilities of its productions. Analogously to a PCFG, a PLCFRS may be induced from a treebank by using relative frequencies as probabilities (Maier and Søgaard, 2008).

DEFINITION 7. The *language* of an LCFRS G is defined as follows (Kallmeyer and Maier, 2013):

1. For every $A \in N$, we define the yield of A, $\mathrm{yield}_G(A)$ as follows:

   (a) For every production $A(t) \rightarrow \varepsilon$ with $t \in T$, $\langle t \rangle \in \mathrm{yield}_G(A)$

---

2 This property is called *ordered* in the RCG literature.

$N = \{\text{SMAIN, PPART, NP, VNW, N, WW, BW}\}$

$V = \{a, b, c, d, e\}$

$\varphi = \{\text{SMAIN} : 1, \text{PPART} : 2, \text{NP} : 1,$
$\quad \text{VNW} : 1, \text{N} : 1, \text{WW} : 1, \text{BW} : 1\}$

$S = \text{SMAIN}$

$P = \{\text{SMAIN}(abcde) \rightarrow \text{WW}(b)\ \text{N}(c)\ \text{BW}(d)\ \text{PPART}(a, e),$
$\quad \text{PPART}(a, b) \rightarrow \text{NP}(a)\ \text{WW}(b),$
$\quad \text{NP}(ab) \rightarrow \text{VNW}(a)\ \text{N}(b),$
$\quad \text{VNW}(\text{Dat}) \rightarrow \varepsilon,\ \text{N}(\text{werkwoord}) \rightarrow \varepsilon,\ \text{WW}(\text{had}) \rightarrow \varepsilon,$
$\quad \text{N}(\text{ze}) \rightarrow \varepsilon,\ \text{BW}(\text{zelf}) \rightarrow \varepsilon,\ \text{WW}(\text{uitgevonden}) \rightarrow \varepsilon\}$

Figure 3.7: The LCFRS $G = \langle N, T, V, P, S \rangle$ extracted from the tree in Figure 3.4.

(b) For every production

$$A(\alpha_1, \ldots \alpha_{\varphi(A)}) \rightarrow B_1(x_1^1, \ldots, x_{\varphi(B_1)}^1) \quad \ldots B_r(x_1^r, \ldots, x_{\varphi(B_r)}^r)$$

and for all $\tau_i \in \text{yield}_G(B_i)$ with $1 \leqslant i \leqslant r$:

$$\langle f(\alpha_1), \ldots f(\alpha_{\varphi(A)}) \rangle \in \text{yield}_G(A)$$

where f is defined as follows:
  i. $f(t) = t$ for all $t \in T$,
  ii. $f(x_j^i) = \tau_i[j]$ for all $1 \leqslant i \leqslant r, 1 \leqslant j \leqslant \varphi(B_i)$, and
  iii. $f(ab) = f(a)f(b)$ for all $a, b \in (T \cup V)^+$.
(c) Nothing else is in $\text{yield}_G(A)$.

2. The language of G is then $L(G) = \text{yield}_G(S)$.

Extracting LCFRS productions from trees

LCFRS productions may be induced from a discontinuous tree, using a procedure described in Maier and Søgaard (2008). We extend this procedure to handle substitution sites, i.e., non-terminals with only variable terminals in their yield, but no lexical items; such nodes occur in tree fragments extracted from a treebank. The procedure is as follows:

Given a discontinuous tree, we extract a grammar production for each non-leaf non-terminal node. The label of the node forms the LHS non-terminal, and the labels of the nodes immediately dominated by it form the RHS non-terminals. The arguments of each RHS non-terminal are based on their yield

tuples. Adjacent variables in the yield of the RHS non-terminals are collapsed into single variables and replaced on both LHS and RHS. Consider the tree fragment in Figure 3.6, which gives the following LCFRS production:

$$\text{SMAIN}(abcde) \rightarrow \text{PPART}(a, e)\ \text{WW}(b)\ \text{N}(c)\ \text{BW}(d)$$

Pre-terminals yield a production with their terminal as a direct argument to the pre-terminal, and an empty RHS. Substitution sites in a tree only appear on the RHS of extracted productions, since it is not known what they will expand to. See Figure 3.7 for examples of LCFRS productions extracted from a discontinuous tree.

3.2.2 Discontinuous Tree-Substitution Grammar

We now employ string-rewriting LCFRS, introduced in the previous section, to replace the CFG foundation of TSGs. Note that the resulting formalism directly rewrites elementary trees with discontinuous constituents, making it an instantiation of the more general notion of a tree-rewriting LCFRS. Tree-rewriting LCFRSs are more general because they allow other rewriting operations besides substitution. However, since we limit the operations in the formalism to substitution, it remains possible to specify a direct mapping to a string-rewriting grammar, as we shall see in the next section. As noted before, a TSG can be seen as a TAG without the adjunction operation. A discontinuous TSG may be related to a special case of set-local multi-component TAG (Weir, 1988; Kallmeyer, 2009). A multi-component TAG is able to specify constraints that require particular elementary trees to apply together; this mechanism can be used to generate the non-local elements of discontinuous constituents.

The following definitions are based on the definition for continuous TSG in Sima'an (1997).

DEFINITION 8. A *probabilistic*, *discontinuous* TSG (PDTSG) is a tuple $\langle N, T, V, S, \mathcal{C}, P \rangle$, where N and T are disjoint finite sets that denote the set of non-terminal and terminal symbols, respectively; V is a finite set of variables; S denotes the start non-terminal; and $\mathcal{C}$ is a finite set of elementary trees. For all trees in $\mathcal{C}$ it holds that for each non-terminal, there is a unique fan-out; this induces a function $\varphi \subset N \times \{1, 2, \ldots\}$ with $\varphi(A)$ being the unique fan-out of $A \in N$. For convenience, we abbreviate $\varphi(\text{root}(t))$ for a tree t as $\varphi(t)$. The function P assigns a value $0 < P(t) \leqslant 1$ (probability) to each elementary tree t such that for every non-terminal $A \in N$, the probabilities of all elementary trees whose root node is labeled A sum to 1.

The tuple $\langle N, T, V, S, \mathcal{C} \rangle$ of a given PDTSG $\langle N, T, V, S, \mathcal{C}, P \rangle$ is called the DTSG underlying the PDTSG.

DEFINITION 9. *Substitution:* The substitution $A \circ B$ is defined *iff* the label of the left-most substitution site of A equals the label of the root node of B. The left-most substitution site of an incomplete tree A is the leaf node containing the first
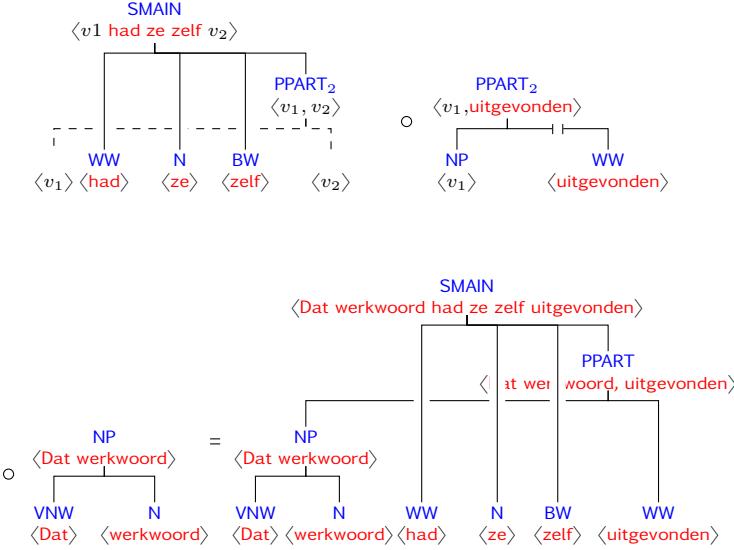
Figure 3.8: A discontinuous tree-substitution derivation of the tree in Figure 3.1. Note that in the first fragment, which has a discontinuous substitution site, the destination for the discontinuous spans is marked in advance, shown with variables (v_n) as placeholders.

occurrence of a variable in the yield of the root of A. When defined, the result of $A \circ B$ equals a copy of the tree A with B substituted for the left-most substitution site of A. In the yield argument of A, each variable terminal is replaced with the corresponding component of one or more contiguous terminals from B. For example, given yield$(A) = \langle l_1 v_2, l_4 \rangle$ and yield$(B) = \langle l_2 l_3 \rangle$ where l_n is a lexical terminal and v_n a variable, yield$(A \circ B) = \langle l_1 l_2 l_3, l_4 \rangle$.

Definition 10. A *left-most derivation* (derivation henceforth) d is a sequence of zero or more substitutions $T = (\ldots (f_1 \circ f_2) \circ \ldots) \circ f_m$, where $f_1, \ldots, f_m \in \mathcal{C}$, root$(T) = $ root$(f_1) = S$, $\varphi(T) = 1$ and T contains no substitution sites. The probability $P(d)$ is defined as:

$$P(f_1) \cdot \ldots \cdot P(f_m) = \prod_{i=1}^{m} P(f_i)$$

Cf. Figure 3.8 for an example.

Definition 11. A *parse* is any tree which is the result of a derivation. A parse can have various derivations. Given the set $D(T)$ of derivations yielding parse T, the probability of T is defined as $\sum_{d \in D(T)} P(d)$.

3.3 Grammar transformations

cfg, lcfrs, and dtsg can be seen as natural extensions of each other. This makes it possible to define transformations that help to make parsing more efficient. Specifically, we define simplified versions of these grammars that can be parsed efficiently, while their productions or labels map back to the original grammar.

3.3.1 A cfg approximation of discontinuous lcfrs parsing

Barthélemy et al. (2001) introduced a technique to guide the parsing of a range concatenation grammar (rcg) by a grammar with a lower parsing complexity. Van Cranenburgh (2012a) applies this idea to probabilistic lcfrs parsing and extends the method to prune unlikely constituents in addition to filtering impossible constituents.

The approximation can be formulated as a tree transformation instead of a grammar transformation. The tree transformation by Boyd (2007) encodes discontinuities in the labels of tree nodes.[3] The resulting trees can be used to induce a pcfg that can be viewed as an approximation to the corresponding plcfrs grammar of the original, discontinuous treebank. We will call this a Split-pcfg.

Definition 12. A Split-pcfg is a pcfg induced from a treebank transformed by the method of Boyd (2007); that is, discontinuous constituents have been split into several non-terminals, such that each new non-terminal covers a single contiguous component of the yield of the discontinuous constituent. Given a discontinuous non-terminal X_n in the original treebank, the new non-terminals will be labeled X_n^{*m}, with m the index of the component, s.t. $1 \leqslant m \leqslant n$.

For example:

$$\text{lcfrs productions: } S(abc) \rightarrow NP(b) \; VP_2(a, c)$$
$$VP_2(a, b) \rightarrow VB(a) \; PRT(b)$$
$$\text{cfg approximation: } S \rightarrow VP_2^{*1} \; NP \; VP_2^{*2}$$
$$VP_2^{*1} \rightarrow VB$$
$$VP_2^{*2} \rightarrow PRT$$
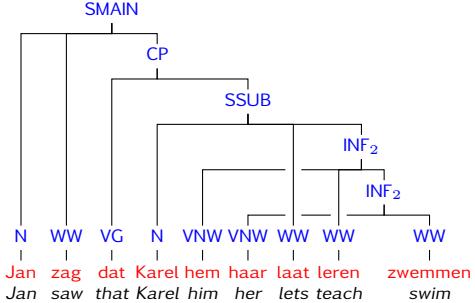
In a postprocessing step, pcfg derivations are converted to discontinuous trees by merging siblings marked with '*'. This approximation overgenerates compared to the respective lcfrs, i.e., it licenses a superset of the derivations of the respective lcfrs. For example, a component VP_2^{*1} may be generated without

---
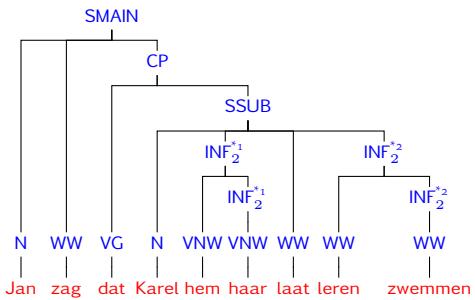
3 Hsu (2010) compares three methods for resolving discontinuity in trees: (*a*) node splitting, as applied here; (*b*) node adding, a simpler version of node splitting that does not introduce new non-terminal labels; and (*c*) node raising, the more commonly applied method of resolving discontinuity. While the latter two methods yield better performance, we use the node splitting approach because it provides a more direct mapping to discontinuous constituents, which, as we shall later see, makes it a useful source of information for pruning purposes.

discontinuous constituents:

$\text{SMAIN}(abc) \rightarrow \text{N}(a) \ \text{WW}(b) \ \text{CP}(c)$

$\text{CP}(ab) \rightarrow \text{VG}(a) \ \text{SSUB}(b)$

$\text{SSUB}(abcd) \rightarrow \text{N}(a)$

$\text{INF}_2(b,d) \ \text{WW}(c)$

$\text{INF}_2(ab,cd) \rightarrow \text{VNW}(a) \ \text{INF}_2(b,d)$

$\text{WW}(c)$

$\text{INF}_2(a,b) \rightarrow \text{VNW}(a) \ \text{WW}(b)$

PCFG approximation:

$\text{SMAIN} \rightarrow \text{N WW CP}$

$\text{CP} \rightarrow \text{VG SSUB}$

$\text{SSUB} \rightarrow \text{N INF}_2^{*_1} \ \text{WW INF}_2^{*_2}$

$\text{INF}_2^{*_1} \rightarrow \text{VNW INF}_2^{*_1}$

$\text{INF}_2^{*_2} \rightarrow \text{WW INF}_2^{*_2}$

$\text{INF}_2^{*_1} \rightarrow \text{VNW}$

$\text{INF}_2^{*_2} \rightarrow \text{WW}$

Figure 3.9: Cross-serial dependencies in Dutch expressed with discontinuous constituents (top); and the same parse tree, after discontinuities have been encoded in node labels (bottom).

generating its counterpart $\text{VP}_2^{*_2}$; such derivations can be filtered in postprocessing. Furthermore, two components $\text{VP}_2^{*_1}$ and $\text{VP}_2^{*_2}$ may be generated which were extracted from different discontinuous constituents, such that their combination could not be generated by the lcfrs.[4] Another problem would occur when productions contain discontinuous constituents with the same label; the following two productions map to the same productions in the cfg approximation:

$$\text{VP}(adceb) \rightarrow \text{VP}_2(a,b) \ \text{CNJ}(c) \ \text{VP}_2(d,e)$$
$$\text{VP}(adcbe) \rightarrow \text{VP}_2(a,b) \ \text{CNJ}(c) \ \text{VP}_2(d,e)$$

However, such productions do not occur in any of the treebanks used in this work. The increased independence assumptions due to rewriting discontinuous components separately are more problematic, especially with nested discontinuous constituents. They necessitate the use of non-local statistical information to select the most likely structures, for instance by turning to tree-substitution

---

4 A reviewer points out that if discontinuous rewriting is seen as synchronous rewriting (synchronous cfgs are equivalent to lcfrss with fan-out 2), the split transformation is analogous to taking out the synchronicity.

grammar (cf. Section 3.2.2). (Note that the issue is not as problematic when the approximation is only used as a source of pruning information).

As a specific example of the transformation, consider the case of cross-serial dependencies. Figure 3.9 shows the parse tree for the example sentence from the previous section, along with the grammar productions for it, before and after applying the CFG approximation of LCFRS. Note that in the approximation, the second level of INF nodes may be rewritten separately, and a context-free grammar cannot place the non-local constraint that each transitive verb should be paired with a direct object. On the other hand, through the use of tree substitution, an elementary tree may capture the whole construction of two verbs cross-serially depending on two objects, and the model needs only to prefer an analysis with this elementary tree. Once an elementary tree contains the whole construction, it no longer matters whether its internal nodes contain discontinuous constituents or indexed node labels, and the complexity of discontinuous rewriting is weakened to a statistical regularity.

A phenomenon which cannot be captured in this representation, not even with the help of tree-substitution, is recursive synchronous rewriting (Kallmeyer et al., 2009). Although this phenomenon is rare, it does occur in treebanks.

3.3.2 TSG COMPRESSION

Using grammar transformations, it is possible to parse with a TSG without having to represent elementary trees in the chart explicitly, but instead work with a parser for the base grammar underlying the TSG (typically a CFG, in our case an LCFRS).

In this section we present such a transformation for an arbitrary discontinuous TSG to a string-rewriting LCFRS. We first look at well-established strategies for reducing a continuous TSG to a CFG, and then show that these carry over to the discontinuous case. Previous work was based on probabilistic TSG without discontinuity; this special case of PDTSG is referred to as PTSG.

Compressing ptsg to pcfg

Goodman (2003) gives a reduction to a PCFG for the special case of a PTSG based on all fragments from a given treebank and their frequencies. This reduction is stochastically equivalent to an all-fragments PTSG after the summation of probabilities from equivalent derivations—however, it does not admit parsing with TSGs consisting of arbitrary sets of elementary trees or assuming arbitrary probability models. Perhaps counter-intuitively, restrictions on the set of fragments increase the size of Goodman's reduction (e.g., depth restriction, Goodman, 2003, p. 134). While Goodman (2003) gives instantiations of his reduction with various probability models, the limitation is that probability assignments of fragments have to be expressible as a composition of the weights of the productions in each fragment. Since each production in the reduction participates in numerous implicit fragments, it is not possible to adjust the probability of an

Elementary tree	Productions	Weight

$$S(ab) \rightarrow S^1(a)\ WW(b) \qquad f/f'$$
$$S^1(ab) \rightarrow S^2(a)\ BW(b) \qquad 1$$
$$S^2(ab) \rightarrow S^3(a)\ N(b) \qquad 1$$
$$S^3(ab) \rightarrow NP(a)\ WW^4(b) \qquad 1$$
$$WW^4(\text{uitgevonden}) \rightarrow \varepsilon \qquad 1$$

$$S(abc) \rightarrow S_2^5(a,c)\ BW^6(b) \qquad f/f'$$
$$S_2^5(ab,c) \rightarrow S_2^7(a,c)\ N(b) \qquad 1$$
$$S_2^7(ab,c) \rightarrow \qquad 1$$
$$PPART_2(a,c)\ WW^8(b)$$
$$WW^8(\text{had}) \rightarrow \varepsilon \qquad 1$$
$$N^7(\text{ze}) \rightarrow \varepsilon \qquad 1$$
$$BW^6(\text{zelf}) \rightarrow \varepsilon \qquad 1$$

$$PPART_2(a,b) \rightarrow \qquad f/f'$$
$$NP(a)\ WW^9(b)$$
$$WW^9(\text{uitgevonden}) \rightarrow \varepsilon \qquad 1$$

Figure 3.10: Transforming a discontinuous tree-substitution grammar into an LCFRS backtransform table. The elementary trees are extracted from the tree in Figure 3.1 with labels abbreviated. The first production of each fragment is used as an index to the backtransform table so that the original fragments in derivations can be reconstructed.

individual fragment without affecting related fragments. We leave Goodman's reduction aside for now, because we would prefer a more general method.

A naive way to convert any TSG is to decorate each internal node of its elementary trees with a globally unique number, which can be removed from derivations in a postprocessing step. Each elementary tree then contributes one or more grammar productions, and because of the unique labels, elementary trees will always be derived as a whole. However, this conversion results in a large number of non-terminals, which are essentially 'inert': they never participate in substitution but deterministically rewrite to the rest of their elementary tree.

A more compact transformation is used in Sangati and Zuidema (2011), which can be applied to arbitrary PTSGs, but adds a minimal number of new non-terminal nodes. Internal nodes are removed from elementary trees, yielding a flattened tree of depth 1. Each flattened tree is then converted to a grammar production. Each production and original fragment is stored in a backtransform table. This table makes it possible to restore the original fragments of a derivation built from flattened productions. Whenever two fragments would map to the same flattened production, a unary node with a unique identifier is added to disambiguate them. The weight associated with an elementary tree carries

over to the first production it produces; the rest of the productions are assigned a weight of 1.

Compressing pdtsg to plcfrs

The transformation defined by Sangati and Zuidema (2011) assumes that a sequence of productions can be read off from a syntactic tree, such as a standard phrase-structure tree that can be converted into a sequence of context-free grammar productions. Using the method for inducing lcfrs productions from syntactic trees given in Section 3.3.2, we can apply the same tsg transformation to discontinuous trees as well.

Due to the design of the parser we will use, it is desirable to have grammar productions in binarized form, and to separate phrasal and lexical productions. We therefore binarize the flattened trees with a left-factored binarization that adds unique identifiers to every intermediate node introduced by the binarization. In order to separate phrasal and lexical productions, a new pos tag is introduced for each terminal, which selects for that specific terminal. A sequence of productions is then read off from the transformed tree. The unique identifier in the first production is used to look up the original elementary tree in the backtransform table.[5]

Figure 3.10 illustrates the transformation of a discontinuous tsg. The middle column shows the productions after transforming each elementary tree. The rightmost column shows how relative frequencies can be used as weights, where f is the frequency of the elementary tree in the treebank, and f' is the frequency mass of elementary trees with the same root label. Note that the productions for the first elementary tree contain no discontinuity, because the discontinuous internal node is eliminated. Conversely, the transformation may also introduce more discontinuity, due to the binarization (but cf. Section 3.5.1 below).

Figure 3.11 presents an overview of the methods of grammar induction presented thus far, as well as the approach for finding recurring fragments that will be introduced in the next section.

3.4   Parsing with plcfrs and pdtsg

We use the fragment extractor introduced in the previous chapter to define the grammar. This yields a Double-dop grammar (2dop; Sangati and Zuidema, 2011). Since our fragment extractor supports discontinuous constituents, we call our model Disco-2dop.

After extracting fragments we augment the set of fragments with all depth 1 fragments, in order to preserve complete coverage of the training set trees. Since depth 1 fragments are equivalent to single grammar productions, this ensures

---

5 Note that only this first production requires a globally unique identifier; to reduce the grammar constant, the other identifiers can be merged for equivalent productions.
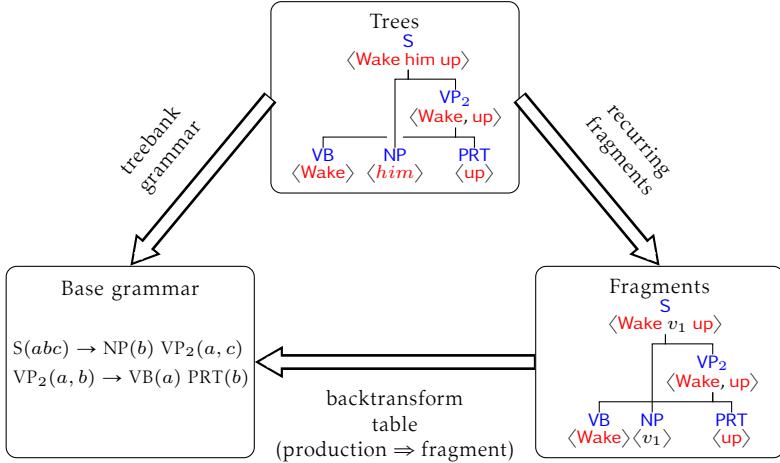
Figure 3.11: Diagram of the methods of grammar induction.

strong equivalence between the TSG and the respective treebank grammar.[6] We then apply the grammar transformation (cf. Section 3.3.2) to turn the fragments into productions. Productions corresponding to fragments are assigned a probability based on the relative frequency of the respective fragment, productions introduced by the transformation are given a probability of 1. For an example, please refer to Figure 3.10.

We parse with the transformed grammar using the disco-dop parser (van Cranenburgh et al., 2011; van Cranenburgh, 2012a). This is an agenda-based parser for PLCFRS based on the algorithm in Kallmeyer and Maier (2010, 2013), extended to produce n-best derivations (Huang and Chiang, 2005) and exploit coarse-to-fine pruning (Charniak et al., 2006).

Parsing with LCFRS can be done with a weighted deduction system and an agenda-based parser. The deduction steps are given in Figure 3.12; for the pseudocode of the parser see Algorithm 4, which is an extended version of the parser in Kallmeyer and Maier (2010, 2013) that obtains the complete parse forest as opposed to just the Viterbi derivation.

In Section 3.4.1 we describe the probabilistic instantiation of DTSG and the criterion to select the best parse. Section 3.4.2 describes how derivations from the compressed TSG are converted back into trees composed of the full elementary trees. Section 3.4.4 describes how coarse-to-fine pruning is employed to make parsing efficient.

---

6 Previous DOP work such as Zollmann and Sima'an (2005) adds all possible tree fragments up to depth 3. Preliminary experiments on 2DOP gave no improvement on performance, while tripling the grammar size; therefore we do not apply this in further experiments.

Lexical: $$\frac{}{p : [A, \langle\langle w_i \rangle\rangle]} \qquad p : A(w_i) \to \varepsilon \in \mathcal{G}$$

Unary: $$\frac{x : [B, \alpha]}{p \cdot x : [A, \alpha]} \qquad \begin{array}{l} p : A(\alpha) \to B(\alpha) \text{ is an instanti-} \\ \text{ated rule from } \mathcal{G} \end{array}$$

Binary: $$\frac{x : [B, \beta], \ y : [C, \gamma]}{p \cdot x \cdot y : [A, \alpha]} \qquad \begin{array}{l} p : A(\alpha) \to B(\beta) \ C(\gamma) \text{ is an in-} \\ \text{stantiated rule from } \mathcal{G} \end{array}$$

Goal: $$[S, \langle\langle w_1 \cdots w_n \rangle\rangle]$$

Figure 3.12: Weighted deduction system for binarized LCFRS.

INPUT: A sentence $w_1 \cdots w_n$, a grammar $\mathcal{G}$
OUTPUT: A chart $\mathcal{C}$ with Viterbi probabilities, a parse forest $\mathcal{F}$.
1: initialize agenda $\mathcal{A}$ with all possible POS tags for input
2: WHILE $\mathcal{A}$ not empty
3:     $\langle I, x \rangle \leftarrow$ pop item with best score on agenda
4:     add $\langle I, x \rangle$ to $\mathcal{C}$
5:     FOR ALL $\langle I', z \rangle$ that can be deduced from $\langle I, x \rangle$ and items in $\mathcal{C}$
6:         IF $I' \notin \mathcal{A} \cup \mathcal{C}$
7:             enqueue $\langle I', z \rangle$ in $\mathcal{A}$
8:         ELSE IF $I' \in \mathcal{A} \wedge z >$ score for I' in $\mathcal{A}$
9:             update weight of I' in $\mathcal{A}$ to z
10:        add edge for I' to $\mathcal{F}$

Algorithm 4: A probabilistic agenda-based parser for LCFRS.

3.4.1 Probabilities and disambiguation

Our probabilistic model uses the relative frequency estimate (RFE), which has shown good results with the Double-DOP model (Sangati and Zuidema, 2011). The relative frequency of a fragment is the number of its occurrences, divided by the total number of occurrences of fragments with the same root node.

In DOP many derivations may produce the same parse tree, and it has been shown that approximating the most probable parse, which considers all derivations for a tree, yields better results than the most probable derivation (Bod, 1995b). To select a parse tree from a derivation forest, we compute tree probabilities on the basis of the 10,000 most probable DOP derivations, and select the tree with the largest probability. Although the algorithm of Huang and Chiang (2005) makes it is possible to extract the exact k-best derivations from a derivation forest, we apply pruning while building the forest.

3.4.2 Reconstructing derivations

After a derivation forest is obtained and a list of k-best derivations has been produced, the backtransform is applied to these derivations to recover their internal structure. This proceeds by doing a depth-first traversal of the derivations, and expanding each non-intermediate[7] node into a template of the original fragment. These templates are stored in a backtransform table indexed by the first binarized production of the fragment in question. The template fragment has its substitution sites marked, which are filled with values obtained by recursively expanding the children of the current constituent.

3.4.3 Efficient discontinuous parsing

We review several strategies for making discontinuous parsing efficient. As noted by Levy (2005, p. 138), the intrinsic challenge of discontinuous constituents is that a parser will generate a large number of potential discontinuous spans.

Outside estimates

Outside estimates (also known as context-summary estimates and figures-of-merit) are computed offline for a given grammar. During parsing they provide an estimate of the outside probability for a given constituent, i.e., the probability of a complete derivation with that constituent divided by the probability of the constituent. The estimate can be used to prioritize items in the agenda. Estimates were first introduced for discontinuous LCFRS parsing in Kallmeyer and Maier (2010, 2013). Their estimates are only applied up to sentences of 30 words. Beyond 30 words the table grows too large.

---

7 An intermediate node is a node introduced by the binarization.

A different estimate is given by Angelov and Ljunglöf (2014), who succeed in parsing longer sentences and providing an A* estimate, which is guaranteed to find the best derivation.

Non-projective dependency conversion

Hall and Nivre (2008), Versley (2014), and Fernández-González and Martins (2015) apply a reversible dependency conversion to the Tiger treebank, and use a non-projective dependency parser to parse with the converted treebank. The method has the advantage of being fast due to the greedy nature of the arc-eager, transition-based dependency parser that is employed. The parser copes with non-projectivity by reordering tokens during parsing. Experiments are reported on the full Tiger treebank without length restrictions.

Reducing fan-out

The most direct way of reducing the complexity of lcfrs parsing is to reduce the fan-out of the grammar.

Maier et al. (2012) introduces a linguistically motivated reduction of the fan-outs of the Negra and Penn treebanks to fan-out 2 (up to a single gap per constituent). This enables parsing of sentences of up to length 40.

Nederhof and Vogler (2014) introduce a method of synchronous parsing with an lcfrs and a definite clause grammar. A parameter allows the fan-out (and thus parsing complexity) of the lcfrs to be reduced. Experiments are reported on sentences of up to 30 words on a small section of the Tiger treebank.

Coarse-to-fine pruning

We will focus on coarse-to-fine pruning, introduced in Charniak et al. (2006) and applied to discontinuous parsing by van Cranenburgh (2012a), who reports parsing results on the Negra treebank without length restrictions. Compared to the previous methods, this method does not change the grammar, but adds several new grammars to be used as preprocessing steps. Compared to the outside estimates, this method exploits sentence-specific information, since pruning information is collected during parsing with the coarser grammars.

Pauls and Klein (2009) present a comparison of coarse-to-fine and (hierarchical A*) outside estimates, and conclude that except when near-optimality is required, coarse-to-fine is more effective as it prunes a larger number of unlikely constituents.

A similar observation is obtained from a comparison of the discontinuous coarse-to-fine method and the outside estimates of Angelov and Ljunglöf (2014): coarse-to-fine is faster with longer sentences (30 words and up), at the cost of not always producing the most likely derivation (Ljunglöf, personal communication).

3.4.4 Coarse-to-fine pipeline

In order to tame the complexity of LCFRS and DOP, we apply coarse-to-fine pruning. Different grammars are used in the sequel, each being an overgenerating approximation of the next. That is, a coarse grammar will generate a larger set of constituents than a fine grammar. Parsing with a coarser grammar is more efficient, and all constituents which can be ruled out as improbable with a coarser grammar can be discarded as candidates when parsing with the next grammar. A constituent is ruled improbable if it does not appear in the k-best derivations of a parse forest. We use the same setup as in van Cranenburgh (2012a); namely, we parse in three stages, using three different grammars:

1. Split-PCFG: A CFG approximation of the discontinuous treebank grammar; rewrites spans of discontinuous constituents independently.
2. PLCFRS: The discontinuous treebank grammar; rewrites discontinuous constituents in a single operation. A discontinuous span $X_n\langle x_1, \ldots, x_n\rangle$ is added to the chart only if all of $X_n^{*m}\langle x_m\rangle$ with $1 \leqslant m \leqslant n$ are part of the k-best derivations of the chart of the previous stage.
3. Disco-DOP: The discontinuous DOP grammar; uses tree fragments instead of individual productions from treebank. A discontinuous span $X_n\langle x_1, \ldots, x_n\rangle$ is added to the chart only if $X_n\langle x_1, \ldots, x_n\rangle$ is part of the k-best derivations of the chart of the previous stage, or if X_n is an intermediate symbol introduced by the TSG compression.

The first stage is necessary because without pruning, the PLCFRS generates too many discontinuous spans, the majority of which are improbable or not even part of a complete derivation. The second stage is not necessary for efficiency but gives slightly better accuracy on discontinuous constituents.

For example, while parsing the sentence "Wake your friend up," the discontinuous VP "Wake … up" may be produced in the PLCFRS stage. Before allowing this constituent to enter into the agenda and the chart, the chart of the previous stage is consulted to see if the two discontinuous components "Wake" and "up" were part of the k-best derivations. In the DOP stage, multiple elementary trees may be headed by this discontinuous constituent, and again they are only allowed on the chart if the previous stage produced the constituent as part of its k-best derivations.

The initial values for k are 10,000 and 50, for the PLCFRS and DOP grammar respectively. These values are chosen to be able to directly compare the new approach with the results in van Cranenburgh (2012a). However, experimenting with a higher value for k for the DOP stage has shown to yield improved performance. In other coarse-to-fine work the pruning criterion is based on posterior thresholds (e.g., Charniak et al., 2006; Bansal and Klein, 2010); the k-best approach has the advantage that it does not require the computation of inside and outside probabilities.

For the initial PCFG stage, we apply beam search as in Collins (1999). The

highest scoring item in each cell is tracked and only items up to 10,000 times less probable are allowed to enter the chart.

Experiments and results are described in Section 3.5 and 3.6.

3.4.5 Discontinuity without lcfrs

The idea up to now has been to generate discontinuous constituents using formal rewrite operations of lcfrs. It should be noted, however, that the pcfg approximation used in the pruning stage reproduces discontinuities using information derived from the non-terminal labels. Instead of using this technique only as a crutch for pruning, it can also be combined with the use of fragments to obtain a pipeline that runs in cubic time. While the cfg approximation increases the independence assumptions for discontinuous constituents, the use of large fragments in the dop approach can mitigate this increase. To create the cfg approximation of the discontinuous treebank grammar, the treebank is transformed by splitting discontinuous constituents into several non-terminal nodes (as explained in Section 3.3.1), after which grammar productions are extracted. This last step can also be replaced with fragment extraction to obtain a dop grammar from the transformed treebank. We shall refer to this alternative approach as 'Split-2dop.' The coarse-to-fine pipeline is now as follows:

1. Split-pcfg: A treebank grammar based on the cfg approximation of discontinuous constituents; rewrites spans of discontinuous constituents independently.
2. Split-2dop grammar: tree fragments based on the same transformed treebank as above.

Since every discontinuous non-terminal is split up into a new non-terminal for each of its spans, the independence assumptions for that non-terminal in a probabilistic grammar are increased. While this representation is not sufficient to express the full range of nested discontinuous configurations, it appears adequate for the linguistic phenomena in the treebanks used in this work, since their trees can be unambiguously transformed back and forth into this representation. Moreover, the machinery of Data-Oriented Parsing mitigates the increase in independence assumptions through the use of large fragments. We can therefore parse using a dop model with a context-free grammar as the symbolic backbone, and still recover discontinuous constituents.

3.5 Experimental setup

In this section we describe the experimental setup for benchmarking our discontinuous Double-dop implementations on several discontinuous treebanks.

3.5.1 TREEBANKS AND PREPROCESSING

We evaluate on three languages: for German, we use the Negra (Skut et al., 1997) and Tiger (Brants et al., 2002) treebanks; for English, we use a discontinuous version of the Penn treebank (Evang and Kallmeyer, 2011); and for Dutch, we use Lassy (Van Noord, 2009) and CGN (van der Wouden et al., 2002) treebanks; cf. Table 3.1. Negra and Tiger contain discontinuous annotations by design, as a strategy to cope with the relatively free word order of German. The discontinuous Penn treebank consists of the WSJ section in which traces have been converted to discontinuous constituents; we use the version used in Evang and Kallmeyer (2011, sec. 5.1–5.2) without restrictions on the transformations. The Lassy treebank is referred to as a dependency treebank but when discontinuity is allowed it can be directly interpreted as a constituency treebank. The *Corpus Gesproken Nederlands* (CGN, Spoken Dutch Corpus; van der Wouden et al., 2002) is a Dutch spoken language corpus with the same syntactic annotations. We use the syntactically annotated sentences from the Netherlands (i.e., without the Flemish part) of up to 100 tokens. The train-dev-test splits we employ are as commonly used for the Penn treebank: sec. 2–21, sec. 24, sec. 23, respectively. For Negra we use the one defined in Dubey and Keller (2003). For Tiger we follow Hall and Nivre (2008) who define sections 0–9 where sentence i belongs to section i mod 10, sec. 0 is used as test, sec. 1 as development, and 2–9 as training. When parsing the Tiger test set, the development set is added to the training set as well; while this is not customary, it ensures the results are comparable with Hall and Nivre (2008). The same split is applied to the CGN treebank but with a single training set. For Lassy the split is our own.[8]

For purposes of training we apply heuristics for head assignment (Klein and Manning, 2003) and binarize the trees in the training sets head-outward

---

8 The Lassy split derives from 80–10–10 partitions of the canonically ordered sentence IDs in each subcorpus (viz. dpc, WR, WS, and wiki). Canonically ordered refers to a 'version sort' where an identifier such as '2.12.a' is treated as a tuple of three elements compared consecutively.

Treebank	train (sents.)	dev (sents.)	test (sents.)
GERMAN			
Negra	1–18,602	19,603–20,602	18,603–19,602
Tiger	40,379 / 45,427	5048	5047
ENGLISH			
PTB: WSJ	39,832	1346	2416
DUTCH			
Lassy small	52,157	6520	6523
CGN	70,277	2000	2000

Table 3.1: The discontinuous treebanks used in the experiments.

with $h = 1, v = 1$ markovization; i.e., n-ary nodes are factored into nodes specifying an immediate sibling and parent. Note that for LCFRS, a binarization may increase the fan-out, and thus the complexity of parsing. It is possible to select the binarization in such a way as to minimize this complexity (Gildea, 2010). However, experiments show that this increase in fan-out does not actually occur, regardless of the binarization strategy (van Cranenburgh, 2012a). Head-outward means that constituents are binarized in a right-factored manner up until the head child, after which the rest of the binarization continues in a left-factored manner.

We add fan-out markers to guarantee unique fan-outs for non-terminal labels, e.g., $\{VP, VP_2, VP_3, \ldots\}$, which are removed again for evaluation.

For the Dutch and German treebanks, punctuation is not part of the syntactic annotations. This causes spurious discontinuities, as the punctuation interrupts the constituents dominating its surrounding tokens. Additionally, punctuation provides a signal for constituent boundaries, and it is useful to incorporate it as part of the rest of the phrase structures. We use the method described in van Cranenburgh (2012a): punctuation is attached to the highest constituent that contains a neighbor to its right. With this strategy there is no increase in the amount of discontinuity with respect to a version of the treebank with punctuation removed. The CGN treebank contains spoken language phenomena, including disfluencies such as interjections and repeated words. In preprocessing, we treat these as if they were punctuation tokens; i.e., they are moved to an appropriate constituent (as defined above) and are ignored in the evaluation.

The complexity of parsing with a binarized LCFRS is $O(n^{3\varphi})$ with φ the highest fan-out of the non-terminals in the grammar (Seki et al., 1991). For a given grammar, it is possible to give a tighter upper bound on the complexity of parsing. Given the unique fan-outs of non-terminals in a grammar, the number of operations it takes to apply a production is the sum of the fan-outs in the production (Gildea, 2010):

$$c(p) = \varphi(A) + \sum_{i=1}^{r} \varphi(B_i)$$

The complexity of parsing with a grammar is then the maximum value of this measure for productions in the grammar. In our experiments we find a worst-case time complexity of $O(n^9)$ for parsing with the DOP grammars extracted from Negra and WSJ. The following sentence from Negra contributes a grammar production with complexity 9. The production is from the VP of *vorgeworfen*; bracketed words are from other constituents, indicating the discontinuities:

(1) Den Stadtteilparlamentariern [ist] immer wieder ["Kirchturmpolitik"] vorgeworfen
     *The district-*MPS *have always again "parochialism" accused*
     [worden], weil sie nicht über die Grenzen des Ortsbezirks hinausgucken
     been, because they not beyond the boundaries of-the local-district look-out
     würden.
     would.

Feature	Description
AC	All capital letters
SC	Initial capital, first word in sentence
C	Initial capital, other position
L, U	Has lower / upper case letter
S	No letters
N, n	All digits / one or more digits
H, P, C	Has dash / period / comma
x	Last character if letter and length > 3

Table 3.2: Unknown word features, Stanford Model 4.

> 'Time and again, the district MPs have been accused of "parochialism" because they would not look out beyond the boundaries of the local district.'

The complexities for Tiger and Lassy are $O(n^{10})$ and $O(n^{12})$ respectively, due to a handful of anomalous sentences; by discarding these sentences, a grammar with a complexity of $O(n^9)$ can be obtained with no or negligible effect on accuracy.

3.5.2 UNKNOWN WORDS

In initial experiments we present the parser with the gold standard part-of-speech tags, as in previous experiments on discontinuous parsing. Later we show results when tags are assigned automatically with a simple unknown word model, based on the Stanford parser (Klein and Manning, 2003). An open class threshold σ determines which tags are considered open class tags; tags that rewrite more than σ words are considered open class tags, and words they rewrite are open class words. Open class words in the training set that do not occur more than 4 times are replaced with signatures based on a list of features; words in the test set which are not part of the known words from the training set are replaced with similar signatures. The features are defined in the Stanford parser as *Model 4*, which is relatively language independent; cf. Table 3.2 for the list of features. Signatures are formed by concatenating the names of features that apply to a word; e.g., 'forty-two' gives _UNK-L-H-o. A probability mass ϵ is assigned for combinations of known open class words with unseen tags. We use $\epsilon = 0.01$. We tuned σ on each training set to ensure that no closed class words are identified as open class words; for English and German we use $\sigma = 150$, and $\sigma = 100$ for Dutch.

3.5.3 Function tags

We investigated two methods of having the parser produce function tags in addition to the usual phrase labels. The first method is to train a separate discriminative classifier that adds function tags to parse trees in a postprocessing step. This approach is introduced in Blaheta and Charniak (2000). We employed their feature set.

Another approach is to simply append the function tags to the non-terminal labels, resulting in, e.g., np-sbj and np-obj for subject and object noun phrases. While this approach introduces sparsity and may affect the performance without function tags, we found this approach to perform best and therefore report results with this approach. Gabbard et al. (2006) and Fraser et al. (2013) use this approach as well. Compared to the classifier approach, it does not require any tuning, and the resulting model is fully generative. We apply this to the Tiger, wsj, and Lassy treebanks.

The Penn treebank differs from the German and Dutch treebanks with respect to function tags. The Penn treebank only has function tags on selected non-terminals (never on preterminals) and each non-terminal may have several function tags from four possible categories; whereas the German and Dutch treebanks have a single function tag on most non-terminals. The tag set also differs considerably; the Penn treebank has 17 function tags (Taylor et al., 2003, Table 1.3), Lassy has 31 (cf. Section A.2), and Tiger has 43 (Smith, 2003, sec. A.4).

3.5.4 Treebank refinements

We apply a set of manual treebank refinements based on previous work. In order to compare the results on Negra with previous work, we do not apply the state splits when working with gold standard pos tags.

For Dutch and German we split the pos tags for the sentence-ending punctuation '.!?'. For all treebanks we add the feature 'year' to the preterminal label of tokens with numbers in the range 1900–2040, and replace the token with 1970. Other numbers are replaced with 000.

Tiger

For Tiger we apply the refinements described in Fraser et al. (2013). Since the Negra treebank is only partially annotated with morphological information, we do not apply these refinements to that treebank.

wsj

We follow the treebank refinements of Klein and Manning (2003) for the Wall Street Journal section of the Penn treebank.

LASSY

The Lassy treebank contains fine-grained part-of-speech tags with morphological features. It is possible to use the full part-of-speech tags as the preterminal labels, but this introduces sparsity. We select a subset of features to add to the preterminal labels:

- nouns: proper/regular;
- verbs: auxiliary/main, finite/infinite;
- conjunctions: coordinating/subordinating;
- pronouns: personal/demonstrative;
- pre- vs. postposition.

Additionally, we percolate the feature identifying finite and infinite verbs to the parents and grandparents of the verb.

For multi-word units (MWU), we append the label of its head child. This helps distinguish MWUs as being nominal, verbal, prepositional, or otherwise.

The last two transformations are based on those for Tiger. Unary NPs are added for single nouns and pronouns in sentential, prepositional and infinitival constituents. For conjuncts, the function tag of the parent is copied. Both transformations can be reversed.

Since the CGN treebank uses a different syntax for the fine-grained POS tags, we do not apply these refinements to that treebank.

3.5.5   METRICS

We employ the exact match and Parseval measures (Black et al., 1992) as evaluation metrics. Both are based on bracketings that identify the label and yield of each constituent. The exact match is the proportion of sentences in which all labeled bracketings are correct. The Parseval measures consist of the precision, recall, and F-measure of the correct labeled bracketings averaged across the treebank. Since the POS accuracy is crucial to the performance of a parser and neither of the previous metrics reflect it, we also report the proportion of correct POS tags.

We use the evaluation parameters typically used with EVALB on the Penn treebank. Namely, the root node and punctuation are not counted towards the score (similar to COLLINS.prm,[9] except that we discount all punctuation, including brackets). Counting the root node as a constituent should not be done because it is not part of the corpus annotation and the parser is able to generate it without doing any work; when the root node is counted it inflates the F-score by several percentage points. Punctuation should be ignored because in the original annotation of the Dutch and German treebanks, punctuation is attached directly under the root node instead of as part of constituents. Punctuation can be re-attached using heuristics for the purposes of parsing, but evaluation should not be affected by this.

___

9 This file is part of the EVALB software, cf. http://nlp.cs.nyu.edu/evalb/
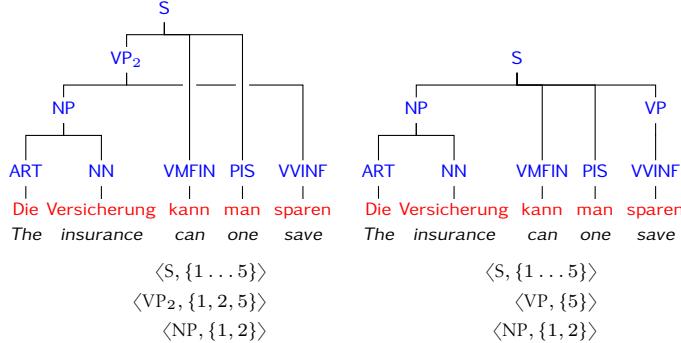
Figure 3.13: Bracketings from a tree with and without discontinuous constituents.

