High Accuracy Rule-based Question Classification using Question Syntax and Semantics
Tayyar Madabushi, Harish; Lee, Mark

Citation for published version (Harvard):

Link to publication on Research at Birmingham portal

General rights
Unless a licence is specified above, all rights (including copyright and moral rights) in this document are retained by the authors and/or the copyright holders. The express permission of the copyright holder must be obtained for any use of this material other than for purposes permitted by law.

• Users may freely distribute the URL that is used to identify this publication.
• Users may download and/or print one copy of the publication from the University of Birmingham research portal for the purpose of private study or non-commercial research.
• Users may use extracts from the document in line with the concept of ‘fair dealing’ under the Copyright, Designs and Patents Act 1988 (?)
• Users may not further distribute the material nor use it for the purposes of commercial gain.

Where a licence is displayed above, please note the terms and conditions of the licence govern your use of this document.

When citing, please reference the published version.

Take down policy
While the University of Birmingham exercises care and attention in making items available there are rare occasions when an item has been uploaded in error or has been deemed to be commercially or otherwise sensitive.

If you believe that this is the case for this document, please contact UBIRA@lists.bham.ac.uk providing details and we will remove access to the work immediately and investigate.
High Accuracy Rule-based Question Classification using Question Syntax and Semantics

Harish Tayyar Madabushi
School of Computer Science,
University of Birmingham,
Birmingham, United Kingdom.
H.T.Madabushi@cs.bham.ac.uk

Mark Lee
School of Computer Science,
University of Birmingham,
Birmingham, United Kingdom.
M.G.Lee@cs.bham.ac.uk

Abstract

We present in this paper a purely rule-based system for Question Classification which we divide into two parts: The first is the extraction of relevant words from a question by use of its structure, and the second is the classification of questions based on rules that associate these words to Concepts. We achieve an accuracy of 97.2%, close to a 6 point improvement over the previous State of the Art of 91.6%. Additionally, we believe that machine learning algorithms can be applied on top of this method to further improve accuracy.

1 Introduction and Motivation

Question Answering (QA) is a task in Natural Language Processing (NLP) that requires the system to provide concise answers to Natural Language questions. Interest in QA has grown dramatically over the past couple of years, in part due to advances in NLP and Machine Learning, that have allowed for significant improvements in QA systems, and in part due to its increased accessibility to the general public via smart-phone applications such as Siri and Google Now.

An important element of QA is Question Classification (QC), which is the task of classifying a question based on the expected answer. As an example, the question “Who is the prime minister?” could be assigned the class “person”, whereas the question “Where is the prime minister?” could belong to the class “location”. Since the task involves identifying the type of answer, it is sometimes referred to as Answer Type Classification. While there do exist QA Systems that do not make use of QC, QC has been shown to significantly improve the performance of QA systems (Hovy et al., 2001).

A priori knowledge of the kind of information that a QA system is required to extract allows for the exploitation of predefined patterns and improved feature selection. For example, consider a QA system provided with the information that the question “How long is the term of office of the Prime Minister?” requires, as an answer, a “number that represents a duration”. Such a system could dramatically reduce its search space, in that it could focus on numbers. The design of a QA system’s search and information extraction components determines the classes that a QC should use. Despite this dependence on how question classes are used, there are some common question classes that are widely accepted as useful. The rules that govern and classes contained in a given question classification are based on the specific Question Taxonomy chosen.

2 Related Work

Work on QC, as in most NLP tasks, can be broadly divided into three categories: a) those that make use of machine learning, b) those that rely purely on rules, and c) those that are a hybrid of the two. With the increased popularity and success of machine learning techniques, most recent work on QC has been limited to methods that make use of machine learning. While there continues to be some exploration into semantic information contained in sentences, such information is often converted into features.

While there are several Question Taxonomies that are available for use in training and testing QC systems, the most popular is the one introduced by Li and Roth (2002). This is because of the 5,500
training questions and corresponding classification they provide, in addition to the classification of the 500 TREC 10 (Voorhees, 2001) questions. Their classification is a two level system which contains a coarse and a fine level of classification for each question. Table 1 lists the classification introduced by them. In this paper, we refer to a specific classes in the following format: coarse:fine. For example, the class animal, contained in the coarse class ENTY, will be referred to as enty:animal.

