A Semantic Distance of Natural Language Queries Based on Question-Answer Pairs

by

Kun Xiong

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2014

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Many Natural Language Processing (NLP) techniques have been applied in the field of Question Answering (QA) for understanding natural language queries. Practical QA systems classify a natural language query into vertical domains, and determine whether it is similar to a question with known or latent answers. Current mobile personal assistant applications process queries, recognized from voice input or translated from cross-lingual queries. Theoretically speaking, all these problems rely on an intuitive notion of *semantic distance*. However, it is neither definable nor computable. Many studies attempt to approximate such a semantic distance in heuristic ways, for instance, distances based on synonym dictionaries. In this paper, we propose a unified algorithm to approximate the semantic distance by a well-defined information distance theory. The algorithm depends on a pre-constructed data structure - *semantic clusters*, which is built from 35 million question-answer pairs automatically. From the semantic measurement of questions, we implement two practical NLP systems, including a question classifier and a translation corrector. Then a series of comparison experiments have been conducted on both implementations. Experimental results demonstrate that our distance based approach produces fewer errors in classification, compared with other academic works. Also, our translation correction system achieves significant improvements on the Google translation results.

Acknowledgements

I would like to express my profound appreciation and thanks to my supervisor Professor Dr. Ming Li. I would like to thank you for your kindly encourage and for valuable suggestions on my research. I would also like to thank you for helping me build my career path. I would like to thank my parents for their cares and supports though all my life. Many thanks to my lab partners Guangyu Feng and Di Wang for their help and inspiration. I would like to thank other co-workers and all the people who made this possible. In the end, I give my sincere gratitude and love to my wife - Yanjun Chen.

Dedication

This thesis is dedicated to the one I love.

Table of Contents

List of Tables **ix**

List of Figures **x**

1 Introduction **1**

1.1 Natural Language Processing and Question Answering 3

1.2 Question Normalization 4

1.3 Natural Language Query Parsing 5

1.4 Contribution 5

1.5 Organization 6

2 Related Work **8**

2.1 Relation Extraction 8

2.2 Question Classification 9

2.3 Semantic Distance 10

3 Observations **12**

3.1 Community QA 12

3.2 Knowledge base 13

3.3 Anchor text of Wikipedia . . . 15
3.4 Challenges . . . 16

4 Semantic Clusters **17**
4.1 Schema matching . . . 18
4.2 Latent Relation Extraction . . . 20
4.3 Clustering . . . 22
4.3.1 Question Template . . . 23
4.3.2 Predicate Path . . . 23

5 Semantic Distance **25**
5.1 Problem Formalization . . . 26
5.2 Approximation . . . 28
5.3 Encoding . . . 30
5.3.1 Encoding Words . . . 31
5.3.2 Encoding Sentences . . . 33

6 Applications and Experiments **36**
6.1 Question Classification . . . 36
6.1.1 System Implementation . . . 37
6.1.2 Experiment Setup . . . 38
6.1.3 Empirical Results . . . 39
6.2 Question Translation Correction . . . 39
6.2.1 System Implementation . . . 40
6.2.2 Experiment Setup . . . 43
6.2.3 Empirical Results . . . 43

7 Conclusion and Future Works 45

APPENDICES 48

A semantic clusters 49

References 54

List of Tables

6.1 Experiment dataset statistics 39
6.2 Comparison of SVM and RSVP on the three domains 40
6.3 Comparison of Google Translation and RSVP 44

List of Figures

1.1 Question template examples labeled by “Spouse” 6

3.1 A example of the CQA web site (Yahoo! Answers) 13
3.2 A example of the knowledge base 14
3.3 Hyperlinks in a Wikipedia article 15

4.1 A radix tree example 19
4.2 The storage structure of DBpedia NTriples 21

5.1 Relationships between distances 30
5.2 Examples of encoding words 33
5.3 Examples of encoding words 35

6.1 Question classification flow chat 37
6.2 RSVP Translation flow chat 42

Chapter 1

Introduction

On recent decades, studies have shown that people rely for their daily information access on search engines and question answering (QA) systems, although, in the 1990s, most of them still depended on reading news or watching televisions. Due to the ceaseless enhancement and study, keyword-based information retrieval (IR) technologies have grown rapidly so that basic search requirements can be satisfied. In consequence, many commercial IR systems (e.g., Google and Yahoo!) arose and achieved great success.

