

# Using a Named Entity Tagger to Generalise Surface Matching Text Patterns for Question Answering

**Mark A. Greenwood and Robert Gaizauskas**

Natural Language Processing Group  
Department of Computer Science  
University of Sheffield, UK

# Outline of Presentation

---

- Introduction and overview
- The basic approach
- Limitations of the approach
- Generalising using an NE tagger
- Results to date
- Discussion and Conclusions
- Future Work

# Introduction and Overview

---

- **The basic question answering system outlined in this talk is similar to those developed by Soubbotin and Soubbotin (2001) and Ravichandran and Hovy (2002).**
  - These systems use a single resource consisting of a large number of surface matching text patterns.
  - These patterns are derived using simple machine learning techniques from a set of question-answer pairs and a corpus of documents in which the answer is known to occur.
- **While agreeing that these systems have enormous potential we aim to highlight, and provided a solution to, a problem that can be caused by overly specific patterns:**
  - They mainly result in systems which may be unable to answer specific types of questions.

# The Basic Approach

---

- **Learning patterns which can be used to find answers involves a two stage process:**
  - The first stage is to learn a set of patterns from a set of question-answer pairs.
  - The second stage involves assigning a precision to each pattern and discarding those patterns which are tied to a specific question-answer pair.
- **To explain the process we will use questions of the form “*When was X born?*”:**
  - As a concrete example we will use “*When was Mozart born?*”.
  - For which the question-answer pair is:
    - Mozart
    - 1756

# The Basic Approach

---

- **The first stage is to learn a set of patterns from the question-answer pairs for a specific question type:**
  - For each example the question and answer terms are submitted to Google and the top ten documents are downloaded.
  - Each document then has the question and answer terms replaced by AnCHoR and AnSWeR respectively.
  - Those sentences which contain both AnCHoR and AnSWeR are retained and joined together to create a single document.
  - This generated document is then used to build a token-level suffix tree, from which repeated strings containing both AnCHoR and AnSWeR and which do not span a sentence boundary are extracted as patterns.

# The Basic Approach

---

- **The result of the first stage is a set of patterns. For questions of the form “*When was X born?*” these may include:**

AnCHor ( AnSWeR -

From AnCHoR ( AnSWeR - 1791 )

AnCHor ( AnSWeR

- **Unfortunately some of these patterns are specific to the question used to generate them.**
- **So the second stage of the approach is concerned with filtering out these specific patterns to produce a set which can be used to answer unseen questions.**

# The Basic Approach

---

- **The second stage of the approach requires a different set of question-answer pairs to those used in the first stage:**
  - Within each of the top ten documents returned by Google, using only the question term: the question term is replaced by AnCHoR and the answer (if it is present) with AnSWeR.
  - Those sentences which contain AnCHoR are retained.
  - All of the patterns from the first stage are converted to regular expressions designed to capture the token which appears in place of AnSWeR.
  - Each regular expression is then matched against each sentence and along with each pattern two counts are maintained:  $C_a$  which is the total number of times this pattern has matched and  $C_c$  which counts the number of times AnSWeR was selected as the answer.
  - After a pattern has been matched against every sentence if  $C_c$  is less than 5 then it is discarded otherwise it's precision is calculated as  $C_c/C_a$  and the pattern is retained only if the precision is greater than 0.1.

# The Basic Approach

---

- **The result of assigning precision to patterns in this way is a set of precisions and regular expressions such as:**

0.967: AnCHoR \ ( ([^ ]+ ) -

0.566: AnCHoR \ ( ([^ ]+ )

0.263: AnCHoR ([^ ]+ ) -

- **These patterns can then be used to answer unseen questions:**
  - The question term is submitted to Google and the top ten returned documents have the question term replaced with AnCHoR.
  - Those sentences which contain AnCHoR are extracted and combined to make a single document.
  - Each pattern is then applied to each sentence to extract possible answers.
  - All the answers found are sorted based firstly on the precision of the pattern which selected it and secondly on the number of times the same answer was found.