<table>
<thead>
<tr>
<th>Coarse</th>
<th>Fine</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABBR</td>
<td>abbreviation, expansion</td>
</tr>
<tr>
<td>DESC</td>
<td>definition, description, manner, reason</td>
</tr>
<tr>
<td>ENTY</td>
<td>animal, body, color, creation, currency, disease, event, food, instrument, language, letter, other, plant, product, religion, sport, substance, symbol, technique, term, vehicle, word</td>
</tr>
<tr>
<td>HUM</td>
<td>description, group, individual, title</td>
</tr>
<tr>
<td>LOC</td>
<td>city, country, mountain, other, state</td>
</tr>
<tr>
<td>NUM</td>
<td>code, count, date, distance, money, order, other, percent, percent, period, speed, temperature, size, weight</td>
</tr>
</tbody>
</table>

Table 1: Question Taxonomy introduced by Li and Roth (2002).

The original method proposed by Li and Roth (2002), relies on machine learning and first classifies questions into coarse classes, before then using the coarse class as a feature in fine grained classification. They also report their results for both the coarse and fine classes. We, however, focus our efforts on fine grained classification.

Metzler and Croft (2005) provide a detailed analysis of statistical methods of QC prior to 2005, while dismissing rule-based systems as “cumbersome and inflexible”, and a more recent survey by Loni (2011) details QC methods using more recent Machine Learning techniques. Work on QC over the last couple of years has involved either reducing the number of features (Pota et al., 2016; Pota et al., 2015), focusing on specific domains (Feng et al., 2015) or using new methods in machine learning such as Convolutional Neural Networks (Kim, 2014) and Skip-Thought Vectors (Kiros et al., 2015).

The previous State of the Art in fine grained classification of Li and Roth (2002)’s data is 91.6% and was achieved by Van-Tu and Anh-Cuong (2016), who base their work on using semantic features in a linear SVM. Of specific relevance to our work is the work by Silva et al. (2011), who first extract headwords, before then mapping these headwords into various categories using WordNet (Miller, 1995) to achieve an accuracy of 90.8%. Previous work by (Huang et al., 2008), which also makes use of both headwords and WordNet, while using slightly different methods, achieves an accuracy of 89.2%.

3 Concepts as a Theoretical Framework for Question Classification

Concepts are generalisations or abstractions that allow the use of previous experience in new situations. For example, questions such as “Who is the actor who . . . ?”, of the form “Who auxilliary_verb (determiner) Concept:Occupation who . . . ?”, can be classified under the class hum:person, if we had information about the Concept “occupation”, because this would enable us to map all questions that use any occupation in this particular pattern to this QC. Similarly, information about the Concept “meaning” would enable us to create a rule to classify questions such as “What is the meaning of the word . . . ?”, and “What does the word . . . mean?” to the question class desc:definition. As can be seen from the latter example, Concepts need not always be associated with nouns.

3.1 Implementing Concepts using Types

As described in the previous section, it is useful to define Concepts as sets of words and to this end, we require a method of generating a large number of words that belong to a particular Concept. To achieve this, we make use of Types (Tayyar Madabushi et al., 2016), which provide a way of defining sections of an ontology to belong to a given Type. While the authors use Types to identify classes of nouns that can be compared when measuring the semantic similarity between two sentences, we use Types to define Concepts. In this work, we modify the definition of types by making use of WordNet hyponyms: $W_1$ is considered a hyponym of $W_2$ if $\forall e \in W_1, e$ is an instance of $W_2$. 
A Type consists of a set of WordNet synsets or words $S$, and represents the set of words whose lemmas belong to the union of the set $S$, and in the case of synsets, the set containing the hyponym closure of the synsets in $S$, and in the case of words, those words. As an example, all words whose lemmas belong to the hyponym closure of the synset ‘occupation.n.01’, such as bookkeeping, acting, and ministry belong to the the Type ‘occupation.n.01’. It is interesting to note that this one definition provides us conceptual information on 283 lemmas (the size of the hyponym closure of occupation.n.01).