In recent years, mobile applications with natural user interfaces, which attempt to accept voice input and cross-language queries, have created new challenges. Speech recognition and machine translation techniques that are used to transform such natural queries into text queries often introduce errors. For example, the speech recognition performs poorly when speakers have strong accents or environments are noisy. Moreover, the problem becomes more serious for machine translation when input questions are rare to their training data. For example, if we try to translate "狗能活多久" (*How long can a dog live?*) into English, Google translation returns "Dogs can live long". In order to identify user intentions, a QA system should be able to tolerate such machine-generated errors.

In a QA system, the question classifier plays a very important role. It categorizes queries into various domains, so they can be analyzed properly by domain-specific methods. Although the text classification problem has been well studied, traditional statistical approaches do not work well in practical QA systems, in which domain settings are usually

vague and even overlap sometimes. For example, in our testing set, the query "*Where is the nearest restaurant?*" belongs to Restaurant domain and Map domain, since it can be answered by both domain APIs, which are Google Map API and Yelp Restaurant Search API. In order to process queries properly, a question classifier in QA systems should have a robust performance even in complicated domain settings.

The problems of machine translation and question classification are considered as related, since they all depend on the calculation of *semantic distance*. However, the semantic distance is very difficult to be defined or to be computed. In this thesis, we propose an approximation method of such a semantic distance using information distance theory ([2]). The algorithm takes advantage of a semantic cluster dataset built from 35 million question-answer pairs (QA pairs) to estimate sentence-level semantic similarities. In practice, we show that the distance measurement can be used to solve previous problems. To correct a translation result, we simply find question templates from our dataset with small semantic distance to it and rewrite it according to the retrieved templates. For question classification, we create positive and negative question examples for each domain. Given an input question and a domain, we classify it as true if it has a smaller semantic distance to positive examples than to negative examples. Otherwise, it is classified to false.

Moreover, with the semantic distance, we are able to collect clues about how to answering it easily. Given a question, if we can find a semantically similar question with known answers, the same answers can be used to answer the question. For example, if we already have an answer to "*What is the population of Toronto*", we just return the same answer to similar questions like "*How many people live in Toronto*".

In our experiments, based on the semantic distance, a translation corrector and a question classifier have been implemented. They are compared, under the same test settings, with several state-of-the-art academic works and popular commercial systems. The experimental results show that our methods are close to the best results of the baseline approaches and outperform them in most cases.

1.1 Natural Language Processing and Question Answering

Natural Language Processing (NLP) is an area of research and application that explores how computers can be used to understand and manipulate natural languages (human languages). Applications of NLP contain a number of fields of studies, such as text classification, machine translation, Part-Of-Speech(POS)-tagging, named entity recognition (NER), relation extraction, and so on.

Question Answering is another area of computer science within the fields of IR and NL, which is interested in building systems that automatically answer natural language questions posted by humans. QA Systems usually generate their answers by querying a structured Knowledge Base[1], or by searching an unstructured collection of natural language documents, e.g., Internet web pages.

Over decades, QA researchers have studied a wide range of question types including *Fact*, *Definition*, *How*, *Why*, *Hypothetical*, and *Cross-lingual* questions. Various methods have been applied. Another classification divides QA implementations into two types: close-domain QA[2] and Open-domain QA[3]. In this thesis, our QA system is an open-domain QA that supports factoid questions and cross-lingual questions.

A standard cross-lingual QA system mainly consists of five steps:

1. Query normalization

   (e.g., voice recognition and cross-language translation)

2. Natural language query parsing

[1] A graph knowledge base is a topical type of the knowledge bases, in which nodes represent ontology entities and a directed edge represents one certain semantic relation between two connected ontologies. For instance, there are two famous implementations - Freebase [5] and DBpedia [27].

[2] Closed-domain QA deals with questions under a specific domain (for example, weather or map), and it can be seen as an easier task because supported types of questions are limited.

[3] Open-domain QA deals with questions within many domains, thus the question parser cannot depend on hand-written rules.

(e.g., POS-tagging, syntactic chunking and question classification)

3. Answer candidates retrieval
4. Answer candidates ranking
5. Answer generation

Apparently, the qualities of Question Normalization and Natural Language Query Parsing are critical in such a pipeline QA system, since it will not find any proper answers when any of the top two steps fails. In this thesis, we only focus on improving these two components.

1.2 Question Normalization

In question normalization, voice queries and cross-lingual queries are transformed into text queries in English by speech recognition and machine translation technologies. In fact, they are not quite helpful in practical QA systems, because they introduce too many errors, either from environmental noises or from over-fitting statistical models they applied.