It is not possible to directly compare evaluation results from discontinuous parsing to existing state-of-the-art parsers that do not produce discontinuous constituents, since parses without discontinuous constituents contain a different set of bracketings; cf. Figure 3.13, which compares discontinuous bracketings to the bracketings extracted from a tree in which discontinuity has been resolved by attaching non-head siblings higher in the tree, as used in work on parsing Negra. Compared to an evaluation of bracketings without discontinuous constituents, an evaluation including discontinuous bracketings is more stringent. This is because bracketings are scored in an all-or-nothing manner, and a discontinuous bracketing includes non-local elements that would be scored separately when discontinuity is removed in a preprocessing step.

For function tags we use two metrics:

1. The non-null metric of Blaheta and Charniak (2000), which is the F-score of function tags on all correctly parsed bracketings. Since the German and Dutch treebanks include function tags on pre-terminals, we also include function tags on correctly tagged words in this metric.

2. A combined F-measure on bracketings of the form $\langle C, F, \text{span} \rangle$, where C is a syntactic category and F a function tag.

3.6 Evaluation

This section presents an evaluation on three languages, and with respect to the use of function tags, tree fragments, pruning, and probabilities.

3.6.1 Main results on three languages

Table 3.3 lists the results for discontinuous parsing of three Germanic languages, with unknown word models. The cited works by Kallmeyer and Maier (2013) and Evang and Kallmeyer (2011) also use LCFRS for discontinuity but employ a treebank grammar with relative frequencies of productions. Hall and Nivre (2008), Versley (2014), and Fernández-González and Martins (2015) use a conversion to dependencies discussed in Section 3.4.3. For English and German our results improve upon the best known discontinuous constituency parsing results. The new system achieves a 16 % relative error reduction over the previous best result for discontinuous parsing on sentences of length ≤ 40 in the Negra test set. In terms of efficiency, the Disco-2DOP model is more than three times as fast as the DOP reduction, taking about 3 hours instead of 10 on a single core. The grammar is also more compact: the Disco-2DOP grammar is only a third of the DOP reduction, at 6 MB versus 18 MB compressed size.

Table 3.3 also includes results from van Cranenburgh and Bod (2013) who do not add function tags to non-terminal labels nor apply the extensive treebank refinements described in Section 3.5.3 and 3.5.4. Although the refinements and some of the function tags improve the score, the rest of the function tags increase sparsity and consequently the resulting F-scores are slightly lower, but this trade-off seems to be justified in order to get parse trees with function tags. The results on CGN show a surprisingly high exact match score. This is due to a large number of interjection utterances, e.g., "uhm."; since such sentences only consist of a root node and POS tags, the bracketing F1-score is not affected by this.

3.6.2 Function tags

Table 3.4 reports an evaluation including function tags. For these three treebanks, the models reproduce most information in the original treebank. The following parts are not yet incorporated. The German and Dutch treebanks contain additional lexical information consisting of lemmas and morphological features. These could be added to the non-terminal labels of the model or obtained from an external POS tagger. Lastly, some non-terminals have multiple parents, referred to as secondary edges in the German and Dutch treebanks.

3.6.3 All-fragments vs. recurring fragments

The original Disco-DOP model (van Cranenburgh et al., 2011) is based on an all-fragments model, while Disco-2DOP is based on recurring fragments. Table 3.5 compares previous results of Disco-DOP to the new Disco-2DOP implementation. The second column shows the accuracy for different values of k, i.e., the number of coarse derivations that determine the allowed labeled spans for the fine stage. While increasing this value did not yield improvements using the DOP reduction, with Disco-2DOP there is a substantial improvement in performance,

Parser, treebank	$|w|$	DEV POS	F1	EX	TEST POS	F1	EX
GERMAN							
Negra, van Cranenburgh (2012a)*	⩽ 40	100	74.3	34.3	100	72.3	33.2
Negra, Kallmeyer and Maier (2013)*†	⩽ 30				100	75.8	
Negra, vanCraBod2013, Disco-2DOP*	⩽ 40	100	77.7	41.5	100	76.8	40.5
Negra, vanCraBod2013, Disco-2DOP	⩽ 40	96.7	76.4	39.2	96.3	74.8	38.7
Tiger, Hall and Nivre (2008)	⩽ 40				97.0	75.3	32.6
Tiger, Versley (2014)	⩽ 40				100	74.2	37.3
Tiger, FeMa2015	⩽ 40					82.6	45.9
Tiger, vanCraBod2013, Disco-2DOP	⩽ 40	97.6	78.7	40.5	97.6	78.8	40.8
Tiger, this work, Disco-2DOP	⩽ 40	96.6	78.3	40.2	96.1	78.2	40.0
Tiger, this work, Split-2DOP	⩽ 40	96.6	78.1	39.2	96.2	78.1	39.0
ENGLISH							
wsj, Evang and Kallmeyer (2011)*†	< 25				100	79.0	
wsj, vanCraBod2013, Disco-2DOP	⩽ 40	96.0	85.2	28.0	96.6	85.6	31.3
wsj, this work, Disco-2DOP	⩽ 40	96.1	86.9	29.5	96.7	87.0	34.4
wsj, this work, Split-2DOP	⩽ 40	96.1	86.7	29.5	96.7	87.0	33.9
DUTCH							
Lassy, vanCraBod2013, Disco-2DOP	⩽ 40	94.1	79.0	37.4	94.6	77.0	35.2
Lassy, this work, Disco-2DOP	⩽ 40	96.7	78.3	36.2	96.3	76.6	34.0
Lassy, this work, Split-2DOP	⩽ 40	96.8	78.0	34.9	96.3	76.2	32.7
CGN, this work, Disco-2DOP	⩽ 40	96.7	72.6	64.1	96.7	73.0	63.8
CGN, this work, Split-2DOP	⩽ 40	96.6	71.2	63.4	96.7	72.2	63.3

Table 3.3: Discontinuous parsing of three Germanic languages. POS is the part-of-speech tagging accuracy, F1 is the labeled bracketing F1-score, EX is the exact match score. Results marked with * use gold POS tags; those marked with † do not discount the root node and punctuation. NB: Kallmeyer and Maier (2013) and Evang and Kallmeyer (2011) use a different test set and length restriction. 'vanCraBod2013' refers to van Cranenburgh and Bod (2013), and 'FeMa2015' to Fernández-González and Martins (2015).

with $k = 5000$ yielding the best score among the handful of values tested. Figure 3.14 shows the average time spent in each stage using the latter model on wsj. The average time to parse a sentence (⩽ 40 words) for this grammar is 7.7 seconds. Efficiency could be improved significantly by improving the PCFG parser using better chart representations such as packed parse forests and bitvectors (Schmid, 2004).

Language, treebank	phrase labels	function tags	combined
German, Tiger	78.2	93.5	68.1
English, wsj	87.0	86.3	82.5
Dutch, Lassy	76.6	92.8	70.0

Table 3.4: Evaluation of function tags on sentences $\leqslant 40$ words, test sets.

Model	k=50 F1 %	k=5000 F1 %
DOP reduction: Disco-DOP	74.3	73.5
Double-DOP: Disco-2DOP	76.3	**77.7**
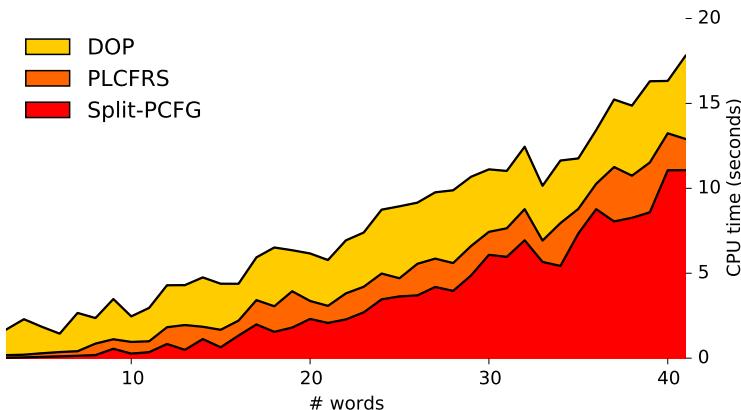
Table 3.5: Comparing F-scores for the DOP reduction (implicit fragments) with Double-DOP (explicit fragments) on the Negra development set with different amounts of pruning (higher k means less pruning); gold POS tags.


Figure 3.14: Average time spent in each stage for sentences by length; disco-2DOP, wsj dev. set.

3.6.4 EFFECTS OF PRUNING

The effects of pruning can be further investigated by comparing different levels of pruning. We first parse the sentences in the Negra development set that are up to 30 words long with a PLCFRS treebank grammar, with $k = 10,000$ and without pruning. Out of 897 sentences, the Viterbi derivation is pruned on only 14 occasions, while the pruned version is about 300 times faster.

Table 3.6 shows results for different levels of pruning on sentences of all lengths. For sentences of all lengths it is not feasible to parse with the unpruned

PLCFRS. However, we can compare the items in the parse forest after pruning and the best derivation to the gold tree from the treebank. From the various measures, it can be concluded that the pruning has a large effect on speed and the number of items in the resulting parse forest, while having only a small effect on the quality of the parse (forest). Note that for the PCFG column, the discontinuous items were not counted incorrectly, leading to the reported percentage being lower than for PLCFRS with $k \geqslant 1000$, when, logically, it should have the highest percentage.

	(PCFG)	$k=100$	$k=1,000$	$k=5,000$	$k=10,000$
CPU time (seconds)	2.461	0.128	0.193	0.444	0.739
Number of items in chart	69,570.5	207.6	282.7	378.2	436.5
Percentage of gold standard items in chart	(94.7)	94.2	97.2	98.1	98.4
F1 score	69.3	69.8	69.9	69.9	69.8

Table 3.6: Results for different levels of pruning; mean over 1000 sentences.

3.6.5 WITHOUT LCFRS

Table 3.3 shows that the Disco-2DOP and Split-2DOP techniques have comparable performance, demonstrating that the complexity of LCFRS parsing can be avoided. Table 3.7 shows the performance in each step of the coarse-to-fine pipelines, with and without LCFRS. Note that the relative difference in scores is not meant to be indicative here (previous experiments with a different setup gave the opposite result), the aim is to establish that LCFRS can be replaced with a weaker formalism while maintaining comparable performance.

To gauge the performance of discontinuous constituents in particular, we can restrict the evaluation of the models in Table 3.3 to discontinuous bracketings; see Table 3.8 for the results. The Split-2DOP model consistently scores slightly lower, but it is conceivable that this difference would disappear given enough data. Also, the manual refinements yield improvements on discontinuous constituents when compared to the results of van Cranenburgh and Bod (2013).

We can conclude that for modeling discontinuous constituents, a formalism that explicitly models discontinuity as an operation is dispensable, and can be replaced with a simpler model in which discontinuities are only modeled probabilistically by encoding them into labels and fragments. This demonstrates that given the use of tree fragments, discontinuous rewriting through LCFRS comes at a high computational cost without a clear benefit over CFG.

Pipeline	F1 %	EX %
Split-PCFG (no LCFRS, no TSG)	65.8	28.0
Split-PCFG $\Rightarrow$ PLCFRS (no TSG)	65.9	28.6
Split-PCFG $\Rightarrow$ PLCFRS $\Rightarrow$ 2DOP	77.7	41.5
Split-PCFG $\Rightarrow$ Split-2DOP (no LCFRS)	78.1	42.0

Table 3.7: Parsing discontinuous constituents is possible without LCFRS (Negra dev. set, gold POS tags; results are for final stage).

3.6.6 THE ROLE OF PROBABILITIES

From the results it is clear that a probabilistic tree-substitution grammar is able to provide much better results than a simple treebank grammar. However, it is not obvious whether the improvement is specifically due to the more fine-grained statistics (i.e., frequencies of more specific events), or generally because of the use of larger chunks. A serendipitous discovery during development of the parser provides insight into this: during an experiment (Negra dev. set, gold POS tags), the frequencies of fragments were accidentally permuted and assigned to different fragments, but the resulting decrease in performance was surprisingly low, from 77.7 to 74.1 F1—suggesting that most of the improvement over the 65.9 F1 score of the PLCFRS treebank grammar comes from memorizing larger chunks, as opposed to statistical reckoning.

3.6.7 PREVIOUS WORK

Earlier work on recovering empty categories and their antecedents in the Penn treebank (Johnson, 2002b; Levy and Manning, 2004; Gabbard et al., 2006; Schmid, 2006; Cai et al., 2011) has recovered non-local dependencies by producing the traces and co-indexation as in the original annotation. If the results include both traces and antecedents (which holds for all but the last work cited), the conversion to discontinuous constituents of Evang and Kallmeyer (2011) could be applied to obtain a discontinuous F-score. Since this would require access to the original parser output, we have not pursued this.

As explained in Section 3.5.5, it is not possible to directly compare the results to existing parsers that do not produce discontinuous constituents. However, the F-measures do give a rough measure, since the majority of constituents are not discontinuous.

For English, there is a result with 2DOP by Sangati and Zuidema (2011) with an F1 score of 87.9. This difference can be attributed to the absence of discontinuous bracketings, as well as their use of the Maximum Constituents Parse instead of the Most Probable Parse; the former optimizes the F-measure instead of the exact match score. Shindo et al. (2012) achieve an F1 score of 92.9% with a Bayesian TSG that uses symbol refinement through latent variables

Parser, treebank	DEV			TEST		
	Rec.	Prec.	F1	Rec.	Prec.	F1
GERMAN						
Tiger, vanCraBod2013, Disco-2DOP	40.0	51.3	44.9	38.0	50.7	43.5
Tiger, this work, Disco-2DOP	39.9	54.7	46.1	38.0	54.0	44.6
Tiger, this work, Split-2DOP	36.6	52.6	43.2	35.4	51.7	42.0
ENGLISH						
WSJ, vanCraBod2013, Disco-2DOP	65.1	66.3	65.7	58.0	73.3	64.8
WSJ, this work, Disco-2DOP	62.1	69.9	65.8	56.6	77.9	65.6
WSJ, this work, Split-2DOP	56.5	66.4	61.0	52.6	77.5	62.7
DUTCH						
Lassy, vanCraBod2013, Disco-2DOP	55.4	62.5	58.7	51.7	58.6	54.9
Lassy, this work, Disco-2DOP	56.2	62.5	59.2	53.5	59.8	56.5
Lassy, this work, Split-2DOP	51.5	61.7	56.2	49.4	60.5	54.4
CGN, this work, Disco-2DOP	13.3	55.6	21.4	15.9	58.0	25.0
CGN, this work, Split-2DOP	9.5	50.3	15.9	13.9	57.3	22.3

Table 3.8: Evaluation restricted to discontinuous constituents. All results for sentences $|w| \leqslant 40$.

(i.e., automatic state splitting).

For German, the best results without discontinuity and no length restriction are F1 scores of 84.2 for Negra (Petrov, 2010) and 76.8 for Tiger (Fraser et al. 2013; note that this result employs a different train-dev-test split than the one in this work).

3.7 SUMMARY

In this chapter we have shown how to parse with discontinuous tree-substitution grammars and presented a practical implementation. We employ a fragment extraction method that finds recurring structures in treebanks efficiently, and supports discontinuous treebanks. This enables a data-oriented parsing implementation that employs a compact, efficient, and accurate model for discontinuous parsing in a generative model that improves upon previous results for this task.

Surprisingly, it turns out that the formal power of LCFRS is not necessary to describe discontinuity, since equivalent results can be obtained with a probabilistic tree-substitution grammar in which non-local relations are encoded in the non-terminal labels. In other words, it is feasible to produce discontinuous constituents without invoking mild context-sensitivity.

We have presented parsing results on three languages. Compared to previous

work on statistical parsing, our models are linguistically richer. In addition to discontinuous constituents, our models also reproduce function tags from the treebank. While there have been previous results on reproducing non-local relations or function tags, in this chapter we reproduced both, using models derived straightforwardly from treebanks, while exploiting ready-made treebank transformations for improved performance.

Part II
Literature

4 *Predictive Modeling Background*

In which we introduce machine learning methodology.

> Natural language processing (NLP) made a lot of progress when we started extracting grammatical knowledge from data, rather than asking experts to write it down for us. We should do the same for stylistic knowledge. The experts do provide great insights into what *kinds* of patterns might be relevant to grammar or style...but their informed hypotheses about what to look for are not a substitute for actual looking.
> — Eisner (2015); emphasis in original

T HIS CHAPTER introduces several background topics related to statistics and machine learning, in particular as applied to text. We first reflect on the differences between statistics and machine learning, and consider pitfalls in statistical inference. The last section provides exposition on aspects of machine learning relevant to this thesis.

4.1 METHODOLOGICAL CONSIDERATIONS

Statistics can be divided in several branches, depending on the kind of answers that are sought. Descriptive statistics summarizes data from a sample, e.g., mean, median, and standard deviation. Inferential statistics draws conclusions from a sample about a population, based on assumptions and hypotheses. We focus on inferential statistics here since it is more difficult to get right.

4.1.1 EXPLANATION VERSUS PREDICTION

A useful distinction to draw within inferential statistics is between explanation and prediction (Breiman, 2001; Shmueli, 2010). Explanation seeks to confirm causal theoretical models; typical examples include statistical hypothesis testing based on Student's t-test, χ^2 tests, and regression models. The aim of predictive models is to learn from data and achieve good generalization performance as

measured on new, unseen data. In an explanatory model the theory is specific and specified in advance, while a predictive model is data oriented; although specific aspects of the data are selected through feature engineering, no claim about the distribution of these features is made in advance.

Explanatory modeling is common in the social sciences such as psychology, sociology, and econometrics. Predictive modeling is less common in the sciences, but it is central to computational linguistics, natural language processing, bioinformatics, and other fields that employ machine learning or forecasting.

In predictive modeling it is crucial to report results on out-of-sample tests (using held-out data or cross-validation). Neither the feature selection nor the training of the model should in any way be based on the out-of-sample data used for evaluation. In contrast with explanatory models, there is no spelled out theory describing the relation between the observations and target values (also known as independent and dependent variables). Therefore, more data is useful to deal with data sparsity, allowing more interactions to be uncovered from the data. In contrast, explanatory models require a sample of sufficient size, but more data would only increase the chances of finding a significant association with small effect size, which is likely to be a false positive. Aside from the sample sizes, there is also a difference in the dimensionality, i.e., the number of features or independent variables. Predictive models can exploit or select from a large set of features, while explanatory models tend to use a smaller, predefined set of variables derived from a theoretical model.

This thesis will employ correlations and predictive models. The nature of the problem arguably makes predictive models more suitable. The set of textual features that determine literariness cannot be enumerated, and it does not seem feasible to define a causal model relating such features to literary ratings.

4.1.2 Simpson's paradox

Simpson's paradox is, properly speaking, neither Simpson's nor a true paradox. The phenomenon was first described by Yule (1903) and popularized by Simpson (1951); it concerns the surprising phenomenon that the association between two variables in a data set may be reversed when subsets of the data are analyzed. The phenomenon is a specific instance of the Omitted-Variable bias. An illustration is shown in Figure 4.1. It is important to be aware of this paradox because it can lead to incorrect conclusions being drawn about correlation and causality. Whenever a correlation is obtained, the effect of subsets should be considered.

A famous case occurred in 1973 at the University of California-Berkeley (Bickel et al., 1975). The university was sued for gender discrimination; 44 % of male applicants had been accepted, compared to only 35 % of female applicants. However, if the rate of acceptance was considered by department, female applicants actually had a small but significant advantage. Concretely, women tended to apply to departments that had more competition. There were more female applicants for the humanities department, and more male applicants to
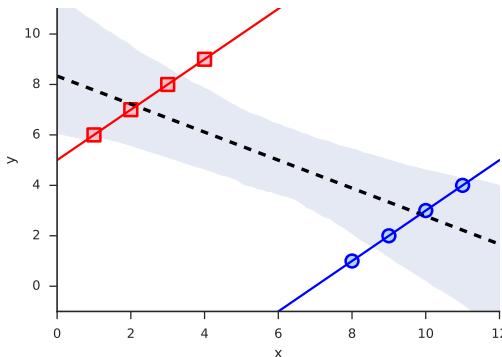
Figure 4.1: An illustration of Simpson's paradox. The blue and red groups both show a positive correlation, while on aggregate, there is a negative correlated (dashed line).

the science department. Furthermore, the humanities department had fewer requirements but also fewer available slots, while the science department had specific requirements but was more likely to accept someone that met them.

What this example reveals is that overlooking an important variable (here the choice of department and its characteristics), and specifically considering data in aggregate, may result in the opposite conclusion compared to the one which should be drawn.

4.2 Machine learning from text

Since much of the rest of this thesis is based on machine learning, we now provide some background. Machine learning techniques include predictive modeling. Applying machine learning methods to text is also known as text mining or text analytics. The document-level tasks are:

information retrieval: find relevant documents,

supervised models: label documents with some predefined set of labels (text categorization, classification), or predict a continuous value (regression),

unsupervised models: automatically induce groupings of documents without predefined criteria (clustering), or summarize the features of documents (dimensionality reduction).

Because of the nature of the task and data at our disposal, we will focus on supervised models.

A central notion in text mining is that of the vector space model, a mathematical abstraction over texts. Given a set of documents and a set of numeric features extracted from them, each document can be represented by a feature vector. This feature vector can be interpreted as the coordinate of the document

in a high-dimensional space. Documents with similar features will be close to each other in this space.

A particularly simple but successful instance of this is the bag-of-words (BoW) model. In the BoW-model the features consists of word counts from the corpus. Note that a bag (also known as a multiset, an unordered set in which each item is associated with a count) abstracts over the original data by throwing away all information on order and consecutively occurring items. The only information that remains are pairs of (word, count) in each document.

Variants replace the word feature with n-grams (n consecutively occurring words) or other linguistic structures than can be counted. In any case it should be noted that the vector space model results in a high-dimensional feature space. It turns out that, in practice, a vector space model of text contains few irrelevant features, while the document vector of a given text may contain many zeros for features it does not contain. A machine learning model of text is therefore learning a "dense" concept (almost all features are relevant for predicting the target) from sparse inputs (each text contains a different subset of features) (Joachims, 1998, p. 139).

Making predictions from data requires a model. Two important classes of predictive models can be distinguished:

Lazy, memory-based: Store instances, and compare instances at test time to make a prediction. For example the nearest neighbor method looks for the most similar instances and derives its prediction from them. Notably, these models are not trained.

Trained models: Fit a model to training data, while aiming to generalize over the data.

Perennial problems with supervised models are overfitting and underfitting. Overfitting occurs when irrelevant particularities of the data such as noise are incorporated in the model, resulting in a model that is successful at predicting the training data, but less so at generalizing over new data. Underfitting is the converse, where useful patterns in the data are ignored, and cannot be applied to new data.

Memory-based models are interesting in that they can base their predictions on arbitrary aspects decided at test time. On the other hand, the model does not learn generalizations that can be inspected, only particular predictions can be inspected. Nearest neighbor methods are prone to overfitting and do not handle high dimensional problems well.

4.2.1 Linear models

One particularly useful and general class of trained models are Linear Models. A linear model assumes that predictions are made from a linear combination of feature values and respective feature weights:

$$\hat{y}_{w,X} = w_0 + w_1 X_1 + ... + w_p X_p$$

Here $\hat{y}$ is the predicted value given weights w and feature values X. Features cannot interact—they make independent contributions to the final prediction. Some classification problems are not linearly separable and thus can never be learned in such a model no matter the amount of data; the xor function is a prototypical case.

This may seem like a severe limitation. The trade off is that these restrictions make it possible to learn from a large number of features, and to do so efficiently. Linear separability is not an issue in practice: "Most text categorization problems are linearly separable" (Joachims, 1998, p. 140). This is due to the high dimensionality. Given p dimensions, the number of data points that can be separated with a linear classifier is $p + 1$ (Hastie et al., 2009, p. 238); a similar result applies for continuous values as well. Moreover, since the model is simple, it is also easy to interpret. Each feature receives a weight which denotes its contribution to the final prediction. Features with both a high weight and high frequency in the data are thus the most important.

Let us briefly consider non-linear models. Non-linear models are strictly more powerful than linear models, since they can model linear patterns as well as particular kinds of non-linear patterns. For certain problems, e.g., when the number of features is small ($p \ll n$), or when the underlying problem is non-linear (e.g., vision), non-linear models tend to be superior. However, the downside of the power of non-linear models is their tendency to overfit if model complexity is not controlled.

For any reasonable machine learning model, theoretical guarantees can be made that as the data grows to infinity, the difference between train and test error becomes arbitrarily small. This means that the choice of model comes down to more practical concerns: speed of convergence, ability to handle noise, robustness to outliers, generalization from limited training data, handling of different types and number of features, and number of parameters that need to be tuned. Taking these concerns into account, as well as some experimentation, leads us to opt for linear models in this thesis. Experiments with non-linear models did not yield improvements, and the simplicity of linear models provides a clear advantage.

A popular kind of linear model are linear Support Vector Machines (svm). svm can be used for classification and regression (predicting a discrete or a continuous target variable, respectively); it is a method of learning that is robust against outliers and overfitting through the use of regularization. This contrast with the standard method for linear regression used in applied statistics, namely Ordinary Least Squares (ols). ols has the property of being underdetermined when the number of features is larger than the number of samples ($p > n$); regularization overcomes this (Hastie et al., 2009, ch. 18).

4.2.2 Challenges

Shalizi (2012) identifies three increasingly difficult challenges in solving regres-

sion problems:

1. Low prediction error
2. Consistent estimates: approximate true feature weights
3. Feature selection

The first is the criterion for good predictions; it is directly estimated by cross-validation, and therefore relatively easy to optimize. The second and third are interesting objectives if we are after more than just predictions and aim to converge on the 'true' model, where true refers to the parameters beyond the data set. This is the aforementioned opposition between prediction and explanation. This thesis will focus only on the first goal. First since it is practical to estimate and evaluate. Second because it does not seem feasible to formulate a potential 'true' set of features and weights for a text mining problem, there are simply too many potential factors. Note that this does not mean feature weights and selection will not concern us, but that when they do, we are considering them with respect to a given data set (within-sample), without knowing whether they would be representative out-of-sample.

Lastly, it should be noted that fitting the model is often not the hard part of machine learning. Given a well-defined problem such as document or image classification, many models can be fitted and tuned, and there will always be some improvement that can be obtained by tweaking (subject to diminishing returns). Another issue is the data: more is typically better (although representative and high quality data is even more important), and it needs to be cleaned and preprocessed properly. Still, this is a practical though not necessarily hard problem. The fundamental, non-technical challenges are to identify and define the (right) problem, and to measure the success of its solutions.

An illustration is the case of the leopard print sofa (Khurshudov, 2015). Convolutional networks have reached impressive accuracy scores in image recognition—they have even been claimed to surpass human performance, for example in distinguishing leopards, jaguars, and cheetahs. However, when presented with a picture of a sofa with a leopard print, the model is fooled and classifies it as a leopard. This illustrates two problems. The first is how to ensure that the model has been evaluated properly; it is not feasible to include every kind of anomalous object in the test set. The second is that it shows that the model is not learning the right thing; in this case, the larger structural leopard features, instead of just the local skin texture.

4.2.3   Explicit, sparse versus implicit, dense representations

In this thesis we work with explicit, sparse representations. Sparse representations, such as the aforementioned BoW-features, are high dimensional where each dimension is an independently measured, predefined feature. The sparsity refers to the fact that only a small number of features may receive a non-zero value in a document, e.g., specialist vocabulary will occur only in certain documents.

The opposite of such representations are implicit, dense features. Distributed word embeddings (e.g., word2vec, Mikolov et al. 2013; GloVe, Pennington et al. 2014) are a popular instance, inspired by so-called 'deep learning' techniques for neural networks. In a model with sparse features, 'blue' and 'red' would be two separate features, represented in different parts (dimensions) of the feature vector. With dense features, each word would be represented by its own vector, and 'red' and 'blue' would have similar vectors (where similar can be defined in terms of a distance metric) reflecting their semantic similarity and tendency to occur in similar contexts. A prototypical example of what can be done with word embeddings is solving word analogies, for example 'king - man + woman = queen.' This would not be impressive if the model were trained on specific knowledge, but in fact this is learned purely from raw text.

Using dense features is outside of the scope of this thesis. The motivation to focus on sparse features is that they are well suited to the amount of text (401 novels), and sparse linear models are easier to interpret. Both word2vec and GloVe are trained on billions of words; the word2vec extension for document classification (doc2vec; Le and Mikolov, 2014), is trained on 100k instances in the experiments presented. There are no off-the-shelf trained models available for Dutch such as for English. Nevertheless, it is an interesting project for future work to apply word embeddings trained on a suitably large reference corpus to the analysis of literature.

4.2.4 Evaluation metrics

As noted before, supervised models consist mainly of classification and regressions problems. For classification, the accuracy metric is the most straightforward; i.e., the percentage of correctly predicted items. For regression the task of evaluation is more involved, since it is not useful to only consider items that were predicted exactly, we rather want to express how close the predictions are to being correct. We will use the following metrics:

Root Mean Squared (rms) error: The mean distance of the predictions from the true values (residuals); has the same scale as the target values; i.e., in our case the error will range from 0–7. Given an arbitrary novel, this metric expresses how far the model's prediction is expected to be from the true value.

R^2, also known as the coefficient of determination. When used on out-of-sample predictions, as in our case, this metric expresses how well the predictions of the model generalize to new data points. The value is based on the ratio of the mean squared error of the predictions divided by the mean squared error of a baseline model which always predicts the mean of the target value:

$$R^2 = 1 - \frac{\text{MSE}(y_{true}, y_{pred})}{\text{MSE}(y_{true}, \text{mean}(y_{true}))}$$
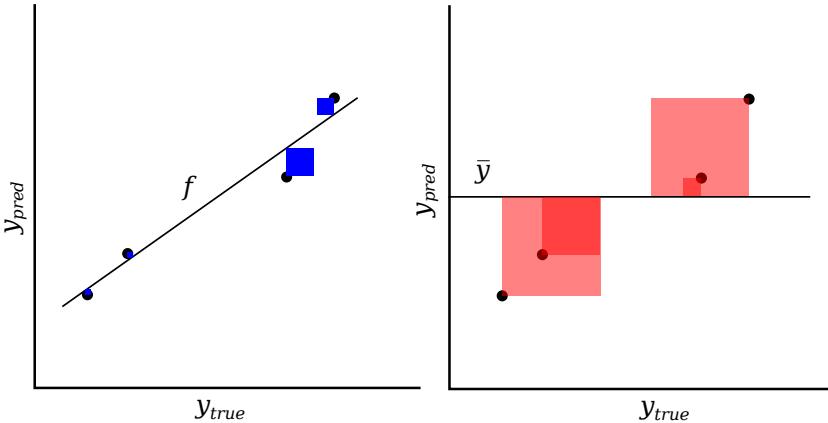
Figure 4.2: An illustration of R^2, the coefficient of determination. The coefficient of determination is one minus the ratio of the area of the blue squares on the left (the predictions) versus the area of the red squares on the right (the baseline). (Figure by Orzetto, CC BY-SA 3.0, via Wikimedia Commons).

See the illustration in Figure 4.2. Note that the result may be negative, since the predictions can be arbitrarily bad compared to the baseline, and despite the name, R^2 is not necessarily the square of any particular quantity. Since this metric can be interpreted as expressing the percentage of variation explained, we report it as a percentage, with 100 % being a perfect score.

KENDALL τ, a rank correlation. Expresses the similarity between the predictions and the target value in terms of ranking, while ignoring the magnitude of differences. The result is a correlation ranging from -1 to 1, with 1 representing a perfect ranking. This metric is useful to consider because by considering only ranks, any noise that might be present in the differences between items is ignored.

5 *Literary Investigations*

In which we introduce the experimental setup of an empirical, computational study of literary language.

> Shakespeare, we say, was one of a group of English dramatists working around 1600, and also one of the great poets of the world. The first part of this is a statement of fact, the second a value-judgment so generally accepted as to pass for a statement of fact. But it is not a statement of fact. It remains a value-judgment, and not a shred of systematic criticism can ever be attached to it.
> — Northrop Frye (1957, p. 21), *Anatomy of Criticism*

LITERATURE consists of written or spoken language that is distinguished in some way. It is usually divided into poetry and prose, fiction and non-fiction, and major genres such as drama, lyric, short story, and novel. This thesis will focus on prose fiction, and novels in particular.

This chapter considers possible definitions of literature, and presents the project to investigate the relation of literary appreciation and textual aspects empirically. The next sections serve to introduce the task of text analysis by providing an overview of style and stylometric aspects, and an introduction to Dutch syntax.

5.1 DEFINITIONS OF LITERATURE

Literature has proven to be difficult to define. In broad strokes, three approaches can be distinguished:[1]

THE VALUE-JUDGMENT DEFINITION, or the *Belles-lettres* tradition, states that what counts as literary is a value judgment (Eagleton, 2008, p. 9), principally of aesthetic quality. This does not mean that literature is synonymous with

---

[1] This is only a brief overview; for a more systematic treatment, cf. Eagleton (2008, p. 1–14). Some of the references and structure of this section is based on the Wikipedia article on literature.

good writing, but that it is the kind of writing that is valued. This definition does imply that in principle any text may come to be viewed as literature, and *vice versa*, any text which is currently viewed as highly literary (e.g., Shakespeare), may lose that status; i.e., it implies that literature is variable, and not an inherent quality of a work (Eagleton, 2008, p. 9).

THE FORMALIST DEFINITION holds that literature is distinguished by its form, through a property called *literariness*:

> The sum of special linguistic and formal properties that distinguish literary texts from non-literary texts, according to the theories of Russian Formalism (Baldick, 2008)

Specifically, foregrounding and defamiliarization distinguishes poetic language from standard language (Mukarovsky, 1964), i.e., the way language draws attention to itself, and the effect of 'making language strange,' respectively. Foregrounding and defamiliarization seem to work best for poetry. On closer inspection, two issues arise: non-standard language can also occur in everyday, non-literary situations such as advertising; and *vice versa*, a literary novel may consist entirely of everyday situations, described in everyday language.

THE SOCIAL DEFINITION states that literature is a matter of prestige decided by the social elite, i.e., publishers and critics (Bourdieu, 1996). Taken to its extreme, this definition predicts that any text could be seen as literary, provided the right social cues exist (compare with the so-called readymades in art), regardless of the content or style of the work in question. Needless to say, this goes against our intuitions. It is, however, the theory that is currently most in vogue among literary scholars in the Netherlands. Adherents believe that social factors explain literary status to a large degree, or sometimes even that it is the sole determining factor.

It is clear that literature presents a demarcation problem: none of these definitions is satisfactory. There is a parallel with the demarcation problem in science. Just as with science, it is not necessary to solve this problem to participate in the production and appreciation of literature. Moreover, it is unlikely that writers and readers will look to literary theorists to adjudicate the boundaries of the literary, despite the apparent normative character of the demarcation problem. It is rather the opposite, a theory of what is literary must adequately describe the value-judgments of readers and writers.

As an alternative to a criterial definition with necessary and sufficient properties, a prototype definition can also be attempted. Meyer (1997, p. 4) suggests that prototypical (Western) literary works:

- are written texts[2]

---

2 Note that there are traditions of oral literature for which this property is not prototypical.

- are marked by careful use of language, including features such as creative metaphors, well-turned phrases, elegant syntax, rhyme, alliteration, meter
- are in a literary genre (poetry, prose fiction, or drama)
- are read aesthetically
- are intended by the author to be read aesthetically
- contain many weak implicatures (are deliberately somewhat open in interpretation)

Since this is a prototype definition, each of these properties allows for exceptions, and may be present to different degrees, and there is no minimum number of properties that must be met. Analogously, a prototype definition of a bird would include flying as a salient property—even though there are clear exceptions (e.g., chickens, penguins, kiwis) which precludes its use in a taxonomic definition. The combination of properties does make an instance more or less of a prototypical example of a concept.

Another avenue is to contrast literary novels with popular novels; it may give insight to consider what kinds of things a literary novel is *not*. In concrete terms it is often said that novels may be divided into *character-driven* and *plot-driven* (e.g., Towey, 2000, p. 137). This often coincides with literary and popular novels, respectively. Where, for example, a popular novel such as a suspense novel may revolve around a murder case and its resolution, a literary novel will often focus on the thoughts and emotional states of its protagonists. This has prompted research that attempts to show that reading literary fiction improves empathy (Bal and Veltkamp, 2013; Kidd and Castano, 2013).

A further opposition is that between the importance of plot versus writing style: the literary novel may primarily be appreciated for the aesthetics or originality of its style, as opposed to the entertainment value of its plot. As the formalists would say, literary language draws attention to itself by deviating from a norm, relative to the expectations of the context (Eagleton, 2008, p. 2–5). Suspense novels are often fast paced both in terms of plot and writing style, and are consequently easy and quick to read (a "page turner"). Literary novels may be valued for being thought provoking and containing more *depth*; Miall and Kuiken (1999) extend the Formalist definition of literature with the condition that a literary work has a transformative effect on the reader. Instead of spelling out all plot details and character motivations, a literary novel leaves more to the imagination (related to the dictum *show, don't tell*), and may be more demanding of the reader; this relates to the implicatures cited by Meyer (1997).

Unfortunately some of these aspects are rather difficult to operationalize, since they hinge on latent variables such as plot, pace, and emotion, and how these interact with the reader. This thesis will focus on stylistic aspects that can be measured computationally from the text.

5.1.1 Historical emergence of the concept of literature

The modern, Western notion of literature as a mark of distinction for the most valuable fine writing emerged in the late eighteenth century, as Heumakers (2015) argues. This is when, according to Heumakers, the 'aesthetic revolution' occurred; before then the question whether a text had enough aesthetic value to be called literature was simply not considered nor would it have been understood. The concept of art and literature that developed within Romanticism is arguably influential to this day.[3]

The central idea is that literature is *autonomous*, i.e., not in the service of moral, political, or religious goals. The concept of autonomy in aesthetics was introduced in Kant's *Kritik der Urteilskraft* (critique of judgment). Kant (1790, §40) introduces the notion of a *sensus communis* (literally common or communal sense) to explain that taste judgments in aesthetics are subjective, but cast with the expectation that others ought to agree with them, and thus having the potential of being valid (normativity). Autonomy, Kant argues, is a pre-condition for such taste judgments.

Before the emergence of autonomous aesthetics, a work of art primarily had an instrumental function to teach through entertainment what is true and good, with beauty in a subordinate role. Since the aesthetic revolution, literature is characterized by originality (creativity), being about more than mere entertainment, not being bound by any genre in terms of style and topics, and social criticism.

5.2 The Project: The Riddle of Literary Quality

The goal of the Riddle of Literary Quality[4] is to investigate the concept of literature empirically, specifically in the form of its textual correlates. There has been little empirical work on literature, while theoretical work tends towards invoking social and cultural factors to explain what comprises literature. The project aims to address this by testing the hypothesis that textual factors are involved, and to estimate the degree to which they influence and explain perceptions of literary value. Although the name of the project refers to 'literary quality,' this notion is problematic since it suggests an essentialistic concept. What we will investigate is more aptly described as the conventions that give rise to the spectrum of literary and non-literary works. For the purposes of the experiments we make use of empirical data on how literary the novels are, namely survey ratings (reader judgments). We then look for the reflection of literary conventions in textual characteristics (various stylistic and topical features), by relating survey and texts.

---

3 The Dutch word for a novel, *roman*, is derived from the same root as Romanticism, although the former predates the latter.

4 Cf. http://literaryquality.huygens.knaw.nl/

5.2.1 The corpus of novels

The corpus consists of 401 recent Dutch novels, whose first Dutch edition was published 2007–2012. Both translated and originally Dutch novels are included (249 and 152, respectively). The corpus contains 210 novels by women writers, and 190 by men. The novels are selected by their popularity in the period 2010–2012. The criteria were being either best selling (382 novels), most often lent from public libraries (19 novels, not counting 189 novels that were already included due to being best selling). While the aim was to restrict the corpus to novel-length fiction texts published 2007–2012, several issues were discovered after the corpus had been finalized:

- Some older novels have been accidentally included; e.g., Noort, *De Eetclub*, 2004.
- Five short-story collections have been included; e.g., King, *Full dark*, *no stars*.
- Five non-fiction texts (not counting novels with autobiographical elements) have been included; e.g., Mak, *Reizen zonder John*.

See Section A.2 for the full list of novels in the corpus.

Each novel has a publisher assigned code, the so-called NUR[5] code. These codes reflect broad prose genre distinctions and are used by book sellers to organize book shops. Note that in general, the NUR code is only mentioned in the fine print of the front matter, so the reader is usually not aware of it directly, but rather indirectly through the book's placement among other books. In some cases publishers choose to mention the genre explicitly on the cover; this is the case with literary thrillers, for example. Table 5.1 shows the NUR codes in the corpus, and the number of novels for each code.

Some NUR codes are questionable. For example, Fifty Shades of Grey by E.L. James is listed as 302, (translated) literary novel, when it clearly is not (it is actually the lowest rated book in the survey, see below). The author Esther Verhoef has four novels in the corpus listed as 305 literary thriller, and one as 301 literary novel. This does not necessarily reflect a true difference in genre. Although it does happen that writers consciously attempt a switch to a different genre, publishers can also decide on their own to change the genre code strictly for marketing purposes.

The case of the literary thriller is interesting. Its code, 305, reflects that it is clustered with other literary novels (301, 302), and not with other thrillers or suspense novels (330, 331, 332). Again, this does not necessarily reflect the actual perceptions of readers, which is an empirical matter, but it is an interesting case of the publisher's influence on the literary market, which might also influence perceptions of literary merit.

Taking into account some of these questionable classifications, and the fact that some of the more fine-grained categories are not useful given the number

---

5 NUR is an acronym for *Nederlandstalige Uniforme Rubriekscode*; roughly, 'uniform categorization of Dutch language [novels].'

code	description	novels
305	literary thriller	99
302	translated literary novel	93
301	literary novel	78
332	thriller	58
331	detective	10
343	romance	10
330	general suspense	9
340	general popular fiction	4
344	regional and family novels	4
342	historical novel (popular)	2
312	popular fiction pockets	1
284	fiction for ages 13-15	1
334	fantasy	1

Table 5.1: A breakdown of the NUR classifications in the corpus.

of novels, we can construct a simplified categorization of four coarser categories: Fiction (148), Suspense (186), Romantic (41), Other (26). NB: All novels are fiction; Suspense and Romantic are genre-novels while Fiction are non-genre novels; we avoid using the term literary novels here because that is a judgment that should be based on the reader opinions. Lastly, Other is a category for novels that do not fit into the previous three categories, but are not numerous enough to form their own category (e.g., science fiction, fantasy, etc), or for texts that turned out to be short story collections or non-fiction.

Two aspects of the corpus selection stand out: selecting by popularity, and selecting contemporary novels. The choice to select the novels by popularity means that it is easier to obtain a large number of judgments from the general public, and it removes a source of bias that would be introduced if the selection was made with subjective criteria (either our own selection, or by critics or a jury). It also means that the corps contains both non-literary and literary novels, which is necessary to be able to learn the difference.

However, popularity selects for a certain kind of work with broad appeal, and this may well exclude the most literary novels.[6] Additionally, the best selling novels may also be the novels for which the publisher's influence and marketing is the strongest, for two reasons: as bestsellers they are *ipso facto* the best marketed novels, and the publisher may have allocated resources based on the previous success of an author or a type of novel. When a particular literary (writing) style gains currency through such publisher influence, the publisher is a confounding factor. That is, our experiments may find an association between

---

6 On a related note, *The Shawshank Redemption* has been the top rated movie on IMDB for years. How many people would actually believe that this movie represents the pinnacle of a century of cinema? It rather seems to be the kind of movie that is easy to agree on.

writing style and literary appreciation, but the question remains how much both were influenced by the publisher, and how much the influences overlap.

The choice to select recent novels removes diachronic factors from the research. It also increases the chances that the opinions by readers are their own, as opposed to a settled consensus influenced by literary critics and awards. While this can never be excluded, the influence would be greater for a celebrated literary classic.

Literary classics would be interesting to study, as they represent the kind of writing that is celebrated for centuries—the literary canon. This corpus cannot help in uncovering what differentiates literary classics from more pedestrian literary novels. The appreciation of literary classics is historically variable and institutional factors would form much more of a confounding variable in a survey on literary classics because the literary status of an established classic has been sedimented over a long time; furthermore, it is more tempting to give socially acceptable answers about the sort of books that 'everyone wants to have read but no one wants to read.'

We therefore focus on literary evaluations of contemporary, popular novels.

5.2.2 The reader survey

The reader survey was held online, from March until September 2013 (Jautze et al., 2016b).[7] Participation was open to anyone and participants were recruited from the general population by advertising on social media and mailing lists. Roughly 14,000 people took the survey; about 138,000 ratings were collected. The main part of the survey was to obtain ratings of the degree to which novels are perceived to be literary and good. Given the list of 401 novels, participants were asked which novels they had read, and for those novels, Likert scale items were presented to give ratings on how literary and how good each novel was. No definition was given for either of the two dimensions, in order not to influence the intuitive judgments of participants. The notion of literature is therefore a pre-theoretical one, reflecting the views of the participants with minimal direction.

It may not be entirely obvious to consider literary and good as separate aspects of a novel; for example, if literature is a matter of aesthetics, or more generally of highly-valued writing, literary and good might seem to coincide, but as argued before, they should be distinguished. It is useful to consider them as separate axes because in this way we may collect data on novels that are appreciated but recognized as non-literary, such as a popular thriller, while conversely allowing for novels recognized as literary (e.g., because of a distinctive style or subject matter) but otherwise disappointing.
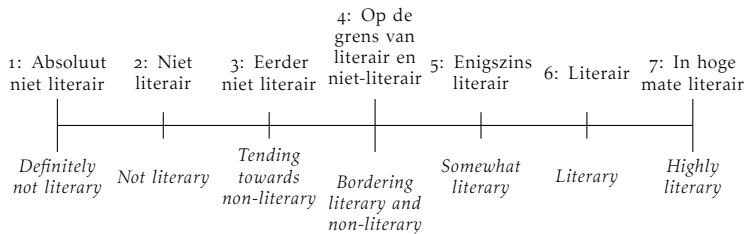
The Likert scale (pronounced "lick-ert") is a tool for the self-reporting of subjective variables. This scale is widely used in social science research, and is

---

7 Cf. http://www.hetnationalelezersonderzoek.nl

also the way in which the ratings for the Riddle of Literary Quality survey were collected. Therefore, it is worth taking a closer look at.