We use types to create a rough approximation of Concepts. We achieve this by manually picking specific synsets within WordNet and associating them and all their hyponyms to a particular QC based on where in a question they appear. Revisiting the first example in Section 3, the Concept “occupation” is defined by creating a Type that includes the word occupation and all hyponyms of the synset ‘occupation.n.01’. Similarly, the synsets ‘people.n.01’, ‘organization.n.01’, ‘university.n.01’, ‘company.n.04’, ‘social_group.n.01’, and all of their hyponyms are assigned to the Question Class “Human Group”. Some words, such as the word “mean” discussed in the second example in Section 3, belong to a particular Type while their hyponyms do not (in the case of “mean”, “aim”, “drive”, and “spell” are hyponyms, which do not imply that a question belongs to the definition class the same way the word “mean” does), and in such cases, we add just the word and not its hyponyms.

The manual process of creating Types is done by looking at all hyponyms of the synset entity.n.01 and assigning them to a Type iff that synset and all its hyponyms represent the same Concept. This sometimes leads to instances wherein the same word is part of different Types because of its different word sense. In such cases, Types are redefined using less general synsets.

Not all of the Types we define are directly associated with a Question Class. For example, we define the Type people_from, consisting of ‘inhabitant.n.01’ and all its hyponyms which enables us to identify the class enty:termeq (i.e. equivalent term). We do this by checking to see if the question asks us what people from a particular place call something, by use of the rule “What auxiliary verb people_from call word?”. As an example, the question “What do Italians call Noodles?” matches this rule and belongs to the QC enty:termeq. We also define groups of verbs as belonging to certain Types, such as the Type of verbs that can only be performed by a person (e.g. sing, invent) and the Type of words that require us to perform a possesive or a prepositional roll (Section 6.1).

4 System Overview

The system presented in this work consists of three parts: a) extracting a Question’s Syntactic Map (defined in Section 5.1), b) identifying the headword, of the noun phrase in the question, while handling Entity Identification and phrase detection, and c) using rules to map words at different positions in the Syntactic Map to identify the QC. These are further broken down into the following steps (programmatically, methods):

<table>
<thead>
<tr>
<th>Syntactic Map Extraction</th>
<th>Question Rewrite</th>
<th>Rewrites questions that are in non-standard form.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parse Tree Analysis</td>
<td></td>
<td>Extract structure information from the question using Constituency-based parse trees</td>
</tr>
<tr>
<td>Word, Phrase and Entity Extraction</td>
<td>Headword Extraction</td>
<td>Extract headwords from noun phrases in the question using a) Possessive Unrolling b) Preposition Rolling c) Entity Identification</td>
</tr>
<tr>
<td></td>
<td>Verb, Wh-word and Adjective Extraction</td>
<td>Extract the Auxiliary and Major Verbs, the Wh-word and all adjectives from the question.</td>
</tr>
<tr>
<td></td>
<td>Match Rules based on the Question Syntax and Word Type</td>
<td>Using a hierarchy of syntactic positions in a question, iteratively check to see if there exists a rule for mapping the word at that position to a QC.</td>
</tr>
</tbody>
</table>

For example, given the question “Name of actress from England in the movie ‘The Titanic’ is what?”, our system identifies its QC as follows: We first identify that this question is not in a form that we can analyse to extract the Syntactic Map and rewrite it as “What is the name of the actress from England in the movie ‘The Titanic’?” (Section 5.2). The question’s parse tree is then analysed to generate the Question’s Syntactic Map (Section 5.1). We then identify the headword to be the noun actress using prepositional
rolling (Section 6.1). At this stage, we have established that the question’s wh-word is “What”, auxiliary verb is “is”, and headword is “actress”. We check for the existence of a rule that classifies this question by iterating through these elements in a predefined order (Section 7.2). This results in the word “actress” matching the rule: ‘occupation.n.01’ and its hyponyms in SQ-NNP when the wh-word is ‘what’ indicate that the question class is hum:ind, so enabling us to classify the question as hum:ind.