# Limitations of the Approach

---

- **Most systems of this kind use questions of the form “*When was X born?*”, often the concrete example “*When was Mozart born?*”, which can be answered from the following text:**

*“Mozart (1756-1791) was a musical genius who...”*
- **A similar question which is also answerable from the same text snippet is “*When did Mozart die?*”. Unfortunately the system just presented is unable to answer this question:**
  - The first stage of the algorithm will generate a pattern such as:  
AnCHoR ( 1756 - AnSWeR )
  - The second stage is, however, unable to assign a precision to this pattern due to the presence of a specific year of birth.
- **What is needed is an approach that can generalise this form of pattern approach to eliminate some or all of the specific text.**

# nlp sheffield Generalising Using an NE Tagger

---

- Many of the question specific words which could appear within the patterns are dates, names, organisations and locations.
- These are exactly the types of words which can be recognised using gazetteers and named entity taggers.
- The approach taken to generalise the answer patterns is therefore to combine the approach with these NLP techniques:
  - A gazetteer and named entity tagger are applied to the documents after the question and answer terms have been replaced with AnCHoR and AnSWeR.
  - Text highlighted as a named entity is then replaced with a tag representing its type, i.e. a date would be replaced by DatE. Processing then continues as before.
  - When using the patterns to answer questions more work is required as any tags selected as answers have to be expanded to their original values.

# d nlp ~~sheffield~~ Generalising Using an NE Tagger

- This results in a system which can now produce generalised patterns, with associated precisions, for questions of the form *“When did X die?”* :

1.000: AnCHoR \ ( DatE - ([^ ]+) \) .

1.000: AnCHoR \ ( DatE - ([^ ]+) \)

0.889: AnCHoR DatE - ([^ ]+)

**It should be clear that these patterns are capable of extracting the answer from the text we saw earlier:**

*“Mozart (1756-1791) was a musical genius who...”*

- This approach also gives rise to more patterns being generated for questions of the form *“When was X born?”*:

0.941: AnCHoR \ ( ([^ ]+) - DatE \)

0.556: AnCHoR ([^ ]+) - DatE

# Results to Date

---

- **A small set of experiments were carried out to see the effect of extending the system in this way:**
- **Three different question types were used:**
  - When was X born?*
  - When did X die?*
  - What is the capital of X?*
- **Six experiments were undertaken, namely the basic and extended system applied to each of the three question types.**
  - For each experiment twenty questions were used for stage 1, twenty questions for stage 2 and the patterns were then tested using a further 100 questions.
- **For each experiment the mean reciprocal rank and confidence weighted scores were calculated as a measure of performance over and above the simple metric of percentage of questions correctly answered.**

# Results to Date

Question Type	System Type	% Correctly Answered	MRR Score	Confidence Weighted
Born	Basic	52%	0.52	0.837
	Extended	53%	0.52	0.843
Died	Basic	0%	0.00	0.000
	<b>Extended</b>	<b>53%</b>	<b>0.53</b>	<b>0.852</b>
Capital City	Basic	24%	0.27	0.259
	Extended	24%	0.28	0.265

# Discussion and Conclusions

---

- **Currently only a small set of experiments have been carried out using this technique, but for these experiments:**
  - Generalising the patterns benefited question types which could not be answered by the basic approach.
  - Generalising the patterns had no detrimental effect on question types which could be answered by the basic approach.
- **A small number of example questions can be used to generate sets of surface matching text patterns – only 40 examples were used to generate and assign precisions to the patterns.**
  - Clearly experiments should be carried out to see if there is a performance advantage in using more examples.
- **There are other ways of generalising the surface matching text patterns which should be investigated.**

# Future Work

---

- **Preliminary work on varying the number of examples used to generate the patterns and their precisions, suggests that using more examples leads to better performance:**
  - This should be continued to try and determine the optimum number of examples to use.
- **We intend to investigate other NLP techniques to generalise and improve the patterns, including:**
  - Name matching and coreference
  - Noun and verb phrase chunking
  - Part of speech tagging
- **We also intend to improve the second stage by experimenting with the document retrieval query**
  - Include keywords such along with the question term, e.g. “*capital*” for questions of the form “*What is the capital of X?*”.

# Any Questions?

**Copies of the paper and these slides can be found at:**

**<http://www.dcs.shef.ac.uk/~mark/phd/work/>**