A Likert scale offers a fixed number of discrete responses, for example, strongly disagree, disagree, neutral, agree, and strongly agree. Strictly speaking, these have to be treated as ordinal categories: it is only known how these categories are ordered, not what the distance between each is, or whether there is a reference point (such as absolute zero with temperature and lengths). In the Riddle survey, the following 7-point scale was used for the literary ratings:

Respondents also had the option to choose 'don't know'; these ratings are not used for the results in this thesis. It can be argued that, under certain conditions, it is reasonable to consider the Likert scale as an interval scale, which justifies taking its mean and applying other parametric methods. These conditions are that the scale is symmetric, that it contains a sufficient number of choices, that a sufficient number of responses are viewed in aggregate, and lastly that the underlying concept is continuous. We will work with these assumptions in this thesis, so as to be able to work with the means of responses per book.

The ratings were collected purely based on the title and author of each novel, and not, for example, based on a sample of the writing in the novel. On the one hand this is a limitation, as the ratings will at best reflect the subjects' association or recollection of the writing style or other aspects of the novel involved, and the survey ratings cannot help in pinpointing what aspect of the novel was appealing. On the other hand it ensures that the hypothesis that the text influences the judgments of the novel is not assumed out of hand without being tested. In particular, it makes it possible to gauge to success of explaining literary ratings from textual features, without prompting the survey participants to consider any of these factors. However, note again that it is ultimately not possible to distinguish textual and social or cultural factors, since they may overlap. For example, textual factors may have influenced a critic who is in turn influential. The goal of the experiment will therefore be to estimate the degree to which literary ratings can be predicted from text.

Another point to note is that the survey seeks out the opinions of non-experts.[8] Other empirical literary research such as Louwerse et al. (2008) employs literary judgments by authors or critics. Both perspectives are interesting

---

8 There may be literary experts such as critics and other professionals part of literary production among the survey participants, but given the large number of participants, the overwhelming majority has to consist of non-experts.

to study. The non-expert readers likely differ from experts in how they judge literary works; this can only be properly investigated by conducting a controlled study on the two groups. Apart from being non-experts, it should be noted that the survey respondents were predominantly highly educated (see below for some demographic figures), and by self-selecting for the survey, tend to be avid readers; any literary-sociological conclusions will therefore be most relevant to this cohort.

Turning to the general public in essence boils down to assuming that literature is a democratic concept, defined by consensus, as opposed to by specialists. Putnam (1975, p. 144) introduces the notion of a linguistic division of labor:

> We could hardly use such words as "elm" and "aluminum" if no one possessed a way of recognizing elm trees and aluminum metal; but not everyone to whom the distinction is important has to be able to make the distinction.

Putnam (1975, p. 146) goes on to hypothesize that:

> Every linguistic community possesses at least some terms whose associated "criteria" are known only to a subset of speakers who acquire the terms, and whose use by other speakers depends upon a structured cooperation between them and the speakers in the relevant subsets.

Is "literature" one of those terms? If it is, a survey of the general public would be a roundabout way of getting at the actual criteria—perhaps even fruitless, if the general public employs fundamentally different criteria or heuristics.

Some of the respondents specifically commented that they had no notion of how literary a novel is, and therefore could not give a rating; one respondent explicitly invokes the explanation that they expect society to judge the novel as literary. On the other hand, many comments do cite specific criteria such as writing style, plot structure, and character development.

As long as we clearly identify the questions that the survey can answer, it has empirical value. The survey outcomes should not be taken to reflect the literary value of the novels, or even an approximation of that. By analogy, the goal of a political poll is not to determine who is the best candidate, regardless of the definition of that term, but simply to estimate the opinions of a particular population.

Figure 5.1 shows the mean literary and quality ratings for the corpus, including the 95 % confidence intervals based on a t-distribution. The confidence interval indicates how accurate the estimation procedure is; note that it cannot predict how close the estimate is to the true value. A 95 % confidence interval gives a range of values such that if the experiment were to be replicated many times, at least 95 % of the time its result would be within this range; i.e., the smaller the confidence interval, the more reliable the result. The novels are
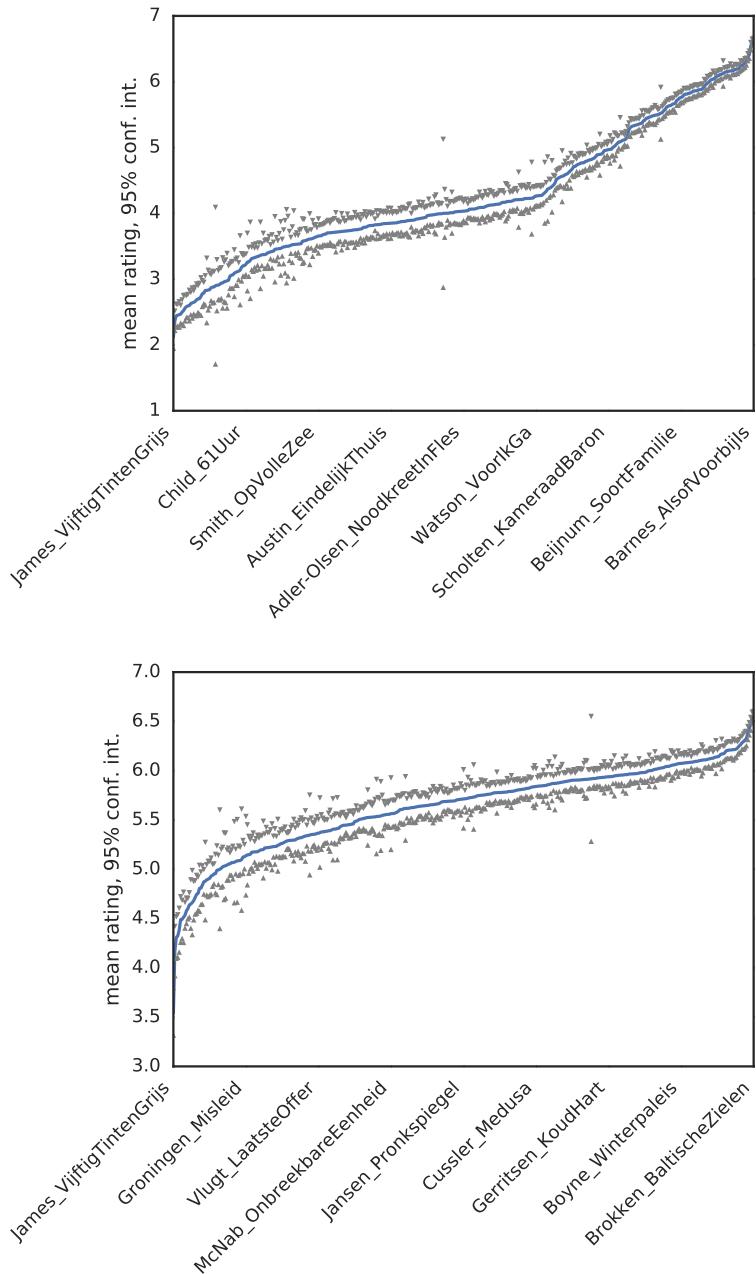
Figure 5.1: Mean literary (above) and quality (below) ratings per novel, with 95 % confidence intervals. Note the range of the *y*-axis for the quality ratings.
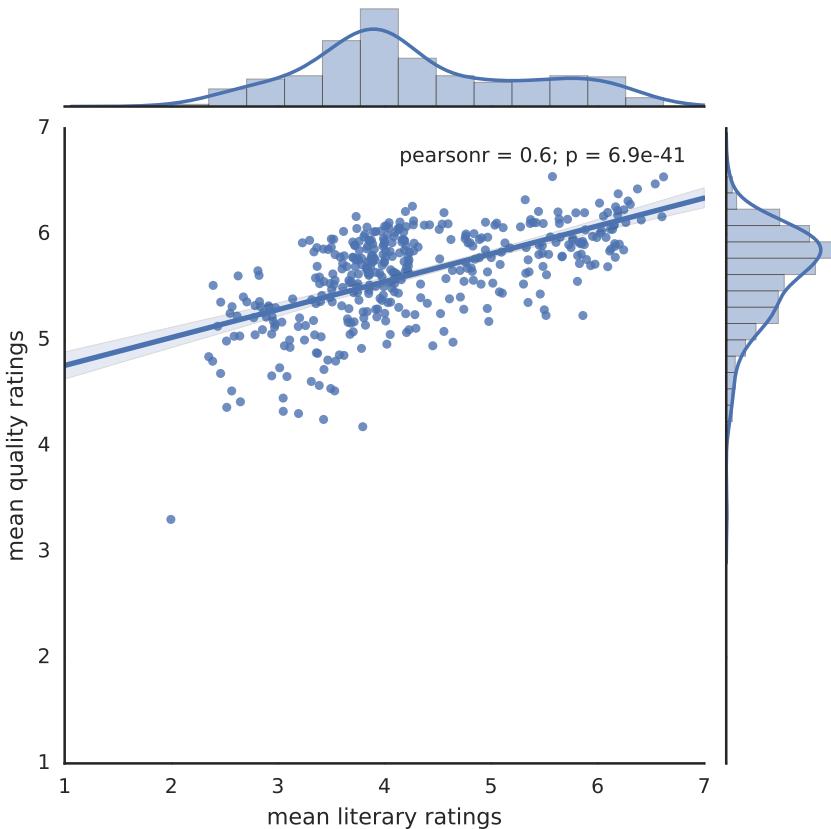
Figure 5.2: A scatter plot of the literary and quality ratings.

ordered by the ratings, with the most highly rated novels on the right. Consider the width of the intervals; i.e., given an interval $[a, b]$, the width is $b - a$. Overall, the width for the literary ratings ranges from 0.08 (816 ratings) to 2.3 (10 ratings). If only novels with at least 50 ratings are considered, the largest confidence interval is 0.9. Of the remaining novels, 91 % has a confidence interval width smaller than 0.5. In practical terms a width of 0.5 means that for a novel with a mean rating of 3, its interval ranges from 2.75 to 3.25.

Interestingly, the highly literary novels on the right of Figure 5.1 appear to have smaller confidence intervals, indicating that there is more consensus for these novels. This makes intuitive sense in that other novels are either borderline cases, or are simply not literary which makes it harder to decide on an appropriate rating. This effect is much less pronounced for the quality ratings. Note also that the quality ratings are concentrated on a smaller range.
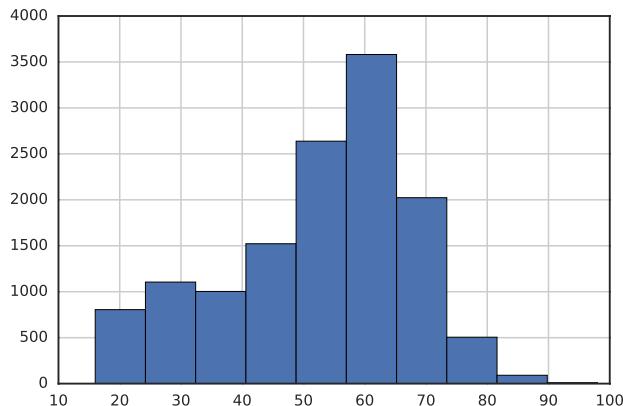
Figure 5.3: Histogram of the age distribution of the survey participants.

The scatter plot in Figure 5.2 shows that there is a significant correlation between the literary and quality ratings ($r = 0.6, p = 6.2 \times 10^{-41}$). The histograms on the sides also visualize the distributions of these two variables. Another salient (and significant) correlation is between the number of literary ratings for a book and its mean literary rating ($r = 0.61, p = 2.5 \times 10^{-42}$).

Let's consider some basic demographics of the survey participants. 71.8 % were women, 27.5 % were men, and 0.72 % gave no answer. The age distribution is shown in Figure 5.3; the age distribution is bimodal, with a large peak around 60, and a smaller peak around 25. The education level is shown in Table 5.2. The majority of participants is highly educated with a college or university diploma.

5.3 Literary stylistics and stylometry

This section reviews methods and previous work in particular applications of text analysis. The common denominator among most of the methods in this section is a focus on style.

There are many possible definitions of style (Berenike Herrmann et al., 2015). A narrow definition of style holds that it consists of largely unconscious choices by the author, on matters that contrast with content aspects. In this thesis we will use a broader (and vaguer) definition:

> Style is a property of texts constituted by an ensemble of formal features which can be observed quantitatively or qualitatively.
> — Berenike Herrmann et al. (2015, p. 44)

answer	translation	count
Geen onderwijs / basisonderwijs	No education / primary school	68
LBO / VBO / VMBO kader of beroeps / MBO 1	Vocational training	180
MAVO / ULO / MULO / VMBO (theoretisch/gemengd)	Vocational training (theoretical/mixed)	664
HAVO OF VWO / HBS / MMS	High school	1652
MBO 2, 3, 4 of MBO oude structuur	Community college	891
HBO	College	4995
wo / postdoctoraal onderwijs	Graduate school	4586
geen opgave	no answer	247

Table 5.2: Education level of survey participants, with approximate American equivalents.

5.3.1 Empirical work on literature

The concept of literary quality is rarely treated directly by literary scholars. An exception is the volume van Peer (2008) which collects various articles that address literary quality, discussing canon formation, stylistic aspects, and conceptual issues in literary evaluation. As for empirical work on literary quality and its possible textual factors, Richards (1929) is a well-known study on the interpretation of poetry; it concludes that the participants generally did not agree on their interpretation or assessment of the 13 poems, even praising obscure poets and finding fault with famous poets. However, Martindale and Dailey (1995) report opposite findings on the same poems by using a questionnaire and quantitative analysis instead of the essay approach of Richards (1929). Miall and Kuiken (1999) present a study in the formalist tradition, based on reader reports of short stories and poems. In addition to the traditional model of foregrounding and defamiliarization, they introduce a third process consisting of the "modification of [the reader's] personal meanings," i.e., the evocative and transformative qualities of the text. These three processes together are claimed to define literariness. The volume Miall (2006) presents an overview of empirical work on literature from the perspective of the reader.

There has also been empirical work that employs computational models. Martindale (1990) presents a sweeping theory of laws about art and its change over time, including results on literature. Chief among these laws is the desire for novelty and a less well-defined concept of "primordial content." Louwerse (2004) studies idiolect and sociolect within texts, across authors, and across genres by measuring semantic variation through Latent Semantic Analysis. The method and claims (chiefly, literature is a deviation from the norm) are interesting, but it is ultimately doubtful whether the conclusions are warranted by the

small amount of data (16 novels). Louwerse et al. (2008) present results with computational models that discriminate literary texts from non-literary texts using Latent Semantic Analysis and bigrams. The corpus is larger (119 novels), but the binary classification task they consider is made artificially easy because the non-literary texts are from obviously different genres such as newswire and sci-fi instead of less-literary novels; they attain a classification accuracy of 87 %. Jockers (2013) presents results with statistical and computational models, a so-called distant reading,[9] applied to a large number of novels, addressing questions on genre and change over time. Underwood (2015) and Underwood and Sellers (2015) present a chronological study of poetic distinction using a predictive model based on word frequencies (bag-of-words). The research question centers on how literary standards change over time, with predictive results on distinguishing poetry that was reviewed in prestigious journals from other poetry. Another development is the workshop Computational Linguistics for Literature, held since 2012 at ACL conferences (Elson et al., 2012, 2013; Feldman et al., 2014, 2015), which welcomes applications of NLP techniques to literary questions.

5.3.2 GENERAL QUALITY AND POPULARITY

Some previous work has specifically looked at identifying quality and predicting popularity of texts.

Bergsma et al. (2012) focus on scientific articles in the field of NLP. They consider the tasks of predicting whether an article was written by a native speaker, what the gender of the author is, and whether the article was published in a workshop or in the main conference. They also consider syntactic features, including tree fragments. However, in contrast to the work in this thesis, their tree fragments are from a predefined set of fragments extracted from a reference corpus. We extract fragments from the training data, which may result in more specific fragments.

Louis and Nenkova (2013) investigate science journalism and model what makes articles interesting and well written. The model is based on various readability, stylometry, and content measures, which are shown to complement each other.

Ashok et al. (2013) work with a corpus of nineteenth century novels from project Gutenberg and predict their download counts, as a proxy for the success of the novels. They try various kinds of features, include simple syntactic features such as POS tags and grammar productions, which achieve accuracy scores between 70 and 80 %.

While not related to text, the work of Salganik et al. (2006) is relevant because they present an interesting way of discovering the influence of social factors

---

9 Distant reading is a term due to Moretti 2005, which contrasts with close reading. Instead of interpreting and discussing select passages at length, distant reading addresses complementary questions using large-scale statistical analysis of digitized texts.

on popularity. They present an experimental study on the popularity of music using an artificial "music market" with the aim of investigating the relationship between quality and success of music, with or without social influence as a factor (showing users music liked by others). They found that social influence increases the unpredictability of popularity in relation to quality. A similar effect likely plays a role in the reader judgments of the survey.

5.3.3 Authorship attribution

One of the earliest applications of quantitative analysis of texts was to establish authorship. The attribution of the Federalist papers by Mosteller and Wallace (1964) is a seminal work. Mosteller and Wallace (1964) were incidentally also the first to introduce Bayesian methods in NLP. Earlier work on authorship attribution by Markov (1913, 1916, as cited in Khmelev and Tweedie 2001) introduced the notion of a Markov chain, a fundamental technique for language models in NLP and other fields modeling sequential processes, where the probability of an event is conditioned on the previous n events.

However, a critique that can be leveled at authorship attribution studies is that a common, rigorous scientific methodology is typically not applied (Chaski, 1997, 2001). In computational linguistics it is normal to benchmark new work on common data sets and to participate in shared tasks. Authorship attribution work is often evaluated on data sets that were collected by the author and not made available. Moreover, there is typically a small, closed set of potential authors or distractors, making the task artificially easy. This makes it difficult to compare different methods and feature sets against each other, because each data set has different characteristics. Fortunately, the recent development of shared tasks in the PAN workshops[10] appears to address these concerns.

5.3.4 Readability

Readability measures estimate how difficult a text is for certain kinds of readers. A common application is the selection of material suitable for children in particular grades. Early methods (e.g., Flesch reading ease, Flesch, 1948) were designed to be computed by hand and in effect consist of a linear regression on variables such as sentence and word lengths. More recent methods compare texts to reference corpora (Collins-Thompson and Callan, 2004; Schwarm and Ostendorf, 2005) and consider more fine-grained linguistic variables (Graesser et al., 2004); the latter has inspired a Dutch version (Pander Maat et al., 2014).

Readability has relevance for literary stylistics because highly literary novels may be more complex and thus less readable than other texts, while conversely genre novels might be more readable, in order to reach a larger audience, or perhaps as a necessary condition for being a page turner.

---

10 Cf. http://pan.webis.de/

5.4 Dutch syntax and its annotation

Since the corpus of novels we will study is Dutch (whether translated or original), and we are particularly interested in syntactic aspects, this section provides some background on the Dutch language. This section is not meant to be comprehensive, but only to give some context for interpreting the syntactic examples and findings that will be presented later.

Dutch is a Germanic language. One interesting property is that it has two word orders: verb second in main declarative clauses and verb final in embedded clauses. The latter is considered to be the primary order, and therefore Dutch is argued to be a subject-object-verb (sov) language (Koster, 1975), despite the fact that main clauses are typically subject-verb-object (svo) like English. The verb-second (V2) property means that in a main declarative clause, the finite verb must appear in second position; this entails that the order is more strict than svo. English is not a V2 language, which makes it an exception among Germanic languages. Consider the following examples:

(1)   a.   Hij *zag* gisteren   deze fiets
           He saw yesterday this bicycle
      b.   Deze fiets   *zag* hij gisteren
           This bicycle saw he yesterday
      c.   Gisteren *zag* hij deze fiets
           Yesterday saw he this bicycle
      d.   *Deze fiets   hij *zag* gisteren.
           This bicycle he saw yesterday.
      e.   *Gisteren hij *zag* deze fiets.
           Yesterday he saw this bicycle.

The finite verb is emphasized. We see that various constituents can be placed in the initial position, as long as the finite verb ends up in the second position (a–c); when two elements precede the verb (d–e), the Dutch sentence becomes ungrammatical, which is not the case for English.

Another property is that Dutch allows long verb clusters, for example in cross-serial dependencies described in Section 1.4. Any non-finite verbs appear in clause-final position, in embedded clauses this includes the finite verb as well. The latter gives rise to two possible orders for embedded clauses in Dutch (not allowed in German or English):

(2)   a.   . . . omdat   ik heb gewerkt
           . . . because I   have worked
           'red order': auxiliary, past participle.

      b.   . . . omdat   ik gewerkt heb
           *. . . because I   worked have
           'green order': past participle, auxiliary.

Interestingly, there is regional variation in the preference for these orders, and additionally, an association with speech versus writing.

In sum, in terms of word-order freedom Dutch and German are somewhere between English (rigid) and Warlpiri (exceptionally free, non-configurational; Hale 1983). Similarly, Dutch has a richer derivational morphology than English, allowing the formation of new words using compounding, which form a single word written without spaces.[11]

The Dutch treebanks follow the same annotation style as the German treebanks, starting with Negra (Skut et al., 1997). This means that discontinuous constituents are possible, and nodes have a function tag in addition to a syntactic category. Terminals have morphological features in addition to a POS category. For the Dutch annotations, we can distinguish two kinds of discontinuous constituents:

1. inherent: similar to other languages, phenomena such as extraposition and relative clauses give rise to inherently non-local relations.
2. incidental: particular features of Dutch and the chosen style of annotation give rise discontinuity that arguably does not raise the complexity of the sentence the way inherent discontinuity does.

The following are examples of incidental discontinuity, with the discontinuous parts emphasized.

The word 'er': 'er' is a versatile function word. It can be used as both pronoun and adverb. When used as part of a separable preposition, it often gives rise to discontinuity.

(3)   Ik geloof *er* niet meer *in*.
        I   believe *it* not longer *in*.
        'I don't believe *in it* anymore.'

The word 'te' (to): When 'te' is used to introduce an infinitival phrase, 'te' causes a discontinuity if the infinitival phrase has any arguments or modifiers that precede 'te.'

(4)   Je   hoeft niet *zo hard*   te *praten*.
        You have not *so loudly* to *talk*
        'You don't have to *talk so loudly*.'

Participle phrases: when a clause has an auxiliary verb and a participle, the participle is put in a separate participle phrase, including its modifiers and

---

11 Interestingly, there is a tendency to add spaces to compounds where Dutch morphology would prescribe none; since this is perceived to be due to the influence of English, this has been called *de Engelse ziekte* (literally the English disease, a term normally denoting rickets). A particular instance that generated some mild controversy was the new logo of the *Rijksmuseum* (national museum), which was rendered with a half space between *Rijks* and *museum* (Sanders, 2013), presumably as a compromise.

arguments. Due to the verb-final nature of embedded clauses, this will cause the auxiliary verb to interrupt the participle verb and its arguments/modifiers.

(5) *Vandaag* heb ik hard *gewerkt*.
     Today   have I hard *worked*.
     '*Today* I have *worked* hard.'

Verb clusters: similar to the cases above, whenever there are two or more verbs, each verb may have arguments or modifiers in a non-adjacent position.

(6) Ik ging weg omdat hij mij anders   *gezien* zou   *hebben*.
     I went away because he me otherwise *seen*   would *have*.
     'I left because otherwise he would *have seen* me.'

5.4.1 Syntactic parsing of Dutch novels

Although we have developed a statistical parser for Dutch in the preceding chapters, we will use the Alpino parser (Bouma et al., 2001) to parse our data set of novels. The motivation for this is that Alpino, being a handwritten broad-coverage parser, will provide better parses than our statistical parser, which has not been trained on novels but on texts of a specific domain. However, we will continue to make use of tree fragments and the extraction method for them.

In order to estimate the quality of the parse trees produced by Alpino for the particular task of parsing novels, we took a random sample of 100 sentences from the corpus of 401 novels, and manually evaluated and corrected each constituent. The evaluation is given in Table 5.3. The results are very good, indicating that there is no particular cause for concern with regards to parse quality. The errors we found relate to PP-attachments (which is arguably a semantic rather than syntactic task) and the clause structure of long sentences.

For comparison, the sentences have also been parsed with the Disco-2DOP model presented in this thesis. As could be expected, the performance is much poorer (an F_1-score of 72.8). There are two reasons why this is not a representative comparison. First, this model is derived solely from the training data, which is very different from the words and constructions in the novels, while Alpino has been hand-tuned for decades on language from various sources. Second, it should be noted that this comparison inherently favors Alpino, since the hand-corrected reference parses were generated by Alpino.

number of sentences:	100
longest sentence:	41
gold brackets (disc.):	669 (35)
cand. brackets (disc.):	662 (32)
labeled recall:	91.93
labeled precision:	92.90
labeled F-measure:	92.41
exact match:	88.00
function tags:	98.26
POS accuracy:	99.41
coverage:	100.00

Table 5.3: Evaluation results for manually evaluated parse trees from novels.

6 Modeling Literary Language

In which we establish a baseline for modeling literary ratings using simple textual features, cliché expressions, and topic modeling.

> Je l'ai toujours dit et senti,
> la véritable jouissance ne se décrit point.
> *I have always said and felt*
> *true enjoyment cannot be described.*
>   — Jean-Jacques Rousseau (1889, p. 78), *Les Confessions*

> One equation can't possibly capture the rich complexity of six hundred fifty years of British poetry. Five or six are probably needed.
>   — Martindale (1990, p. 117), *The clockwork muse: The predictability of artistic change*

As a preamble to a full predictive model, this chapter explores several explanatory variables of literariness and language. Our ambitions are more modest than those of Martindale (1990), since we will be content with correlations that partially explain the data.

6.1 Modeling literariness with basic textual features

Before looking at more complex features and models we first construct a simpler model to gauge how difficult the task of predicting literary ratings from texts is.

We consider a chunk of 1000 sentences of each novel, to control for length effects. To avoid the particularities of the start of the novels, we look at sentences 1000–2000. We leave out 18 novels with less than 2000 sentences.

An often heard writing advice is to minimize the number of adverbs and adjectives. Since they are grammatically optional, they may be viewed as unnecessary; leaving the reader to intuit the specifics of the situation may be preferable. Liberman (2015a) responds to an instance of such writing advice

	Literary	Quality
% adverbs	-0.18*	-0.24*
% adjectives	-0.02	-0.17*
% verbs	-0.35*	-0.09
% nouns	0.29*	0.20*

Table 6.1: Correlation coefficients for POS tags with survey ratings. * indicates a significant result with $p < 0.001$.

	% Adj	%Adv	%Noun	%Verb
Liberman (2015a), English texts:				
'8 novels by admired authors'	6.46	7.54	20.34	20.19
Brown, Da Vinci code	6.9	5.7	28.3	19.8
bad writing samples	10.47	5.39	28.40	14.40
Riddle corpus, Dutch:				
383 novels	3.98	5.30	17.90	20.50
- Fiction	4.02	5.16	18.58	19.92
- Suspense	3.92	5.14	17.82	20.90
- Romantic	4.13	5.98	16.56	21.33

Table 6.2: Comparison of results on mean POS tag proportions.

by measuring their occurrences in various samples of writing, demonstrating that good writing does not contain less adverbs and adjectives, *vis à vis* bad writing (the texts were samples of literature and purple prose, respectively, and included non-fiction as well).[1] We can repeat this experiment on our corpus, with the added benefit of being able to directly correlate the results with survey ratings. It should be noted that the survey ratings and corpus we use provide a different contrast from the good and bad writing that Liberman (2015a) uses, since all of the novels in our corpus were highly successful and professionally edited. Furthermore, our corpus is larger, with 383 instead of 18 texts.

Table 6.1 lists the correlations. Surprisingly, we do find correlations, and the results are the opposite of Liberman (2015a), to the extent that the experiments are comparable. See Table 6.2 for a comparison of POS tag proportions with the results of Liberman (2015a). Some of the difference is due to different annotation practices for English and Dutch POS tags, but it stands out that the Dutch POS tags are much more consistent across genres, as compared to the two samples that Liberman (2015a) reports. But it is not obvious that this can be attributed to the good-bad contrast one way or the other. The small sample size and large difference in genre and domain in the data set of Liberman (2015a) could well be drowning out any markers of good and bad writing.

---

1 I am grateful to Stella Frank for bringing this blog post to my attention.

Except for adjectives versus literary ratings, and verbs versus quality, the correlations we find are significant, and point in the direction predicted by the writing advice of avoiding adverbs and adjectives. However, the correlations are relatively weak. Incidentally, the phrasal equivalents of adverbs and adjectives, AP and ADVP respectively, yield weak, non-significant correlations.

Nouns and verbs do have relatively strong correlations. However, this may be explained to an extent by the mean sentence length, which will be reported shortly. Namely, a regular sentence requires a main verb, so when the sentences are shorter, a larger proportion of words consists of main verbs; conversely, longer sentences leave more room for nouns and other parts of speech.

For now we conclude that the notion that adverbs and adjectives are a predictor of non-literary or bad writing cannot be rejected. Obviously, any piece of good writing can be ruined by a generous helping of gratuitous adverbs and adjectives, but the strength of the correlation suggests that the majority of adverbs and adjectives do not detract from the quality of writing. The advice of avoiding them needs qualification; which adjectives or adverbs should be avoided in what situation and context? The answer will almost surely not fit in a writing advice slogan.

There are many more features beyond POS tags that could be considered; e.g., Pander Maat et al. (2014) has a rather comprehensive list, used in the context of readability assessment of Dutch; inspired by a similar system for English (Graesser et al., 2004). Somewhat arbitrarily, we will consider the following set of easily-extracted features.

WORDS PER SENTENCE i.e., mean sentence length. Despite the name, we count all tokens as words.

COMMON VOCABULARY the percentage of tokens that are part of the 3000 most common words in a large reference corpus. We use Sonar 500 (Oostdijk et al., 2013), a 500 million word corpus composed of various domains. The use of an external corpus should give a more reliable indicator of complex vocabulary than proxies such as word length or type-token ratio which are not defined relative to general language use.

DIRECT SPEECH the percentage of sentences with direct speech punctuation.

MODIFYING PPs the percentage of prepositional phrases with the MOD function. This label indicates an adverbial modifier, which are grammatically optional elements (in the same way as adverbs and adjectives).

AVERAGE DEPENDENCY LENGTH the distance (in words) between head words and their dependents (arguments or modifiers). Longer dependencies are claimed to be more difficult to process (Gibson, 2000). We follow the definition of Liu (2008): punctuation is ignored, as well as dependencies on the root node; the distance of a dependency between words with index a and b is $|a - b|$; the total of all dependency lengths in a text is summed before computing the mean.

COMPRESSION RATIO the number of bytes when the text is compressed divided by the uncompressed size. This expresses, in a technical sense, how repetitive

	Literature	Quality
Words per sentence	0.40*	0.25*
Common vocabulary	-0.31*	-0.17*
Modifying PPs	0.45*	0.23*
Direct speech	-0.38*	-0.03
Avg. dependency length	0.38*	0.24*
Compression ratio	0.32*	0.05

Table 6.3: Correlation coefficients for basic textual features and survey ratings. * indicates a significant result with $p < 0.001$.

a text is. We use the bzip2 compression scheme; this performed best (lowest compression ratios) compared to the two other evaluated algorithms, gzip and lzma (all applied with highest compression settings). For this feature we use the first 100k bytes of each text, instead of 1000 sentences, since this feature works not on the level of sentences but on bytes.

Most of the traditional readability measures (e.g., Flesch reading ease) are based on sentence length and a measure of word length or complexity; we therefore include such variables (the latter in the form of common vocabulary). Note that the above features are not independent of each other. Longer sentences will tend to contain more modifier constituents. While more direct speech tends to lower the mean sentence length because dialogue often contains shorter sentences. When the results are correlated with the survey outcomes, the results in Table 6.3 obtain; i.e., literary texts have longer sentences, less common words, are less repetitive, have more optional PPs, and more direct speech.

The modifying PPs illustrate an instance of the Simpson's paradox: if all instances of the modifier function tag are counted, a positive correlation obtains, but as shown before, adverbs, which are also modifiers, show a negative correlation. Further investigation of the MOD tag confirms this. While PP and REL (relative clause) have a significant positive correlation, the other phrasal and POS tags with which MOD occurs, AP, ADVP, CP and adjectives, show a very slight or almost no correlation with literary ratings.

When these features are normalized and combined into an Ordinary Least Squares model of the literary ratings with 5-fold cross-validation,[2] we obtain the prediction performance in Table 6.4. Each line shows the effect of adding another feature to the model. Note that the negative R^2 in the first line means that this feature by itself performs worse than the baseline of predicting the mean rating. The results are not very impressive. The rest of this thesis will show how much improvement can be obtained from more refined textual features.

---

2 A regression analysis as used in applied statistics is commonly evaluated using within-sample metrics, which leads to substantially higher scores. However, because of the focus on predictive modeling in this thesis, we strictly report out-of-sample results.

	RMS error	R^2	Kendall tau
% adjectives, adverbs	1.001	-0.5	0.098
+ % nouns	0.971	5.4	0.160
+ % verbs	0.912	16.5	0.264
+ words per sentence	0.886	21.2	0.290
+ Common vocabulary	0.886	21.2	0.287
+ Modifying PPs	0.858	26.1	0.333
+ Direct speech	0.844	28.6	0.339
+ Avg. dependency length	0.838	29.6	0.345
+ Compression ratio	0.835	30.0	0.348

Table 6.4: Linear regression with 5-fold cross-validation on the baseline features.

6.2 Cliché expressions in literary and genre novels

An often heard argument among literary critics is that the cliché is one of the most prominent characteristics of "low brow" literature. Clichés can concern the story line, the characters or the style of writing. We focus on clichéd expressions, stock phrases which form a type of multi-word expressions. We present a corpus study in which we examine to what extent clichéd expressions can be attested in a corpus of various kinds of fiction.

We define cliché expressions as follows:

Definition 13. A *cliché expression* is a fixed, conventionalized yet compositional multi-word expression which has become overused to the point of losing its original meaning or effect.

Let's unpack the four important terms in this definition:

fixed: the expression in the form that is recognized cannot be changed, or only to a limited degree by filling in specified open slots.

conventionalized: i.e., the phrase is recognized by many speakers as a unit, instead of being put together word for word.

compositional: the meaning is the sum of its parts following the regular process in the language; this is in contrast to syntactically anomalous phrases such as "by and large", or semantically non-compositional figurative expressions such as "kick the bucket".

overused: this aspect is subjective and therefore harder to pin down. Many other multi-word expressions are accepted as a normal part of the lexicon, while cliché expressions are marked as informal, tired, unoriginal, etc. Furthermore, "A cliché is a kind of ersatz novelty or creativity that is, *ipso facto*, unwelcome or deprecated by the reader" (Cook and Hirst, 2013, emphasis in original). The term overused might suggest that there is some range of acceptable frequency for items, but this limit seems hard to determine; the cliché-hood of an expression rests on a contingent, cultural judgment.

The combination of conventionalized and compositional is interesting. While these expressions are semantically regular, we know that the speaker did not choose any part of the expression separately, but only the expression as a whole.

We use a data set of 6,641 Dutch cliché expressions provided to us.[3] We determine the frequencies of the clichés in the corpus of 401 novels and relate them to the survey results. The aim is to see whether the prevalence of clichéd expressions offers insights into literary evaluations.

As reference corpora we will also look at Lassy Small and cgn. Lassy Small consists mainly of news paper and Wikipedia text. cgn is a corpus of spoken language.

---

3 We are grateful to Wouter van Wingerden and Pepijn Hendriks for providing us with their data set of clichés. This data set is the source for their published collection of clichés (van Wingerden and Hendriks, 2015).

This work can be compared with Cook and Hirst (2013), who also measure to what extent texts are cliched. However, they use a method based on n-gram frequencies from a reference corpus, which does not require an explicit list of cliché expressions. This means that their method can only estimate the amount of cliché expressions, but not inspect the expressions themselves or analyze the number of types and counts for each expression.

6.2.1 Preprocessing

We tokenize both the cliches and the novels. We convert the cliches into regular expressions. A handful are removed because they are too generic and generate too many matches. The cliches contain several kinds of notation which we translate using regular expression operators:

(1) a. Optional elements:
         (Kijk,) dat bedoel ik nou.
         (Look,) that's what I mean.
     b. Open slots:
         Geen [bier] meer voor jou!
         No more [beer] for you!
     c. Variables:
         Y, zoals X dan zou zeggen.
         Y, as X would say.
     d. Alternatives:
         Daar zit een boek/artikel in!
         That's material for a book/paper!

Some aspects cannot be translated precisely. When the alternatives span multiple words, the scope was not specified, so these have been edited manually. For lack of more specific criteria, we allow any phrase of 1 to 3 words in open slots. Lastly, some clichés involve mini-dialogues; since we match on a per-sentence basis in the novels, these clichés will never be found.

After translation we remove duplicates. The resulting list of 5,771 patterns are counted across the whole corpus. We use Google's RE2 library to match the patterns efficiently using Deterministic Finite-State Automata. The result is a document-pattern matrix with occurrence counts.

Some examples with high frequency:

Dat dacht ik al \.	125
Is dat alles \? \S+	124
Nergens voor nodig \.	114
Waar of niet \?	112
Wat kan ik voor je doen \?	102
Waarom denk je dat \?	101
Komt allemaal goed \.	84
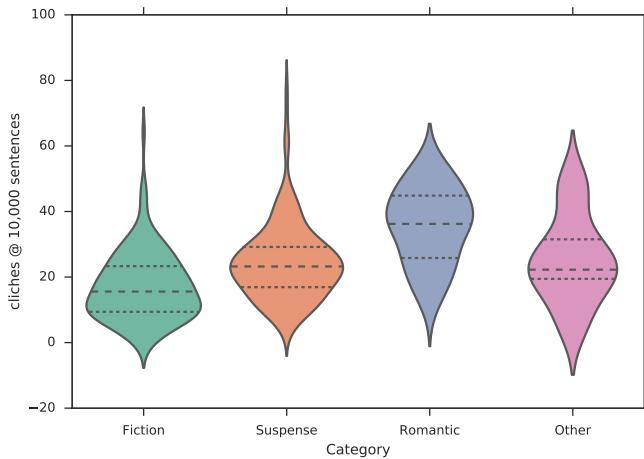
Figure 6.1: A violin plot of the number of clichés by genre.

6.2.2 Results

For purposes of analysis, we sum the counts for all clichés in each document, and compute the correlation with the target value. See Figure 6.2 and 6.3 for the results. For both the literary ratings and quality there is a significant correlation, which is stronger than the correlations obtained with the baseline features of the previous section. The plots show that most novels with a high number of clichés are non-literary; the highly literary novel *De Buurman* (the neighbor) by Voskuil is the strongest exception to this. On the other hand, novels with few clichés may or may not be literary. In other words, clichés are most useful as a negative marker of literariness. For example, *50 shades of Grey*, the least literary novel, has relatively few cliché expressions for novels with a similar rating, and falls below the regression line. Two novels do not have any matches at all.

To compare the rate of clichés across genres and domains, Table 6.5 shows an overview aggregated across the main genres in the corpus and two reference corpora. Especially the Romantic genre contains a large number of clichés, and judging by the type-token ratio, it contains the most repetition of cliché expressions (a ratio of 1 indicates that each type of cliché expression occurs only once; the lower the ratio, the more repetition). The violin plot in Figure 6.1 illustrates the genre differences.

The reference corpora contain a much lower rate of clichés, which is probably attributable to their domain and either lack of informal dialogue (Lassy Small), or transcription of disfluencies preventing matches (CGN).
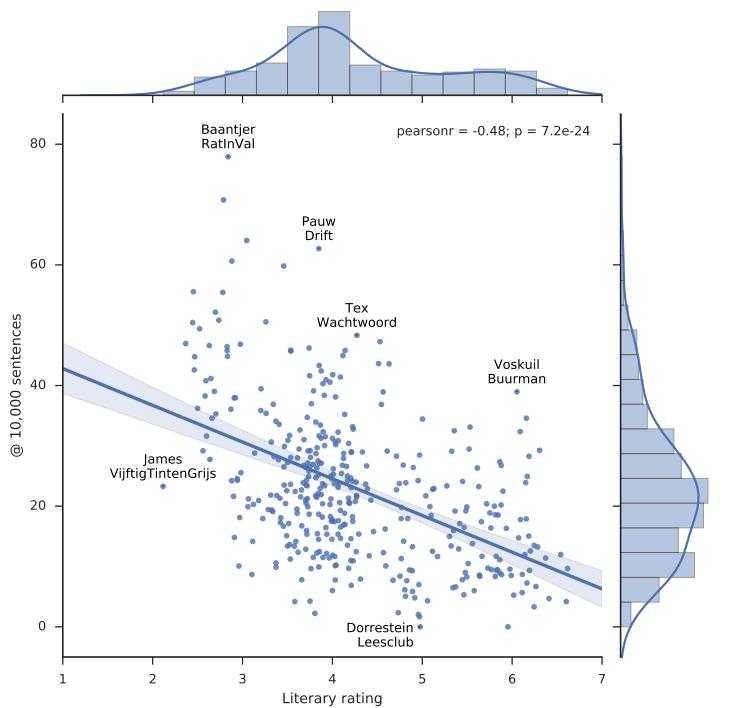
Figure 6.2: Correlation of the number of clichés with literary ratings.

	texts	sentences	matches @ 10,000 sentences	@ 10,000 sentences, freq > 1	type-token ratio
Novels	369	9882.41	23.22	5.06	0.9
- Fiction	137	8611.09	18.11	3.01	0.94
- Suspense	177	10519.9	23.35	5.39	0.88
- Other	17	10733.2	23.84	6.36	0.89
- Romantic	38	11116	36.72	8.92	0.87
Lassy Small	1	52157	0.38	0	1
CGN	1	70277	9.25	4.41	0.63

Table 6.5: Overview of cliché occurrences. The rows with novels show the mean.
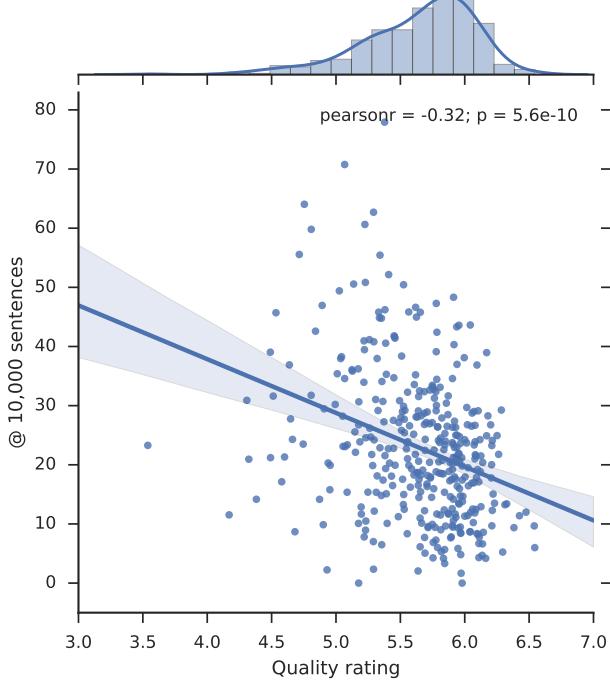
Figure 6.3: Correlation of the number of clichés with general quality ratings.

6.3 Literariness and genre in a topic model

Topic models are an unsupervised technique for text analysis. A topic model aims to automatically discover topics in a collection of documents. We apply Latent Dirichlet Allocation to our corpus of novels to investigate several literary questions.

Instead of examining topics on a macro-scale in a geographical or historical interpretation (Jockers and Mimno, 2013; Riddell, 2014), we take a new perspective: whether novels have a dominant topic in their topic distributions (mono-topicality), and whether certain topics may express an explicit or implicit genre in the corpus. We hypothesize that there is a relationship between these aspects of the topic distributions and perceptions of literary quality. We then interpret the model by taking a closer look at the topics in a selection of the novels.

6.3.1 Latent Dirichlet allocation

Latent Dirichlet Allocation (LDA, Blei et al., 2003) is a topic model that automatically induces topics from a corpus of documents. LDA is based on the BoW-assumption. LDA assumes that documents were generated from a mixture of a small number of topics, where a "topic" is a distribution over words;[4] the 'latent' in the name refers to revealing this hidden structure. 'Dirichlet' refers to a distribution that generates other distributions. Given a number of topics, LDA produces two kinds of distributions for the data: word distributions for each topic, and topic distributions for each document. These two distributions interact because LDA prefers to associate each document with few topics, while at the same time allocating the majority of weight in each topic to a small number of words. The result is that frequently co-occurring words tend to receive a high probability in one particular topic, while similar documents will have a high probability for topics describing their similarities.

6.3.2 Experimental setup

We preprocess the 401 novels of the Riddle of Literary Quality by lemmatizing[5] the words, removing punctuation, function words and names, and splitting the remaining text in chunks of 1000 tokens. We use Mallet (McCallum, 2002) to create a topic model with 50 topics. Figure 6.4 shows an overview of the topics with their proportion across the corpus.