4.1 Methodology

To avoid bias, we use the 5,500 questions and their respective question classes provided as training data by Li and Roth (2002) for exploration and rule discovery, and ensure that the 500 TREC questions, which consist of the test set, are not observed during the creation of rules (although the system is, at regular intervals, tested on this set to ensure progress). Once we complete the analysis of a question’s parse tree, not all words in the question are of further relevance to the task of QC. However, so as to maximise the number of words that we have rules for, we try to create rules for all words that appear in training set.

5 Syntactic Maps

Previous work that has made use of parse trees includes that by Silva et al. (2011), who used Collin’s Rules (Collins, 1999) to extract headwords and work by Shen and Lapata (2007) who made use of FrameNet (Baker et al., 1998). Unlike these works, we first extract, what we call, a Question’s Syntactic Map, before creating rules that depend on the position of words in this Map.

A Syntactic Map (SM), unlike a parse tree, is a fixed structure that we fill in with information from a question’s parse tree and can contain empty or “None” elements. It is a generic template for all the different kinds of questions that we can classify, and any question that we cannot convert to a Syntactic Map, cannot be classified using our system. Crucially, the SM contains the following five elements of a question: a) the question’s wh-word b) the noun phrase (if any) contained in the WHNP sub-tree and its internal phrase structure, and from the SQ sub-tree of the parse tree: c) the Auxiliary Verb (AVP) d) the noun phrase (if any) and its internal phrase structure, and e) the Main Verb (MVP) (if any). Noun phrases including possessives, and prepositional phrases are extracted into similar fixed structures. Programmatically, a SM is a class (object-oriented programming), as are the constituent noun phrases, prepositional phrases, and verbs. The generic structure of a SM, along with the structure of its constituents is shown in Table 2.

<table>
<thead>
<tr>
<th>WH Word</th>
<th>WHNP</th>
<th>WHNP Phrase in WHNP</th>
<th>WHNP Phrase in SQ</th>
<th>AVP</th>
<th>Auxiliary Verb of SQ</th>
<th>AVP Phrase</th>
<th>NNP</th>
<th>Noun Phrase in WHNP</th>
<th>NNP Phrase in SQ</th>
<th>MVP</th>
<th>First Main Verb of SQ</th>
</tr>
</thead>
</table>

Table 2: The fixed structure of a Syntactic Map (left), and the constituent phrase structures (right).

In the question “How much does the President get paid?”, it is the adverb “much” that allows us to infer that the expected answer is a number and additionally, the word “paid” allows us to infer that the number, in fact, represents money hence resulting in the question class num:money.

In the questions “What is a golf ball made of?” and “What does gringo mean?” the verbs after the noun (the first Main Verb or MVP) provide us with important clues on which question class these questions belong to (in this case entry:substance and desc:def). It is for this reason that we move beyond conventional headword extraction and focus on populating Syntactic Maps, which capture more information about the question. Although Silva et al. (2011) consider words other than nouns, they do so only when the questions contain certain exact phrases.

5.1 Syntactic Map Extraction

The first step in SM extraction is the extraction of the “WHNP” and “SQ” sections of a question from its constituent parse tree, which we generating using the Stanford CoreNLP toolkit Manning et al. (2014).
The WHNP sub-tree represents the Wh-noun Phrase and the SQ sub-tree the main clause of a wh-question. In cases where there is neither (e.g. Name the highest mountain.), we use the first noun phrase as the SQ sub-tree. From the WHNP and the SQ sections of the parse tree, we extract the various elements of the SM as shown in Table 2. This requires the parsing of noun, prepositional, possessive and verb phrases. Due to space constraints, we only provide an overview of each of these below. Additionally, extracting each of these elements is done recursively as sentences often contain possessive phrases or prepositional phrases within one another. Table 3 illustrates one such scenario in which a question has two recursive possessive phrases.