A previous work in [42] has stated a similar method correcting such errors in speech recognition. They simply applied the same method on machine translation without noticing that errors of speech recognition are significantly different from errors of machine translation. The speech recognition often returns a correctly structured sentence with few mistaken words or phrases. For example, "How many legs does a spider have?" may be recognized as "**Hole** many legs does a spider have?". However, the machine translation usually translates words/phrases correctly but assembles them in a bad order. For example, a Chinese question "狗能活多久" (*How long can a dog live?*) could be mistakenly translated into "Dogs can live long". In this thesis, we design a general semantic measurement, which can significantly reduce errors in both situations. Also, we construct a dataset of semantic clusters, which is used to measure semantic closeness of two questions directly. Since the problem of speech recognition has been introduced in [42] already, we only evaluate the correction of machine translation in this thesis.

1.3 Natural Language Query Parsing

A natural language query is an information access request in a form of human language, especially in text. Informally, it has the same meaning with the term of "*question*". The greatest challenge of building QA systems is to understand and to process input questions. Given a possibly noisy query, the system needs to classify it into the correct question categories(we call them vertical domains in the field of QA), or to determine whether it is semantically similar to a question with known answers. Traditional QA systems used to have rule-based query parsers, which are extremely hard to scale up and cost too much human efforts on maintenance. Although statistical text classifiers get rid of the trivial manual works, they preform poorly on practical domain settings. In our QA system, to overcome such difficulties, we developed a more efficient question classifier by using a unified algorithm of semantic distance approximation.

The semantic measurement relies on a dataset of *semantic clusters* constructed from a very large QA dataset, in which each cluster represents a question template annotated by possible semantic intentions. To construct such clusters, we first extract DBpedia entries from questions and their answers, and secondly discover relations over the DBpedia graph between question entries and answer entries. Finally, we cluster questions with same templates and with same relations respectively to generate the semantic clusters. Figure 1.1 lists several template examples in the cluster of *Spouse* relation.

The clusters dataset is a very helpful tool to measure semantic similarity between two questions in very different forms. In such clusters, we are able to calculate the semantic distance between "*What is the population of Toronto*" and "*How many people live in Toronto*" by comparing the relations of their representative templates. In this case, their distance is relatively small because their templates are within one semantic cluster.

1.4 Contribution

Our three main contributions are:

who is <PlaceHolder>'s wife
who was <PlaceHolder>'s wife
who is the wife of <PlaceHolder>
who was <PlaceHolder>'s first wife
what was <PlaceHolder>'s wife's name
what is <PlaceHolder>'s wife's name
what is <PlaceHolder>'s wife called
who is <PlaceHolder> married to
who is married to <PlaceHolder>
who married <PlaceHolder>
who was married to <PlaceHolder>
...

Figure 1.1: Question template examples labeled by "Spouse"

- We generate a dataset of semantic clusters without any manual efforts, which contains a great number of annotated question templates. It is not only useful for measuring the semantic distance but also helpful for developing other question-understanding methods.
- We propose a unified algorithm that approximates the semantic distance by information distance theory.
- We develop two practical applications: a translation corrector and a question classifier, which have been aggregated into our RSVP QA system.

1.5 Organization

In this chapter, we briefly introduce technical background, motivation, and procedure of our research. Related works are included in chapter 2. Chapter 3 provides important observations in related techniques and useful datasets; moreover, the main challenges are

summarized here as well. In chapter 4, we construct a dataset of semantic clusters by processing the QA pairs dataset. The approximation of the semantic distance has been introduced in chapter 5. In chapter 6, we apply our semantic measurement into two popular NLP applications and conduct comparison experiments respectively. Conclusions and future work are included in the final chapter 7.

Chapter 2

Related Work

2.1 Relation Extraction

Relation extraction is a sub-area of NLP that aims to detect and classify relationships between two latent concepts in natural language text. Many types of relation extraction approaches have been explored. They mainly includes bootstrapping, supervised classification, unsupervised classification and distant supervision approaches.

Bootstrapping starts with a small set of labeled relations (or templates), iteratively discovers templates from these initial seeds, and searches the templates to discover more seeds ([6]). This approach may suffer from semantic drifting since it is hard to eliminate ambiguity while iterating. Moreover, the method may have a low recall because of the limited initial cases.

Supervised learning approaches are able to extract more general templates ([23]; [13]), but depend on a large set of labeled data. In fact, due to lack of annotated data, most works have been evaluated on small datasets.

Unsupervised approaches do not require labeled data. They usually take advantage of the arguments of dependency paths [1]to find semantic relations. For example, Lin and

[1]Dependency path is the path linking a node to another in a dependency parsing tree.

Pantel's work [32] used distributional similarity of dependency paths to discover different representations of the same semantic relation. However, these approaches usually suffer from model over-fitting and parsing mistakes in informal written English.