We have attempted to identify topics for novels with high literary ratings, and topics specific for suspense and romantic novels. According to Jockers and Mimno (2013), the topics can be used to identify literary themes. They use the terms "theme" and "topic" as "proxies for [...] a type of literary content that is semantically unified and recurs with some degree of frequency or regularity

---

4 Note that this notion of topic is unrelated to more usual notions such as discourse or narrative topic.
5 We use the lemmas that are part of the output of the Alpino parser.
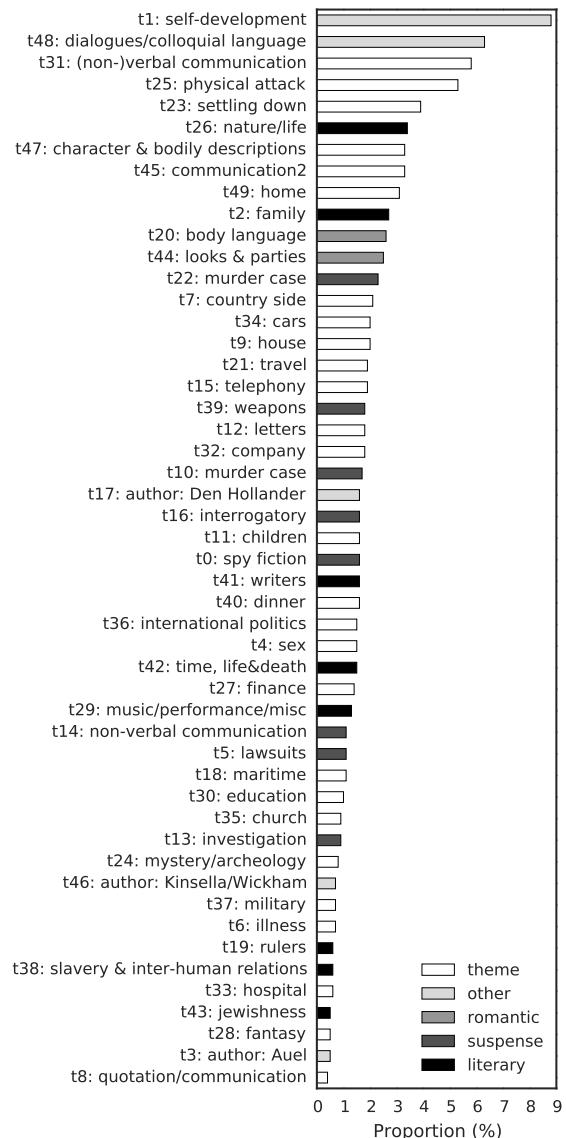
Figure 6.4: Overview of topics, sorted by proportion of the corpus.

throughout and across a corpus" (p. 751). We found that three topics are specific to a single author (for instance t3), and about a third seem genre specific. By inspecting the most important words for each topic we found that most topics (genre related or not) indeed cohere with certain themes (cf. Figure 6.4). This suggests that the choice for 50 topics is neither too small nor too high.

6.3.3 Quantitative analysis

We aim to gain insight into the distribution of topics in relation to the literary ratings of the novels (predicting literary ratings is not the goal for now). In order to interpret the topic distributions, we introduce the concept of mono-topicality.[6] A mono-topical novel contains little diversity in topic distribution, which means that one or two topics are dominant throughout the novel. A novel which shows more variation in topics has a more even distribution of topics, i.e., such a novel has a larger topic diversity. Figure 6.5 shows an example of both cases.

The x-axis shows the distribution of topics, sorted from least to most prevalent. In John Grisham's The Appeal, topic 5 ("lawsuits") has a proportion of 47.8 % of all 50 topics. This novel is more mono-topical than the Franzen's Corrections, which has a more balanced distribution of topic proportions.

We hypothesize that the less mono-topical a novel is, the higher the literary ratings by readers will be. And indeed, Figure 6.6 shows that there is a statistically significant correlation between the diversity of topics of a book and its literary ratings. Books with a single, highly prominent topic, such as Grisham's, tend to be seen as less literary.

6.3.4 Interpretation

There are several possible explanations for the correlation. Genre novels could have a tendency to single out certain topics, as they deal with more 'typical' or genre-specific subject matter than do general novels. If this were the case, we would simply be finding that genre novels are considered to be less literary than general novels, and this would tell us little about literary quality in a more general sense. General novels, on the other hand, deal with all sorts of subjects and themes, across and beyond 'genre' subjects, and therefore a topic model may not be able to single out thematic groups of words common to these novels, and thus may not find one single prominent topic. A third explanation could be that highly literary novels do deal with specific themes and subjects which are also part of genre novels, but that these are described in wordings that are more implicit or subtle, and therefore do not come up as single, clear topics. If this were the case, that would bring us closer to an explanation of what topics have to do with literary quality. These explanations are not mutually exclusive

---

6 Algee-Hewitt et al. (2015) independently introduced a similar concept under the same name.
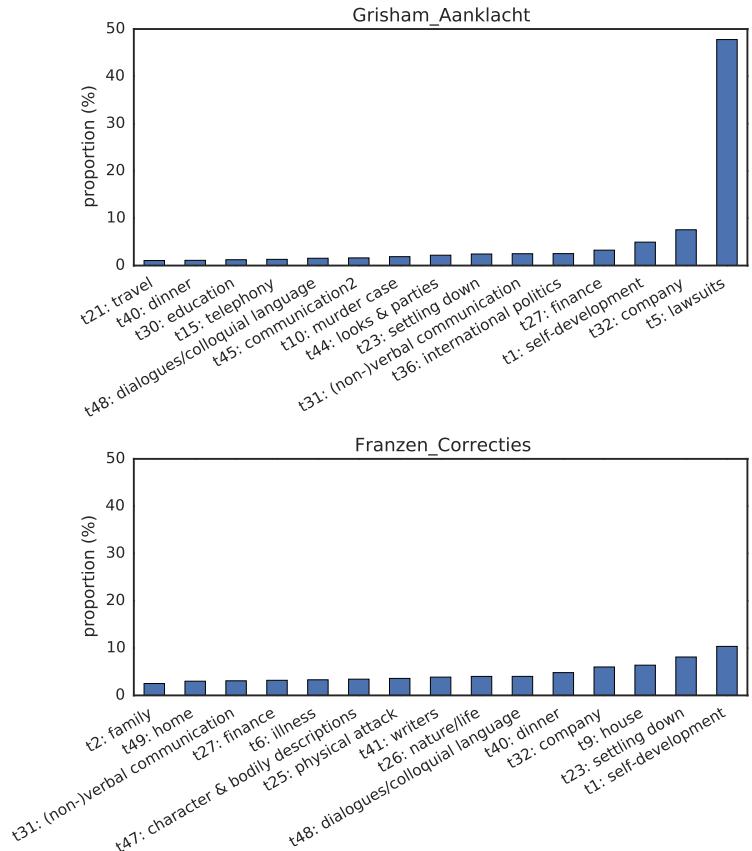
Figure 6.5: Distribution of 15 prominent topics in two novels with high (top) and low (bottom) mono-topicality.
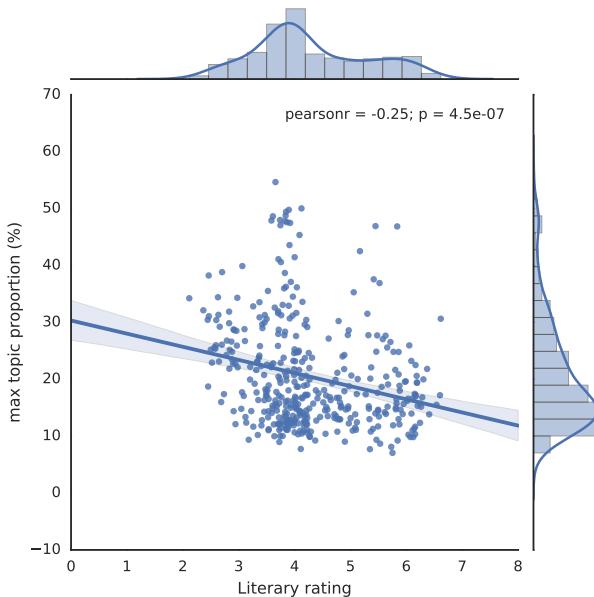
Figure 6.6: Correlation between share of the most prominent topic per book and mean literary ratings.

and we will explore the topic model here to examine the value of the second and third explanation.

The topic that shows the highest correlation ($r = 0.49$) with literary appreciation is topic 29; cf. Figure 6.7. This topic is most prominent in fifteen originally Dutch general novels. The twenty words in topic 29 with the highest weights are begin, music, play, occasion, first, the first, sing, only, year, one, stay, sometimes, even, new, own, always, high, exact(ly), bike, appear. They show little coherence, making it hard to interpret their context, although 'music and performance' appears to be part of it. To find out more about the novels in which this topic is prominent, we consult a Dutch website maintained by librarians called *Literatuurplein*, which provides information on the themes and content of Dutch novels.

Most of these novels show similarities in themes, such as family relationships. In ten of the novels the protagonist has an artistic profession: a couple of writers, a painter and a stand-up comedian. None of them has a musical or acting career, despite the 'music and performance' words; and *vice versa*, none of the twenty most prominent words concern writing. All in all, at first glance topic 29 seems not to address the themes and content of the novels, whereas most other topics in the model do concern specific themes (cf. Figure 6.4).

For instance, topic 2 and 11 address family relations, topic 12 and 41 are
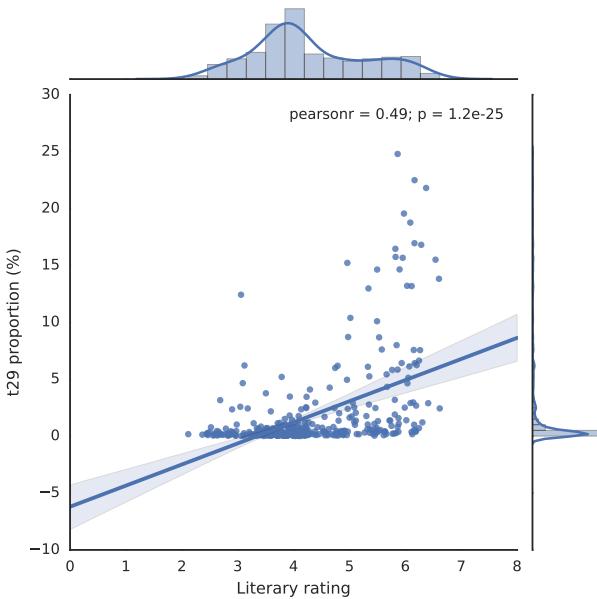
Figure 6.7: Correlation between topic 29 proportion and mean literary ratings.

about writing novels, and topic 6 and 33 concern health issues. These topics are present, but as smaller topics. This shows that the second explanation, of the general novels not sharing themes, is not valid. It could be an indication though that the highly literary novels indeed use a more subtle way of describing themes similar to other novels in our corpus, our third explanation. As a final note, in topic 29 there are proportionally more adverbs than in the other topics mentioned, which contain more nouns. Perhaps this shows that style is a more shared element in literary novels than the choice of words. In other words, this brief analysis shows that there is merit to our third explanation. This will therefore become a new hypothesis for further research.

6.3.5 Topic weights over chunks: Plot lines

Since the topic model was constructed over 1000 word chunks, it is possible to inspect the prominence of topics across a novel. Especially topics related to events are suited to this, such as murders, police investigations, and telephone calls.

For a given topic, we plot its weights with text time (as opposed to story time) on the x-axis. This axis could be normalized so that all novels start at the same point and end at the same point, or calibrated to the amount of time that elapses in each story. However, we choose to leave the time dimension as is, such that the x-axis represents the same amount of text for each novel. The y-axis
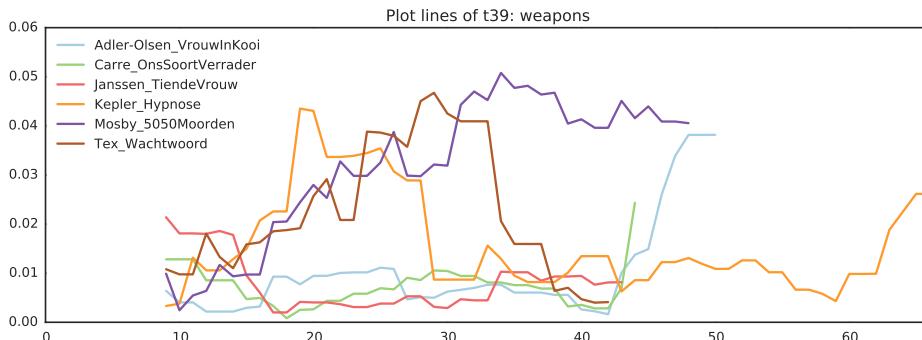
Figure 6.8: Plot line of topic 39 (weapons) in 6 novels.

	RMS error	R^2
Baseline features (OLS)	0.84	30.0
Topics (SVM)	0.69	50.27

Table 6.6: Regression results with topic model weights.

shows the prominence of the topic in a specific chunk in each novel, relative to the other possible topics. We apply a rolling mean of 10 chunks to smooth the plot. Figure 6.8 show such a plot, for the topic on weapons in 6 thrillers.

6.3.6   TOPICS AS FEATURES FOR PREDICTION

In the next chapter we will use bigrams and tree fragments as features to predict the ratings of novels in the survey. These models rely on thousands of features. As a preview, here we report an experiment on a predictive model using just topic weights. The topic model is, like the bigrams, also based on a bag-of-words model, but reduces the word-dimensions to topic-dimensions. We can represent each document as a feature vector with its 50 topic weights.

On the one hand this a dramatic reduction in information in terms of the number of variables. On the other, the topics capture relationships between words that cannot be capture by a linear bag-of-words model. As a concrete example, in a linear BoW-model, synonyms have independently trained feature weights, while they can share a topic in a topic model.

A methodological remark that should be made is that the topic model is based on all text, including the texts that are used as test set in each of the 5 cross-validation folds. However, the target values (literary ratings) are not seen during training, therefore this is still an out-of-sample test with respect to the values that are being predicted. Table 6.6 and Figure 6.9 show the results of the regression experiment. In this experiment we used the same chunks as for the topic model. Each novel is therefore predicted and evaluated multiple times.

r2_score = 0.5
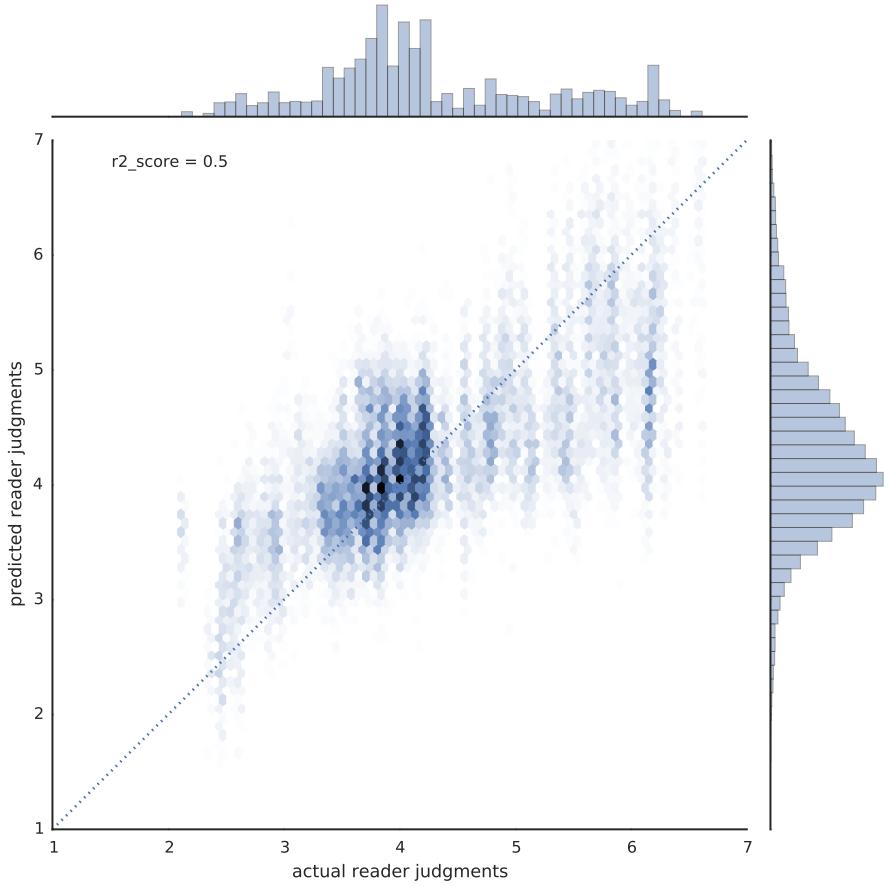
Figure 6.9: Regression results with topics as features

6.3.7 Summary

We have explored a topic model of contemporary novels in relation to genre and literariness, and shown that topic diversity correlates with literary ratings. Most topics express a clear theme or genre. However, topic 29, the most literary topic, does not. It rather appears to be associated with a particular Dutch literary writing style.

7 *Predictive Models of Literature*

In which we model authorship and what makes texts literary by exploiting an ensemble of lexical and syntactic patterns in a predictive model.

T HIS CHAPTER is about learning to predict aspects of literature, authorship and literariness. We explore both simple textual features, word n-grams, as well as richer, syntactic features, namely tree fragments, which form the building blocks of the Tree-Substitution Grammars as used in the previous part on parsing.

7.1 AUTHORSHIP ATTRIBUTION WITH TREE FRAGMENTS

We present a method of authorship attribution and stylometry that exploits hierarchical information in phrase structures. Contrary to much previous work in stylometry, we focus on content words rather than function words. Texts are parsed to obtain phrase-structure trees, and compared with texts to be analyzed. An efficient tree-kernel method identifies common tree fragments among data of known authors and unknown texts. These fragments are then used to identify authors and characterize their styles. Our experiments show that the structural information from fragments provides complementary information to the baseline trigram model.

7.1.1 INTRODUCTION

The task of authorship attribution (for an overview cf. Stamatatos, 2009) is typically performed with superficial features of texts such as sentence length, word frequencies, and use of punctuation & vocabulary. While such methods attain high accuracy scores (e.g., Grieve, 2007), the models make purely statistical decisions that are difficult to interpret. To overcome this we could turn to higher-level patterns of texts, such as their syntactic structure.

Syntactic stylometry was first attempted by Baayen et al. (1996), who looked at the distribution of frequencies of grammar productions.[1] More recently, Raghavan et al. (2010) identified authors by deriving a probabilistic grammar

---

[1] A grammar production is a rewrite rule that generates a constituent.
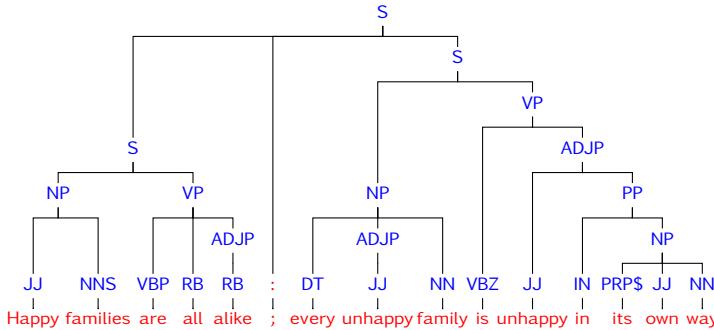
Figure 7.1: A phrase-structure tree produced by the Stanford parser.

for each author and picking the author grammar that can parse the unidentified text with the highest probability. There is also work that looks at syntax on a more shallow level, such as Hirst and Feiguina (2007), who work with partial parses; Wiersma et al. (2011) looked at n-grams of part-of-speech (POS) tags, and Menon and Choi (2011) focused on particular word frequencies such as those of 'stop words,' attaining accuracy scores well above 90 % even in cross-domain tasks.

In this section we also aim to perform syntactic stylometry, but we analyze syntactic parse trees directly, instead of summarizing the data as a set of grammar productions or a probability measure. The unit of comparison is tree fragments. Our hypothesis is that the use of fragments can provide a more interpretable model compared to one that uses fine-grained surface features such as word tokens.

7.1.2 Method

We investigate a corpus consisting of a selection of novels from a handful of authors. The corpus was selected to contain works from different time periods from authors with a putatively distinctive style. In order to analyze the syntactic structure of the corpus we use hierarchical phrase structures, which divide sentences into a series of constituents that are represented in a tree structure; cf. Figure 7.1 for an example. We analyze phrase structures using the notion of tree fragments (referred to as subset trees by Collins and Duffy, 2002). This notion is taken from the framework of Data-Oriented Parsing (Scha, 1990), which hypothesizes that language production and comprehension exploits an inventory of fragments from previous language experience that are used as building blocks for novel sentences. In our case we can surmise that literary authors might make use of a specific inventory in writing their works, which characterizes their style. Fragments can be characterized as follows:

DEFINITION 14. A fragment f of a tree T is a connected subset of nodes from T,
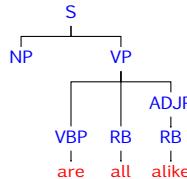
Figure 7.2: A phrase-structure fragment from the tree in Figure 7.1.

with $|f| \geqslant 2$, such that each node of f has either all or none of the children of the corresponding node in T.

When a node of a fragment has no children, it is called a frontier non-terminal; in a parsing algorithm such nodes function as substitution sites where the fragment can be combined with other fragments. Cf. Figure 7.2 for an example of a fragment. An important consideration is that fragments can be of arbitrary size. The notion of fragments captures anything from a single context-free production such as

(1) $S \rightarrow NP\ VP$

... to complete stock phrases such as

(2)   Come with me if you want to live.

In other words, instead of making assumptions about grain size, we let the data decide. This is in contrast to n-gram models where n is an *a priori* defined sliding window size, which must be kept low because of data-sparsity considerations.

To obtain phrase structures of the corpus we employ the Stanford parser (Klein and Manning, 2003), which is a treebank parser trained on the Wall Street journal (wsj) section of the Penn treebank (Marcus et al., 1993). This unlexicalized parser attains an accuracy of 85.7 % on the wsj benchmark ($|w| \leqslant 100$). Performance is probably much worse when parsing text from a different domain, such as literature; for example dialogue and questions are not well represented in the news domain on which the parser is trained. Despite these issues we expect that useful information can be extracted from the latent hierarchical structure that is revealed in parse trees, specifically in how patterns in this structure recur across different texts.

We preprocess all texts manually to strip away dedications, epigraphs, prefaces, tables of contents, and other such material. We also verified that no occurrences of the author names remained.[2] Sentence and word-level tokenization is done by the Stanford parser. Finally, the parser assigns the most likely parse tree for each sentence in the corpus. No further training is performed; as our method is memory based, all computation is done during classification.

---

2 Exception: War and Peace contains a character with the same name as its author. However, since this occurs in only one of the works, it cannot affect the results.

In the testing phase the author texts from the training sections are compared with the parse trees of texts to be identified. To do this we modified the fragment extraction algorithm of Sangati et al. (2010) to identify the common fragments among two different sets of parse trees.[3] This is a tree-kernel method (Collins and Duffy, 2002) which uses dynamic programming to efficiently extract the maximal fragments that two trees have in common. We use the variant reported by Moschitti (2006b) which runs in average linear time in the number of nodes in the trees.

To identify the author of an unknown text we collect the fragments which it has in common with each known author. In order to avoid biases due to different sizes of each author corpus, we use the first 15,000 sentences from each training section. From these results all fragments which were found in more than one author corpus are removed. The remaining fragments which are unique to each author are used to compute a similarity score.

We have explored different variations of similarity scores, such as the number of nodes, the average number of nodes, or the fragment frequencies. A simple method which appears to work well is to count the total number of content words.[4] Given the parse trees of a known author A and those of an unknown author B, with their unique common fragments denoted as $A \sqcap B$, the resulting similarity is defined as:

$$f(A, B) = \sum_{x \in A \sqcap B} \text{content\_words}(x)$$

However, while the number of sentences in the training sets has been fixed, they still diverge in the average number of words per sentence, which is reflected in the number of nodes per tree as well. This causes a bias because statistically, there is a higher chance that some fragment in a larger tree will match with another. Therefore we also normalize for the average number of nodes. The author can now be guessed as:

$$\arg\max_{A \in \text{Authors}} \frac{f(A, B)}{1/|A| \sum_{t \in A} |t|}$$

Note that working with content words does not mean that the model reduces to an n-gram model, because fragments can be discontiguous; e.g., "he said X but Y." Furthermore the fragments contain hierarchical structure while n-grams do not. To verify this contention, we also evaluate our model with trigrams instead of fragments. For this we use trigrams of word & part-of-speech pairs, with words stemmed using Porter's algorithm. With trigrams we simply count the number of trigrams that one text shares with another. Raghavan et al. (2010) have observed that the lexical information in n-grams and the

---

3 The code used in the experiments is available at http://github.com/andreasvc/authident.
4 Content words consist of nouns, verbs, adjectives, and adverbs. They are identified by the part-of-speech tags that are part of the parse trees.

structural information from a PCFG perform a complementary role, achieving the highest performance when both are combined. We therefore also evaluate with a combination of the two.

Author (sentences)	Works (year of first publication)
Conrad, Joseph (25,889)	Heart of Darkness (1899), Lord Jim (1900), Nostromo (1904), The Secret Agent (1907)
Hemingway, Ernest (40,818)	A Farewell To Arms (1929), For Whom the Bell Tolls (1940), The Garden of Eden (1986), The Sun Also Rises (1926)
Huxley, Aldous (23,954)	Ape and Essence (1948), Brave New World (1932), Brave New World Revisited (1958), Crome Yellow (1921), Island (1962), The Doors of Perception (1954), The Gioconda Smile (1922)
Salinger, J.D. (26,006)	Franny & Zooey (1961), Nine Stories (1953), The Catcher in the Rye (1951), Short stories (1940–1965)
Tolstoy, Leo (66,237)	Anna Karenina (1877); transl. Constance Garnett, Resurrection (1899); transl. Louise Maude, The Kreutzer Sonata and Other Stories (1889); transl. Benjamin R. Tucker, War and Peace (1869); transl. Aylmer Maude & Louise Maude

Table 7.1: Texts in the authorship attribution corpus. Note that the works by Tolstoy are English translations from project Gutenberg; the translations are contemporaneous with the works of Conrad.

20 sentences	trigrams	fragments	combined	100 sentences	trigrams	fragments	combined
Conrad	83.00	87.00	94.00	Conrad	100.00	100.00	100.00
Hemingway	77.00	52.00	81.00	Hemingway	100.00	100.00	100.00
Huxley	86.32	75.79	86.32	Huxley	89.47	78.95	89.47
Salinger	93.00	86.00	94.00	Salinger	100.00	100.00	100.00
Tolstoy	77.00	80.00	90.00	Tolstoy	95.00	100.00	100.00
average:	83.23	76.16	**89.09**	average:	96.97	95.96	**97.98**

Table 7.2: Accuracy in % for authorship attribution with test texts of 20 or 100 sentences.

7.1.3 EVALUATION & DISCUSSION

Our data consist of a collection of novels from five authors. See Table 7.1 for a specification. We perform cross-validation on 4 works per author. We evaluate on two different test sizes: 20 and 100 sentences. We test with a total of 500 sentences per work, which gives 25 and 5 data points per work given these sizes.

	Conrad	Hemingway	Huxley	Salinger	Tolstoy
Conrad	94	1		2	3
Hemingway	3	81		11	5
Huxley	5	2	82	1	5
Salinger	1	2	3	94	
Tolstoy	8		2		90

Table 7.3: Confusion matrix for attribution of 20-sentence chunks with trigrams and fragments combined. The rows are the true authors, the columns the predictions of the model.

As training sets only the works that are not tested on are presented to the model. The training sets consist of 15,000 sentences taken from the remaining works. Evaluating the model on these test sets took about half an hour on a machine with 16 cores, employing less than 100 MB of memory per process. The similarity functions were explored on a development set, the results reported here are from a separate test set.

The authorship attribution results are in Table 7.2. It is interesting to note that even with three different translators, the works of Tolstoy can be successfully identified; i.e., the style of the author is modeled, not the translator's.

Gamon (2004) also classifies chunks of 20 sentences, but note that in his methodology data for training and testing includes sentences from the same work. Recognizing the same work is easier because of recurring topics and character names.

Grieve (2007) uses opinion columns of 500–2,000 words, which amounts to 25–100 sentences, assuming an average sentence length of 20 words. Most of the individual algorithms in Grieve (2007) score much lower than our method, when classifying among 5 possible authors like we do, while the accuracy scores are similar when many algorithms are combined into an ensemble. Although the corpus of Grieve is carefully controlled to contain comparable texts written for the same audience, our task is not necessarily easier, because large differences within the works of an author can make classifying that author more challenging.

Table 7.3 shows a confusion matrix when working with 20 sentences. It is striking that the errors are relatively asymmetric: if A is often confused with B, it does not imply that B is often confused with A. This appears to indicate that the similarity metric has a bias towards certain categories which could be removed with a more principled model.

Here are some examples of sentence-level and productive fragments that were found:

(3)   Conrad: [PP [IN] [NP [NP [DT] [NN sort]] [PP [IN of] [NP [JJ] [NN]]]]]

(4)   Hemingway: [$_{VP}$ [$_{VB}$ have] [$_{NP}$ [$_{DT}$ a] [$_{NN}$ drink]]]

(5)   Salinger: [$_{NP}$ [$_{DT}$ a] [$_{NN}$] [$_{CC}$ or] [$_{NN}$ something]]

(6)   Salinger: [$_{ROOT}$ [$_{S}$ [$_{NP}$ [$_{PRP}$ I]] [$_{VP}$ [$_{VBP}$ mean] [$_{SBAR}$]] [. .]]]

(7)   Tolstoy: [$_{ROOT}$ [$_{SINV}$ [„ "] [$_{S}$] [, ,] [„ "] [$_{VP}$ [$_{VBD}$ said]] [$_{NP}$] [, ,] [$_{S}$ [$_{VP}$ [$_{VBG}$ shrugging] [$_{NP}$ [$_{PRP\$}$ his] [$_{NNS}$ shoulders]]]]] [. .]]]

It is likely that more sophisticated statistics, for example methods used for collocation detection, or general machine learning methods to select features such as support vector machines would allow to select only the most characteristic fragments.

On the task of attributing the disputed and co-authored Federalist papers, the fragment model attributes 14 out of 15 papers to Madison (in line with the commonly held view); the 19th paper is (incorrectly) attributed to Jay. To counter the imbalance in training data lengths, we normalized on the number of nodes in the fragments common to training & test texts.

7.1.4   DISCUSSION

We conclude from the above that deep syntactic features show potential for stylometric tasks. Our method of syntactic stylometry is conceptually simple—we do not resort to sophisticated statistical inference or an ensemble of algorithms—and takes sentence-level hierarchical phenomena into account. Contrary to much previous work in stylometry, we included content words rather than just function words. The experiments show that lexical and syntactic features perform particularly well when combined. Having demonstrated the feasibility of analyzing literary syntax through parse tree fragments, it becomes possible to apply these techniques to address other literary questions. Another direction is to avoid manually defining the similarity function, and instead employ a more sophisticated machine learning algorithm, such as support vector machines.
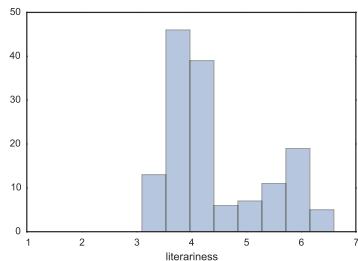
Figure 7.3: A histogram of the mean literary ratings.

7.2 Predicting literariness from a bag-of-bigrams model

We will consider the task of predicting the literary and quality ratings of the contemporary Dutch novels in the Riddle corpus. In this pilot study, we will work with a subset of the novels and consider two kinds of word bigram features, content and style bigrams. The use of support vector machines is explored with classification and regression tasks.

7.2.1 Survey data and novels

	Original	Translated
Thrillers	0	31
Literary thrillers	26	29
Literary fiction	27	33

Table 7.4: The number of books in each category. These categories were assigned by the publishers.

The data set used for this pilot study contains a selection of 146 books from the 401 included in the Riddle survey; see Table 7.4 and 7.5. Both translated and original (Dutch) novels are included. The data set contains three genres, as indicated by the publisher: literary novels, literary thrillers and thrillers. There are no Dutch thrillers in the corpus. Note that these labels are ones that the publishers have assigned to the novels. We will not be using these labels in our experiments—save for one where we interpret genre differences—we base ourselves on reader judgments. In other words: when we talk about highly literary texts, they (in theory) could be part of any of these genres, as long as readers judged them to be highly literary.

	Original	Translated
literature	Bernlef, Dis, Dorrestein, Durlacher, Enquist, Galen, Giphart, Hart, Heijden, Japin, Kluun, Koch, Kroonenberg, Launspach, Moor, Mortier, Rosenboom, Scholten, Siebelink, Verhulst, Winter	Auel, Avallone, Baldacci, Binet, Blum, Cronin, Donoghue, Evans, Fragoso, George, Gilbert, Giordano, Harbach, Hill, Hodgkinson, Hosseini, Irving, James, Krauss, Lewinsky, Mastras, McCoy, Pick, Picoult, Rosnay, Ruiz Zafón, Sansom, Yalom
literary thrillers	Appel, Dijkzeul, Janssen, Noort, Pauw, Terlouw, Verhoef, Vermeer, Visser, Vlugt	Coben, Forbes, French, Gudenkauf, Hannah, Haynes, Kepler, Koryta, Lackberg, Larsson, Läckberg, Nesbo, Patterson, Robotham, Rosenfeldt, Slaughter, Stevens, Trussoni, Watson
thrillers		Baldacci, Clancy, Cussler, Forsyth, Gerritsen, Hannah, Hoag, Lapidus, McFadyen, McNab, Patterson, Roberts, Rose

Table 7.5: Authors in the data set.

7.2.2 Experimental setup

Three aspects of a machine learning model can be distinguished: the target of its predictions, the features which predictions are based on, and the kind of model and predictions it produces.

Machine learning tasks

We consider two tasks:

1. Literary ratings
2. Bad/good (general quality)

The target of the classification model is a binary classification whether a book is within the 25 % judged to be the most literary, or good. Figure 7.3 shows a histogram of the literary judgments. This cutoff divides the two peaks in the histogram, while ensuring that the number of literary novels is not too small.

A more difficult task is to try to predict the average rating for literariness of each book. This not only involves the large differences between thrillers and literary novels, but also smaller differences within these genres.

Textual features

The features used to train the classifier are based on a bag-of-words model with relative frequencies. Instead of single words we use word bigrams. Bigrams are occurrences of two consecutive words observed in the texts. The bigrams are restricted to those that occur in between 60 % and 90 % of texts used in the model, to avoid the sparsity of rare bigrams on the one hand, and the most

frequent function bigrams on the other. No limit is placed on the total number of bigram features. We consider two feature sets:
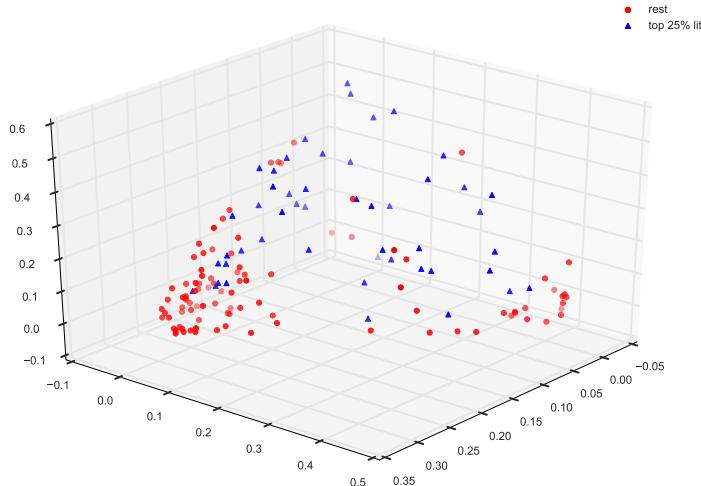
Figure 7.4: Non-negative matrix factorization based on style bigrams (literary novels are the blue triangles).

CONTENT BIGRAMS: Content words contribute meaning to a sentence and are thus topic related; they consist of nouns, verbs, adjectives, and adverbs. Content bigrams are extracted from the original tokenized text, without further preprocessing.

STYLE BIGRAMS: Style bigrams consist of function words, punctuation, and part-of-speech tags of content words (similar to Bergsma et al. 2012). In contrast with content words, function words determine the structure of sentences (determiners, conjunctions, prepositions) or express relationships (pronouns, demonstratives). Function words are identified by a selection of part-of-speech tags and a stop word list. Function words are represented with lemmas, e.g., auxiliary verbs appear in uninflected form. Lemmas and part-of-speech tags were automatically assigned by the Alpino parser (Bouma et al., 2001).

MODELS

All machine learning experiments are performed with scikit-learn (Pedregosa et al., 2011). The classifier is a linear Support Vector Machine (SVM) with regularization tuned on the training set. The cross-validation is 10-fold and stratified (each fold has a distribution of the target class that is similar to that of the whole data set).

For regression the same setup of texts and features is used as for the classification experiments, but the machine learning model is a linear Support Vector Regression model.

7.2.3 Results

Before we train machine learning models, we consider a dimensionality reduction of the data. Figure 7.4 shows a non-negative matrix factorization of the style bigrams. In other words, this is a visualization of a decomposition of the bigram counts, without taking into account whether novels are literary or not (i.e., an unsupervised model). Notice that most of the non-literary novels (red) cluster together in one corner, while the literary books (blue) show more variation. When content bigrams are used, a similar cluster of non-literary books emerges, but interestingly, this cluster only consists of translated works. With style bigrams this does not occur.

Features	Literary	Bad/good
Content bigrams	90.4	63.7
Style bigrams	89.0	63.0

Table 7.6: Accuracy (percentage correct) of classifying literary and good novels using bigrams.

This result seems to suggest that non-literary books are easier to recognize than literary books, since the literary novels show more variation. However, note that this decomposition present just one way to summarize and visualize the data. The classification model, when trained specifically to recognize literary and non-literary texts, can still identify particular discriminating features.

Classification

Table 7.6 shows the evaluation of the classification models. The content bigrams perform better than the style bigrams. The top-ranked bigram features of the model for literary classification are shown in Table 7.8.

If we look only at the top 20 bigrams that are most predictive of literary texts according to our model and plot how often they occur in each genre as specified by the publishers, we see that these bigrams occur significantly more often in literary texts; cf. the plot in Figure 7.6. This indicates that there are features specific to literary texts, despite the variance among literary texts shown in Figure 7.4.

When trained on the bad/good dimension, the classification accuracy is around 60 %, compared to around 90 % for literariness, regardless of whether the features consist of content or style bigrams. This means that the bad/good judgments are more difficult to predict from these textual features. This is not due to the variance in the survey responses themselves. If literariness were a

Features	Literary	Bad/Good
Content bigrams	61.3 (0.65)	33.5 (0.49)
Style bigrams	57.0 (0.67)	22.2 (0.52)

Table 7.7: Evaluation of the regression models; R^2 scores (percentage of variation explained), root mean squared error in parentheses (1–7).

more clearly defined concept for the survey participants than general quality, we would expect there to be less consensus and thus more variance on the latter dimension. But this is not what we find; in fact the mean of the standard deviations of the bad/good responses is lower than for the literariness responses (1.08 vs. 1.33).[5] Prediction performance depends not so much on the variance in the responses themselves, but rather on the degree to which the features available to the model provide information on the responses. It is likely that the bad/good dimension depends on higher-level, plot-related characteristics, or text-extrinsic social factors.

Regression

The regression results cannot be evaluated with a simple 'percentage correct' accuracy metric, because it is not feasible to predict a continuous variable exactly. Instead we report the coefficient of determination (R^2). This metric captures the percentage of variation in the data that the model explains by contrasting the errors of the model predictions with those of the null model which always predicts the mean of the data. R^2 can be contrasted with the root mean squared error, also known as the standard error of the estimate, or the norm of residuals, which measures how close the predictions are to the target on average. In contrast with R^2, this metric has the same scale as the original data, and lower values are better.

The regression scores are shown in Table 7.7. Predicting the bad/good scores is again more difficult. The regression results for literariness predictions are visualized in Figure 7.5. Each data point represents a single book. The x-axes show the literariness ratings from survey participants, while the y-axes show the predictions from the model. The diagonal line shows what the perfect prediction would be, and the further the data points (novels) are from this line, the greater the error. On the sides of the graphs the histograms show the distribution of the literariness scores. Notice that the model based on content bigrams mirrors the bimodal nature of the literariness ratings, while the histogram of predicted literariness scores based on style bigrams shows only a single peak.

---

5 However, it should be noted that while both ratings are on a 1–7 Likert scale, the observed range and distribution for the bad/good versus literary responses are arguably not comparable (cf. the histograms on the axes of Figure 5.2); i.e., few bad ratings ($\leqslant 4$) were reported, while many novels were consistently rated in this range on the literary dimension.
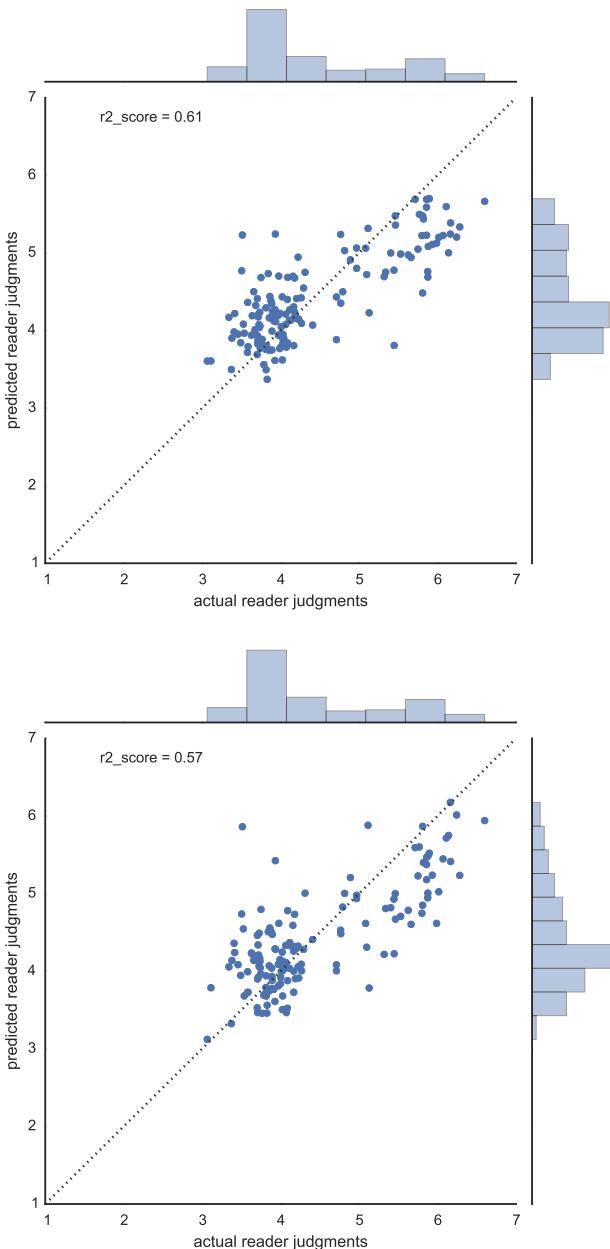
Figure 7.5: Scatter plot of predicted literary judgments using content bigrams (above) and style bigrams (below).
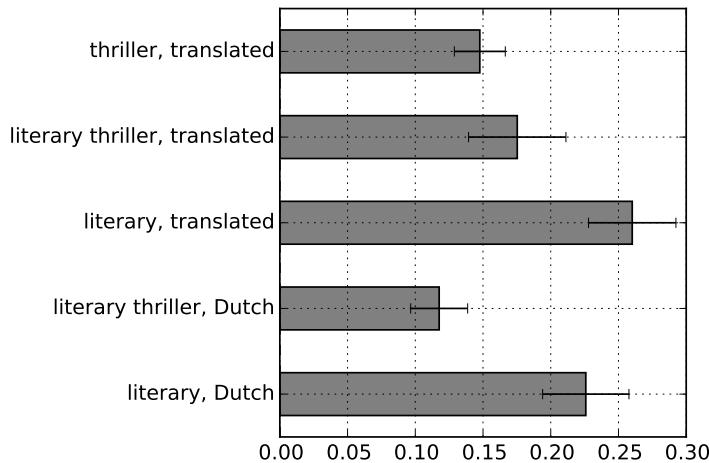
Figure 7.6: A bar plot of the number of occurrences (% of total tokens) of the top 20 most important literary features (cf. Table 7.8) across the genres given by the publisher (error bars show 95 % confidence interval).

Figure 7.7 shows the same regression results with the publisher-assigned genres highlighted. The graph shows that predicting the literariness of thrillers is more difficult than predicting the literariness of the more literary rated novels. Most thrillers have ratings between 3.4 and 4.3, while the model predicts a wider range of ratings between 3.3 and 5.0; i.e., the model predicts more variation than actually occurs. For the literary novels both the predicted and actual judgments show a wide range between 4.5 and 6.5. The actual judgments of the literary novels are about 0.5 points higher than the predictions. However, there are novels at both ends of this range for which the ratings are well predicted. Judging by the dispersion of actual and predicted ratings of the literary novels compared to the thrillers, the model accounts for more of the variance within the ratings of literary novels.

It should be noted that while in theory 100 % is the perfect score, the practical ceiling is much lower due to the fact that the model is trying to predict an average rating—and because part of the variation in literariness will only be explainable with richer features, text-extrinsic sociological influences, or random variation.

7.2.4 INTERPRETATION

As the experiments show, there are textual elements that allow a machine learning model to distinguish between works that are perceived as highly literary as opposed to less literary ones—at least for this data set and survey. We now take a closer look at the features and predictions of the literary classification
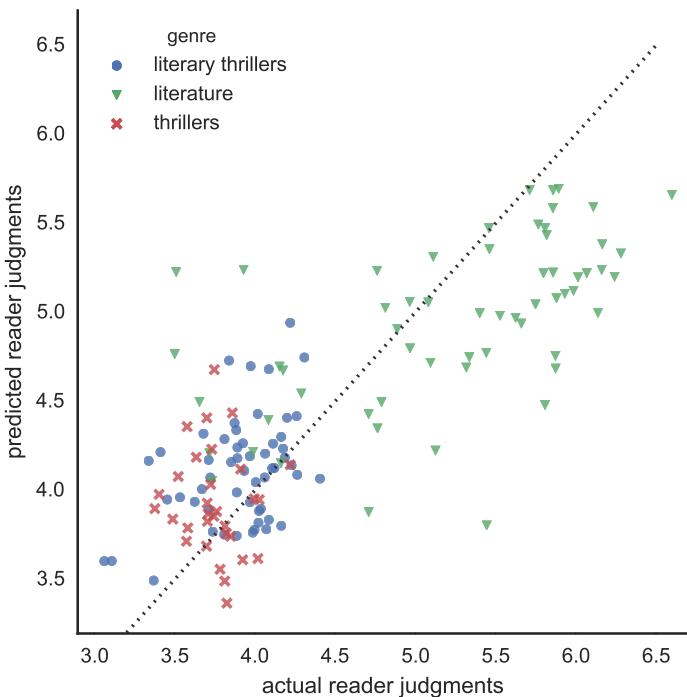
Figure 7.7: Scatter plot of predicted literary judgments using content bigrams, with data points distinguished by the publisher-assigned genre.

task to interpret its success.

Content

When we look at the forty bigrams that perform best and worst for the literary novels (cf. Table 7.8), we can identify a few tendencies.