<table>
<thead>
<tr>
<th>Parse Tree</th>
<th>Extracted Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROOT</td>
<td></td>
</tr>
<tr>
<td>SBARQ</td>
<td></td>
</tr>
<tr>
<td>WHNP</td>
<td></td>
</tr>
<tr>
<td>SQ</td>
<td></td>
</tr>
<tr>
<td>WP</td>
<td></td>
</tr>
<tr>
<td>VP</td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>?</td>
<td></td>
</tr>
<tr>
<td>NNP</td>
<td>Dudley</td>
</tr>
<tr>
<td>NNP</td>
<td>Do-Right</td>
</tr>
<tr>
<td>POS</td>
<td>horse</td>
</tr>
<tr>
<td>NNP</td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>WH Word</td>
<td>What</td>
</tr>
<tr>
<td>WHNP</td>
<td></td>
</tr>
<tr>
<td>NNP</td>
<td>None</td>
</tr>
<tr>
<td>SQ</td>
<td></td>
</tr>
<tr>
<td>AVP</td>
<td>['is']</td>
</tr>
<tr>
<td>NNP</td>
<td>(Possessive) Dudley</td>
</tr>
<tr>
<td>MVP</td>
<td>Do-Right (Possessive)horse</td>
</tr>
<tr>
<td>name</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: The Parse Tree and Extracted SM of a Question Consisting of a Nested Structure.

We make the conscious decision of stopping the SM extraction process after reaching the first main verb. This is because we observed that there were very few questions that require structural information beyond this point.

Our method of analysing noun phrases handles the extraction of adjectives, possessive phrases, prepositions and trailing adjectives but ignores all determiners. Prior to analysing parse trees of noun phrases, we first modify certain parse tree patterns that noun phrases occur in. The resultant Constituency-based parse trees are not always valid but greatly simplify the analysis of noun phrases. Two examples of the modifications we perform to noun phrase sub-trees are illustrated in Table 4.

<table>
<thead>
<tr>
<th>Parse Tree</th>
<th>Extracted Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>ADJP</td>
<td></td>
</tr>
<tr>
<td>RB</td>
<td></td>
</tr>
<tr>
<td>JJ</td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>PP</td>
<td></td>
</tr>
<tr>
<td>NNS</td>
<td>Word_1</td>
</tr>
<tr>
<td>RB</td>
<td>Word_2</td>
</tr>
<tr>
<td>JJ</td>
<td>Word_3</td>
</tr>
<tr>
<td>Prepositional Phrase</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parse Tree</th>
<th>Extracted Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>CC</td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>DT</td>
<td></td>
</tr>
<tr>
<td>NN</td>
<td></td>
</tr>
<tr>
<td>and</td>
<td></td>
</tr>
<tr>
<td>DT</td>
<td></td>
</tr>
<tr>
<td>NN</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Some of the Parse Tree Modifications that are Performed on Noun Phrases.

This simplification process leaves us with the task of extracting information from noun phrases that belong to a much smaller set of sub-tree patterns. Some of the more common noun phrase patterns are illustrated in Table 5. Possessive phrases are treated as nouns that must have, attached to them, yet another noun. When we identify a preposition phrase or a verb phrase, that sub-tree is passed to either the preposition or verb analysis method respectively.

Similarly, we extract information from prepositional sub-trees based on their structure, which nearly
always belong to one of the following three patterns: A preposition phrase with one child that is the preposition and the other that is one of either a noun phrase, verb phrase or another prepositional phrase (e.g. “name of the prime minister of U.K.”). These patterns are illustrated in Table 6. Just as in the case of noun phrases, we pass on any sub-trees of phrases that are of a different kind to the appropriate analysis module, which enables us to generate a recursive SM.

Table 5: Some Common Sub-tree Patterns that Noun Phrases occur in.

Table 6: Some Common Sub-Tree Patterns that Prepositional Phrases occur in.

5.2 Question Rewrites

There are some questions that do not belong to the standard structure of questions such as “A corgi is a kind of what?” and “In 139 the papal court was forced to move from Rome to where?”. We identify several of these structures and create rewrite rules (e.g. \(x\) is/was \(y\) in/of \(z\)?) to rewrite these questions to a form that we can parse. We use regular expressions instead of parse tree analysis as these structures are very easy to identify and so the overhead of parsing is not justified. Using these rules the above two questions will be rewritten as “What is a corgi a kind of?” and “To where was the papal court forced to move from Rome in 139?”.