The book, *a book*, *a letter*, and *to write* are also part of the most important features, as well as *the bar*, *a cigarette*, and *the store*. This suggests a certain pre-digital situatedness, as well as a reflection on the writing process. Interestingly enough, in contrast to *the book* and *letter* that are most discriminating, negative indicators contain words related to modern technology: *mobile phone* and *the computer*. Inspection of the novels shows that the literary novels are not necessarily set in the pre-digital age, but that they have fewer markers of recent technology. This might be tied to the adage in literary writing that good writing should be 'timeless'—which in practice means that at the very least a novel should not be too obvious in relating its settings to the current day. It could also show a hint of nostalgia, perhaps connected to a romantic image of

weight	literary features, content		weight	non-literary features, content	
12.1	*de oorlog*	the war	-6.1	*de moeder*	the mother
8.1	*het bos*	the forest	-5.1	*keek op*	looked up
8.1	*de winter*	the winter	-4.9	*mijn hoofd*	my head
6.6	*de dokter*	the doctor	-4.9	*haar moeder*	her mother
5.8	*zo veel*	so much	-4.7	*mijn ogen*	my eyes
4.8	*nog altijd*	yet still	-4.7	*ze keek*	she looked
4.5	*de meisjes*	the girls	-4.5	*mobiele telefoon*	mobile telephone
4.3	*zijn vader*	his father	-4.2	*de moord*	the murder
4.0	*mijn dochter*	my daughter	-4.0	*even later*	a while later
3.9	*het boek*	the book	-3.8	*nu toe*	(until) now
3.8	*de trein*	the train	-3.5	*zag ze*	she saw
3.7	*hij hem*	he him	-3.4	*ik voel*	I feel
3.7	*naar mij*	at me	-3.3	*mijn man*	my husband
3.5	*zegt dat*	says that	-3.2	*tot haar*	to her
3.5	*het land*	the land	-3.2	*het gebouw*	the building
3.5	*een sigaret*	a cigarette	-3.2	*liep naar*	walked to
3.4	*haar vader*	her father	-3.1	*we weten*	we know
3.4	*een boek*	a book	-3.1	*enige wat*	only thing
3.2	*de winkel*	the shop	-3.1	*en dus*	and so
3.1	*elke keer*	each time	-3.0	*in godsnaam*	in god's name

weight	literary features, style		weight	non-literary features, style	
21.8	*! WW*	! VERB ,	-13.8	*nu toe*	until now
20.5	*u ,*	you (FORMAL) ,	-13.4	*en dus*	and so
18.0	*haar haar*	her her	-13.4	*achter me*	behind me
16.5	*SPEC :*	NAME :	-13.2	*terwijl ik*	while I
15.4	*worden ik*	become I	-13.1	*tot nu*	until now

Table 7.8: The top 20 most important content bigrams and top 5 most important style bigrams of literary (left), and non-literary texts (right), respectively.

the writer.

In the negative features, we find another time-related tendency. The first is indications of time—*little after*, and in Dutch 'tot nu' and 'nu toe', which are part of the phrase 'tot nu toe' (*so far* or *up until now*), *minutes after* and *ten minutes*; another indicator that awareness of time, albeit in a different sense, is not part of the 'literary' discourse. Indicators of location are *the building*, *the garage/car park*, and *the location*, showing a different type of setting than the one described above. We also see indicators of homicide: *the murder*, and *the investigation*. Some markers of colloquial speech are also found in the negative markers: *for god's sake* and *thank you*, which aligns with a finding of Jautze et al. (2013), where indicators of colloquial language were found in low-brow literature.

It is possible to argue, that genre is a more important factor in this classification than literary style. However, we state that this is not particular to

this research, and in fact unavoidable. The discussion of how tight genre and literariness are connected, has been held for a long time in literary theory and will probably continue for years to come. Although it is not impossible for so called 'genre novels' to gain literary status (cf. Margaret Atwood's sci-fi(-like) work for instance—although she objects to such a classification; Hoby 2013), it is the case that certain topics and genres are considered to be less literary than others. The fact that the literary novels are apparently not recognized by proxy, but on an internal coherence (cf. Section 7.2.3), does make an interesting case for the literary novel to be a genre on its own. Computational research into genre differences has proven that there are certain markers that allow for a computer to make an automated distinction between them, but it also shows that interpretation is often complex (Moretti, 2005; Allison et al., 2011; Jautze et al., 2013). Topic modeling might give some more insight into our findings.

Style

A stronger case against genre determining the classification is the success of the function words in the task. Function words are not directly related to themes or topics, but reflect writing style in a more general sense. Still, the results do not rule out the existence of particular conventions of writing style in genres, but in this case the distinction between literariness and genre becomes more subtle. Function words are hard to interpret manually, but we do see in the top 20 (Table 7.8 shows the top 5) that the most discriminating features of less literary texts contain more question marks (and thus questions), and more numerals (TW)—which can possibly be linked to the discriminative qualities of time-indications in the content words. Some features in the less-literary set appear to show more colloquial language again, such as *ik mezelf* ('I myself'), *door naar* ('through/on to'; an example can be found in the sentence '*Heleen liep door naar de keuken.*', which translates to 'Heleen walked on to the kitchen', a sound grammatical construction in Dutch, but perhaps not a very aesthetically pleasing one). A future close reading of the original texts will give more information on this intuition.

In future work, more kinds of features should be applied to the classification of literature to get more insight. Many aspects could be studied, such as readability, syntax, semantics, discourse relations, and topic coherence. Given a larger data set, the factors genre and translation/original can be controlled for.

The general question which needs to be answered is whether a literary interpretation of a computational model is even possible. The material to work with (the features), consist of concise sets of words or even part-of-speech tags, which are not easy to interpret manually; and they paint only a small part of the picture. The workings of the machine learning model remain largely hidden to the interpreter. This is an instance of the more general problem of the interpretability of results in computational humanities (Bod, 2013). In the specific case of literature, we can observe that readers of literature follow a

similar pattern: literature can be recognized and appreciated, but it is hard to explain what makes texts literary, let alone to compose a highly literary work.

Good and bad predictions

In Figure 7.7, we can see both outliers and novels that are well predicted by the regression model. Here we discuss a few and suggest why the model does or does not account for their perceived literariness.

Emma Donoghue - Room A literary novel that is rated as highly literary (5.5), but with a lower prediction (3.8). This may be because this novel is written from the perspective of a child, with a correspondingly limited vocabulary.

Elizabeth Gilbert - Eat, Pray Love A novel with a low literariness rating (3.5), but a high prediction (5.2) by the model. This novel may be rated lower due to the perception that it is a novel for women, dealing with new age themes, giving it a more specific audience than the other novels in the data set.

Charles Lewinsky - Melnitz A novel that is both rated (5.7) and predicted (5.7) as highly literary. This novel chronicles the history of a Jewish family including the events of the second world war. This subject, and the plain writing style makes it stand out from the other novels.

Erwin Mortier - While the Gods Were Sleeping The most highly rated (6.6) literary novel in the data set, with a high prediction (5.7). A striking feature of this novel is that it consists of short paragraphs and short, often single line sentences. It features a lot of metaphors, analogies, and generally a poetic writing style. This novel also deals with war, but the writing style contrasts with Lewinsky, which may explain why the model's prediction is not as close for this novel.

7.2.5 Summary

Our experiments have shown that literary novels share significant commonalities, as evidenced by the performance of machine learning models. It is still a challenge to understand what these literary commonalities consist of, since a large number of word features interact in our models. General quality is harder to predict than literariness.

Features related to genre (e.g., *the war* in literary novels and *the homicide* in thrillers) indicate that genre is a possible confounding factor in the classification, but we find evidence against the notion that the results are solely due to genre. One aspect that stood out in our analysis of content features, which is not necessarily restricted to genre (or which might indicate that the literary novel is a genre in and of itself), is that setting of space and time rank high among the discriminating features. This might be indicative of a 'timeless quality'

that is expected of highly literary works (where words as *book* and *letter* are discriminative)—as opposed to more contemporary settings in less literary novels (*computer* and *mobile phone*). Further study is needed to get more insight into these themes and to what extent these are related to genre differences or a literary writing style.

The good performance of style features shows the importance of writing style and indicates that the classification is not purely based on topics and themes. Although genres may also have particular writing styles and thus associated style features, the fact that good results are obtained with two complementary feature sets suggests that the relation between literariness and text features is robust.

Finally, the regression on content and function words shows that the model accounts for more than just genre distinctions. The predictions within genres are good enough to show that it is possible to distinguish highly literary works from less literary works. This is a result that merits further investigation.

7.3 Mining literary tree fragments

Text mining and stylometry is typically based on simple, frequent textual features extracted from surface forms such as word or character n-grams, as in the previous section. As shown in Section 7.1, an alternative to such simple features is to extract features from syntactic parse trees, namely using tree fragments: arbitrarily-sized connected subgraphs of parse trees. Post and Bergsma (2013) provide an overview of previous work on using tree fragments as predictive features. These fragments are flexible in that they can capture both stylistic (syntactic) and topical (lexical) aspects, and can be both general (small), or specific (large).

We can distinguish implicit and explicit fragment models. In an implicit fragment model, tree fragments are not represented individually, and it is not possible to inspect the weight a fragment contributes to making predictions with the model. However, through the use of a kernel, the contribution of all fragments (under a given definition) can be computed. This method is known under the rubric of tree kernels (Moschitti, 2006b). Explicit fragments are trained on a predefined matrix of document-fragment counts. With explicit fragments, it is necessary to define the set of fragments. Previous work has used tree-substitution grammars extracted from newswire treebanks (Post and Bergsma, 2013), or extracted fragments at test time from the training corpus in a memory-based setup (this chapter, Section 7.1).

Here we consider the challenge of selecting and extracting relevant tree fragments from the corpus itself, at training time, so as to get more specific and interpretable features. Since we will select features at training time, they need to generalize to unseen documents without knowing anything about the documents in the test set. The approach is similar to Swanson and Charniak (2013), in that we aim to discover the most predictive fragments using a relevancy metric. There are two main differences. First, instead of sampling fragments from a Bayesian mixture model, we use the method of recurring fragment extraction. Second, instead of ranking fragments on relevancy with respect to a discrete set of classes using entropy-based measures, we use a continuous target value, namely literary ratings of novels, and use the Pearson correlation coefficient as metric.

7.3.1 The task: predicting literary ratings

We consider the regression problem of predicting the literary ratings of a set of novels, on a scale of 1 to 7. The task we consider is to predict the mean rating for each novel. We exclude novels that have been rated by less than 50 participants; 16 novels are excluded through this constraint.

Since we want to extract relevant features from the texts themselves and the number of novels is relatively small, it is important to apply cross-validation, so as to exploit the data to the fullest extent while still maintaining an out-of-sample approach. The tree fragments can be highly specific to particular novels,
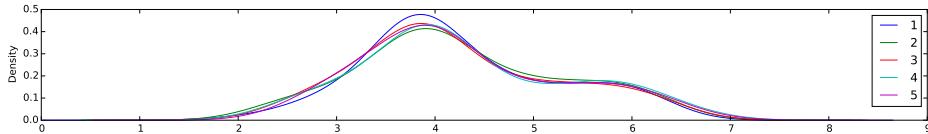
Figure 7.8: A kernel density estimation plot of literary ratings of the 5-fold division of the corpus.

therefore feature selection should be part of the cross-validation; i.e., we aim to select features that are representative across multiple training folds. We divide the corpus in 5 folds, with the following constraints:

- Novels by the same author must be assigned to the same fold.
- Each fold should have a distribution of literary ratings that is similar to the overall distribution (stratification).
- Each fold should be roughly the same size.

Each fold contains around 80 novels. Figure 7.8 visualizes the distribution of literary ratings in the folds using a kernel density estimation (i.e., similar to a histogram but continuous). The shape illustrates the bimodal distribution of the literary ratings: a large number of novels have a rating of 4, with a smaller peak of literary novels at 6. The plot shows a very similar shape for the 5 folds, which ensures that each fold holds a representative sample of the target distribution.

7.3.2 Motivation for rich syntactic features

The bigrams performed rather well in the previous section, so why do we need or want syntax? More generally, it has been observed that simple n-gram features often exhibit superior performance in various tasks:

> A variety of "discouraging results" in the text categorization literature have shown that simple bag-of-words (BoW) representations usually perform better than "more sophisticated" ones (e.g. using syntax)
> — Bergsma et al. (2012)

Still, there are at least four reasons why n-grams are limited and investigating syntactic features is worthwhile:

1. Phrases longer than n tokens, phrases with gaps and open slots. Such phrases include multi-word expressions. Consider:

    (8)   a.   in light of
               b.   so there's that
               c.   Get NP off the ground

Notice that none of the bigrams in the above signal that there is a multi-word expression, only the expression as a whole identifies it as a fixed expression. The open slot NP stands for a constituent of any length.

2. Non-lexical information, e.g., function tags. Consider these simple noun phrases:

(9)   a. NP-SBJ → DT NN
          The apple is a deciduous tree.
     b. NP-OBJ → DT NN
          John eats *an apple*.

Even though this is one of the most common grammar productions, the distinction in grammatical function is a strong predictor; cf. Figure 7.9. The *y*-axis shows the proportion of simple subject/object NPs, divided by the total number of simple NPs. The difference may be explained by the observation that in everyday language, it is more common to have an animate subject (e.g., proper noun or pronoun), as opposed to a determiner-noun NP which is more likely to be inanimate, as in the examples of (9).

3. Larger syntactic units or constructions. Consider the following sentence:

(10)   Langzaam besefte hij *dat* hij zijn broers niet had moeten laten ophangen, *dat* zijn daad veel vijandigheid in het land zou veroorzaken, *dat* God hem daarvoor hard zou straffen. (Abdolah-Koning) 'Slowly he realized *that* he should not have had his brothers executed, *that* his act would cause much animosity in his country, *that* God would punish him harshly for it.'

This sentence contains three that-clauses, that are all an argument to the same verb (besefte, *realized*). To capture and generalize this stylistic device requires an abstract, syntactic feature containing just the three 'that' instances and their supporting structure.

4. discontinuities, non-local dependencies. Certain syntactic phenomena can neither be described by *n*-grams, nor by the traditional syntactic representations. These have been the topic of part I of this thesis.

7.3.3 PREPROCESSING

We parse the novels with the Alpino parser (Bouma et al., 2001). We preprocess the trees in the same manner as was applied for the Disco-DOP tree-substitution grammar in Chapter 3. Namely, the trees include discontinuous constituents,
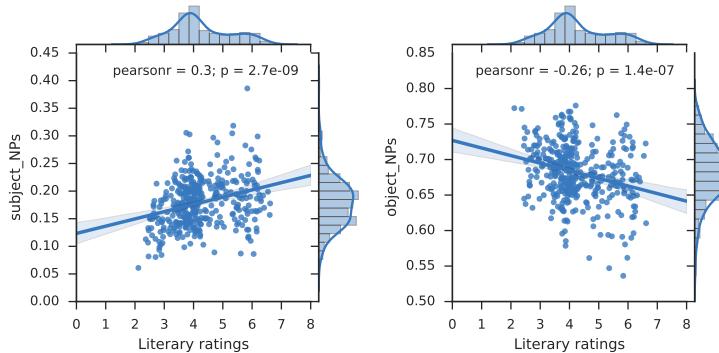
Figure 7.9: Simple noun phrases as subject (left) versus object (right).

non-terminal labels consist of both syntactic categories and function tags, se-lected morphological features, and constituents are binarized head-outward with a markovization of $h = 1, v = 1$.

For a fragment to be attested in a pair of parse trees, its labels need to match exactly, including the aforementioned categories, tags, and features. The binarization implies that fragments may contain partial constituents; i.e., a contiguous sequence of children from an n-ary constituent.

Cf. Figure 7.10 for an example parse tree; for brevity, this tree is rendered without binarization. The non-terminal labels consist of a syntactic category (shown in red), followed by a dash and a function tag (green). Some labels contain percolated morphological features, prefixed by a colon. The part-of-speech tags additionally have morphological features (black) in square brackets.

To remove length effects, and potential particularities of the start of novels, we consider sentence numbers 1000–2000 of each novel. There are 18 novels with less than 2000 sentences; these are excluded. Together with the constraint of at least 50 ratings, this brings the total number of novels we will consider at 369.

7.3.4 ALTERNATIVE FEATURE SELECTION METHODS

Before outlining our approach, we first consider some alternative feature selec-tion methods. There are two methods for feature selection based on Support Vector Machines (SVM). The standard SVM formulation uses L_2 (quadratic) reg-ularization; this produces dense models in which all features are used. The formula for fitting an epsilon-insensitive support vector regression model is as follows:

$$\min_{w} \frac{1}{2} w^T w + C \sum_{i=1}^{n} (\max(0, |y_i - w^T X_i| - \epsilon))$$
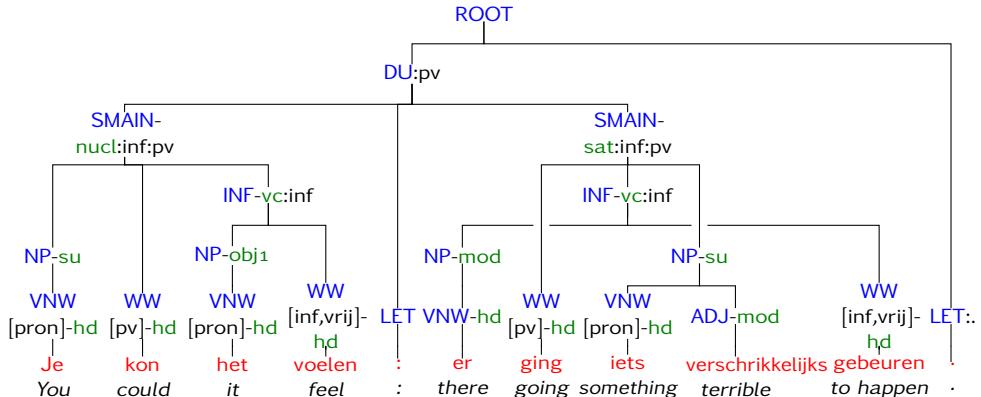
Figure 7.10: A parse tree from Franzen, The Corrections. Translation: "You could feel it: something terrible was about to happen."

where X are the training vectors, y are the target values, and n is the number of samples. Two parameters C and ϵ control the amount of regularization. The result is w, the feature weights (coefficients). L_1-regularized linear models produce sparse models, in which only a subset of features receives nonzero weights. For example, fitting a Lasso model optimizes the following objective function (p is the number of features, and t a regularization parameter):

$$\min_{w_0,w} \left\{ \frac{1}{n} \sum_{i=1}^{n} (y_i - w_0 - X_i^T n)^2 \right\} \text{ subject to } \sum_{j=1}^{p} |w_j| \leqslant t$$

Recursive feature elimination iteratively trains models while keeping only the features with the highest weights from the previous step. Since this method requires repeated model fitting, it is not feasible when the feature set grows beyond a certain size. For example, from the 40k trees of the Penn treebank, about 1 million recurring fragments are extracted, which is not feasible to train an svm on; we aim for a selection procedure to reduce the potential fragments to tens of thousands of features.

Another approach is to integrate the parse trees with the model as is done in tree-kernel support vector machines. A Tree-kernel svm (Collins and Duffy, 2002; Moschitti, 2006b) defines a predictive model based on parse-tree similarity. Instead of specifying a pre-defined feature vector for each instance, the classifier works with parse trees directly. The similarity of parse trees is based on the implicitly defined set of tree fragments that they share. This means that feature engineering is automated, although the features cannot be directly inspected, nor is it possible to control which features are included or excluded.[6] Pighin and

---

6 It is possible to change the definition of tree fragments, e.g., subtrees, subset trees, or partial trees; however, it is not possible to select an arbitrary subset.
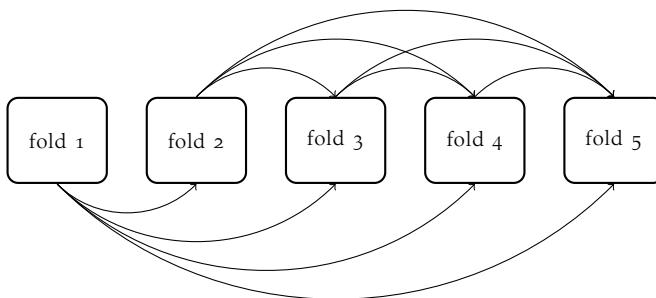
Figure 7.11: Cross-validated fragment extraction.

Moschitti (2010) present a procedure which extracts the most important features from such a model. However, their algorithm assumes that each instance is a single sentence, rather than a document. Therefore it is not directly applicable to the problem in this chapter. Moreover, Post and Bergsma (2013) report that tree-kernel methods are computationally expensive (prohibitively so, given enough training data), and explicit fragment models achieve better performance in a fraction of training time.

7.3.5 FRAGMENT MINING

We present an approach based on heuristics using simple statistics. The general idea is based on Yu and Liu (2004), and also followed by Swanson and Charniak (2013). The procedure is divided in two parts. Fragment extraction:

1. Given texts divided in folds $F_1 \ldots F_n$, each C_i is the set of parse trees obtained from parsing all texts in F_i. Extract the largest common fragments of the parse trees in all pairs of folds $\langle C_i, C_j \rangle$ with $i < j$. A common fragment f of parse trees t_1, t_2 is a connected subgraph of both t_1 and t_2. The result is a set of initial candidates that occur in at least two different texts, stored separately for each pair of folds $\langle C_i, C_j \rangle$.

2. Count occurrences of all fragments in all novels.

Fragment selection is done separately w.r.t. each test fold. Given test fold i, we consider the fragments found in training folds $\{1..n\} \setminus i$; e.g., given $n = 5$, for test fold 1 we select only from the fragments and their counts as observed in training folds 2–5. Given a set of fragments from training folds, selection proceeds as follows:

1. Zero count threshold: remove fragments that occur in less than 5 % of novels (too specific); frequency threshold: remove fragments that occur

recurring fragments	3,193,952
after threshold: occurs in $> 5\%$ of texts	375,514
after threshold: freq > 50	98,286
relevance threshold: correlated s.t. $p < 0.05$	30,044
redundancy threshold: $|r| < 0.5$	7,642

Table 7.9: The number of fragments in folds 2–5 after each step.

less than 50 times across the corpus (too rare). The purpose of this step is to filter out fragments for which there is too little information to reliably determine a correlation with the literary ratings.

2. Relevance threshold: select fragments by considering the correlation of their counts with the literary ratings of the novels. Apply a simple linear regression based on the Pearson correlation coefficient[7], and use an F-test to filter out fragments whose p-value[8] > 0.05. The F-test determines significance based on the number of data points N, and the correlation r; the effective threshold is around $|r| > 0.11$.

3. Redundancy removal: greedily select most relevant fragment and remove other fragments that are too similar to it. Similarity is measured by computing the correlation coefficient between the feature vectors of two fragments, with a cutoff of $|r| > 0.5$.

   Preliminary experiments where this step was not applied indicated that it does improve performance.

Table 7.9 lists the number of fragments in folds 2–5 after each of these steps.

7.3.6 Qualitative analysis of selected fragments

Figure 7.12 visualizes three highly ranked fragments according to the correlation based metric, extracted from the first fold. Note the different scales on the y-axis.

The fragments can be categorized by the presence of lexical items and discontinuous constituents. Table 7.10 shows a breakdown of fragment types in the first fold. The following lists some examples of each of these types. Each fragment is followed by an example sentence from the novel where the fragment is most common (including my translations).

Purely syntactic fragments:

---

7 Note that the Pearson correlation coefficient assumes a linear relationship, but on the other hand it offers an interpretable effect size. Other relevancy metrics are based on Mutual Information (e.g., Torkkola, 2003; Kraskov et al., 2004) and repeated sampling of features and instances (e.g., Partition Retention; Chernoff et al. 2009, and Stability Selection; Meinshausen and Bühlmann 2010). Evaluating such measures is left for future work.

8 If we were actually testing hypotheses we would need to apply Bonferroni correction to avoid the Family-Wise Error due to multiple comparisons; however, since the regression here is only a means to an end, we leave the p-values uncorrected.
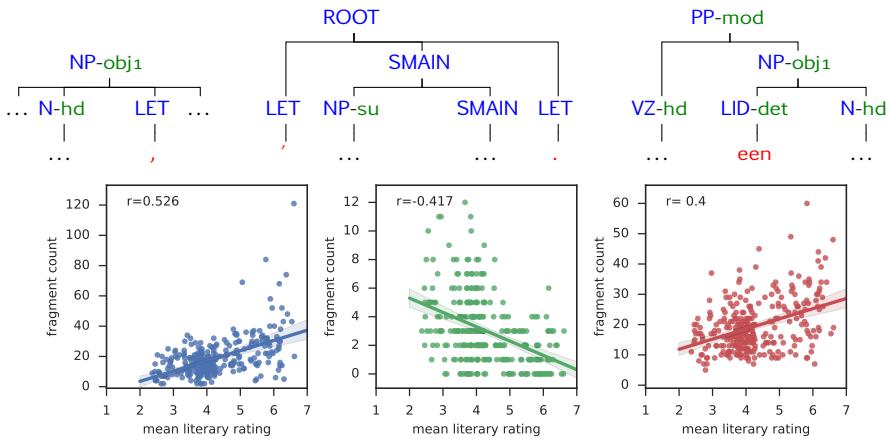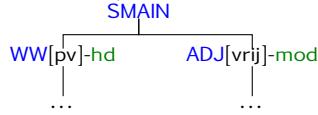
Figure 7.12: Three fragments whose frequencies in the first fold have a high correlation with the literary ratings. From left to right; Blue: complex NP with comma; Green: quoted speech. Red: Adjunct PP with indefinite article.

fully lexicalized	1,321
syntactic (no lexical items)	2,283
mixed	4,038
discontinuous	684
discontinuous substitution site	396
total	7,642

Table 7.10: Breakdown of fragment types selected in the first fold.
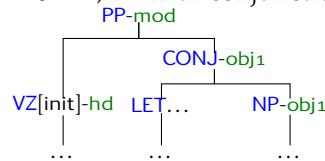
(11) a. $r = -0.435$; verb with modifier; max freq=35; sum freq=4117

SMAIN
WW[pv]-hd   ADJ[vrij]-mod
  …   …

Rufus *keek stoïcijns* voor zich uit. (Vermeer-Apres-ski:1105)
Rufus *looked* ahead *stoically*.
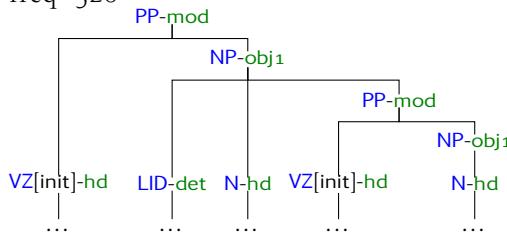
  b. $r = 0.412$; PP with conjunction; max freq=5; sum freq=206

PP-mod
           CONJ-obj1
VZ[init]-hd   LET…   NP-obj1
   …   …   …

Ik verwarmde hem *tegen mijn hart*, *mijn mond*. (Durlacher-Held:1611)
I warmed him *against my heart*, *my mouth*.

  c. $r = 0.352$; nested chain of PP, NP, PP, NP; max freq=8; sum freq=520

PP-mod
     NP-obj1
              PP-mod
                    NP-obj1
VZ[init]-hd LID-det N-hd VZ[init]-hd N-hd
   …   …   …   …   …

'Eén fles maar?' vroeg de auteur *van Het streven naar geluk*, terwijl hij zijn nek strekte naar het etiket. (Houellebecq-KaartEnGebied:1508)
'Just one bottle?' asked the author *of The striving for happiness*, while he stretched his neck towards the label.

The first fragment captures the pattern that is advised against with the slogan *show, don't tell*. The verb-adverb pattern spells out that something is said or done in a certain way, instead of leaving it up to the context to suggest. The second and third fragments are correlated with higher literary ratings and represent arguably more complex syntactic structures, through conjunction and nesting. Note how the second fragment includes an incomplete constituent, indicated by LET…; this is made possible by the binarization scheme.

Most of the fully lexicalized fragments consist of 1 or 2 words, with 18 fragments of 3 words. Finding larger lexicalized fragments such as multi-word expressions would require lifting the frequency thresholds, or obtaining candidate expressions from an external corpus or list, as was done in Section 6.2. The 3-word fragments are all fixed multi-word expressions, here is the top 4:
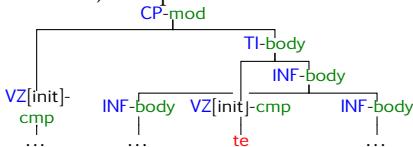
(12) a. r= 0.195: voor het eerst (first time)
      b. r= 0.175: bij wijze van (by way of)

c. r=-0.156: achter de rug (over with)
d. r= 0.156: in de lucht (in the air)

The following shows some of the discontinuous fragments. Note that not all of these fragments show a crossing branch; those fragments would show a crossing branch once they are part of a complete tree. The discontinuous parts are emphasized.
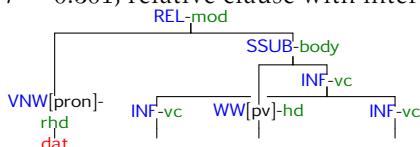
(13) a. *r* = 0.355; complementized to-infinitive clause

```
                    CP-mod
              ┌───────┴───────┐
              │           TI-body
              │         ┌─────┴──────┐
              │         │        INF-body
              │         │      ┌────┴─────┐
          VZ[init]-   INF-body VZ[init]-cmp  INF-body
            cmp        │        │           │
            …          …        te          …
```

'Bent u niet bang om te worden afgebrand *door u met mij* te asso-
ciëren?' (Houellebecq-KaartEnGebied:1280)
'Are you not afraid to be burned *by associating with me*?'
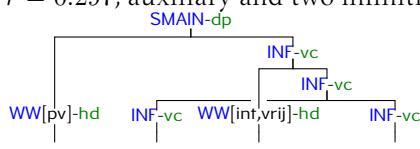
b. *r* = 0.301; relative clause with interrupted infinitive clause

```
                    REL-mod
              ┌───────┴───────┐
              │          SSUB-body
              │        ┌─────┴──────┐
              │        │         INF-vc
              │        │       ┌────┴─────┐
        VNW[pron]-   INF-vc  WW[pv]-hd   INF-vc
           rhd        │        │          │
           dat        …        …          …
```

Jacob dacht vooral aan Adriana en zijn bezoek, dat *nu een andere
lading* zou *krijgen*. (Smit-Vloed:1900)
Jacob mainly thought about Adriana and his visit, which would
now be seen in a different light.
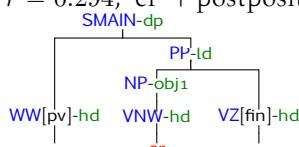
c. *r* = 0.297; auxiliary and two infinitives

```
                   SMAIN-dp
             ┌───────┴──────┐
             │           INF-vc
             │         ┌────┴─────┐
             │         │        INF-vc
             │         │      ┌───┴────┐
        WW[pv]-hd   INF-vc  WW[inf,vrij]-hd  INF-vc
           │         │        │              │
           …         …        …              …
```

Nou ja het is mijn eigen schuld, ik had *eerder op uw mails* moeten
reageren. (Houellebecq-KaartEnGebied:1326)
Well it is my own fault, I should have *replied to your mails sooner*.

d. *r* = 0.294; 'er' + postposition

```
                SMAIN-dp
           ┌───────┴──────┐
           │            PP-ld
           │         ┌────┴─────┐
           │      NP-obj1       │
           │         │          │
      WW[pv]-hd   VNW-hd     VZ[fin]-hd
          │         │           │
          …         er          …
```
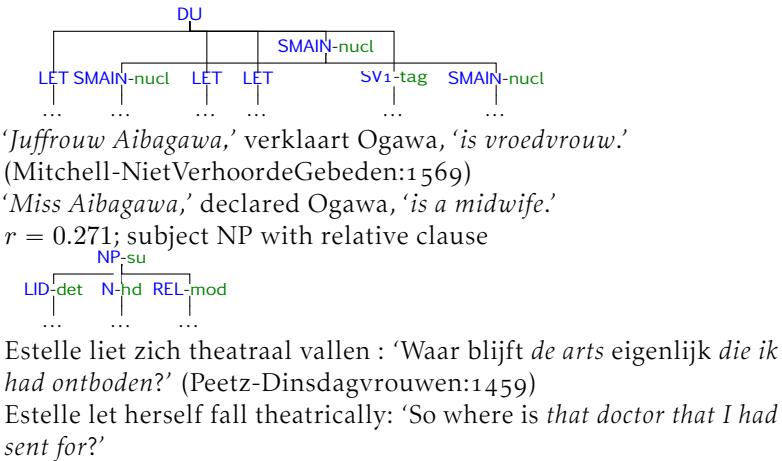
Ze gingen van boord, Diederik *zette er* gelijk de pas *in*. (Rosenboom-
ZoeteMond:1931)
They disembarked, Diederik immediately *started walking rapidly*.

e. *r* = 0.273; direct speech with medial tag sentence

DU
SMAIN-nucl
LET SMAIN-nucl LET LET SV1-tag SMAIN-nucl
... ...

'*Juffrouw Aibagawa*,' verklaart Ogawa, '*is vroedvrouw.*'
(Mitchell-NietVerhoordeGebeden:1569)
'*Miss Aibagawa*,' declared Ogawa, '*is a midwife.*'

f. $r = 0.271$; subject NP with relative clause

NP-su
LID-det N-hd REL-mod
...   .

Estelle liet zich theatraal vallen : 'Waar blijft *de arts* eigenlijk *die ik
had ontboden*?' (Peetz-Dinsdagvrouwen:1459)
Estelle let herself fall theatrically: 'So where is *that doctor that I had
sent for*?'

It is striking that in most of the examples, the discontinuity is not present
in the English translation; nor, perhaps, in the original language. Regardless of
this, the discontinuity need not be inherent for it to be a discriminating feature—
the fragments are still indicative of higher complexity because of verb clusters
and relative clauses. Moreover, it turns out that the majority of discontinuous
fragments has a positive correlation with the literary ratings.

7.3.7 Quantitative analysis of selected fragments

Figure 7.13 shows a breakdown by fragment size (defined as number of non-
terminals), distinguishing fragments that are positively versus negatively corre-
lated with the literary ratings.

Note that sizes 1 and 3 are special cases corresponding to lexical and bi-
nary grammar productions, respectively. The fragments with 2, 4, and 6 non-
terminals are not as common because an even number implies the presence of
unary nodes. Except for fragments of size 1, the frontier of fragments can consist
of either substitution sites or terminals (note that the latter are not counted to-
wards the size). On the one hand smaller fragments corresponding to one or two
grammar productions are most common, and are predominantly positively cor-
related with the literary ratings. On the other hand there is a significant negative
correlation between fragment size and literary ratings ($r = -0.2, p = 3.3 \times 10^{-73}$);
i.e., larger fragments tend to be negatively correlated with the literary ratings.

It is striking that the number of positively correlated fragments is larger than
the number of negatively correlated fragments. There are less literary novels in
the corpus, so they might be expected to contribute proportionally less fragments
compared to the less literary novels. Additionally, from the breakdown by size
it is clear that there are more positively correlated fragments because of a large
number of small fragments of size 3 and 5; however, combinatorially, the number
of possible fragment types grows exponentially with size (and this is reflected
in the initial set of recurring fragments), so larger fragment types would be
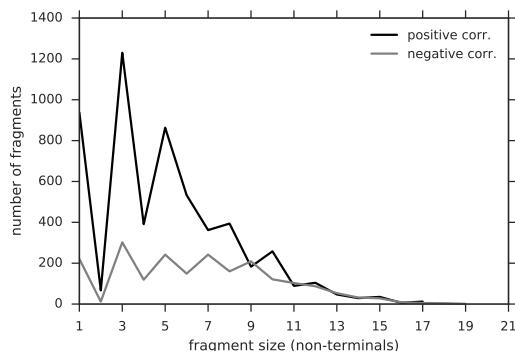
Figure 7.13: Breakdown by fragment size (number of non-terminals).

expected to be more numerous. In effect, the selected negatively correlated fragments ignore this distribution by being relatively uniform with respect to size, while the literary fragments actually show the reverse distribution.

To investigate this, Figure 7.14 shows the distribution of fragments at the various steps of the fragment selection process. The plots on the left show that the peak in small positively correlated fragments is introduced when adding the frequency > 50 threshold, and becomes more pronounced when fragments are selected by correlation. In the last step (redundancy removal) the negatively correlated fragments also become relatively uniformly distributed with respect to size. The plots on the right show a histogram with respect to correlation. The curve for negatively correlated fragments is mirrored (i.e., the absolute value of the correlation coefficient is shown) to highlight the asymmetry in the distribution.

What could explain the peak of positively correlated, small fragments? At first sight, the following seems plausible:

HYPOTHESIS 1: There is a larger number of types of positively correlated fragments, with a smaller set of more frequent negatively correlated fragments.

However, when the fragments with 3 non-terminals are histogrammed in terms of frequency, the positively correlated fragments are more numerous across all frequencies (cf. Figure 7.15). The following is an alternative hypothesis:

HYPOTHESIS 2: Literary language invokes a larger set of syntactic constructions when compared to the language of non-literary novels, and therefore more variety is observed in the fragments that are correlated with literature.

This is supported by the tree fragments discussed up to now. In order to investigate the peak of small fragments, we inspect the 40 fragments of size 3
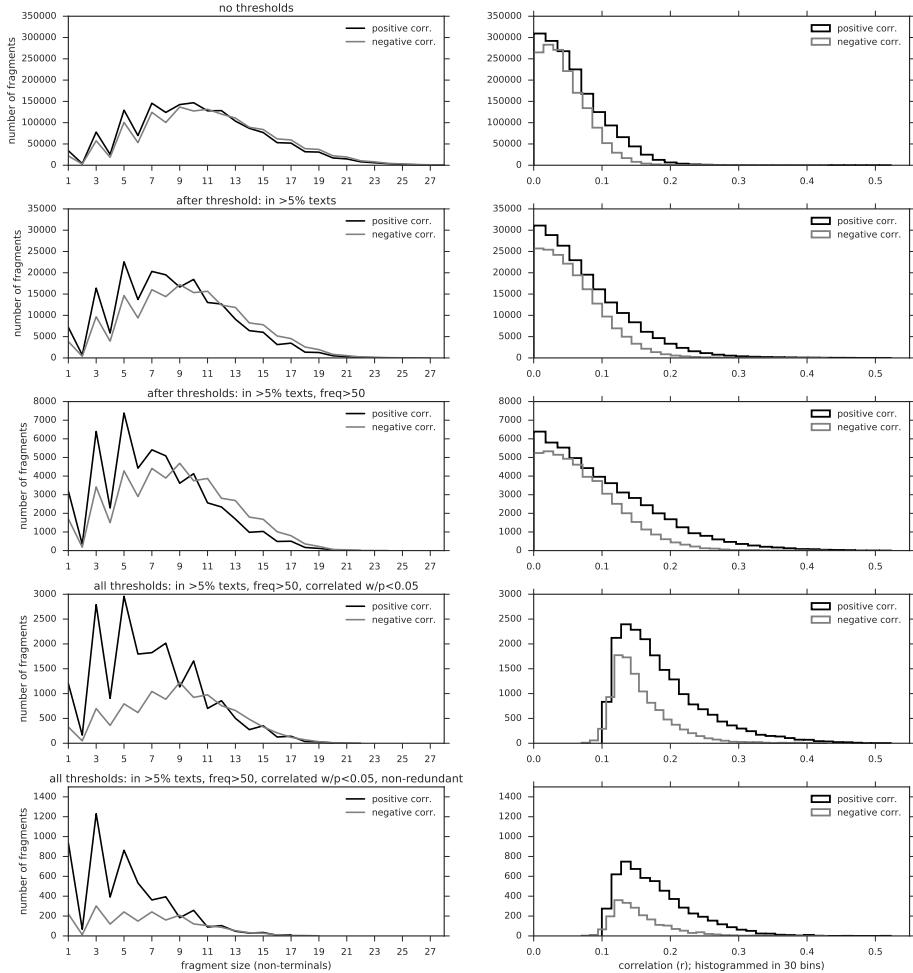
Figure 7.14: Left: Breakdown by fragment size (number of non-terminals); right: histogram by correlation. Each row corresponds to an additional step in the fragment selection.
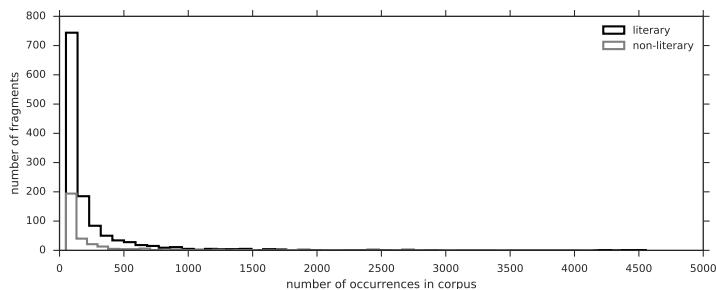
Figure 7.15: A histogram of the frequencies of fragments with 3 non-terminals.
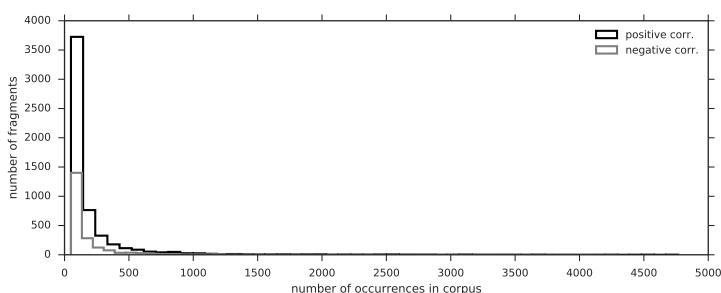
Figure 7.16: Breakdown by fragment frequency.

with the highest correlations. These fragments contain indicators of unusual or more complex sentence structure:

- DU, dp: discourse phenomena for which no specific relation could be detected (e.g., discourse relations beyond the sentence level).
- NPs in appositive relation, e.g., 'John the artist.'
- a complex NP, e.g., containing punctuation, or NPs and NPs.
- an NP containing an adjective used nominally or an infinitive verb.

On the other hand, most non-literary fragments are top-level productions containing ROOT or clause-level labels, for example to introduce direct speech.

Another way of analyzing the selected fragments is by frequency. When we consider the total frequencies of selected fragments across the corpus, there is a range of 50 to 107,270. Figure 7.16 shows a histogram of the total frequencies. The histogram has been truncated at a frequency of 5000, to highlight the shape for the lower frequencies. The bulk of fragments have a low frequency (before fragment selection 2 is by far the dominant frequency), but the tail is very long. Except for the fact that there is a larger number of positively correlated fragments, the histograms have a very similar shape.

Lastly, Figure 7.17 and 7.18 show a breakdown by the syntactic categories
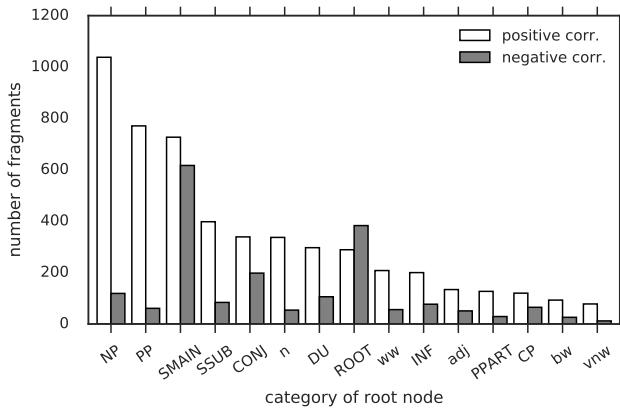
Figure 7.17: Breakdown by category of fragment root (top 15 labels).
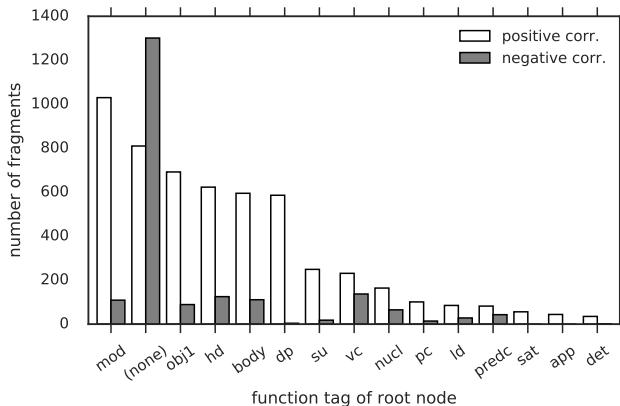
Figure 7.18: Breakdown by function tag of fragment root (top 15 labels).
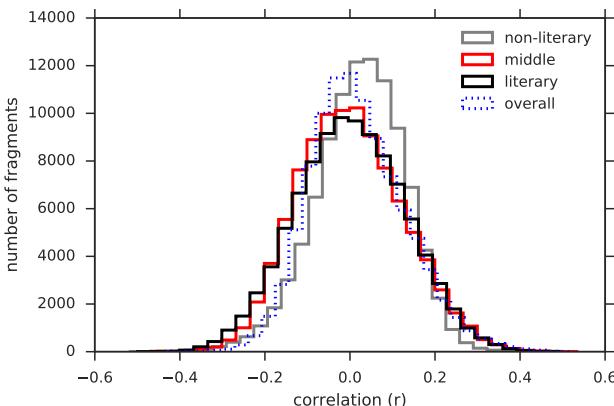
Figure 7.19: Histograms of the correlations of fragments for different subsets of the novels.

and function tags of the root node of the fragments. The positively correlated fragments are spread over a larger variety of both syntactic categories and function tags. This means that for most labels, the number of positively correlated fragments is higher; the exceptions are ROOT, SV1 (a verb-initial phrase, not part of the top 15), and the absence of a function tag (indicative of a non-terminal directly under the root node). All of these exceptions point to a tendency for negatively correlated fragments to represent (templates of) complete sentences.

Up to now we have been considering positively and negatively correlated fragments. It may be tempting to think of these as literary and non-literary fragments, but strictly speaking, this is not correct. The correlation only tells us whether we expect the literary ratings to increase or decrease as a function of the frequency of a given fragment. The strength of the correlation does not tell us about scale or slope, i.e., whether the fragment is useful to predict a broad difference between literary and less literary novels, or a difference on a smaller scale such as between low brow and middle brow novels in terms of literary ratings.

To investigate this, we divide the set of novels into three categories: non-literary (mean rating < 4), literary (rating > 5), and middle brow (the rest). Figure 7.19 shows a histogram of the fragments with respect to the correlations to each of these subsets. The middle and literary subsets show a very similar distribution. The non-literary subset is markedly different, showing a higher peak for fragments with no correlation, and relatively less negatively correlated fragments compared to the other subsets. This means that there is a subset of fragments which is indifferent to the literary ratings of non-literary novels; these fragments therefore correlate specifically with differences in literary ratings in the range 4–7.