6 Concept Identification

In this section, we provide details on methods we use for identifying relevant Concepts, which we extract by analysing the SM.

6.1 Preposition Rolling and Possessive Unrolling

Rolling and Unrolling refer to the selective moving forward through a preposition, or backwards through a possessive noun. Consider the question “What is the quantity of American soldiers still unaccounted for from the Vietnam war?” from which we extract \(\text{quantity(PP)}\) of \(\text{PP-NN: (JJ)American soldiers}\), and the question “What are the different types of plastic?” from which we extract \(\text{(JJ)different types(PP)}\) of \(\text{PP-NN: plastic}\). In the second instance, we must roll through the preposition to reach the relevant word “plastic”, whereas, in the first instance, we must not, so identifying “quantity”.

Similarly, consider the question “What game’s board shows the territories of Irkutsk, Yakutsk and Kamchatka?” from which we extract the noun phrase \((\text{Possessive})\text{game board}\), and the question “Name Alvin’s brothers.” from which we extract \((\text{Possessive})\text{Alvin brothers}\). In the first instance we need to unroll through the possessive to reach the relevant word “game”, whereas in the second case we must not. We call this selective process of moving forward through a preposition “Rolling”, and the process of selectively moving backwards through a possessive “Unrolling”. Rolling and Unrolling are achieved through a list of rules that depend on the Type of the target and source of the Roll or Unroll.

6.2 Headword and Phrase Extraction

Consider the question “What mystery writer penned ‘...the glory that was Greece, and the grandeur that was Rome’?”. The relevant noun phrase that we extract from the SM is “mystery writer” and the head of
this noun phrase is “writer”, the last noun in the noun phrase. This is often the case, and some previous
works have used only this to identify the head of a noun phrase (Metzler and Croft, 2005). Unfortunately,
this is not always the case, and does not always provide the word that is most useful for QC. For example,
the noun phrase extracted from “What crop failure caused the Irish Famine?” is “crop failure” and the
relevant noun is “crop”. Although it can be argued that the head noun in this phrase is “failure”, qualified
by “crop”, this would not aid us in classification, as “crops” are a form of food and the expected Question
Class is enty:food, while “failure” is a very different Concept.

We automatically identifying the head noun by identifying Verb Nouns and Descriptive Nouns starting
at the right of the noun phrase and ignoring such nouns. We define Verb Nouns as nouns that have a more
common verb form (e.g. fail) or verbs that are “acts”, which we identify by parsing the definition of the
verb. Similarly, we define Descriptive Nouns as nouns that belong to a Type we define as descriptive
which includes, for example, hyponyms of the synset ‘digit.n.01’.

6.3 Entity Identification
Let us now consider the question “What is bipolar disorder?”. The correct Question Class for this
question is desc:definition, however, it is easy to miss-classify this question as belonging to the class
enty:dismed (entity, disease or medicine), because the word “bipolar” is tagged as an adjective. To get
around this we require a method of identifying that “bipolar disorder” must be considered as a single
entity.

Even in instances wherein it is relatively easy to identify an entity, as in the case of phrases that
consist of consecutive nouns, it is important to be able to convert these phrases to a form that appears in
WordNet. For example, the phrase “equity securities” can be identified as a single entity, however, it is
listed in WordNet under the entry “Shares”.

We identify these phrases using a method called Wikification (Mihalcea and Csomai, 2007), which is
the process of linking words and phrases in a piece of text to titles of Wikipedia entries. The intuition
behind this is that a phrase that appears as a Wikipedia Article title must be important enough to be con-
sidered as a single Entity. We base our method of Wikification on the original, while replacing the process
of keyword identification with SM and that of Word Sense Disambiguation with the method detailed in
Section 7.1. For example, there is an article on Wikipedia titled “Bipolar Disorder” on Wikipedia and
the Wikified term for “equity securities” is “Shares”.

7 Question Classification using Syntactic Maps
Once we have the SM of a question, we use rules to identify the relevant QC. However, before we can
match appropriate words, we require a way of identifying the correct sense of a word.