	R^2	Kendall τ	RMS error
Fragments	55.5 (2.5)	0.511 (0.023)	0.668 (0.17)
Bigrams	56.5 (2.6)	0.532 (0.026)	0.66 (0.174)
Combined	57.4 (2.7)	0.532 (0.033)	0.654 (0.173)
Interpolated	58.0 (2.4)	0.538 (0.025)	0.649 (0.165)

Table 7.11: Regression evaluation of predicting literary ratings with fragments and bigrams. Each score is the mean over 5-fold cross validation, with the standard error in parentheses.

7.3.8 REGRESSION RESULTS

We perform 5-fold cross-validation with linear support vector regression. Feature counts are transformed to *tf-idf* weights. The model contains two hyperparameters: C determines the regularization, and ϵ is a threshold beyond which predictions are considered good enough during training. We tune these parameters on a development set and settle for $C = 100$ and $\epsilon = 0$.

In order to contrast the performance of tree fragments with a simple but strong baseline, we consider bigram features as well. Bigrams present a good trade off in terms of informativeness (a bigram frequency is more specific than the frequency of an individual word) and sparsity (three or more consecutive words results in a large number of n-gram types with low frequencies). This is supported by the results of Wang and Manning (2012), who find that in sentiment classifications tasks, word bigrams are superior to unigrams and trigrams.

We train separate models on both types of features. Training a single model on the combined set of features does give an improvement, although difference is slight when considering the cross-validation standard error. Interpolating the predicted values of the fragment and bigram models yields an even better score; i.e., we use the predictions of separately trained models, and combine them by taking the arithmetic mean. Cf. Table 7.11 for the scores and Figure 7.20 for a visualization of the predictions using fragments and bigrams in a scatter plot.

Notice that the scatter plot shows that the model has a bias in that the predictions for highly literary novels tend to be too low, and *vice versa* for non-literary novels. Consider that there are always many features with both positive and negative weights affecting the resulting prediction, so that on average, the prediction tends toward the mean, which is a conservative default. In the pathological case, when the features are completely unhelpful in predicting the target, all predictions will simply equal the mean; in the ideal case, the features give enough information so that the prediction error tends to zero. The bias in Figure 7.20 is therefore an indication of the degree to which the features reflect the target variable.

Another issue is that the corpus consists of a few prominent genres (thriller,
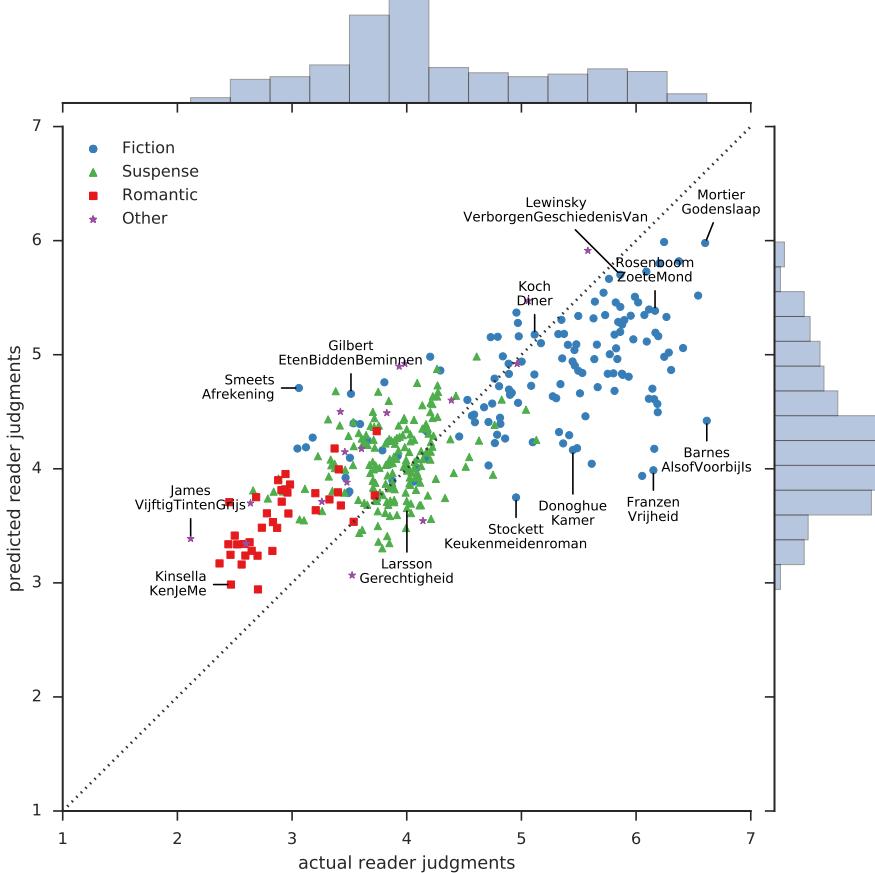
Figure 7.20: A scatter plot of regression predictions with respect to the actual literary ratings. Showing interpolated predictions using bigrams and fragments.
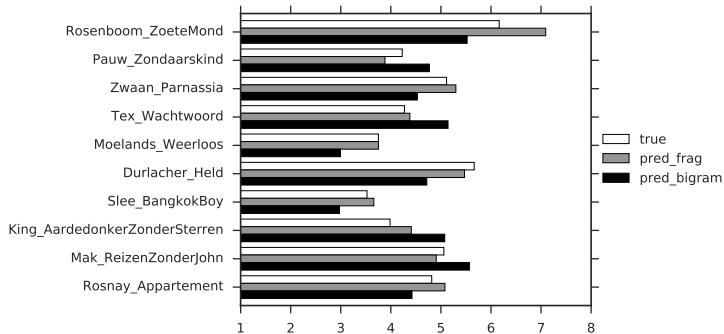
Figure 7.21: The ten novels for which the prediction with fragments differs the most from the prediction with bigrams.
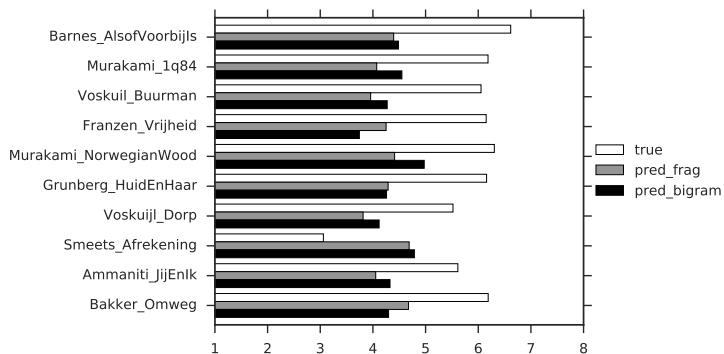

Figure 7.22: The ten novels with the largest prediction error (using both fragments and bigrams).

romantic fiction, general fiction). Figure 7.20 shows that genre is highly correlated with the literary ratings. The model is therefore likely picking up on genre markers. On the other hand, within the general fiction category, which contains the literary novels, the model is also correctly predicting some of the higher literary ratings. It would be interesting to incorporate the genre categories directly into the model, or to train the model on just the literary novels, especially if more data were available.

To analyze the difference in predictions between the two feature sets, we consider the top 10 novels for which the predictions diverge the most; cf. Figure 7.21. The prediction using fragments is too high for 4 novels, compared to 4 with bigrams (of which one overlaps). The prediction with fragments is too low with 3 novels, versus 6 times with bigrams (of which one overlaps). The prediction with fragments is very close (< 0.15) in case of Tex, Moelands, and Slee; the bigrams have no close prediction in the ten novels.

Figure 7.22 shows a similar bar plot with the ten novels with the largest prediction error when the predictions of the fragment and bigram models are interpolated. Of these novels, 9 are highly literary, which are underestimated by the model. For the other novel (Smeets-Afrekening) the literary rating is overestimated by the model. Since this top 10 is based on the mean prediction from both models, the error is large for both models. This does not change when the top 10 errors using only fragments or bigrams is inspected; i.e., the hardest novels to predict are hard with both feature types.

What could explain these errors? At first sight, there does not seem to be an obvious commonality between the literary novels that are predicted well, or between the ones with a large error. For example, whether the novels have been translated or not does not explain the error. A possible explanation is that the successfully predicted literary novels share a particular (e.g., rich) writing style that sets them apart from other novels, while the literary novels that are underestimated by the model do not distinguish themselves in this way by their writing style. It is difficult to confirm this directly by inspecting the model, since each prediction is the sum of several thousand features, and the contributions of these features form a long tail. If we define the contribution of a feature as the absolute value of its weight times its *tf-idf* value in the document, then in case of Barnes-AlsofVoorbijIs, the top 100 features contribute only 34 % of the total prediction.

An alternative is to look at the baseline features introduced in Section 6.1. If we take the top 4 literary novels with the largest error and contrast them with 4 literary novels which are well predicted, we get the results shown in Figure 7.23. The most striking difference is sentence length: the underestimated literary novels have shorter sentences. Voskuil and Franzen have a higher proportion of direct speech (they are in fact the only literary novels in the top 10 novels with the most direct speech). Lastly, the underestimated novels have a higher proportion of common words (lower vocabulary richness). These observations are compatible with the explanation suggested above, that a subset of the literary novels share a simple, readable writing style with non-literary novels. Such a style may be more difficult to detect than a literary style with long and complex sentences, or rich vocabulary and phraseology, because a simple, well-crafted sentence may not offer overt surface markers of stylization.

To gain insight in how the performance is affected by the amount of data and the number of features, we can look at learning curves. Figure 7.24 shows learning curves for restricting the number of novels in the training set and for restricting the number of features that are used. In both cases the novels or features are shuffled once, so that initial segments represent random samples. The novels and features are sampled in 5 % increments (i.e., 20 models are trained in each case). The graphs show the cross-validated scores; the training scores were also computed, but these are of limited utility since the performance is unrealistically good (consistently 99 %). Consider that a trivial memory-based model can simply store all training examples and attain a 100 % score; given this
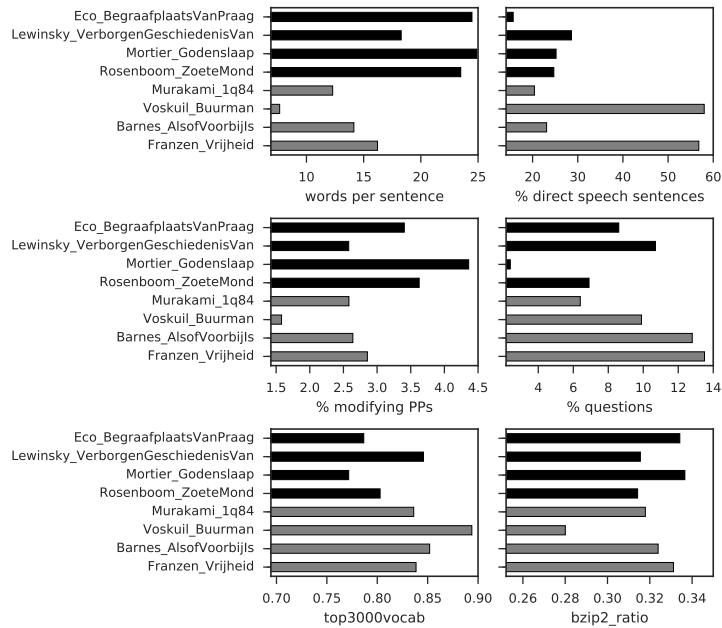
Figure 7.23: Comparison of baseline features for literary novels with good (black) and bad (gray) predictions. Note that the x-axis does not start at 0, to highlight the differences.
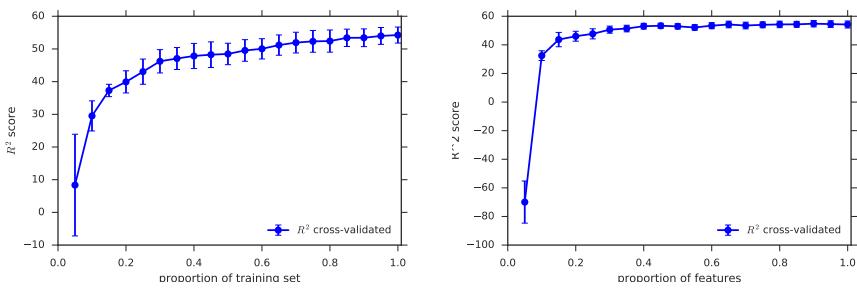

Figure 7.24: Learning curve when restricting: training set (left), features (right). The error bars show the standard error.

possibility, the training score is no more than a soundness check of the learning algorithm.

The graphs show that increasing the number of novels has a large effect on performance. The curve is steep up till 30 % of the training set, and the performance keeps improving steadily but more slowly up till the last data point. Since the performance is relatively flat starting from 85 %, we can conclude that the k-fold cross-validation with $k = 5$ provides an adequate estimate of the model's performance if it were trained on the full data set; if the model was still gaining performance significantly with more training data, the cross-validation score would underestimate the true prediction performance.

For the number of features the story is different. The performance at 40 % is already high with an R^2 of 53.0 %, and grows more slowly from that point. Curiously, the performance drops slightly after this point, and picks up again at 60 % of features. This indicates that the feature selection is not optimal, although there probably is not much to be gained from tuning with this set. It is likely that similar performance can be attained with a much smaller feature set.

7.3.9 PREDICTIVE FEATURES

The following lists the top 10 features with the highest absolute weight in the SVM model trained on folds 1–4. Negative weights indicate fragments predictive of less literary novels and *vice versa*; unlike with correlation coefficients, there is no fixed range of $[-1, 1]$.
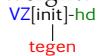
(14) a. weight: 1.75585; corr: 0.246439; 'book'

N-hd
|
boek

Ze sloeg het boek dicht. (Siebelink-LichaamVanClara:1213)
She closed the book.
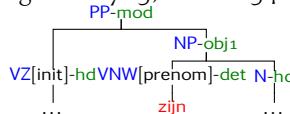
b. weight: 1.92342; corr: 0.244592; preposition 'against', 'to'.

VZ[init]-hd
|
tegen

'Ongure slaaf,' zei ze tegen me. (Irving-InMens:1246)
'Loathsome slave,' she said to me.
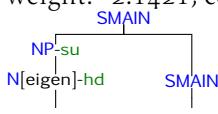
c. weight: 2.0763; corr: 0.340082; preposition + 'his' + noun

PP-mod
VZ[init]-hd VNW[prenom]-det N-hd
NP-obj1
... zijn ...

In zijn wimpers hangen klonters. (Beijnum-SoortFamilie:1356)
His eye lashes contain lumps.

d. weight: -2.1421; corr: -0.281452; name + finite verb

SMAIN
NP-su
N[eigen]-hd SMAIN
... ...
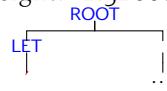
Jeffrey knikte weer. (Slaughter-Onaantastbaar:1197)
Jeffrey nodded again.

e. weight: -2.27645; corr: -0.346132; intensifier (really)

ADJ[vrij]-
mod
echt
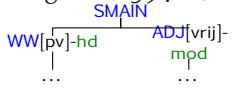
'Dit gaat niet echt goed.' (Mansell-DrieIsTe:1582)
'This is not going well at all.'

f. weight: -2.32608; corr: -0.391185; direct speech

ROOT
LET
…
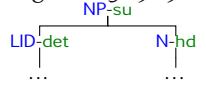
'Ja hè? (Mansell-SmaakTePakken:1706)
'Isn't it?

g. weight: -2.39402; corr: -0.493574; verb + postverbial adjective

SMAIN
WW[pv]-hd   ADJ[vrij]-
…   mod
            …
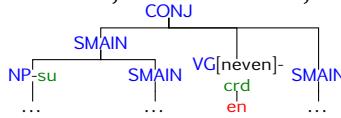
Ik moet plotseling weg. (Wickham-Vraagprijs:1728)
I suddenly get the urge for going.

h. weight: 2.58989; corr: 0.456933; determiner + noun subject

NP-su
LID-det   N-hd
…   …
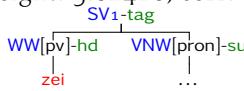
De sjah zweeg. (Abdolah-Koning:1183)
The sjah remained silent.

i. weight: -2.71746; corr: -0.363539; sentence conjunction where
second conjunct has no subject

CONJ
SMAIN
NP-su   SMAIN   VG[neven]-   SMAIN
…   …   crd   …
                  en

De chauffeur lacht schaterend en trekt op. (Vlugt-LaatsteOffer:1340)
The chauffeur laughs out loud and takes off.

j. weight: 3.02426; corr: 0.167564; tag question 'he said'

SV1-tag
WW[pv]-hd   VNW[pron]-su
zei   …

'Jezus,' zei ze. (Patterson-Partnerruil:1046)
'My God,' she said.

We can validate the feature selection by comparing the relevancy metric
(correlation) to the model weights. Figure 7.25 shows a hexbin plot (i.e., a
combination of a scatter plot and histogram) of these two variables. There is a
relatively strong and significant correlation (r=0.52) between these variables;
that is, in the majority of cases the feature weight has the same sign as the
correlation, and the magnitude of the correlation is related to the feature weight
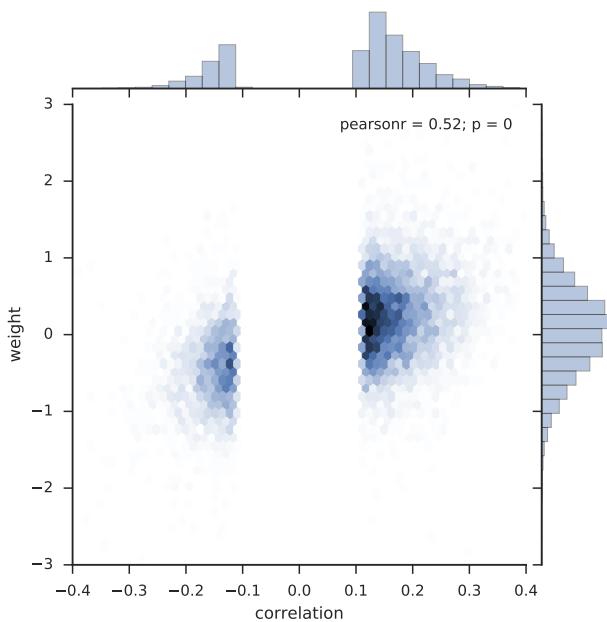
Figure 7.25: Hexbin plot of fragment correlation versus feature weight in the predictive model.

as well.

7.3.10 Summary

> As data coverage grows, some may worry that models of syntax will be superseded by better n-gram models. This study suggests that hierarchic syntax retains its value even in a world of big data.
>
> — van Schijndel and Schuler (2015, p. 1604), Hierarchic syntax improves reading time prediction.

We have presented a fully data-oriented approach to mining syntactic tree fragments for a predictive model. Although we did not use our discontinuous parser directly for parsing literature, we did exploit its capability to extract discontinuous fragments of arbitrary size. The fragments are not extracted from an external data set, but mined from the training data itself, taking the target variable into account. The data-oriented fragment mining approach comes close to the level of accuracy of the bigram baseline model, while offering more possibilities for inspection and interpretation.

An individual tree fragment is easier to interpret than a bigram feature, especially with larger fragments and specific syntactic labels. However, interpreting the decisions of a model with thousands of features remains difficult, as the predictions are the result of the interaction of each of these features.

Looking at the positively and negatively correlated tree fragments, as well as comparing the novels that are easy and difficult to predict, supports the hypothesis that literary language, especially the kind that the model picks up well, tends to use a larger set of syntactic constructions than the language of non-literary novels.

The model has certain restrictions. The fragments we have considered are relatively frequent, which was motivated by the choice of extracting predictive fragments from the corpus itself. With a different method or an external data set, rarer fragments may be exploited, such as larger multi-word expressions (as seen in Section 6.2). A more fundamental limitation is the bag-of-features representation. Only the total number of occurrences of each feature in a text is considered, without regard for the order and context in which those features occur. However, overcoming this is difficult and part of the success of simpler models is due to ignoring much of this complexity.

Still, a clear contribution of our model is that it includes rich syntactic information such as non-local dependencies and function tags, and exploits recurring syntactic and lexical patterns of arbitrary size. This information allows the syntactic fragments to capture phenomena that are not captured by Bag-of-Words models.
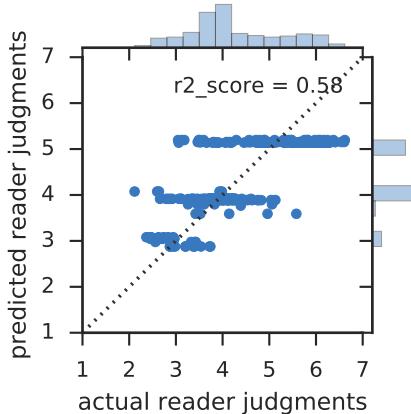
Figure 7.26: Predictions using only genre as feature.

7.4   Putting it all together

The strategy of combining features can be extended further. Table 7.12 presents results with an ensemble of both simple and computationally intensive features, as well as three metadata variables. The features are combined into a ridge regression model, with the same 5-fold cross-validation as in the previous section. The results are presented incrementally, to illustrate the contribution of each feature relative to features before it. Figure 7.27 shows a visualization of the predictions in a scatter plot. cliches is the number of cliché expressions in the texts (Section 6.2). topics is a set of 50 LDA topic weights induced from the corpus (Section 6.3). fragments and bigrams are the predictions from the models of the previous section. We also include three (categorical) metadata features not extracted from the text, but directly related to the book in question: Translated, Author gender and Genre. Genre is the coarse genre classification Fiction, Suspense, Romantic, Other; Genre alone is already a strong predictor, with an R^2 of 58.3 on its own. However, this score is misleading, because the predictions are very coarse due to the discrete nature of the feature; cf. Figure 7.26. Another striking result is that the metadata variables Author gender and Translated increase the score, but only when they are both present.

	RMS error	R^2
mean sent. len.	0.916	16.4
+ % direct speech sentences	0.880	22.9
+ top3000vocab	0.876	23.5
+ bzip2_ratio	0.871	24.4
+ cliches	0.839	29.9
+ topics	0.693	52.2
+ bigrams	0.647	58.2
+ % modifying PPs	0.645	58.6
+ avg. dependency length	0.656	57.1
+ fragments	0.636	59.7
+ Genre	0.511	74.0
+ Translated	0.513	73.8
+ Author gender	0.490	76.1

Table 7.12: Incremental results with an ensemble of features. The divisions indicate the grouping of features into: Lexical/General, Syntactic, Metadata.
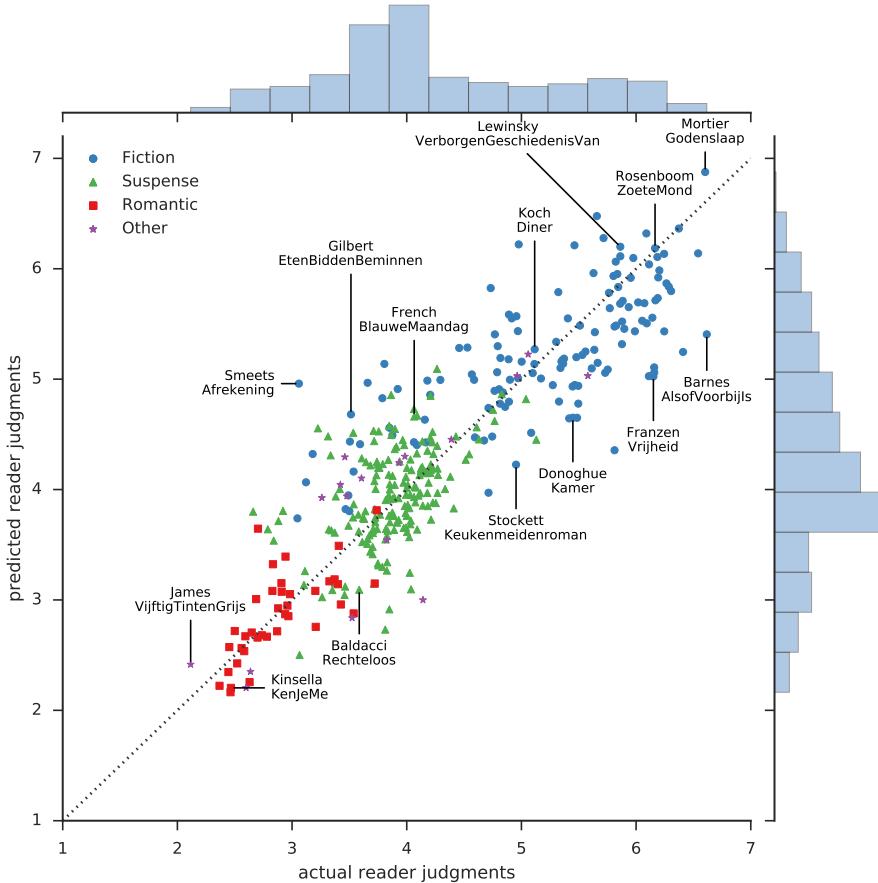
Figure 7.27: A scatter plot of regression predictions with respect to the actual literary ratings, using the ensemble model.

Novels with original or translated title: Kinsella, *Remember me?*; James, *50 shades of grey*; Gilbert, *Eat pray love*; French, *Blue monday*; Smeets, *Redemption*; Koch, *The dinner*; Lewinsky, *Johannistag*; Rosenboom, *Sweet tooth*; Mortier, *While the gods were sleeping*; Barnes, *Sense of an ending*; Franzen, *Freedom*; Donoghue, *Room*; Stockett, *The help*; Baldacci, *Hell's corner*.

Conclusion

To recapitulate, we have developed richer Data-Oriented Parsing models and applied them to the modeling of literary language.

T HIS CONCLUDES our investigation into syntax and literature. We have shown that rich syntactic analyses can be learned from data, and are useful in characterizing a particular kind of language, literature. A common thread in this thesis has been the use of syntactic tree fragments, and more specifically, syntactic trees enriched with non-local and functional relations. We have shown how to extract such fragments from large treebanks, and applied them to establish the following.

Our efficient method for recurring fragment extraction enables applications on larger corpora. We have shown how to parse with discontinuous tree-substitution grammars and presented a practical implementation. These components can be combined to form a compact, efficient, and accurate data-oriented parsing model that produces discontinuous constituents and function tags.

Procedures for automatically inducing grammars from data can be adapted to yield more detailed analyses. If this is done carefully, this may be done without requiring more powerful formalisms. Specifically, we showed that the question of finding the formalism that exactly fits the desired generative capacity can be sidestepped. Similarly, the choice between constituency and dependency representations is unnecessary, since the inclusion of discontinuous constituents and function tags offers the strengths of both.

A limitation that remains is the reliance on configuration. Tree fragments capture a particular realization of a linguistic construction, with a fixed word order and morphological inflection. The way that the tree-substitution grammar is induced and employed in parsing relies on exact, all-or-nothing matching of structure and non-terminal labels. It remains a challenge to overcome this in a data-driven fashion.

In the second part of this thesis we clearly established that literature can be recognized from textual features, and the predictions are quite successful. Our experiments give an estimate to what degree the text contributes to the literary character of a novel. The estimate is a lower bound: much more sophisticated textual analyses may be performed, which could explain an even

larger proportion of the variation in literary appreciation. It is impressive how much we already managed to explain using stylistic measures and markers. While it is not possible to assign these textual features as the cause of the ratings from the survey participants, this result clearly rules out the notion that these value-judgments of literary merit were arbitrary, or predominantly determined by factors beyond the text.

Still, it is obvious that many aspects have not been taken into account. Our focus has been on writing style, exploiting surface structure or syntactic aspects that are still relatively close to it. This should be expanded to included effects beyond the sentence level, and to include more fine-grained effects than the occurrence count of features across the novels. Deeper analysis could consider the narrative and characters of the novels, or more nuanced textual aspects such as the implicatures and aesthetics of the language.

Much remains to be done.

A The corpus of novels

A.1 Preprocessing the corpus

Natural Language Processing (NLP) tools require texts to be preprocessed and cleaned to various degrees. This chapter describes the automated processing steps to clean up and identify paragraph, sentence, and word boundaries in texts from a collection of Dutch ebooks in various source formats.

A.1.1 Conversion to plain text

Although the texts may contain markup such as italics, bold, and underlined text, the NLP tools only handle plain text, and this markup must be discarded. Note that emphasis expressed using all capitals or accents is preserved.

The following shows the tools and command line options used to convert texts to plain text. We assume a file name without extension given in the variable $a.

EPUB

For the epub format we use Calibre.[1] The conversion proceeds as follows:

```
ebook-convert $a.epub $a.txt
```

Note that we deliberately do not enable -heuristic-processing, since we will fix various issues by exploiting linguistic knowledge not available to Calibre.

WORD

We use antiword[2] to convert files in Microsoft Word format. The antiword utility offers to option to output text with a single paragraph per line, which helps to avoid mistakes in paragraph identification later on:

```
antiword -w o $a.doc > $a.txt
```

1 Calibre 2.5.0, cf. http://calibre-ebook.com/
2 antiword 0.37, cf. http://www.winfield.demon.nl/index.html

PDF

Since pdf is a page-oriented format, this format requires the most care. Heuristic processing is required to reconstruct paragraphs by removing line breaks and hyphenation. Some extraneous material, such as headers and footers, can by filtered at the outset by specifying a cropping area. Inspecting some of the pdf files shows that they include headers and footers in a so-called 'bleed' area that is trimmed from the final printed book. This area is about 30 Postscript points on each side (roughly 1.06 cm).

For pdf we use the pdftotext utility.[3] We pass an option to preserve as much of the original layout as possible in the output, which helps further processing. The crop area is specified relative to the page size $width and $height:

```
pdftotext -layout -r 72 -x 30 -y 30 -W $[ $width - 60]
          -H $[ $height - 60 ] $a.pdf $a.txt
```

A.1.2   Normalization of punctuation and other special characters

Unicode punctuation is collapsed into matching ASCII versions; e.g., left and right quotes are converted to plain single or double quotes; various dashes such as em- and en-dashes are converted to plain hyphens with added space around them, while variants of hyphens are converted to hyphens without extra white space. Ligatures (e.g., 'ff' as single character) are expanded into ASCII characters. So-called discretionary or soft hyphens and zero width spaces are removed.

Two single quotes are replaced by one double quote; this means that the distinction between single and double quotes is maintained, but fixes possible OCR errors where double quotes are represented as two single quotes. Space after sentence-ending punctuation is ensured.

Whenever the Dutch genitive contraction for *des* (*'s*) is detected, it is ensured that it forms a separate token (e.g., *'s middags*). Conversely, where the plural marker *'s* is preceded by an acronym, any white space in between is removed (e.g., *de SUV 's*).

Characters denoting scene breaks are removed. Sequences of empty lines are reduced to a single empty line.

For dashes that are at the start of a line (such as when dashes indicate dialogue) or preceded by a space, a space is appended, to ensure that it forms a separate punctuation token. This does not affect dashes in conjunctions (e.g., *linker- en rechterzijde*).

A.1.3   Page numbers, running heads

The running heads (containing file names, dates, chapter name) in the corpus have already been taken care of by cropping during the pdf conversion. Page numbers are not in a fixed location, so cropping them would require tuning the

---

3 pdftotext 0.26.5, cf. http://poppler.freedesktop.org

cropping parameters by hand for each file. It turns out that it is easier to remove them by regular expressions.

Page numbers are defined as lines with only numbers and white space. If more than 50 such numbers are detected, they are removed; this threshold is applied to preserve chapter numbers in a file that does not contain page numbers.

The white space around page numbers is replaced by a single new line, such that later on it can be decided whether their surrounding lines form a paragraph or contain a paragraph break coinciding with a page break.

A.1.4 Hyphenation

Hyphenation at the end of line can usually be removed, but not always. A compound that needs to be hyphenated regardless may end up at the end of a line:

Amsterdam is not the capital of North- Holland.	⇒	. . . North-Holland . . .

Moreover, since potential hyphens are sometimes marked even when not at the end of a line, every single hyphenation character needs to be reconsidered, not just at the end of lines. Based on previous work, a dehyphenation strategy based on word counts with and without hyphens is used (Bauge, 2012). These word counts could be based solely on the document being processed, but this will include hyphenated word forms, and may not have enough coverage (the dehyphenated form may be less frequent or may not even occur in the text).

Therefore, the dictionary is augmented with word counts based on a large corpus. In order to avoid making the results for a text dependent on the word counts of other texts, only the word counts of a given text and the reference corpus are used to determine the most likely alternative for each hyphenated token. The reference corpus is the 500 million word Sonar corpus (Oostdijk et al., 2013); supplemented with a word list (without counts, so counts of 1 are assumed) from the Dutch OpenTaal project (version 2.10; cf. http://www.opentaal.org/).

Hyphens bordering on digits are never removed, nor are hyphens between pairs of vowels that form vowel clashes in Dutch. Vowel clashes consist of vowels that form diphthongs; e.g., the hyphen in *zee-egel* will not be considered for removal. When a word contains multiple hyphens, all combinations of hyphens are considered:

Noord-Hol- land	⇒	NoordHolland, NoordHol-land, Noord-Hol-land, Noord-Holland *(most common)*
ge-wel-dig	⇒	geweldig *(most common)*, ge-weldig, gewel-dig, ge-wel-dig
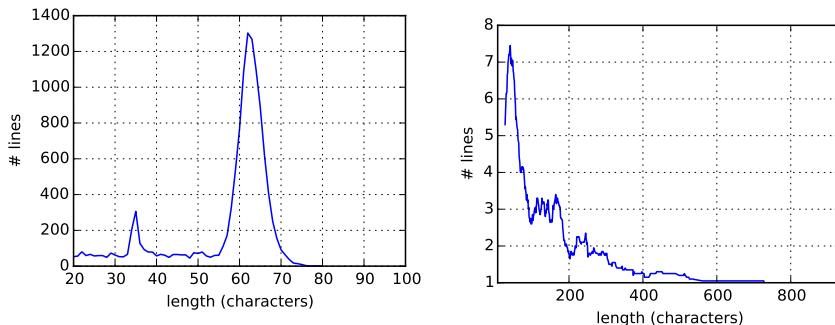
Figure A.1: Comparison of line lengths in a text with (left) and without (right) fixed-width formatting. Plots are smoothed with rolling mean with a window size of 20.

When none of the alternatives appear in the dictionary, the hyphens are left as is, except when the hyphen appears at the end of the line, in which case the default action is to remove the hyphens.

An issue is that the spacing around hyphens is not always correct; sometimes a space is missing before or after, sometimes it should be left out. These cases can be reduced by properly treating (i.e., removing) the aforementioned Unicode characters indicating discretionary hyphens and zero width spaces, but problems remain. Some examples:

REMOVE SPACE, KEEP HYPHEN: cia- directeur ⇒ cia-directeur
REMOVE SPACE & HYPHEN: verschil -lende ⇒ verschillende
A PARENTHETICAL:
    Valmorain wist niet - want hij   ⇒   […] gevraagd - dat […]
    had er nooit naar gevraagd
    -dat in Saint-Lazare […]

On the other hand, some hyphens should appear before or after a space:

CONJUNCTIONS: "linker- en rechterzijde", "CIA-observateurs en -agenten"
INTENSIFIERS: "aller- allerliefste"

Since there is considerable ambiguity in these cases, and the number of occurrences is manageable, a list of manual corrections was compiled. These corrections are applied before any other processing occurs.

A.1.5 PARAGRAPHS

Paragraphs[4] consist of one or more sentences and are the smallest unit of discourse organization in a text.

---

4 NB: the Dutch term for paragraph is *alinea*; the false friend *paragraaf* refers to a section.

```
Weer later trok Dzjengis Khan van het oosten naar het westen,
waarbij hij het land van de Perzen vernielde zodat er niets meer
overbleef van hun oude glorie. Pas in de tijd van de Safaviden
richtte het land zich weer op. Het was slechts voor een korte pe-


                              15
^L
riode van glorie. Het land raakte in verval. De stammen vochten
met elkaar om de macht.

Aan het begin van de negentiende eeuw wist een van die stam-
men de heerschappij te veroveren.
  Dit verhaal gaat over een koning van deze stam: prins Naser.


                              16
^L
                    2. Prins Naser

Er was eens een Perzische prins die later toen hij koning was Pa-
rijs bezocht.
```

Figure A.2: A text fragment spanning multiple pages from Kader Abdolah, *Koning*, before processing. ^L indicates a page break. The vertical white space before page number 16 has been abbreviated.

```
Weer later trok Dzjengis Khan van het oosten naar het westen ,
waarbij hij het land van de Perzen vernielde zodat er niets meer
overbleef van hun oude glorie .  ↵
Pas in de tijd van de Safaviden richtte het land zich weer op .
↵
Het was slechts voor een korte periode van glorie .  ↵
Het land raakte in verval .  ↵
De stammen vochten met elkaar om de macht .  ↵

Aan het begin van de negentiende eeuw wist een van die stammen de
heerschappij te veroveren .  ↵

Dit verhaal gaat over een koning van deze stam :  prins Naser .
↵

2 .  ↵
Prins Naser ↵

Er was eens een Perzische prins die later toen hij koning was
Parijs bezocht .  ↵
```

Figure A.3: The text fragment after processing. Empty lines indicate paragraph breaks, ↵ indicates a sentence boundary, and spaces indicate word/punctuation boundaries.

In preprocessing we aim to preserve paragraphs by formatting the output to contain one paragraph per line. For texts with a page layout that is justified or wrapped at a fixed line length, lines will have a (relatively) fixed width. This applies to texts converted from a page-oriented format such as `pdf`. In these cases paragraphs need to be reconstructed.

We assume that paragraphs breaks are signaled in three ways:

1. Separation by one or more empty lines.

2. Given a text with fixed-width formatting, a paragraph-ending line is shorter than typical lines.

3. Indentation may indicate a paragraph-opening line.

While empty lines can be trivially identified, the lengths of lines and indentation need to be judged heuristically. We use similar thresholds as used by Calibre, but with different methods.

A text is detected as having fixed-width formatting if 80 % of lines are 100 characters or shorter.[5] Due to the use of proportional fonts, the exact number of characters in lines varies, even for justified layouts; some variance has to be accommodated.

If fixed-width formatting is detected, the paragraph detection using method 2 and 3 is enabled and thresholds x and y are computed to detected paragraphs based on these two criteria:

1. a paragraph ends when its length is less than $x - 10$ where x is the smallest line length such that at least 80 % of lines are $< x$.

2. a new paragraph starts when it is indented by y spaces. The value for y is the most common indentation relative indentation length (the indentation of a line minus that of the previous line).

Figure A.1 shows the line lengths in texts with and without fixed width formatting. The peak in the graph on the left indicates a text with fixed width formatting, since the lengths of a majority of lines fall under a small interval.

In order to avoid spurious paragraph breaks due to indentation, it is important that any margins are removed, so that normal lines have 0 indentation. For example, a book may be formatted to have differently sized margins on even and odd pages, but this extra indentation should not be counted as introducing paragraphs. Incidentally, the `pdf` cropping method ensures this is the case.

Figures A.2 and A.3 show an example of paragraph detection and dehyphenation.

---

5 Calibre employs a similar heuristic, but stores the line lengths in a histogram with predefined buckets. By defining the buckets in advance, there is the possibility that the most common line lengths fall into two buckets leading to a false negative.

A.1.6 Tokenization, parsing

After considering ucto (van Gompel et al., 2012), elephant (Evang et al., 2013), and Alpino's[6] tokenizer, the last appears to perform best on various edge cases involving hyphenated words and quoted speech, and additionally supports preserving paragraphs. A comparison of the word and sentence tokenization (detected sentence boundaries are indicated by ↵):

Test sentences 'Echt waar?' fluisterde ze in zijn hals. Hij schoot op de JP8-brandstof toen de Surface-to-Air (sam)-missiles op hem af kwamen. 81 procent van de schoten was raak.

ucto Echt waar ? ↵
 ' fluisterde ze in zijn hals . ↵
 Hij schoot op de JP 8-brandstof toen de Surface-to-Air (sam)-missiles op hem af kwamen . 81 procent van de schoten was raak . ↵

elephant ' Echt waar ? ' fluisterde ze in zijn hals . ↵
 Hij schoot op de JP8-brandstof toen de Surface-to-Air (sam)-missiles op hem af kwamen . ↵
 81 procent van de schoten was raak . ↵

Alpino ' Echt waar ? ' fluisterde ze in zijn hals . ↵
 Hij schoot op de JP8-brandstof toen de Surface-to-Air (sam)-missiles op hem af kwamen . ↵
 81 procent van de schoten was raak . ↵

ucto separates the sentences with quoted speech incorrectly, separates JP and 8, and also fails to detect the sentence boundary before "81 procent."

elephant separates the opening parenthesis of "(sam)-missiles," which should remain a single token.

Some issues with Alpino's tokenizer were discovered, but they can be ameliorated by pre- and postprocressing. An ellipses after a period ending the previous sentence is merged by the tokenizer, so a temporary separator is added to prevent this. Square brackets have special meaning to the parser (to specify syntactic structure), so they are replaced with parentheses.

Each sentence is assigned an identifier of the form n–m, where n is a paragraph number, and m is the line number within that paragraph (both starting from 1).

The Alpino parser is used to assign part-of-speech (pos) tags and construct parse trees for the texts. See Figure A.4 for an example. Not shown in this visualization are detailed morphological tags (e.g., definite vs. indefinite, singular vs. plural) and grammatical function tags (e.g., subject, object) at each node, which are also part of Alpino's output.

---

6 Alpino, cf. http://www.let.rug.nl/vannoord/alp/Alpino/
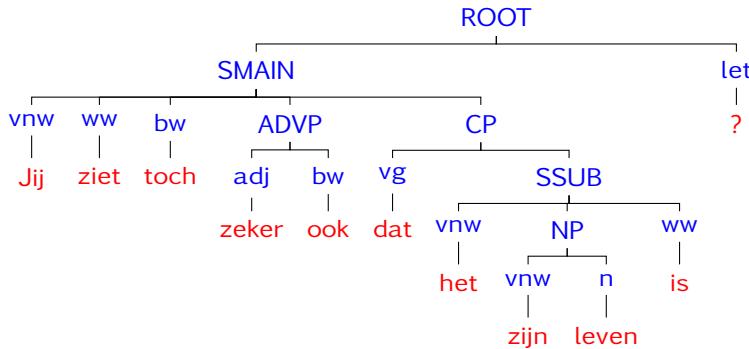
Figure A.4: A parse tree from Alpino of a sentence from Arthur Japin, Vaslav.

A.2 The list of novels

Table A.1 and A.2 list some of the novels with the lowest and highest ratings. The full list of the 401 novels is as follows. An English title is given where available. Each novel is listed with a coarse genre category (Fiction, Thriller, Romantic, Other). Translated novels are indicated as such; other novels were originally written in Dutch. Most novels (365) were included through the bestseller list. Exceptions are indicated as follows:

boekenweekgeschenk (national book week gift, 4 novels) novels commissioned for the national book week that are complimentary with book purchases.
Literaire Juweeltjes (literary jewels, 13 novels) a series of short novels by well known authors in low price editions to promote reading.[7]
LIBRARY (19 novels) the novel was added because of its popularity in public libraries, while not appearing on the bestseller list.

Abdolah, Kader. 2011: *De koning* (The King), Fiction. 2011: *De kraai*, Fiction, *boekenweekgeschenk*.

Adler-Olsen, Jussi. 2010: *De fazantenmoordenaars* (The Absent One: A Department Q Novel), Suspense, translated. 2010: *De noodkreet in de fles* (A Conspiracy of Faith: A Department Q Novel), Suspense, translated. 2010: *De vrouw in de kooi* (The Keeper of Lost Causes), Suspense, translated. 2011: *De bedrijfsterrorist*, Suspense, translated. 2011: *Dossier 64*, Suspense, translated. 2012: *Het Washingtondecreet*, Suspense, translated.

Allende, Isabel. 2010: *Het eiland onder de zee* (Island Beneath the Sea), Fiction, translated. 2011: *Het negende schrift van Maya* (Maya's Notebook), Fiction, translated.

Amirrezvani, Anita. 2007: *Dochter van Isfahan* (The Blood Of Flowers), Fiction, translated.

---

7 Cf. http://nl.wikipedia.org/wiki/Literaire_Juweeltjes

	Literary	Quality
James-VijftigTintenGrijs	2.12	3.54
Worthy-JamesWorthy	3.79	4.17
Kluun-Haantjes	3.50	4.31
Royen-Mannentester	3.18	4.32
Vermeer-Cruise	3.11	4.38
Proper-GooischeVrouwen	2.69	4.49
Vermeer-Apres-ski	3.07	4.49
James-VijftigTintenDonkerder	2.60	4.51
Bezaz-Vinexvrouwen	3.54	4.53
Gilbert-EtenBiddenBeminnen	3.51	4.58
…		
Nesbo-Pantserhart	4.27	6.26
Murakami-NorwegianWood	6.31	6.28
Binet-Hhhh	6.02	6.29
Heijden-Tonio	6.29	6.31
Hill-Negerboek	5.32	6.33
Mitchell-NietVerhoordeGebeden	6.19	6.38
Lanoye-Sprakeloos	6.37	6.43
Mortier-GestameldLiedboek	6.54	6.48
Barnes-AlsofVoorbijIs	6.62	6.54
Brokken-BaltischeZielen	5.58	6.54

Table A.1: The 10 novels with the lowest and highest mean quality ratings.

	Literary	Quality
James-VijftigTintenGrijs	2.12	3.54
Kinsella-ShopaholicBaby	2.37	4.89
Kinsella-MagIkJe	2.44	5.52
Weisberger-ChanelChic	2.45	4.71
Kinsella-MiniShopaholic	2.46	4.84
Kinsella-KenJeMe	2.47	5.35
Mansell-VersierMeDan	2.50	5.17
Weisberger-ChampagneInChateau	2.52	5.03
Fforde-Trouwplannen	2.56	5.33
Mansell-DrieIsTe	2.58	5.04
...		
Eco-BegraafplaatsVanPraag	6.24	5.84
Bernlef-GeleendeLevens	6.25	6.23
Enter-Grip	6.26	6.11
Heijden-Tonio	6.29	6.31
Murakami-NorwegianWood	6.31	6.28
Lanoye-Sprakeloos	6.37	6.43
Houellebecq-KaartEnGebied	6.41	6.14
Mortier-GestameldLiedboek	6.54	6.48
Mortier-Godenslaap	6.60	6.16
Barnes-AlsofVoorbijIs	6.62	6.54

Table A.2: The 10 novels with the lowest and highest mean literary ratings.