7.1 Word Sense Disambiguation
SMs often provide us with a single word that represents the object that the question expects as an answer.
The question “What album put The Beatles on the cover of Time in 1967?”, for example, requires that the
answer consists of an “album”. However, it is unclear whether album refers to “one or more recordings
issued together” or “a book of blank pages with pockets or envelopes”. Huang et al. (2008) address this
problem by use of the Lesk Algorithm (Lesk, 1986).

Our use of SM allows for implicit Word Sense Disambiguation as it is rare for the same word to appear
at the same syntactic location but in different senses. When this does happen however, we identify the
sense of a word based on the Types of the surrounding elements of the SM. For example, “How much
does it cost to fly to Japan?” and “How much does a plane weigh?” both have the word “much” at the
same position and so require us to identify the Types of associated words (i.e. “cost” and “weigh”) to be
able to disambiguate the relevant Concept.

7.2 Mapping Question Classes
The intuition behind the mapping process is that words or phrases at certain positions in the SM trigger
certain Concepts, which gives away the question class. To this end, we use Types defined for each differ-
ent position in the SM to map questions to question classes. For example, the word “do” appearing as the
Algorithm 1: A Simplified Algorithm showing the Mapping of the Syntactic Map to Question Classes

```
Data: Syntactic Map, Type Definitions, Classes associated with Type Definitions.
Result: Question Class
1 if Preposition Rolling Possible then
  2 Perform Preposition Roll
3 if Possessive Unrolling Possible then
  4 Perform Possessive Unroll
5 Initialise head_noun_class to None; /* head_noun_class is a Tuple Consisting of the Major and Minor Question Type */
6 head_noun ← Extract Head Noun from Syntactic Map ;
7 head_noun_adjectives ← Extract Head Noun adjectives from Syntactic Map ;
8 for reversed( head_noun_adjectives ) do
  9 if adjective has Type Defined then
     10 head_noun_class ← Class associated with Type;
   11 if head_noun_class is None then
     12 if head_noun has Type Defined then
        13 head_noun_class ← Class associated with Type;
   14 if head_noun_class[0] == "ABBR" then
     15 if head_noun is an Abbreviation then
        16 return ( 'ABBR', 'exp' )
     17 return head_noun_class
18 if All of the following elements in the Syntactic Map are Empty: WHNP-NNP, SQ-MVP, head_noun_adjectives then
19 if There has been no Rolling or Unrolling then
   20 if AVP is one of “is”, “are”, “was”, “were” then
     21 if WH_Word is “What” then
        22 return ( 'DESC', 'def' )
     23 if WH_Word is “Who” then
        24 return ( 'HUM', 'desc' )
   25 for reversed( head_noun_adjectives ) do
     26 if adjective has WSD Type Defined then
        27 return Class associated with WSD Type;
     28 wh_word ← Extract What Word from Syntactic Map ;
   29 if wh_word == "define" then
     30 if head_noun_class[0] == "DESC" then
        31 return head_noun_class
     32 return ( "DESC", "def" )
   33 if wh_word == "how" then
     34 if head_noun_class[0] == "DESC" then
        35 return head_noun_class
     36 return ( "DESC", "manner" )
   /* Similar restrictions are imposed on other possible wh_words (i.e. "where", "whose", "whose", "describe", "when", "why", "name", and "what") */
37 main_verb ← Extract Main Verb from Syntactic Map ;
38 auxiliary_verb ← Extract Auxiliary Verb from Syntactic Map ;
39 for verb in [ main_verb, auxiliary_verb ] do
   40 if verb has Type Defined then
     41 return Class associated with Type;
   42 if verb has WSD Type Defined then
     43 return Class associated with WSD Type;
   44 if head_noun_class is None then
     45 return ( "ENTY", "other" )
   46 return head_noun_class
```

auxiliary verb is handled differently from when it appears as the main verb in the SM. The order in which different sections of the SM are considered determines which word is finally used during classification.

There are some special words, such as “much”, “do”, “name” and “call”, that require more complex classification rules. The adjective “much” for example could indicate the class num:money or num:weight depending on whether the other sections of the SM contain the Type “money” or the Type “weight”. As in the case of WSD, we define disambiguation rules for each such word.