Ammaniti, Niccolò. 2007: *Het laatste oudejaar van de mensheid*, Fiction, translated. 2007: *Zo God het wil* (As God Commands), Fiction, translated. 2010: *Jij en ik* (Me And You), Fiction, translated. 2010: *Laat het feest beginnen* (Let the Games Begin), Fiction, translated.

Appel, René. 2008: *Weerzin*, Suspense. 2010: *Van twee kanten*, Suspense.

Auel, Jean Marie. 2011: *Het lied van de grotten* (The Land of Painted Caves: Earth's Children), Fiction, translated.

Austin, Lynn. 2008: *Eindelijk thuis* (Until We Reach Home), Fiction, translated, library.

Avallone, Silvia. 2010: *Staal*, Fiction, translated.

Baantjer, Appie / Waal, Simon de. 2010: *Een dief in de nacht*, Suspense. 2010: *Een lijk in de kast*, Suspense. 2011: *Een rat in de val*, Suspense. 2011: *Een schot in de roos*, Suspense. 2012: *Een mes in de rug*, Suspense.

Bakker, Gerbrand. 2010: *De omweg*, Fiction.

Baldacci, David. 2007: *Geniaal geheim* (Simple Genius), Suspense, translated. 2008: *Niets dan de waarheid* (The Whole Truth), Suspense, translated. 2008: *De rechtvaardigen* (Divine Justice), Suspense, translated, library. 2009: *Familieverraad* (First Family), Suspense, translated. 2009: *In het geheim* (True Blue), Suspense, translated. 2010: *Rechteloos* (Hell's Corner), Suspense, translated. 2010: *Verlos ons van het kwaad* (Deliver Us From Evil), Suspense, translated. 2011: *Die zomer* (One Summer), Fiction, translated. 2011: *De provocatie* (Zero Day), Suspense, translated. 2011: *De zesde man* (The Sixth Man), Suspense, translated. 2012: *Onschuldig* (The Innocent), Suspense, translated.

Barnes, Julian. 2011: *Alsof het voorbij is* (The Sense of an Ending), Fiction, translated.

Barr, Emily. 2007: *Klasgenoten* (Out Of My Depth), Fiction, translated.

Beijnum, Kees van. 2010: *Een soort familie*, Fiction.

Berg, Greetje van den. 2009: *Ergens achteraan*, Other, library.

Bernlef, J.. 2010: *Geleende levens*, Fiction. 2011: *De een zijn dood*, Fiction.

Bezaz, Naima El. 2010: *Vinexvrouwen*, Fiction.

Binchy, Maeve. 2009: *Hart & Ziel* (Heart and Soul), Romantic, translated, library. 2011: *En toen kwam Frankie* (Minding Frankie), Fiction, translated.

Binet, Laurent. 2010: *HhhH* (HhhH), Fiction, translated.

Blake, Sarah. 2010: *De laatste brief* (The Postmistress), Fiction, translated.

Blum, Jenna. 2010: *Het Familieportret* (Those Who Save Us), Fiction, translated. 2011: *In tweestrijd* (The Stormchasers), Fiction, translated.

Boyne, John. 2011: *Het winterpaleis* (The House Of Special Purpose), Fiction, translated.

Brijs, Stefan. 2011: *Post voor mevrouw Bromley*, Fiction.

Brokken, Jan. 2010: *Baltische zielen*, Other; non-fiction.

Brown, Janelle. 2008: *Alles wat wij wilden was alles* (All We Ever Wanted Was Everything), Fiction, translated.

Brown, Dan. 2009: *Het Verloren Symbool* (The Lost Symbol), Suspense, translated.

Burgers-Drost, Julia. 2008: *Tussen hart en verstand*, Other, library.

Bushnell, Candace. 2008: *1 Fifth Avenue* (One Fifth Avenue), Romantic, translated.

Buwalda, Peter. 2010: *Bonita Avenue* (Bonita Avenue), Fiction.

Campert, Remco. 2007: *Dagboek van een poes*, Fiction.

Carré, John le. 2010: *Ons soort verrader* (Our Kind Of Traitor), Suspense, translated.

Casey, Jane. 2010: *Spoorloos* (The Missing), Suspense, translated.

Child, Lee. 2010: *61 Uur* (61 Hours), Suspense, translated. 2010: *Tegenspel* (Worth Dying For), Suspense, translated. 2011: *De affaire* (The Affair), Suspense, translated.

Clancy, Tom. 2010: *Op leven en dood* (Dead or Alive), Suspense, translated. 2012: *In het vizier* (Locked On), Suspense, translated.

Clancy, Tom / Telp, Peter. 2011: *De ogen van de vijand* (Against All Enemies), Suspense, translated.

Cleave, Chris. 2009: *Kleine Bij* (The Other Hand), Fiction, translated.

Coben, Harlan. 2009: *Verloren* (Long Lost), Suspense, translated. 2010: *Verzoeking* (Caught), Suspense, translated. 2011: *Levenslijn* (Live Wire), Suspense, translated. 2012: *Blijf dichtbij* (Stay Close), Suspense, translated.

Coelho, Paulo. 2010: *De beschermengel*, Fiction, translated. 2011: *Aleph* (Aleph), Fiction, translated.

Collins, Suzanne. 2010: *De hongerspelen* (The Hunger Games), Other, translated. 2010: *Spotgaai* (Mockingjay), Other, translated. 2011: *Vlammen* (Catching Fire), Other, translated.

Cornwell, Patricia. 2010: *Mortuarium* (Port Mortuary), Suspense, translated. 2010: *De Scarpetta factor* (The Scarpetta Factor), Suspense, translated. 2011: *Rood waas* (Red Mist), Suspense, translated.

Cronin, Justin. 2010: *De oversteek* (The Passage), Fiction, translated.

Cussler, Clive / Brul, Jack Du. 2011: *Dodenschip* (Plague Ship), Suspense, translated.

Cussler, Clive / Cussler, Dirk. 2010: *Duivelsadem* (Arctic Drift), Suspense, translated. 2011: *Wassende maan* (Crescent Dawn), Suspense, translated.

Cussler, Clive. 2010: *Medusa* (Medusa), Suspense, translated.

Dewulf, Bernard. 2010: *Kleine dagen*, Fiction.

Dijkshoorn, Nico. 2012: *Nooit ziek geweest*, Fiction.

Dijkzeul, Lieneke. 2007: *Koude lente*, Suspense. 2010: *Gouden bergen*, Suspense, *Literaire Juweeltjes*. 2011: *Verloren zoon*, Suspense.

Dis, Adriaan van. 2010: *Tikkop*, Fiction.

Donoghue, Emma. 2010: *Kamer* (Room), Fiction, translated.

Dorrestein, Renate. 2010: *De leesclub*, Fiction. 2011: *De stiefmoeder*, Fiction. 2012: *De zondagmiddagauto*, Fiction, *Literaire Juweeltjes*.

Duenas, Maria. 2012: *Het geluid van de nacht*, Romantic, translated.

Durlacher, Jessica. 2010: *De held*, Fiction.

Eco, Umberto. 2011: *Begraafplaats van Praag* (The Prague Cemetery), Fiction, translated.

Eggers, Dave. 2007: *Wat is de wat* (What is the What), Fiction, translated.

Enquist, Anna. 2011: *De verdovers*, Fiction.

Enter, Stephan. 2011: *Grip*, Fiction.

Evans, Nicholas. 2010: *De vergeving* (The Brave), Fiction, translated.

Falcones, Ildefonso. 2010: *De hand van Fatima*, Fiction, translated.

Fallon, J.. 2007: *Op de man af* (Getting Rid of Matthew), Romantic, translated.

Fforde, Katie. 2010: *Trouwplannen* (Wedding Season), Romantic, translated, library.

Fielding, Joy. 2009: *Roerloos* (Still Life), Fiction, translated.

Follett, Ken. 2010: *Val der titanen* (Fall of Giants), Fiction, translated.

Folsom, Allan. 2009: *Dag van ontmaskering* (The Hadrian Memorandum), Suspense, translated.

Forbes, Elena. 2010: *Sterf met mij* (Die With Me), Suspense, translated.

Forsyth, Frederick. 2010: *De cobra* (The Cobra), Suspense, translated.

Fragoso, Margaux. 2011: *Tijger, tijger* (Tiger, tiger), Fiction, translated.

Franck, Julia. 2008: *De middagvrouw*, Fiction, translated.

Franzen, Jonathan. 2010: *De correcties* (The Corrections), Fiction, translated. 2010: *Vrijheid* (Freedom), Fiction, translated.

French, Nicci. 2007: *Tot het voorbij is* (Until It's Over), Suspense, translated. 2008: *Wat te doen als iemand sterft* (What to Do When Someone Dies), Fiction, translated. 2009: *Medeplichtig* (Complicit), Suspense, translated. 2011: *Blauwe maandag* (Blue Monday), Suspense, translated. 2012: *Dinsdag is voorbij* (Tuesday's Gone), Suspense, translated.

Galen, Alex van. 2012: *Süskind*, Fiction.

Gastel, Chantal van. 2008: *Zwaar verliefd!*, Romantic. 2010: *Geknipt voor jou*, Romantic.

George, Elizabeth. 2010: *Lichaam van de dood* (This Body of Death), Suspense, translated. 2012: *Een duister vermoeden* (Believing the Lie), Suspense, translated.

Gerrard, Nicci. 2008: *Het weerzien* (The Middle Place), Fiction, translated.

Gerritsen, Tess. 2007: *De Mefisto Club* (The Mephisto Club), Suspense, translated. 2007: *Verdwijn* (Vanish), Suspense, translated. 2008: *Koud hart* (The Bone Garden), Suspense, translated. 2009: *Het aandenken* (The Keepsake), Suspense, translated. 2010: *Sneeuwval* (Ice Cold), Suspense, translated. 2011: *Het stille meisje* (The Silent Girl), Suspense, translated.

Gilbert, Elizabeth. 2009: *Eten, bidden, beminnen* (Eat, Pray, Love), Fiction, translated. 2010: *Toewijding* (Committed), Fiction, translated.

Giordano, Paolo. 2009: *De eenzaamheid van de priemgetallen* (The Solitude of Prime Numbers), Fiction, translated.

Giphart, Ronald. 2010: *IJsland*, Fiction. 2011: *Zeven jaar goede seks*, Fiction, *Literaire Juweeltjes*.

Grisham, John. 2008: *De aanklacht* (The Appeal), Suspense, translated. 2010: *De bekentenis* (The Confession), Suspense, translated. 2010: *De wettelozen* (Ford County), Suspense, translated. 2011: *Het proces* (The Litigators), Suspense, translated.

Groningen, Merel van. 2008: *Misleid*, Other.

Grunberg, Arnon. 2010: *Huid en haar*, Fiction. 2011-2009: *Selmonosky's droom*, Fiction, *Literaire Juweeltjes*. 2012: *De man zonder ziekte*, Fiction.

Gudenkauf, Heather. 2010: *In stilte gehuld* (The Weight of Silence), Suspense, translated.

Hannah, Sophie. 2007: *Kleine meid* (Little Face), Suspense, translated. 2009: *Moederziel* (The Point of Rescue), Other, translated.

Harbach, Chad. 2011: *De kunst van het veldspel* (The Art of Fielding), Fiction, translated.

Hart, Maarten 't. 2009: *Verlovingstijd*, Fiction.

Hayder, Mo. 2009: *Huid* (Skin), Suspense, translated. 2010: *Diep* (Gone), Suspense, translated. 2011: *Rot* (Hanging Hill), Suspense, translated.

Haynes, Elizabeth. 2011: *Waarheen je ook vlucht* (Into the Darkest Corner), Suspense, translated.

Heijden, A.F.Th van der. 2011: *Tonio*, Fiction.

Hill, Lawrence. 2011: *Het negerboek* (The Book of Negroes), Fiction, translated.

Hoag, Tami. 2011: *Dieper dan de doden* (Deeper than the Dead), Suspense, translated.

Hodgkinson, Amanda. 2012: *Britannia Road 22* (22 Britannia Road), Fiction, translated.

Hollander, Loes den. 2007: *Naaktportret*, Suspense, library. 2008: *Broeinest*, Suspense, library. 2008: *Dwaalspoor*, Suspense, library. 2009: *Driftleven*, Suspense, library. 2010: *De kat op zolder*, Fiction, *Literaire Juweeltjes*. 2010: *Krachtmeting*, Suspense. 2010: *Het scherventapijt*, Fiction. 2010: *Vluchtgedrag*, Suspense. 2010: *Wisselgeld*, Suspense. 2011: *Glansrol*, Suspense. 2011: *Uitglijder*, Suspense. 2011: *Zielsverwanten*, Suspense. 2012: *Troostkind*, Suspense.

Hosseini, Khaled. 2007: *Duizend schitterende zonnen* (A Thousand Splendid Suns), Fiction, translated.

Houellebecq, Michel. 2011: *De kaart en het gebied*, Fiction, translated.

Indriðason, Arnaldur. 2010: *Onderstroom*, Suspense, translated. 2011: *Doodskap*, Suspense, translated.

Irving, John. 2010: *De laatste nacht in Twisted River* (Last Night in Twisted River), Fiction, translated. 2012: *In een mens* (In One Person), Fiction, translated.

Jackson, Lisa. 2007: *De zevende doodzonde* (Shiver), Suspense, translated.

James, Erica. 2008: *Schaduwleven* (Tell it to the Skies), Romantic, translated. 2008: *Zussen voor altijd* (Love and Devotion), Romantic, translated. 2009: *De kleine dingen* (It's the Little Things), Romantic, translated.

James, E.L.. 2012: *Vijftig tinten donkerder* (Fifty Shades Darker), Other, translated. 2012: *Vijftig tinten grijs* (Fifty Shades of Grey), Other, translated. 2012: *Vijftig tinten vrij* (Fifty Shades Freed), Other, translated.

Jansen, Suzanna. 2010: *De pronkspiegel*, Other, *Literaire Juweeltjes*; non-fiction.

Janssen, Roel. 2007: *De tiende vrouw*, Suspense.

Japin, Arthur. 2007: *De overgave*, Fiction. 2010: *Vaslav*, Fiction. 2012: *Dooi*, Fiction, *Literaire Juweeltjes*.

Jonasson, Jonas. 2011: *De 100-jarige man die uit het raam klom en verdween* (The 100-Year-Old Man Who Climbed Out the Window and Disappeared), Fiction, translated.

Kelly, Cathy. 2010: *Eens in je leven* (Once in a lifetime), Romantic, translated.

Kepler, Lars. 2010: *Hypnose* (The Hypnotist), Suspense, translated. 2011: *Contract*, Suspense, translated. 2012: *Getuige*, Suspense, translated.

King, Stephen. 2009: *Gevangen* (Under the Dome), Suspense, translated. 2010: *Aardedonker, zonder sterren* (Full Dark, No Stars), Suspense, translated; short stories. 2011: *22/11/63* (11/22/63), Suspense, translated. 2011: *Eenmalige zonde* (Blockade Billy), Suspense, translated; short stories.

Kingsbury, Karen. 2008: *Laatste dans* (A Time to Dance), Fiction, translated, library. 2009: *Nooit te laat* (Redemption), Fiction, translated, library.

Kinsella, Sophie. 2009: *Ken je me nog?* (Remember me?), Romantic, translated. 2009: *Shopaholic & Baby* (Shopaholic and Baby), Romantic, translated. 2009: *Wat spook jij uit?* (Twenties Girl), Romantic, translated. 2010: *Mini Shopaholic* (Mini Shopaholic), Romantic, translated. 2012: *Mag ik je nummer even?* (I've Got Your Number), Romantic, translated.

Kluun.   2010: *Haantjes*, Fiction.

Koch, Herman.   2009: *Het diner* (The Dinner), Fiction.   2011: *Zomerhuis met zwembad* (Summerhouse with Swimming Pool), Fiction.

Kooten, Kees van.   2013: *De verrekijker*, Other, *boekenweekgeschenk*; non-fiction.

Koryta, Michael.   2008: *Begraven* (A Welcome Grave), Suspense, translated.

Krauss, Nicole.   2010: *Het grote huis* (Great House), Fiction, translated.

Kroonenberg, Yvonne.   2011: *De familieblues*, Fiction; short stories.

Kwast, Ernest van der.   2010: *Mama Tandoori*, Fiction.

Läckberg, Camilla.   2007: *Predikant* (The Preacher), Suspense, translated.   2008: *IJsprinses* (Ice Princess), Suspense, translated.   2008: *Steenhouwer* (The Stonecutter), Suspense, translated.   2008: *Zusje* (The Gallows Bird), Suspense, translated.   2009: *Oorlogskind* (The Hidden Child), Suspense, translated.   2010: *Sneeuwstorm en amandelgeur* (The Scent of Almonds), Suspense, translated.   2010: *Zeemeermin* (The Drowning), Suspense, translated.   2011: *Vuurtorenwachter* (The Lost Boy), Suspense, translated.   2012: *Engeleneiland* (The Angel Maker's Wife), Suspense, translated.

Lanoye, Tom.   2009: *Sprakeloos*, Fiction.   2012: *Heldere hemel*, Fiction, *boekenweekgeschenk*.

Lapidus, Jens.   2009: *Snel geld* (Easy Money), Suspense, translated.   2010: *Bloedlink*, Suspense, translated.   2011: *Val dood*, Suspense, translated.

Larsson, Stieg.   2008: *Gerechtigheid* (The Girl Who Kicked the Hornets' Nest), Suspense, translated.   2008: *Mannen die vrouwen haten* (The Girl with the Dragon Tattoo), Suspense, translated.   2008: *De vrouw die met vuur speelde* (The Girl Who Played with Fire), Suspense, translated.

Launspach, Rik.   2009: *1953*, Fiction.

Lavender, Will.   2008: *Het verborgen raadsel* (Obedience), Suspense, translated.

Lehane, Dennis.   2008: *Gone Baby Gone* (Gone, baby, gone), Suspense, translated.

Lewinsky, Charles.   2007: *Het lot van de familie Meijer* (Melnitz), Fiction, translated.   2010: *De verborgen geschiedenis van Courtillon* (Johannistag), Fiction, translated.

Lindell, Unni.   2008: *Honingval*, Suspense, translated.

Loo, Tessa de.   2012: *De grote moeder*, Fiction.

Ludlum, Robert / Mills, Kyle.   2012: *Het Ares akkoord* (The Ares Decision), Suspense, translated.

Luiten, Hetty.   2009: *Je blijft altijd welkom*, Other, library.

Macdowell, Heather / Macdowell, Rose.   2008: *Diner voor 2* (Turning Tables), Romantic, translated.

Mak, Geert.   2012: *Reizen zonder John*, Other; non-fiction.

Mankell, Henning.   2007: *Kennedy's brein* (Kennedy's Brain), Suspense, translated.   2008: *De Chinees* (The Man from Beijing), Suspense, translated.   2009: *De Daisy Sisters* (Daisy Sisters), Fiction, translated.   2010: *De gekwelde man* (The Troubled Man), Suspense, translated.   2011: *De geschiedenis van een gevallen engel*, Fiction, translated.

Mansell, Jill.   2008: *Scherven brengen geluk* (An Offer You Can't Refuse), Romantic, translated.   2009: *Eenmaal andermaal verliefd* (Rumour Has It), Romantic, translated.   2010: *Versier me dan* (Take a Chance on Me), Romantic, translated.   2011: *Drie is te veel* (Two's Company), Romantic, translated.   2011: *De smaak te pakken* (To the Moon and Back), Romantic, translated.   2012: *Vlinders voor altijd* (A Walk in the Park), Romantic, translated.

Marlantes, Karl. 2011: *Matterhorn* (Matterhorn), Fiction, translated.

Mastras, George. 2009: *Tranen over Kashmir* (Fidali's Way), Fiction, translated.

McCoy, Sarah. 2012: *De bakkersdochter* (The Baker's Daughter), Fiction, translated.

McFadyen, Cody. 2010: *Tijd om te sterven* (Abandoned), Suspense, translated.

McNab, Andy. 2010: *Onbreekbare eenheid* (Seven Troop), Suspense, translated. 2011: *Oorlogswond* (Exit Wound), Suspense, translated.

Meer, Vonne van der. 2009: *Het zingen*, *het water*, *de peen*, Fiction, *Literaire Juweeltjes*. 2011: *De vrouw met de sleutel*, Fiction.

Meyer, Deon. 2012: *13 uur*, Suspense, translated.

Middelbeek, Mariette. 2009: *Single en sexy*, Romantic. 2009: *Turbulentie*, Romantic.

Mitchell, David. 2010: *De niet verhoorde gebeden van Jacob de Zoet* (The Thousand Autumns of Jacob de Zoet), Fiction, translated.

Moelands, Kim. 2010: *Weerloos*, Suspense.

Montefiore, Santa. 2009: *In de schaduw van het Palazzo* (The Italian Matchmaker), Romantic, translated. 2010: *De affaire* (The Affair), Romantic, translated. 2011: *Villa magdalena* (The House by the Sea), Romantic, translated. 2012: *Fairfield park* (The Summer House), Romantic, translated.

Moor, Marente de. 2010: *De Nederlandse maagd*, Fiction.

Moor, Margriet de. 2010: *De schilder en het meisje*, Fiction.

Mortier, Erwin. 2008: *Godenslaap*, Fiction. 2011: *Gestameld liedboek*, Fiction.

Mosby, Steve. 2008: *50/50 Moorden* (The 50/50 Killer), Suspense, translated.

Murakami, Haruki. 2007: *Norwegian Wood* (Norwegian Wood), Fiction, translated. 2010: *1q84* (1Q84), Fiction, translated.

Neill, Fiona. 2009: *Vriendschap*, *liefde en andere stommiteiten* (Friends, Lovers and Other Indiscretions), Romantic, translated.

Nesbø, Jo. 2010: *Het pantserhart*, Suspense, translated.

Nesbo, Jo. 2008: *De verlosser*, Suspense, translated. 2009: *De sneeuwman*, Suspense, translated. 2011: *De schim*, Suspense, translated.

Nesser, Håkan. 2012: *De man zonder hond*, Suspense, translated.

Newman, Ruth. 2008: *Vleugels* (Twisted Wing), Suspense, translated. 2010: *Schaduwkant* (The Company of Shadows), Suspense, translated.

Noort, Saskia. 2007: *Afgunst*, Suspense, library. 2004: *De eetclub* (The Dinner Club), Suspense. 2009: *De verbouwing*, Suspense. 2011: *Koorts* (Fever), Suspense.

Paolini, Christopher. 2011: *Erfenis* (Inheritance), Other, translated.

Patterson, James. 2008: *De affaire* (The Quickie), Suspense, translated.

Patterson, James / Ledwige, Michael. 2012: *Hitte* (Now you see her), Suspense, translated.

Patterson, James / Marklund, Liza. 2010: *Partnerruil* (The Postcard Killers), Suspense, translated.

Pauw, Marion. 2008: *Daglicht*, Suspense. 2008: *Drift*, Suspense. 2008: *Villa Serena*, Suspense. 2009: *Zondaarskind*, Suspense. 2010: *Jetset*, Suspense.

Peetz, Monika. 2011: *De dinsdagvrouwen*, Fiction, translated.

Pick, Alison. 2011: *Donderdagskind* (Far to Go), Fiction, translated.

Picoult, Jodi. 2008: *Negentien minuten* (Nineteen Minutes), Fiction, translated.

Proper, Emile / Eynden, Sabine van den. 2008: *Gooische vrouwen*, Romantic.

Ravelli. 2011: *De Vliegenvanger*, Fiction.

Rendell, Ruth. 2010: *De dief - Kattenkruid!* (The Thief), Suspense, translated.

Rijn, Linda van. 2010: *Last Minute*, Suspense. 2012: *Blue Curacao*, Suspense.

Roberts, Nora. 2009: *Eerbetoon* (Tribute), Romantic, translated, library.

Robotham, Michael. 2008: *Gebroken* (Shatter), Suspense, translated.

Rose, Karen. 2010: *Moord voor mij* (Kill For Me), Suspense, translated.

Rosenboom, Thomas. 2009: *Zoete mond*, Fiction. 2010: *Mechanica*, Fiction, *Literaire Juweeltjes*.

Rosenfeldt, Hjorth. 2011: *Wat verborgen is*, Suspense, translated.

Rosnay, Tatiana de. 2007: *Haar naam was Sarah* (Sarah's Key), Fiction, translated. 2009: *Die laatste zomer* (A Secret Kept), Fiction, translated. 2010: *Kwetsbaar* (Moka), Fiction, translated. 2011: *Het huis waar jij van hield* (The House I Loved), Fiction, translated. 2012: *Het appartement*, Fiction, translated.

Rowling, J.K.. 2007: *Harry Potter en de Relieken van de Dood* (Harry Potter and the Deathly Hallows), Other, translated, library.

Royen, Heleen van. 2009: *De mannentester*, Fiction. 2012: *Sabine*, Fiction, *Literaire Juweeltjes*.

Ruiz Zafón, Carlos. 2008: *De schaduw van de wind* (The Shadow of the Wind), Fiction, translated. 2009: *Het spel van de engel* (The Angel's Game), Fiction, translated. 2012: *De gevangene van de hemel* (The Prisoner of Heaven), Fiction, translated.

Sambeek, Ciel van. 2008: *Koninginnenrit*, Fiction. 2010: *Bloed zaad en tranen*, Fiction.

Sansom, Christopher John. 2007: *Winter in Madrid* (Winter in Madrid), Suspense, translated.

Scholten, Jaap. 2010: *Kameraad Baron*, Fiction.

Sedaris, David. 2010: *Van je familie moet je het hebben*, Fiction, translated; short stories.

Shah, Hannah. 2009: *De dochter van de iman* (The Imam's Daughter), Fiction, translated.

Siebelink, Jan. 2010: *Het lichaam van Clara*, Fiction. 2011: *Oscar*, Fiction.

Slaughter, Karin. 2007: *Onaantastbaar* (Skin Privilege), Suspense, translated. 2008: *Versplinterd* (Fractured), Suspense, translated. 2009: *Genesis* (Genesis), Suspense, translated. 2010: *Ongezien* (The Unremarkable Heart), Suspense, translated. 2010: *Verbroken* (Broken), Suspense, translated. 2011: *Gevallen* (Fallen), Suspense, translated. 2012: *Genadeloos* (Criminal), Suspense, translated.

Slee, Carry. 2009: *Bangkok Boy*, Other. 2010: *Fatale liefde*, Other.

Smeets, Mart. 2010: *De afrekening*, Fiction. 2011: *Een koud kunstje*, Fiction, *Literaire Juweeltjes*; short stories.

Smit, Susan. 2010: *Vloed*, Fiction.

Smith, Wilbur. 2011: *Op volle zee* (Those in Peril), Suspense, translated.

Spijker, Rita. 2007: *Tussen zussen*, Fiction.

Springer, F.. 2010: *Quadriga*, Fiction.

Steel, Danielle. 2010: *De weg van het hart* (Matters of the Heart), Romantic, translated. 2011: *Door dik en dun* (Big Girl), Romantic, translated.

Stevens, Chevy. 2010: *Vermist* (Still Missing), Suspense, translated.

Stockett, Kathryn. 2010: *Een keukenmeidenroman* (The Help), Fiction, translated.

Sundstøl, Vidar. 2010: *Land van dromen*, Suspense, translated.

Terlouw, Jan / Terlouw, Sanne. 2011: *Hellehonden*, Suspense.

Tex, Charles den. 2009: *Spijt*, Suspense. 2010: *Wachtwoord*, Suspense.

Theorin, Johan. 2008: *Schemeruur*, Suspense, translated.

Thomése, P.F.. 2010: *De weldoener*, Fiction.

Treur, Franca. 2009: *Dorsvloer vol confetti*, Fiction.

Trussoni, Danielle. 2010: *Het uur van de engelen* (Angelology), Suspense, translated.

Verhoef, Esther. 2007: *Close-up*, Suspense. 2008: *Alles te verliezen*, Suspense. 2009: *Erken mij*, Suspense, library. 2010: *Déjà vu*, Suspense. 2012: *Tegenlicht*, Fiction.

Verhulst, Dimitri. 2010: *De laatste liefde van mijn moeder*, Fiction.

Vermeer, Suzanne. 2007: *De vlucht*, Suspense. 2008: *Zomertijd*, Suspense. 2009: *Après-ski*, Suspense. 2009: *Cruise*, Suspense. 2010: *De suite*, Suspense. 2011: *Bella Italia*, Suspense. 2011: *Zwarte piste*, Suspense. 2012: *Bon Bini beach*, Suspense. 2012: *Noorderlicht*, Suspense.

Visser, Judith. 2008: *Stuk*, Suspense.

Vlugt, Simone van der. 2007: *Het laatste offer*, Suspense. 2008: *Blauw water*, Suspense. 2009: *Herfstlied*, Suspense. 2009: *Jacoba, Dochter van Holland*, Fiction. 2010: *Op klaarlichte dag*, Suspense. 2011: *In mijn dromen*, Suspense. 2012: *Rode sneeuw in december*, Other.

Voskuijl, Anouschka. 2011: *Dorp*, Suspense.

Voskuil, J.J.. 2012: *De buurman*, Fiction.

Vuijsje, Robert. 2008: *Alleen maar nette mensen*, Fiction. 2012: *Alleen maar foute mensen*, Fiction, *Literaire Juweeltjes*.

Wageningen, Gerda van. 2008: *In de schemering*, Other, library.

Watson, S.J.. 2011: *Voor ik ga slapen* (Before I Go to Sleep), Suspense, translated.

Weiner, Jennifer. 2009: *Sommige meisjes* (Certain Girls), Romantic, translated.

Weisberger, Lauren. 2008: *Chanel Chic* (Chanel Chic), Romantic, translated. 2010: *Champagne in Chateau Marmont* (Last Night At Chateau Marmont), Romantic, translated.

Wickham, Madeleine. 2009: *Zoete tranen* (The Gatecrasher), Romantic, translated. 2010: *De cocktailclub* (Cocktails for Three), Romantic, translated. 2011: *Het zwemfeestje* (Swimming Pool Sunday), Romantic, translated. 2012: *De vraagprijs* (A Desirable Residence), Romantic, translated.

Wieringa, Tommy. 2009: *Caesarion*, Fiction. 2011: *Portret van een heer*, Other, *Literaire Juweeltjes*; non-fiction.

Winter, Leon de. 2008: *Recht op terugkeer*, Fiction. 2012: *VSV of daden van onbaatzuchtigheid*, Fiction.

Wisse, Clemens. 2009: *De jonge boerin van Madezicht*, Other, library.

Worthy, James. 2011: *James Worthy*, Fiction.

Yalom, Irvin D.. 2012: *Het raadsel spinoza* (The Spinoza Problem), Fiction, translated.

Zwaan, Josha. 2010: *Parnassia*, Fiction.

Zwagerman, Joost. 2010: *Duel*, Fiction, *boekenweekgeschenk*.

Bibliography

Abouelhoda, Mohamed Ibrahim, Stefan Kurtz, and Enno Ohlebusch (2004). Replacing suffix trees with enhanced suffix arrays. *Journal of Discrete Algorithms*, 2(1):53–86.

Aiolli, Fabio, Giovanni Da San Martino, Alessandro Sperduti, and Alessandro Moschitti (2007). Efficient kernel-based learning for trees. In *Proceeding of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 308–315.

Algee-Hewitt, Mark, Ryan Heuser, and Franco Moretti (2015). On paragraphs. scale, themes, and narrative form. Stanford Literary Lab pamphlet 10. http://litlab.stanford.edu/LiteraryLabPamphlet10.pdf.

Allamanis, Miltiadis and Charles Sutton (2014). Mining idioms from source code. ArXiv e-print. http://arxiv.org/abs/1404.0417.

Allison, Sarah, Ryan Heuser, Matthew L. Jockers, Franco Moretti, and Michael Witmore (2011). Quantitative formalism: an experiment. Stanford Literary Lab pamphlet. http://litlab.stanford.edu/LiteraryLabPamphlet1.pdf.

Aloni, Maria, Andreas van Cranenburgh, Raquel Fernández, and Marta Sznajder (2012). Building a corpus of indefinite uses annotated with fine-grained semantic functions. In *The eighth international conference on Language Resources and Evaluation (LREC)*. http://www.lrec-conf.org/proceedings/lrec2012/pdf/362_Paper.pdf.

Angelov, Krasimir and Peter Ljunglöf (2014). Fast statistical parsing with parallel multiple context-free grammars. In *Proceedings of EACL*, pages 368–376. http://aclweb.org/anthology/E14-1039.

Ashok, Vikas, Song Feng, and Yejin Choi (2013). Success with style: using writing style to predict the success of novels. In *Proceedings of EMNLP*, pages 1753–1764. http://aclweb.org/anthology/D13-1181.

Baayen, Harold, H. van Halteren, and F. Tweedie (1996). Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution. *Literary and Linguistic Computing*, pages 121–132.

Bal, P. Matthijs and Martijn Veltkamp (2013). How does fiction reading influence empathy? an experimental investigation on the role of emotional transportation. *PLoS ONE*, 8(1):1–12. http://dx.doi.org/10.1371/journal.pone.0055341.

Baldick, Chris (2008). Literariness. In *The Oxford Dictionary of Literary Terms*. Oxford University Press, USA.

Bansal, Mohit and Dan Klein (2010). Simple, accurate parsing with an all-fragments grammar. In *Proceedings of ACL*, pages 1098–1107. http://aclweb.org/anthology/P10-1112.

Barthélemy, François, Pierre Boullier, Philippe Deschamp, and Éric de la Clergerie (2001).

Guided parsing of range concatenation languages. In *Proceedings of ACL*, pages 42–49. http://aclweb.org/anthology/P01-1007.

Bauge, Ola S. (2012). Dehyphenation: Some empirical methods. Master's thesis, University of Oslo. https://www.duo.uio.no/bitstream/handle/123456789/9043/Bauge2012-dehyphenation.pdf.

Behnel, Stefan, Robert Bradshaw, Craig Citro, Lisandro Dalcin, Dag Sverre Seljebotn, and Kurt Smith (2011). Cython: the best of both worlds. *Computing in Science and Engineering*, 13:31–39. http://doi.ieeecomputersociety.org/10.1109/MCSE.2010.118.

Bender, Emily M., Dan Flickinger, and Stephan Oepen (2002). The grammar matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In *Proceedings of the 2002 Workshop on Grammar Engineering and Evaluation*, pages 1–7. http://aclweb.org/anthology/W02-1502.

Berenike Herrmann, J., Karina van Dalen-Oskam, and Christof Schöch (2015). Revisiting style, a key concept in literary studies. *Journal of Literary Theory*, 9(1):25–52.

Bergsma, Shane, Matt Post, and David Yarowsky (2012). Stylometric analysis of scientific articles. In *Proceedings of NAACL*, pages 327–337. http://aclweb.org/anthology/N12-1033.

Bickel, Peter J., Eugene A. Hammel, J. William O'Connell, et al. (1975). Sex bias in graduate admissions: Data from berkeley. *Science*, 187(4175):398–404.

Bikel, Daniel M. (2004). Intricacies of Collins' parsing model. *Computational Linguistics*, 30(4):479–511. http://aclweb.org/anthology/J04-4004.

Black, Ezra, John Lafferty, and Salim Roukos (1992). Development and evaluation of a broad-coverage probabilistic grammar of English-language computer manuals. In *Proceedings of ACL*, pages 185–192. http://aclweb.org/anthology/P92-1024.

Blaheta, Don and Eugene Charniak (2000). Assigning function tags to parsed text. In *Proceedings of NAACL*, pages 234–240. http://aclweb.org/anthology/A00-2031.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan (2003). Latent Dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022. http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf.

Bod, Rens (1992). A computational model of language performance: Data-oriented parsing. In *Proceedings COLING*, pages 855–859. http://aclweb.org/anthology/C92-3126.

——— (1995a). *Enriching Linguistics with Statistics: Performance Models of Natural Language*. PhD thesis, University of Amsterdam.

——— (1995b). The problem of computing the most probable tree in data-oriented parsing and stochastic tree grammars. In *Proceedings of EACL*, pages 104–111. http://aclweb.org/anthology/E95-1015.

——— (2000). Parsing with the shortest derivation. In *Proceedings of COLING*, pages 69–75.

——— (2003). An efficient implementation of a new DOP model. In *Proceedings of EACL*, volume 1, pages 19–26. http://aclweb.org/anthology/E03-1005.

——— (2013). Who's afraid of patterns?: The particular versus the universal and the meaning of humanities 3.0. *BMGN – Low Countries Historical Review*, 128(4). http://www.bmgn-lchr.nl/index.php/bmgn/article/view/9351/9785.

Bod, Rens, Remko Scha, and Khalil Sima'an, editors (2003). *Data-Oriented Parsing*. The University of Chicago Press.

Bonnema, Remko, Paul Buying, and Remko Scha (1999). A new probability model for data oriented parsing. In *Proceedings of the 12th Amsterdam Colloquium*, pages 85–90.

Boullier, Pierre (1998). Proposal for a natural language processing syntactic backbone.

Technical Report RR-3342, INRIA-Rocquencourt, Le Chesnay, France. http://www.inria.fr/RRRT/RR-3342.html.

Bouma, Gosse, Gertjan Van Noord, and Robert Malouf (2001). Alpino: Wide-coverage computational analysis of Dutch. *Language and Computers*, 37(1):45–59. http://www.let.rug.nl/vannoord/papers/alpino.pdf.

Bourdieu, Pierre (1996). *The rules of art: Genesis and structure of the literary field*. Stanford University Press.

Boyd, Adriane (2007). Discontinuity revisited: An improved conversion to context-free representations. In *Proceedings of the Linguistic Annotation Workshop*, pages 41–44. http://aclweb.org/anthology/W07-1506.

Brants, Sabine, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith (2002). The Tiger treebank. In *Proceedings of the workshop on treebanks and linguistic theories*, pages 24–41. http://www.bultreebank.org/proceedings/paper03.pdf.

Breiman, Leo (2001). Statistical modeling: The two cultures. *Statistical Science*, 16(3):199–231. With comments and a rejoinder by the author. http://dx.doi.org/10.1214/ss/1009213726.

Bresnan, Joan, Ronald M. Kaplan, Stanley Peters, and Annie Zaenen (1982). Cross-serial dependencies in Dutch. *Linguistic Inquiry*, 13(4):613–635.

Bybee, Joan L. (2007). From usage to grammar: The mind's response to repetition. *Language*, 82(4):711–733.

Cai, Shu, David Chiang, and Yoav Goldberg (2011). Language-independent parsing with empty elements. In *Proceedings of ACL-HLT*, pages 212–216. http://aclweb.org/anthology/P11-2037.

Chambi, Samy, Daniel Lemire, Owen Kaser, and Robert Godin (2016). Better bitmap performance with Roaring bitmaps. *Software: Practice and Experience*, 46(5):709–719. http://arxiv.org/abs/1402.6407.

Charniak, Eugene (1996). Tree-bank grammars. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1031–1036.

Charniak, Eugene and Mark Johnson (2005). Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of ACL*, pages 173–180. http://aclweb.org/anthology/P05-1022.

Charniak, Eugene, Mark Johnson, M. Elsner, J. Austerweil, D. Ellis, I. Haxton, C. Hill, R. Shrivaths, J. Moore, M. Pozar, et al. (2006). Multilevel coarse-to-fine PCFG parsing. In *Proceedings of NAACL-HLT*, pages 168–175. http://aclweb.org/anthology/N06-1022.

Chaski, Carole E. (1997). Who wrote it? steps toward a science of authorship identification. *National Institute of Justice Journal*, 233:15–22.

——— (2001). Empirical evaluations of language-based author identification techniques. *Forensic Linguistics*, 8:1–65.

Chernoff, Herman, Shaw-Hwa Lo, and Tian Zheng (2009). Discovering influential variables: a method of partitions. *The Annals of Applied Statistics*, pages 1335–1369. http://www.jstor.org/stable/27801550.

Chiang, David (2000). Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proceedings of ACL*, pages 456–463. http://aclweb.org/anthology/P00-1058.

Choi, Jinho D. and Martha Palmer (2010). Robust constituent-to-dependency conversion for English. In *Proceedings of the 9th International Workshop on Treebanks and Linguistic Theories*, pages 55–66. http://hdl.handle.net/10062/15934.

Chomsky, Noam (1956). Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113–124.

———— (1965). *Aspects of the Theory of Syntax*. MIT press.

Church, Kenneth Ward (1988). A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of Applied Natural Language Processing*, pages 136–143. http://aclweb.org/anthology/A88-1019.

Cohn, Trevor, Phil Blunsom, and Sharon Goldwater (2010). Inducing tree-substitution grammars. *Journal of Machine Learning Research*, 11(Nov):3053–3096.

Cohn, Trevor, Sharon Goldwater, and Phil Blunsom (2009). Inducing compact but accurate tree-substitution grammars. In *Proceedings of NAACL-HLT*, pages 548–556. http://aclweb.org/anthology/N09-1062.

Collins, Michael (1999). *Head-driven statistical models for natural language parsing*. PhD thesis, University of Pennsylvania.

Collins, Michael and Nigel Duffy (2001). Convolution kernels for natural language. In *Advances in neural information processing systems*, pages 625–632.

Collins, Michael and Nigel Duffy (2002). New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of ACL*. http://aclweb.org/anthology/P02-1034.

Collins, Michael and T. Koo (2005). Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1):25–70.

Collins-Thompson, Kevyn and James P. Callan (2004). A language modeling approach to predicting reading difficulty. In *HLT-NAACL*, pages 193–200. http://aclweb.org/anthology/N04-1025.

Cook, Paul and Graeme Hirst (2013). Automatically assessing whether a text is clichéd, with applications to literary analysis. In *Proceedings of the 9th Workshop on Multiword Expressions*, pages 52–57. http://aclweb.org/anthology/W13-1008.

Crammer, Koby, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer (2006). Online passive-aggressive algorithms. *The Journal of Machine Learning Research*, 7:551–585. http://jmlr.org/papers/volume7/crammer06a/crammer06a.pdf.

Daum, Michael, Kilian A. Foth, and Wolfgang Menzel (2004). Automatic transformation of phrase treebanks to dependency trees. In *Proceedings of LREC*. http://www.lrec-conf.org/proceedings/lrec2004/pdf/489.pdf.

de Montaigne, Michel (1958). *Complete Essays of Montaigne*. Stanford U. Press. translated by Donald M. Frame.

Dienes, Peter and Amit Dubey (2003). Deep syntactic processing by combining shallow methods. In *Proceedings of ACL*, pages 431–438. http://aclweb.org/anthology/P03-1055.

Dubey, Amit and Frank Keller (2003). Probabilistic parsing for German using sister-head dependencies. In *Proceedings of ACL*, pages 96–103. http://aclweb.org/anthology/P03-1013.

Eagleton, Terry (2008). *Literary Theory: an Introduction*. University of Minnesota Press, Minneapolis.

Eisner, Jason M. (1996). Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of COLING*, pages 340–345. http://aclweb.org/anthology/C96-1058.

Eisner, Jason M. (2015). Comment on a Language Log post, March 27. http://languagelog.ldc.upenn.edu/nll/?p=18398#comment-1493059.

Elson, David, Anna Kazantseva, Rada Mihalcea, and Stan Szpakowicz, editors (2012).

Proceedings of the NAACL-HLT 2012 Workshop on Computational Linguistics for Literature. Association for Computational Linguistics, Montréal, Canada. http://aclweb.org/anthology/W12-25.

Elson, David, Anna Kazantseva, and Stan Szpakowicz, editors (2013). *Proceedings of the Workshop on Computational Linguistics for Literature*. Association for Computational Linguistics, Atlanta, Georgia. http://aclweb.org/anthology/W13-14.

Evang, Kilian, Valerio Basile, Grzegorz Chrupała, and Johan Bos (2013). Elephant: Sequence labeling for word and sentence segmentation. In *Proceedings of EMNLP*, pages 1422–1426. http://aclweb.org/anthology/D13-1146.

Evang, Kilian and Laura Kallmeyer (2011). PLCFRS parsing of English discontinuous constituents. In *Proceedings of IWPT*, pages 104–116. http://aclweb.org/anthology/W11-2913.

Everett, Daniel L. (2005). Cultural constraints on grammar and cognition in Pirahã. *Current anthropology*, 46(4):621–646. http://www.jstor.org/stable/10.1086/431525.

——— (2009). *Don't sleep, there are snakes: Life and language in the Amazonian jungle*. Vintage Books.

Feldman, Anna, Anna Kazantseva, and Stan Szpakowicz, editors (2014). *Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLFL)*. Association for Computational Linguistics, Gothenburg, Sweden. http://aclweb.org/anthology/W14-09.

Feldman, Anna, Anna Kazantseva, Stan Szpakowicz, and Corina Koolen, editors (2015). *Proceedings of the Fourth Workshop on Computational Linguistics for Literature*. Association for Computational Linguistics, Denver, Colorado. http://aclweb.org/anthology/W15-07.

Fernández-González, Daniel and André F. T. Martins (2015). Parsing as reduction. In *Proceedings of ACL*, pages 1523–1533. http://aclweb.org/anthology/P15-1147.

Flesch, Rudolph (1948). A new readability yardstick. *Journal of applied psychology*, 32(3):221.

Fraser, Alexander, Helmut Schmid, Richárd Farkas, Renjing Wang, and Hinrich Schütze (2013). Knowledge sources for constituent parsing of German, a morphologically rich and less-configurational language. *Computational Linguistics*, 39(1):57–85. http://aclweb.org/anthology/J13-1005.

Frye, Northrop (1957). *Anatomy of criticism: Four essays by Northrop Frye*. Princeton University Press.

Gabbard, Ryan, Mitchell Marcus, and Seth Kulick (2006). Fully parsing the Penn treebank. In *Proceedings of NAACL-HLT*, pages 184–191. http://aclweb.org/anthology/N06-1024.

Gamon, Michael (2004). Linguistic correlates of style: authorship classification with deep linguistic analysis features. In *Proceedings of COLING*.

Gazdar, Gerald, Ewan Klein, Geoffrey Pullum, and Ivan Sag (1985). *Generalized phrase structure grammar*. Harvard University Press, Cambridge, Mass.

Geman, Stuart and Mark Johnson (2004). Probability and statistics in computational linguistics, a brief review. In Johnson, Mark, Sanjeev P. Khudanpur, Mari Ostendorf, and Roni Rosenfeld, editors, *Mathematical foundations of speech and language processing*, pages 1–26. Springer.

Gibson, Edward (2000). The dependency locality theory: A distance-based theory of linguistic complexity. *Image, language, brain*, pages 95–126.