Algorithm 1, while not exhaustive in listing the mapping rules (due to space constraints), provides a simplified overview of the mapping of Semantic Maps to Question Classes. It takes as input the SM,
the Type definitions and associated Question Classes and returns a tuple consisting of the Major and Minor question classes. Just over 230 Type definitions and 10 special Word Sense Disambiguation definitions cover the entire test set, and at the time of writing, these have been expanded to around 600 Type definitions and 70 WSD definitions.

8 Results

We achieve an accuracy of 97.2% on the TREC 10 dataset which translates to an incorrect tagging of 14 of the 500 questions in the dataset. This is close to a 6 point improvement over the previous state of the art of 91.6% (Van-Tu and Anh-Cuong, 2016). We list our accuracy against that of various other works that have reported results on the TREC 10 dataset in Table 7.

<table>
<thead>
<tr>
<th>Study Classifiers</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>This Work</td>
<td>None</td>
</tr>
<tr>
<td>Van-Tu and Anh-Cuong (2016)</td>
<td>Linear SVM</td>
</tr>
<tr>
<td>Pota et al. (2016; Pota et al. (2015)</td>
<td>Linear SVM</td>
</tr>
<tr>
<td>Kim (2014)</td>
<td>Convolutional Neural Networks</td>
</tr>
<tr>
<td>Kiros et al. (2015)</td>
<td>Skip-Thought Vectors</td>
</tr>
<tr>
<td>Silva et al. (2011)</td>
<td>Linear SVM</td>
</tr>
<tr>
<td>Loni et al. (2011)</td>
<td>Linear SVM</td>
</tr>
<tr>
<td>Merkel and Klakow (2007)</td>
<td>Language Modelling</td>
</tr>
<tr>
<td>Li and Roth (2006)</td>
<td>SNoW</td>
</tr>
<tr>
<td>Li and Roth (2002)</td>
<td>SNoW</td>
</tr>
</tbody>
</table>

Table 7: Results Achieved by this Work alongside some other Works that use the same Dataset.

8.1 Error Analysis

Table 8 provides a list of some of the questions that we misclassify along with the reason for this. One of the advantages of a purely rule-based system is the ability to pinpoint the exact reason for an incorrect classification.

<table>
<thead>
<tr>
<th>Question</th>
<th>Correct Class</th>
<th>Classified As</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>What are the twin cities?</td>
<td>LOC city</td>
<td>DESC def</td>
<td>We classify both these as definitions because we (correctly) identify “twin cities” and “speed of light” as entities. The presence of the word “the” however requires information about the entity instead of a definition for the entity - a rule that requires to be added.</td>
</tr>
<tr>
<td>What is the speed of light?</td>
<td>NUM speed</td>
<td>DESC def</td>
<td></td>
</tr>
<tr>
<td>What is compounded interest?</td>
<td>DESC def</td>
<td>DESC desc</td>
<td>Our Wikification system fails to identify “compounded interest” to be the same as the entity “compound interest”.</td>
</tr>
<tr>
<td>What is the spirometer test?</td>
<td>DESC def</td>
<td>ENTY instru</td>
<td>The word “test”, has a natural verb form so forcing the system to identify “spirometer” as the head noun. Some modifications to the function identifying Verb Nouns are required to rectify this.</td>
</tr>
</tbody>
</table>

Table 8: An analysis of some of the questions that we fail to classify correctly.

9 Conclusion and Future Work

We presented a purely rule-based system for QC which exploits decades of research into the structure of language and Concepts. Although this method has focused on a particular type of questions, we believe that a similar method can be applied to classifying questions of a different type, and we intend to extend our work to include those datasets. We also note that these are a common and important kind of questions, which are similar to those handled by most modern smartphone interactive systems such as Google Now (Ristovski, 2016).

Finally, we intend to implement a QA system that leverages QC to explore the true impact of high-accuracy question classification. We also intend to make this system available through a simple Application Programming Interface (API) \(^1\) so other QA systems can benefit from this work.

\(^1\)API available at: http://www.harishmadabushi.com/research/questionclassification/
References