Gildea, Daniel (2010). Optimal parsing strategies for linear context-free rewriting

systems. In *Proceedings of NAACL-HLT*, pages 769–776. http://aclweb.org/anthology/N10-1118.

Goodman, Joshua (2003). Efficient parsing of DOP with PCFG-reductions. In Bod et al. (2003), pages 125–146.

Graesser, Arthur C., Danielle S. McNamara, Max M. Louwerse, and Zhiqiang Cai (2004). Coh-Metrix: analysis of text on cohesion and language. *Behavior research methods, instruments, & computers*, 36(2):193–202.

Grieve, Jack (2007). Quantitative authorship attribution: An evaluation of techniques. *Literary and Linguistic Computing*, 22(3):251–270. http://llc.oxfordjournals.org/content/22/3/251.abstract.

Groenink, Annius V. (1997). Mild context-sensitivity and tuple-based generalizations of context-grammar. *Linguistics and Philosophy*, 20(6):607–636. http://www.jstor.org/stable/25001685.

Hale, Ken (1983). Warlpiri and the grammar of non-configurational languages. *Natural Language & Linguistic Theory*, 1(1):5–47.

Hall, Johan and Joakim Nivre (2008). Parsing discontinuous phrase structure with grammatical functions. In Nordström, Bengt and Aarne Ranta, editors, *Advances in Natural Language Processing*, volume 5221 of *Lecture Notes in Computer Science*, pages 169–180. Springer. http://dx.doi.org/10.1007/978-3-540-85287-2_17.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). *The Elements of Statistical Learning*. Springer Series in Statistics. Springer-Verlag, New York.

Henderson, James (2004). Discriminative training of a neural network statistical parser. In *Proceedings of ACL*, pages 95–102. http://aclweb.org/anthology/P04-1013.

Heumakers, Arnold (2015). *De Esthetische Revolutie: Ontstaan en ontwikkeling van het moderne kunstbegrip in Verlichting en Romantiek*. PhD thesis, University of Amsterdam. Translation: *The Aesthetic Revolution: Origin and development of the modern conception of art in the Enlightenment and Romanticism*. http://hdl.handle.net/11245/1.444695.

Hirst, Graeme and Ol'ga Feiguina (2007). Bigrams of syntactic labels for authorship discrimination of short texts. *Literary and Linguistic Computing*, 22(4):405–417. http://llc.oxfordjournals.org/content/22/4/405.abstract.

Hoby, Hermione (2013). Margaret Atwood: interview. The Telegraph, Aug 18. http://www.telegraph.co.uk/culture/books/10246937/Margaret-Atwood-interview.html.

Hoogweg, Lars (2003). Extending DOP with insertion. In Bod et al. (2003), pages 317–335.

Hsu, Yu-Yin (2010). Comparing conversions of discontinuity in PCFG parsing. In *Proceedings of Treebanks and Linguistic Theories*, pages 103–113. http://hdl.handle.net/10062/15954.

Huang, Liang and David Chiang (2005). Better k-best parsing. In *Proceedings of IWPT*, pages 53–64. NB corrected version on author homepage: http://www.cis.upenn.edu/~lhuang3/huang-iwpt-correct.pdf.

Huybregts, Marinus A.C. (1976). Overlapping dependencies in Dutch. *Utrecht Working Papers in Linguistics*, 1:24–65.

Jackendoff, Ray and Eva Wittenberg (2014). What you can say without syntax: a hierarchy of grammatical complexity. In Newmeyer, F.J. and L. Preston, editors, *Measuring Grammatical Complexity*. Oxford University Press, Oxford.

Jautze, Kim, Corina Koolen, Andreas van Cranenburgh, and Hayco de Jong (2013). From high heels to weed attics: a syntactic investigation of chick lit and literature. In *Proc. of workshop Computational Linguistics for Literature*, pages 72–81. http://aclweb.org/

anthology/W13-1410.

Jautze, Kim, Andreas van Cranenburgh, and Corina Koolen (2016a). Topic modeling literary quality. In *Digital Humanities 2016: Conference Abstracts*, pages 233–237, Krákow, Poland. http://dh2016.adho.org/abstracts/95.

Jautze, Kim, Karina van Dalen-Oskam, Erica Nagelhout, Hayco de Jong, Corina Koolen, and Gertjan Filarksi (2016b). The development of a large online survey of readers' opinions. in submission.

Jiménez, Aí da, Fernando Berzal, and Juan-Carlos Cubero (2010). Frequent tree pattern mining: A survey. *Intelligent Data Analysis*, 14(6):603–622.

Joachims, Thorsten (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of ECML*, pages 137–142. Springer Berlin Heidelberg. http://dx.doi.org/10.1007/BFb0026683.

Jockers, Matthew L. (2013). *Macroanalysis: Digital methods and literary history*. University of Illinois Press.

Jockers, Matthew L. and David Mimno (2013). Significant themes in 19th-century literature. *Poetics*, 41(6):750–769. http://dx.doi.org/10.1016/j.poetic.2013.08.005.

Johnson, Mark (2002a). The DOP estimation method is biased and inconsistent. *Computational Linguistics*, 28(1):71–76. http://aclweb.org/anthology/J02-1005.

———— (2002b). A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of ACL*, pages 136–143. http://aclweb.org/anthology/P02-1018.

Joshi, Aravind K. (1985). How much context sensitivity is necessary for characterizing structural descriptions: Tree adjoining grammars. In Dowty, David R., Lauri Karttunen, and Arnold M. Zwicky, editors, *Natural language parsing: Psychological*, *computational and theoretical perspectives*, pages 206–250. Cambridge University Press, New York.

Kaeshammer, Miriam and Vera Demberg (2012). German and English treebanks and lexica for tree-adjoining grammars. In *Proceedings of LREC*, pages 1880–1887. http://www.lrec-conf.org/proceedings/lrec2012/pdf/398_Paper.pdf.

Kallmeyer, Laura (2009). A declarative characterization of different types of multicomponent tree adjoining grammars. *Research on Language and Computation*, 7(1):55–99.

———— (2010). *Parsing Beyond Context-Free Grammars*. Cognitive Technologies. Springer. http://dx.doi.org/10.1007/978-3-642-14846-0.

Kallmeyer, Laura and Wolfgang Maier (2010). Data-driven parsing with probabilistic linear context-free rewriting systems. In *Proceedings of COLING*, pages 537–545. http://aclweb.org/anthology/C10-1061.

Kallmeyer, Laura and Wolfgang Maier (2013). Data-driven parsing using probabilistic linear context-free rewriting systems. *Computational Linguistics*, 39(1):87–119. http://aclweb.org/anthology/J13-1006.

Kallmeyer, Laura, Wolfgang Maier, and Giorgio Satta (2009). Synchronous rewriting in treebanks. In *Proceedings of IWPT*. http://aclweb.org/anthology/W09-3810.

Kant, Immanuel (1790). *Critique of Judgement*. Macmillan, London. translated by J.H. Bernard (1914, 2nd ed. revised). http://oll.libertyfund.org/titles/1217.

Kaplan, Ronald M. and Joan Bresnan (1982). Lexical-functional grammar: A formal system for grammatical representation. In Bresnan, Joan, editor, *The mental representation of grammatical relations*, pages 173–281. The MIT Press, Cambridge: MA.

Karlsson, Fred (2007). Constraints on multiple center-embedding of clauses. *Journal of Linguistics*, 43(2):365–392.

Khmelev, Dmitri V. and Fiona J. Tweedie (2001). Using Markov chains for identification

of writer. *Literary and Linguistic Computing*, 16(3):299–307. http://llc.oxfordjournals.org/content/16/3/299.abstract.

Khurshudov, Artem (2015). Suddenly, a leopard print sofa appears. Blog post, May 27. http://rocknrollnerd.github.io/ml/2015/05/27/leopard-sofa.html.

Kidd, David Comer and Emanuele Castano (2013). Reading literary fiction improves theory of mind. *Science*, 342(6156):377–380.

Kirby, Simon (2002). Learning, bottlenecks and the evolution of recursive syntax. In Briscoe, T., editor, *Linguistic Evolution through Language Acquisition: Formal and Computational Models*, pages 173–204. Cambridge University Press.

Klein, Dan and Christopher D. Manning (2003). Accurate unlexicalized parsing. In *Proceedings of ACL*, volume 1, pages 423–430. http://aclweb.org/anthology/P03-1054.

Koenig, Jean-Pierre and Karin Michelson (2014). Deconstructing SYNtax. In Müller, Stefan, editor, *Proceedings of the 21st International Conference on Head-Driven Phrase Structure Grammar*, pages 114–134, Stanford, CA. CSLI Publications.

Koster, Jan (1975). Dutch as an SOV language. *Linguistic analysis*, 1(2):111–136.

Kraskov, Alexander, Harald Stögbauer, and Peter Grassberger (2004). Estimating mutual information. *Physical review E*, 69(6):066138. http://journals.aps.org/pre/abstract/10.1103/PhysRevE.69.066138.

Kuhlmann, Marco (2013). Mildly non-projective dependency grammar. *Computational Linguistics*, 39(2):355–387. http://aclweb.org/anthology/J13-2004.

Kuhlmann, Marco and Giorgio Satta (2009). Treebank grammar techniques for non-projective dependency parsing. In *Proceedings of EACL*, pages 478–486. http://aclweb.org/anthology/E09-1055.

Le, Quoc V. and Tomas Mikolov (2014). Distributed representations of sentences and documents. In *Proceedings of ICML*, pages 1188–1196. http://jmlr.org/proceedings/papers/v32/le14.pdf.

Levy, Roger (2005). *Probabilistic models of word order and syntactic discontinuity*. PhD thesis, Stanford University.

Levy, Roger and Christopher D. Manning (2004). Deep dependencies from context-free statistical parsers: correcting the surface dependency approximation. In *Proceedings of ACL*, pages 327–334. http://aclweb.org/anthology/P04-1042.

Liberman, Mark (2015a). Moar verbs. Language Log post, March 24. http://languagelog.ldc.upenn.edu/nll/?p=18398.

———— (2015b). Reproducible computational experiments. Presentation at workshop Statistical Challenges in Assessing and Fostering the Reproducibility of Scientific Results, February 26. http://languagelog.ldc.upenn.edu/myl/LibermanCATS02262015.pdf.

Liu, Haitao (2008). Dependency distance as a metric of language comprehension difficulty. *Journal of Cognitive Science*, 9(2):159–191. http://j-cs.org/gnuboard/bbs/board.php?bo_table=__vol009i2&wr_id=3.

Lopez, Adam (2007). Hierarchical phrase-based translation with suffix arrays. In *Proceedings of EMNLP-CoNLL*, pages 976–985. http://aclweb.org/anthology/D07-1104.

Louis, Annie and Ani Nenkova (2013). What makes writing great? first experiments on article quality prediction in the science journalism domain. *Transactions of the Association for Computational Linguistics*, 1:341–352. http://aclweb.org/anthology/Q13-1028.

Louwerse, Max (2004). Semantic variation in idiolect and sociolect: Corpus linguistic evidence from literary texts. *Computers and the Humanities*, 38(2):207–221.

Louwerse, Max, Nick Benesh, and Bin Zhang (2008). Computationally discriminating

literary from non-literary texts. In Zyngier, S., M. Bortolussi, A. Chesnokova, and J. Auracher, editors, *Directions in empirical literary studies: In honor of Willie Van Peer*, pages 175–191. John Benjamins Publishing Company, Amsterdam.

Maier, Wolfgang, Miriam Kaeshammer, Peter Baumann, and Sandra Kübler (2014). Discosuite – a parser test suite for German discontinuous structures. In *Proceedings of LREC*. http://www.lrec-conf.org/proceedings/lrec2014/pdf/230_Paper.pdf.

Maier, Wolfgang, Miriam Kaeshammer, and Laura Kallmeyer (2012). PLCFRS parsing revisited: Restricting the fan-out to two. In *Proceedings of TAG*, volume 11. http://wolfgang-maier.net/pub/tagplus12.pdf.

Maier, Wolfgang and Timm Lichte (2011). Characterizing discontinuity in constituent treebanks. In *Proceedings of Formal Grammar 2009*, pages 167–182. Springer.

Maier, Wolfgang and Anders Søgaard (2008). Treebanks and mild context-sensitivity. In *Proceedings of Formal Grammar 2008*, pages 61–76.

Manning, Christopher D. and Hinrich Schütze (1999). *Foundations of statistical natural language processing*. MIT Press.

Marcus, Mitchell P., Mary Ann Marcinkiewicz, and Beatrice Santorini (1993). Building a large annotated corpus of English: The Penn treebank. *Computational linguistics*, 19(2):313–330. http://aclweb.org/anthology/J93-2004.

Markov, Andrei A. (1913). Primer statisticheskogo issledovanija nad tekstom "Evgenija Onegina" illjustrirujuschij svjaz' ispytanij v tsep (an example of statistical study on the text of Eugene Onegin illustrating the linking of events to a chain). *Izvestija Imp. Akademii nauk*, serija VI(3):153–162.

——— (1916). Ob odnom primenenii statisticheskogo metoda (on some application of statistical method). *Izvestija Imp. Akademii nauk*, serija VI(4):239–242.

Martens, Scott (2010). Varro: an algorithm and toolkit for regular structure discovery in treebanks. In *Proceedings of COLING*, pages 810–818. http://aclweb.org/anthology/C10-2093.

Martindale, Colin (1990). *The clockwork muse: The predictability of artistic change*. Basic Books, New York.

Martindale, Colin and Audrey Dailey (1995). I.a. richards revisited: Do people agree in their interpretations of literature? *Poetics*, 23(4):299–314. http://www.sciencedirect.com/science/article/pii/0304422X94000252.

Matsuzaki, Takuya, Yusuke Miyao, and Jun'ichi Tsujii (2005). Probabilistic CFG with latent annotations. In *Proceedings of ACL*, pages 75–82.

McCallum, Andrew Kachites (2002). Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

McCawley, James D. (1982). Parentheticals and discontinuous constituent structure. *Linguistic Inquiry*, 13(1):91–106. http://www.jstor.org/stable/4178261.

Meinshausen, Nicolai and Peter Bühlmann (2010). Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473.

Menon, Rohith K. and Yejin Choi (2011). Domain independent authorship attribution without domain adaptation. In *Proceedings of Recent Advances in Natural Language Processing*, pages 309–315.

Meyer, Jim (1997). What is literature. *Work Paper of the Summer Institute of Linguistics*, 41:40–53. http://files.eric.ed.gov/fulltext/ED461270.pdf.

Miall, David S. (2006). *Literary reading: empirical & theoretical studies*. Peter Lang.

Miall, David S. and Don Kuiken (1999). What is literariness? three components of literary reading. *Discourse Processes*, 28(2):121–138.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Gregory S. Corrado, , and Jeffrey Dean (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, pages 3111–3119.

Moretti, Franco (2005). *Graphs, maps, trees: abstract models for a literary history*. Verso.

Moschitti, Alessandro (2006a). Efficient convolution kernels for dependency and constituent syntactic trees. In *ECML*, pages 318–329. Proceedings of ECML.

———— (2006b). Making tree kernels practical for natural language learning. In *Proceedings of EACL*, pages 113–120. http://aclweb.org/anthology/E06-1015.

Moschitti, Alessandro, Daniele Pighin, and Roberto Basili (2008). Tree kernels for semantic role labeling. *Computational Linguistics*, 34(2):193–224. http://aclweb.org/anthology/J08-2003.

Mosteller, Frederick and David Wallace (1964). *Inference and disputed authorship: The Federalist*. Addison-Wesley.

Mukarovsky, Jan (1964). Standard language and poetic language. *A Prague School reader on aesthetics, literary structure, and style*, pages 17–30.

Napoles, Courtney, Matthew Gormley, and Benjamin Van Durme (2012). Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 95–100. http://aclweb.org/anthology/W12-30.

Nederhof, Mark-Jan and Heiko Vogler (2014). Hybrid grammars for discontinuous parsing. In *Proceedings of COLING*, pages 1370–1381. http://aclweb.org/anthology/C14-1130.

Nevins, Andrew, David Pesetsky, and Cilene Rodrigues (2009). Pirahã exceptionality: A reassessment. *Language*, 85(2):355–404.

Nivre, Joakim, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi (2007). MaltParser: a language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135. http://dx.doi.org/10.1017/S1351324906004505.

O'Donnell, Timothy J., Joshua B. Tenenbaum, and Noah D. Goodman (2009). Fragment grammars: Exploring computation and reuse in language. Technical Report MIT-CSAIL-TR-2009-013, MIT CSAIL. http://hdl.handle.net/1721.1/44963.

Ojeda, Almerindo E. (1988). A linear precedence account of cross-serial dependencies. *Linguistics and Philosophy*, 11(4):457–492.

Okakura, Kakuzō (1906). *The Book of Tea*. Duffield & Company.

Oostdijk, Nelleke, Martin Reynaert, Véronique Hoste, and Ineke Schuurman (2013). The construction of a 500-million-word reference corpus of contemporary written dutch. In *Essential speech and language technology for Dutch*, pages 219–247. Springer.

Pander Maat, Henk, Rogier Kraf, Antal van den Bosch, Nick Dekker, Maarten van Gompel, Suzanne Kleijn, Ted Sanders, and Ko van der Sloot (2014). T-Scan: a new tool for analyzing Dutch text. *Computational Linguistics in the Netherlands Journal*, 4:53–74. http://www.clinjournal.org/sites/clinjournal.org/files/05-PanderMaat-etal-CLIN2014.pdf.

Pauls, Adam and Dan Klein (2009). Hierarchical search for parsing. In *Proceedings of NAACL-HLT*, pages 557–565. http://aclweb.org/anthology/N09-1063.

Pedregosa, Fabian, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). Glove: Global

vectors for word representation. In *Proceedings of EMNLP*, volume 14, pages 1532–1543.

Peters, P. Stanley and R. W. Ritchie (1973). On the generative power of transformational grammars. *Information Sciences*, 6:49–83. http://dx.doi.org/10.1016/0020-0255(73)90027-3.

Petrov, Slav (2010). Products of random latent variable grammars. In *Proceedings of NAACL-HLT*, pages 19–27. http://aclweb.org/anthology/N10-1003.

Petrov, Slav, Leon Barrett, Romain Thibaux, and Dan Klein (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of ACL*, pages 433–440. http://aclweb.org/anthology/P06-1055.

Pighin, Daniele and Alessandro Moschitti (2010). On reverse feature engineering of syntactic tree kernels. In *Proceedings of CoNLL*, pages 223–233. http://aclweb.org/anthology/W10-2926.

Pike, Kenneth L. (1943). Taxemes and immediate constituents. *Language*, 19(2):65–82. http://www.jstor.org/stable/409840.

Post, Matt and Shane Bergsma (2013). Explicit and implicit syntactic features for text classification. In *Proceedings of ACL*, pages 866–872. http://aclweb.org/anthology/P13-2150.

Post, Matt and Daniel Gildea (2009). Bayesian learning of a tree substitution grammar. In *Proceedings of the ACL-IJCNLP 2009 Conference*, *Short Papers*, pages 45–48. http://aclweb.org/anthology/P09-2012.

Prescher, Detlef, Remko Scha, Khalil Sima'an, and Andreas Zollmann (2003). On the statistical consistency of DOP estimators. In *Proceedings CLIN 2003*. http://www.cnts.ua.ac.be/clin2003/proc/07Prescher.pdf.

Putnam, Hilary (1975). The meaning of 'meaning'. *Minnesota Studies in the Philosophy of Science*, 7:131–193.

Raghavan, Sindhu, Adriana Kovashka, and Raymond Mooney (2010). Authorship attribution using probabilistic context-free grammars. In *Proceedings of ACL*, pages 38–42. http://aclweb.org/anthology/P10-2008.

Ranta, Aarne (2011). *Grammatical framework: Programming with multilingual grammars*. CSLI Publications.

Richards, I.A. (1929). *Practical criticism: A study of literary judgment*. Harcourt Brace Jovanovich, New York.

Riddell, Allen (2014). How to read 22,198 journal articles: Studying the history of german studies with topic models. In Erlin, Matt and Lynne Tatlock, editors, *Distant Readings: Topologies of German culture in the long nineteenth century*, pages 91–114. Camden House, Rochester, New York.

Rieck, Konrad, Tammo Krueger, Ulf Brefeld, and Klaus-Robert Müller (2010). Approximate tree kernels. *Journal of Machine Learning Research*, 11:555–580. http://jmlr.org/papers/volume11/rieck10a/rieck10a.pdf.

Roark, Brian, Kristy Hollingshead, and Nathan Bodenstab (2012). Finite-state chart constraints for reduced complexity context-free parsing pipelines. *Computational Linguistics*, 38(4):719–753. http://aclweb.org/anthology/J12-4002.

Roorda, Dirk, Gino Kalkman, Martijn Naaijer, and Andreas van Cranenburgh (2014). LAF-fabric: a data analysis tool for linguistic annotation framework with an application to the hebrew bible. *Computational Linguistics in the Netherlands Journal*, 4:105–120. http://clinjournal.org/sites/clinjournal.org/files/08-Roorda-etal-CLIN2014.pdf.

Rousseau, Jean-Jacques (1782/1889). *Les Confessions*, volume Tome 2. Launette, Paris.

https://fr.wikisource.org/wiki/Les_Confessions_(Rousseau)/Livre_VIII.

Salganik, Matthew J., Peter Sheridan Dodds, and Duncan J. Watts (2006). Experimental study of inequality and unpredictability in an artificial cultural market. *Science*, 311(5762):854–856.

Sanders, Ewoud (2013). De meest besproken spa tie ooit. NRC Handelsblad, April 22. *The most debated spa ce ever*, opinion piece, accessed February 27th, 2016. http://www.nrc.nl/handelsblad/2013/04/22/de-meest-besproken-spa-tie-ooit-1236528.

Sangati, Federico and Andreas van Cranenburgh (2015). Multiword expression identification with recurring tree fragments and association measures. In *Proceedings of the 11th Workshop on Multiword Expressions*, pages 10–18. http://aclweb.org/anthology/W15-0902.

Sangati, Federico and Willem Zuidema (2011). Accurate parsing with compact tree-substitution grammars: Double-DOP. In *Proceedings of EMNLP*, pages 84–95. http://aclweb.org/anthology/D11-1008.

Sangati, Federico, Willem Zuidema, and Rens Bod (2010). Efficiently extract recurring tree fragments from large treebanks. In *Proceedings of LREC*, pages 219–226. http://lrec-conf.org/proceedings/lrec2010/pdf/613_Paper.pdf.

Santorini, Beatrice and Anthony Kroch (2007). The syntax of natural language: An online introduction using the trees program. http://www.ling.upenn.edu/~beatrice/syntax-textbook.

Scha, Remko (1990). Language theory and language technology; competence and performance. In de Kort, Q.A.M. and G.L.J. Leerdam, editors, *Computertoepassingen in de Neerlandistiek*, pages 7–22. LVVN, Almere, the Netherlands. Original title: Taaltheorie en taaltechnologie; competence en performance. English translation: http://iaaa.nl/rs/LeerdamE.html.

Schabes, Yves and Richard C. Waters (1995). Tree insertion grammar: cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics*, 21(4):479–513. http://aclweb.org/anthology/J95-4002.

Scheepers, Christoph, Patrick Sturt, Catherine J. Martin, Andriy Myachykov, Kay Teevan, and Izabela Viskupova (2011). Structural priming across cognitive domains from simple arithmetic to relative-clause attachment. *Psychological Science*, 22(10):1319–1326.

Schmid, Helmut (2004). Efficient parsing of highly ambiguous context-free grammars with bit vectors. In *Proceedings of COLING '04*. http://aclweb.org/anthology/C04-1024.

———— (2006). Trace prediction and recovery with unlexicalized PCFGs and slash features. In *Proceedings of COLING-ACL*, pages 177–184. http://aclweb.org/anthology/P06-1023.

Schuler, William, Samir AbdelRahman, Tim Miller, and Lane Schwartz (2010). Broad-coverage parsing using human-like memory constraints. *Computational Linguistics*, 36(1):1–30. http://aclweb.org/anthology/J10-1001.

Schuler, William, David Chiang, and Mark Dras (2000). Multi-component TAG and notions of formal power. In *Proceedings of ACL*, pages 448–455. http://aclweb.org/anthology/P00-1057.

Schwarm, Sarah E. and Mari Ostendorf (2005). Reading level assessment using support vector machines and statistical language models. In *Proceedings of ACL*, pages 523–530. http://aclweb.org/anthology/P05-1065.

Seki, Hiroyuki, Takahashi Matsumura, Mamoru Fujii, and Tadao Kasami (1991). On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229.

Shalev-Shwartz, Shai, Yoram Singer, Nathan Srebro, and Andrew Cotter (2011). Pegasos: Primal estimated sub-gradient solver for SVM. *Mathematical programming*, 127(1):3–30.

Shalizi, Cosma Rohilla (2012). Review of peter bühlmann and sara van de geer (2011), statistics for high-dimensional data: Methods, theory and applications. The Bactra Review 161, September 2. Accessed March 1, 2016. http://bactra.org/reviews/stats-for-high-d-data/.

Shieber, Stuart M. (1985). Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8:333–343.

Shindo, Hiroyuki, Yusuke Miyao, Akinori Fujino, and Masaaki Nagata (2012). Bayesian symbol-refined tree substitution grammars for syntactic parsing. In *Proceedings of ACL*, pages 440–448. http://aclweb.org/anthology/P12-1046.

Shmueli, Galit (2010). To explain or to predict? *Statistical Science*, 25(3):289–310. http://dx.doi.org/10.1214/10-STS330.

Sima'an, Khalil (1997). Efficient disambiguation by means of stochastic tree substitution grammars. In Jones, D. and H. Somers, editors, *New Methods in Language Processing*, pages 178–198. UCL Press, UK.

—— (2000). Tree-gram parsing lexical dependencies and structural relations. In *Proceedings of ACL*, pages 37–44.

—— (2002). Computational complexity of probabilistic disambiguation. *Grammars*, 5(2):125–151.

Simpson, Edward H. (1951). The interpretation of interaction in contingency tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 13(2):238–241. http://www.jstor.org/stable/2984065.

Skut, Wojciech, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit (1997). An annotation scheme for free word order languages. In *Proceedings of ANLP*, pages 88–95. http://aclweb.org/anthology/A97-1014.

Smith, George (2003). A brief introduction to the tiger treebank, version 1. Technical report, Potsdam Universität. http://www.uni-potsdam.de/u/germanistik/ls_dgs/tiger1-intro.pdf.

Socher, Richard, John Bauer, Christopher D. Manning, and Andrew Y. Ng (2013). Parsing with compositional vector grammars. In *Proceedings of ACL*, pages 455–465.

Stamatatos, Efstathios (2009). A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556. http://dx.doi.org/10.1002/asi.21001.

Steedman, Mark (2000). *The Syntactic Process*. The MIT Press.

Swanson, Ben and Eugene Charniak (2013). Extracting the native language signal for second language acquisition. In *Proceedings of NAACL-HLT*, pages 85–94. http://aclweb.org/anthology/N13-1009.

Swanson, Ben, Elif Yamangil, Eugene Charniak, and Stuart Shieber (2013). A context free TAG variant. In *Proceedings of the ACL*, pages 302–310. http://aclweb.org/anthology/P13-1030.

Swanson, Benjamin and Eugene Charniak (2012). Native language detection with tree substitution grammars. In *Proceedings of ACL*, pages 193–197. http://aclweb.org/anthology/P12-2038.

Taylor, Ann, Mitchell Marcus, and Beatrice Santorini (2003). The Penn treebank: an

overview. In Abeillé, Anne, editor, *Treebanks: Building and using parsed corpora*, pages 5–22. Springer. http://dx.doi.org/10.1007/978-94-010-0201-1_1.

Telljohann, Heike, Erhard Hinrichs, and Sandra Kübler (2004). The Tüba-D/Z treebank: Annotating German with a context-free backbone. In *Proceedings of LREC*, pages 2229–2235. http://www.lrec-conf.org/proceedings/lrec2004/pdf/135.pdf.

Telljohann, Heike, Erhard W. Hinrichs, Sandra Kübler, Heike Zinsmeister, and Kathrin Beck (2012). Stylebook for the Tübingen treebank of written German (TüBa-D/Z). Technical report, Seminar für Sprachwissenschaft, Germany. http://www.sfs.uni-tuebingen.de/fileadmin/static/ascl/resources/tuebadz-stylebook-1201.pdf.

Torkkola, Kari (2003). Feature extraction by non parametric mutual information maximization. *Journal of Machine Learning Research*, 3:1415–1438. http://www.jmlr.org/papers/volume3/torkkola03a/torkkola03a.pdf.

Towey, Cathleen A. (2000). Flow. *The Acquisitions Librarian*, 13(25):131–140. http://dx.doi.org/10.1300/J101v13n25_11.

Trautwein, Marten H. (1995). *Computational pitfalls in tractable grammar formalisms*. PhD thesis, University of Amsterdam. http://www.illc.uva.nl/Research/Publications/Dissertations/DS-1995-15.text.ps.gz.

Underwood, Ted (2015). The literary uses of high-dimensional space. *Big Data & Society*, 2(2). http://bds.sagepub.com/content/2/2/2053951715602494.

Underwood, Ted and Jordan Sellers (2015). How quickly do literary standards change? In: *figshare*. https://dx.doi.org/10.6084/m9.figshare.1418394.

van Cranenburgh, Andreas (2012a). Efficient parsing with linear context-free rewriting systems. In *Proceedings of EACL*, pages 460–470. Corrected version: http://andreasvc.github.io/eacl2012corrected.pdf.

———— (2012b). Extracting tree fragments in linear average time. Technical Report PP-2012-18, Institute for Logic, Language and Computation (ILLC). http://dare.uva.nl/en/record/421534.

———— (2012c). Literary authorship attribution with phrase-structure fragments. In *Proceedings of CLFL*, pages 59–63. Revised version: http://andreasvc.github.io/clfl2012.pdf.

———— (2014). Extraction of phrase-structure fragments with a linear average time tree kernel. *Computational Linguistics in the Netherlands Journal*, 4:3–16. http://www.clinjournal.org/sites/default/files/01-Cranenburgh-CLIN2014.pdf.

van Cranenburgh, Andreas and Rens Bod (2013). Discontinuous parsing with an efficient and accurate DOP model. In *Proceedings of IWPT*, pages 7–16. http://www.illc.uva.nl/LaCo/CLS/papers/iwpt2013parser_final.pdf.

van Cranenburgh, Andreas and Rens Bod (2017). A data-oriented model of literary language. In *Proceedings of EACL*, pages 1228–1238. http://aclweb.org/anthology/E17-1115.

van Cranenburgh, Andreas and Corina Koolen (2015). Identifying literary texts with bigrams. In *Proc. of workshop Computational Linguistics for Literature*, pages 58–67. http://aclweb.org/anthology/W15-0707.

van Cranenburgh, Andreas, Remko Scha, and Rens Bod (2016). Data-oriented parsing with discontinuous constituents and function tags. *Journal of Language Modelling*, 4(1):57–111. http://dx.doi.org/10.15398/jlm.v4i1.100.

van Cranenburgh, Andreas, Remko Scha, and Federico Sangati (2011). Discontinuous data-oriented parsing: A mildly context-sensitive all-fragments grammar. In *Proceedings of SPMRL*, pages 34–44. http://aclweb.org/anthology/W11-3805.

van der Beek, Leonoor, Gosse Bouma, Robert Malouf, and Gertjan van Noord (2002). The Alpino dependency treebank. *Language and Computers*, 45(1):8–22. http://www.let.rug.nl/vannoord/papers/treebank.pdf.

van der Wouden, Ton, Heleen Hoekstra, Michael Moortgat, Bram Renmans, and Ineke Schuurman (2002). Syntactic analysis in the spoken Dutch corpus (CGN). In *Proceedings of LREC*, pages 768–773. http://www.lrec-conf.org/proceedings/lrec2002/pdf/71.pdf.

van Eynde, Frank (2005). Part of speech tagging en lemmatisering van het D-COI corpus. *POS tagging and lemmatization of the D-COI corpus*, tech report, July 2005. http://www.ccl.kuleuven.ac.be/Papers/DCOIpos.pdf.

van Gompel, Maarten, Ko van der Sloot, and Antal van den Bosch (2012). Ucto: Unicode tokeniser. http://ilk.uvt.nl/ucto/ucto_manual.pdf.

Van Noord, Gertjan (2009). Huge parsed corpora in Lassy. In *Proceedings of TLT7*, Groningen, The Netherlands. LOT.

van Noord, Gertjan, Ineke Schuurman, and Gosse Bouma (2011). Lassy syntactische annotatie. *Lassy Syntactic Annotation*, tech report, revision 19455. http://www.let.rug.nl/vannoord/Lassy/sa-man_lassy.pdf.

van Peer, Willie, editor (2008). *The Quality of Literature: Linguistic studies in literary evaluation*, volume 4 of *Linguistic Applications to Literature*. John Benjamins Publishing.

van Schijndel, Marten and William Schuler (2015). Hierarchic syntax improves reading time prediction. In *Proceedings of NAACL*, pages 1597–1605. http://aclweb.org/anthology/N15-1183.

van Wingerden, Wouter and Pepijn Hendriks (2015). *Dat Hoor Je Mij Niet Zeggen: De allerbeste taalclichés*. Thomas Rap, Amsterdam. Transl.: You didn't hear me say that: The very best linguistic clichés.

Versley, Yannick (2014). Experiments with easy-first nonprojective constituent parsing. In *Proceedings of SPMRL-SANCL 2014*, pages 39–53. http://aclweb.org/anthology/W14-6104.

Vijay-Shanker, K., David J. Weir, and Aravind K. Joshi (1987). Characterizing structural descriptions produced by various grammatical formalisms. In *Proceedings of ACL*, pages 104–111. http://aclweb.org/anthology/P87-1015.

Wang, Sida and Christopher Manning (2012). Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of ACL*, pages 90–94. http://aclweb.org/anthology/P12-2018.

Weir, David J. (1988). *Characterizing mildly context-sensitive grammar formalisms*. PhD thesis, University of Pennsylvania. http://repository.upenn.edu/dissertations/AAI8908403/.

Wells, Rulon S. (1947). Immediate constituents. *Language*, 23(2):81–117. http://www.jstor.org/stable/410382.

Wiersma, Wybo, John Nerbonne, and Timo Lauttamus (2011). Automatically extracting typical syntactic differences from corpora. *Literary and Linguistic Computing*, 26(1):107–124. http://llc.oxfordjournals.org/content/26/1/107.abstract.

Xia, Fei, Chung-Hye Han, Martha Palmer, and Aravind Joshi (2001). Automatically extracting and comparing lexicalized grammars for different languages. In *Proceedings of IJCAI*, pages 1321–1330.

Yamangil, Elif and Stuart Shieber (2012). Estimating compact yet rich tree insertion grammars. In *Proceedings of ACL*, pages 110–114. http://aclweb.org/anthology/P12-2022.

Yáñez-Bouza, Nuria (2006). Prescriptivism and preposition stranding in eighteenth-century prose. *Historical Sociolinguistics and Sociohistorical Linguistics*, 6. http://www.let.leidenuniv.nl/hsl_shl/preposition%20stranding.htm.

Yu, Lei and Huan Liu (2004). Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224. http://jmlr.org/papers/volume5/yu04a/yu04a.pdf.

Yule, G. Udny (1903). Notes on the theory of association of attributes in statistics. *Biometrika*, 2(2):121–134. http://www.jstor.org/stable/2331677.

Zhang, Kaizhong, Rick Statman, and Dennis Shasha (1992). On the editing distance between unordered labeled trees. *Information processing letters*, 42(3):133–139.

Zollmann, Andreas (2004). A consistent and efficient DOP estimator. Master's thesis, University of Amsterdam. http://www.science.uva.nl/pub/theory/illc/researchreports/MoL-2004-02.text.pdf.

Zollmann, Andreas and Khalil Sima'an (2005). A consistent and efficient estimator for data-oriented parsing. *Journal of Automata Languages and Combinatorics*, 10(2/3):367–388. https://staff.fnwi.uva.nl/k.simaan/D-Papers/JALCsubmit.pdf.

Zuidema, Willem (2006). Theoretical evaluation of estimation methods for data-oriented parsing. In *Proceedings of EACL*, pages 183–186.

——— (2007). Parsimonious data-oriented parsing. In *Proceedings of EMNLP-CoNLL*, pages 551–560. http://aclweb.org/anthology/D07-1058.

Glossary of acronyms

Parse tree labels

The following describes the set of labels used in the syntactic annotation of the Dutch treebank Lassy. Based on van Noord et al. (2011); my translations.

Phrasal Label	Description
AP	adjectival phrase
ADVP	adverbial phrase
AHI	'aan het …' infinitival clause (continuous aspect)
CONJ	conjunctive clause
CP	complementizer phrase with subordinating conjunction
DETP	determiner phrase
DU	discourse unit
INF	bare infinitive clause
NP	noun phrase
OTI	'om te …' infinitival, reduced relative clause
PPART	perfect/passive participle
PP	prepositional phrase
PPRES	present participle
REL	relative clause
SMAIN	declarative clause (verb second)
SSUB	subordinating clause (verb final)
SVAN	clause prefixed by 'van' (quotative)
SV1	verb-initial clause (yes/no-question, imperative)
TI	infinitival clause prefixed by 'te' (to)
WHREL	relative clause with wh-antecedent
WHSUB	wh-constituent: subordinate clause
WHQ	wh-constituent: main clause

Function Tag	Description
APP	appositive
BODY	body (of complementizer phrase)
CMP	complementizer
CNJ	conjunct
CRD	coordinating conjunction
DET	determiner
DLINK	discourse link
DP	discourse part
HD	head
HDF	final element of circumposition
LD	locative or directional complement
ME	measure (duration, weight, . . .) complement
MOD	adverbial modifier
MWP	part of multi-word unit
NUCL	sentence nucleus
OBCOMP	comparative complement
OBJ1	direct object
OBJ2	secondary object
PC	verbal complement prefixed by preposition
POBJ1	dummy direct object
PREDC	predicative complement
PREDM	predication of state during action
RHD	head of relative clause
SAT	satellite, initial or final
SE	compulsory reflexive object
SU	subject
SUP	dummy subject
SVP	separable verb part
TAG	interjections, reduced question/imperative, direct speech introduction
VC	verbal complement
WHD	head of question

The following describes the DCOI tag set, taken from van Eynde (2005). The tags consist of a coarse POS tag, listed below, and a set of fine-grained morphological features that combine with them. We list only the selection of morphological features that is used in this work.

POS Tag	Description
let	punctuation
spec	part of proper noun
bw	adverb
vg	conjunction
lid	determiner
vnw	pronoun
tw	cardinal
ww	verb
adj	adjective
n	noun
tsw	interjection
vz	preposition

Morphological Feature	Description
eigen	proper noun
pron, det	personal/reflexive, possessive pronoun
init, fin	pre- vs postposition
neven, onder	coordinating, subordinating conjunction
prenom, nom, vrij	prenominal, nominal, or free standing
pv, inf	finite, infinite verb
vd, od	past, present participle

Other acronyms

Acronym	Expanded version
2DOP	Double-DOP, DOP based on recurring fragments; Section 3.4
BOW	Bag-of-Words; Section 4.2
CFG	Context-Free Grammar; Section 1.2
DOP	Data-Oriented Parsing; Section 1.3
DOP1	All-fragments DOP with RFE
DTSG	Discontinuous TSG; Section 3.2.2
EWE	Equal Weights Estimate; Section 1.3.2
LCFRS	Linear Context-Free Rewriting Systems; Section 3.2.1
LDA	Latent Dirichlet Allocation; Section 6.3
MPD	Most Probable Derivation; Section 1.3.1
MPP	Most Probable Parse; Section 1.3.1
NLP	Natural Language Processing
NUR	*Nederlandstalige Uniforme Rubriekscode* (roughly, 'uniform categorization of Dutch language [novels].'); Section 5.2.1
OLS	Ordinary Least Squares; Section 4.2
R^2	Coefficient of Determination; Section 4.2.4
RFE	Relative Frequency Estimate; Section 1.3.2
RMS	Root Mean Square error; Section 4.2.4
PDTSG	Probabilistic DTSG; Section 3.2.2
PLCFRS	Probabilistic Linear Context-Free Rewriting Systems; Section 3.2.1
PTB	Penn Treebank
PTSG	Probabilistic Tree-Substitution Grammar; Section 1.3.1
POS	Part-of-Speech
SVM	Support Vector Machines; Section 4.2
TSG	Tree-Substitution Grammar; Section 1.3.1
WSJ	Wall Street Journal section of PTB
PCFG	Probabilistic Context-Free Grammar; Section 1.2

Abstract

Tʜɪs ᴛʜᴇsɪs applies the Data-Oriented Parsing framework in two areas: parsing & literature. The data-oriented approach rests on the assumption that re-use of chunks of training data can be detected and exploited at test time. Syntactic tree fragments form the common thread in the thesis.

Chapter 2 presents a method to efficiently extract them from treebanks, based on heuristics of re-occurrence. This method is thus able to discover the potential building blocks of large corpora. Chapter 3 then develops a multi-lingual statistical parser based on tree-substitution grammar that handles discontinuous constituents and function tags. We show how a mildly context-sensitive grammar can be employed to produce discontinuous constituents, and then compare this to an approximation that stays within the efficiently parsable context-free framework. The conclusion from the empirical evaluation is that tree fragments allow the grammar to adequately capture the statistical regularities of non-local relations, without the need for the increased generative capacity of mildly context-sensitive grammar.

The second part investigates what separates literary from other novels. Aside from an introduction in Chapter 4 to machine learning we discuss the difference between explanation and prediction.

Chapter 5 discusses the data used for this investigation. We work with a corpus of novels and a reader survey with ratings of how literary novels are perceived to be. While considerable questions remain with respect to whether a survey of the general public is an appropriate instrument to probe the concept of literature, when viewed as a barometer of public opinion we may consider the basic question of whether such opinions are at all predictable. The first goal is therefore to find out the extent to which the literary ratings can be predicted from the texts; the second, more challenging goal is to characterize the kind of patterns that are predictors of more or less literary texts.

Chapter 6 establishes baselines for this question. We show that literary novels contain less adjectives and adverbs than non-literary novels, and present several simple measures that are significantly correlated with the literary ratings, such as vocabulary richness and text compressibility. Cliché expressions is established as a negative marker of literary language. A topic model is developed of the

corpus, revealing a number of clearly interpretable themes in the novels.

Special attention is given in Chapter 7 to syntactic aspects, as investigated in the first part. The syntactic methods are contrasted with lexical baselines based on bigrams (sequences of two consecutive words). The combination of lexical and syntactic features gives an improvement, and the syntactic features are more interpretable.

In the end, the literary ratings are predictable from textual features to a large extent. While it is not possible to infer a causal relation between these textual features and the ratings from the survey participants, this result clearly rules out the notion that these value-judgments of literary merit were arbitrary, or predominantly determined by factors beyond the text.

Samenvatting

D IT PROEFSCHRIFT past Data-Geörienteerd Ontleden toe op twee proble-
men: ontleden & literatuur. De Data-Geörienteerde aanpak exploiteert
de aanname dat hergebruik van brokstukken uit eerdere taalervarin-
gen kan worden gedetecteerd en toegepast op nieuwe zinnen. Syntactische
boomfragmenten vormen de gemene deler in dit proefschrift.

Hoofdstuk 2 presenteert een methode om ze efficiënt te extraheren uit verza-
melingen parseerbomen, gebaseerd op de heuristiek dat relevante fragmenten
meerdere malen voorkomen. Deze methode kan zodoende potentiële bouwste-
nen ontdekken van grote corpora. Hoofdstuk 3 zet vervolgens een methode
uiteen om deze fragmenten te gebruiken bij de ontwikkeling van een multi-
linguaal statistisch model van parseren, gebruikmakend van een zogenaamde
boomsubstituerende grammatica die boomfragmenten samenvoegt tot volledige
analyses. Deze grammatica produceert analyses met discontinue constituenten
en grammaticale functierelaties. We laten zien hoe een mild contextgevoelige
grammatica gebruikt kan worden om discontinue constituenten te produceren,
en vergelijken dit model vervolgens met een benadering die binnen het efficiënt
parseerbare contextvrije formalisme blijft. De conclusie van de empirische eva-
luatie is dat boomfragmenten het mogelijk maken voor de grammatica om de
statistische regulariteit van niet-lokale afhankelijkheden adequaat te omvatten,
zonder daarbij de toegevoegde generatieve capaciteit van mild contextgevoelige
grammatica's nodig te hebben.

Het tweede deel behandelt de vraag wat literaire van andere romans onder-
scheid. Hoofdstuk 5 behandelt de data die wordt gebruikt voor dit onderzoek.
We gebruiken een corpus van romans en een lezersonderzoek met lezersmenin-
gen over hoe literair romans bevonden worden. Het eerste doel is te kwantifice-
ren in hoeverre 'literariteit' kan worden voorspeld aan de hand van tekstuele
kenmerken; het tweede doel is te karakteriseren welke kenmerken voorspellend
zijn voor literariteit.

Hoofdstuk 6 toont enkele basale modellen voor deze vraag. We laten zien dat
literaire romans minder bijvoeglijke en bijwoordelijke naamwoorden bevatten
dan niet-literaire romans, en presenteren verscheidene simpele maten met
een significante correlatie met de literaire oordelen, zoals de rijkheid van het

vocabulaire en de comprimeerbaarheid van de tekst. Cliché uitdrukkingen worden ingezet als een negatieve marker van literaire taal. Een zogeheten 'topic' model van het corpus wordt ontwikkeld, wat laat zien dat er een aantal duidelijk interpreteerbare thema's in de romans aanwezig zijn.

Speciale aandacht wordt in Hoofdstuk 7 besteed aan syntactische kenmerken, zoals behandeld in het eerste deel. De syntactische methoden worden gecontrasteerd met simpelere lexicale methoden gebaseerd op bigrammen (sequenties van twee opeenvolgende woorden). De combinatie van lexicale en syntactische kenmerken geeft een verbetering, en de syntactische kenmerken zijn beter te interpreteren.

Uiteindelijk is de conclusie dat de literaire oordelen in grote mate voorspelbaar zijn op basis van tekstkenmerken. Hoewel het niet mogelijk is om een direct oorzakelijk verband aan te wijzen tussen de tekstkenmerken en de oordelen van proefpersonen, is toch duidelijk aangetoond dat de waardeoordelen over literariteit geenszins arbitrair zijn, noch in meerderheid bepaald door factoren buiten de tekst.