Distant supervision is supervised by a knowledge base, in stead of human annotations. It builds the training data by matching its relational entities though the text corpus, and extracts new entities using the trained classifiers ([35]; [10]).

Our problem is extracting relations from question-answer pairs and a different method has been applied. In order to interpret intention of a question, we need establish relationships between words/phrases in questions and words/phrases in their answers. To get rid of human efforts, the DBpedia knowledge base has been used to supervise the extraction process. Firstly we discover every potential relation candidate. Next, we rank all candidates by a co-reference confidence and statistical significance over 35 million questions.

2.2 Question Classification

Question classification has long been studied in the field of NLP and QA. Different works stick on different categories or domains. From the prospective of question interests, [30] introduces a taxonomy, including Abbreviation, Description, Entity, Human, Location and Numeric Values. Questions are categorized as Solution, Reason, Fact, etc. in [7] by different content types. Recently, many works have build on a popular question taxonomy of TREC. However, we classify the questions into more concrete *vertical domains*, such as Weather, Restaurants and Maps, hence classified questions of each domain can be answered in each domain much more easily by domain answering engines (e.g., Yelp Restaurant Search API or Google Weather API).

Content features of question text are very useful in the intent-identification purpose, such as lexical features (e.g., bag-of-words, n-gram), syntactic features (e.g., POS-tagging, parse trees [18, 37]) and semantic features (e.g., synonyms from WordNet [34]). A standard approach [29] makes use of the language model, while another study [41] only involves language-independent features. Recent decades, supervised machine learning methods have

commonly been used to solve the clasification problem and achieve a great progress. Syntactic and semantic features combined with popular learning models (e.g., logistic regression and SVM) are applied to classify questions in [30, 46]. To decrease the annotation efforts, unsupervised methods or semi-supervised methods, such as latent semantic indexing ([15]) and latent Dirichlet allocation ([4]), have been studied. They turn out to be efficient and useful in documentation cluster analysis. But they are not able to establish reliable probability distributions over short documents as questions. In practice, questions are usually not proper English and domain settings can be complicated. Traditional supervised and unsupervised classifiers do not have a good solution for these situations . However, our semantic distance-based question classifier performs much better.

2.3 Semantic Distance

The semantic distance is intuitive but difficulty to be computed. Over decades, many researchers in computational linguistics have tried to explore certain semantic measurements for various purposes. Using notations in [9], let c_1 and c_2 be two synonym sets (synsets), $L(c_1, c_2)$ be the WordNet path length from c_1 to c_2, and $lso(c_1, c_2)$ be the lowest super-ordinate of c_1 and c_2. Many similar studies are given briefly as following.

1. At the word level, Hirst and St-Onge proposed a measure of semantic relatedness defined as
$$rel(c_1, c_2) = C - L(c_1, c_2) - k \times d$$
where d is the number of times the path changes direction on the WordNet tree, and C and k are constants.
2. At the word level, Leacock and Chodorow approximated the semantic similarity by
$$\text{similarity}(c_1, c_2) = -\log \frac{L(c_1, c_2)}{2D}$$
where the length function L uses only hyponymy links (i.e., "is-a" relations), and D is the overall depth of the taxonomy.

3. At the word level, Resnik introduces *information contents* to the measure by defining

$$\text{similarity}(c_1, c_2) = -\log P(lso(c_1, c_2))$$

where $P(\cdot)$ is the probability of encountering an instance of a synset c in some specific corpus.

4. At the word level, Jiang and Conrath also used *shared information content*:

$$\text{dist}(c_1, c_2) = 2\log P(lso(c_1, c_2)) - [\log P(c_1) + \log P(c_2)]$$

5. At the word level, Lin's semantic similarity measure is

$$\text{similarity}(c_1, c_2) = \frac{2\log P(lso(c_1, c_2))}{\log P(c_1) + \log P(c_2)}$$

To some extent, these approaches provide useful semantic similarities in word-level, but none of them gives a comprehensive semantic distance of natural language sentences. Our previous papers succeed in applying information distance theory ([2]) to approximate the semantic distance. The quanta QA system by [47] use it to measure relevancy of an answer candidate to a given question. In addition, [42] implements a speech recognition correction system, in which such distance is exploited to compute the closest question template candidates. They both infer semantic distance of sentences from word-level distance of WordNet. In this paper, we follow the same theory but propose a better sentence-encoding method, which measure semantics of question directly. We also improve the the word-level encoding by covering more English words from Yago2 and DBpedia knowledge bases.

Chapter 3

Observations

3.1 Community QA

Recent years, Community Question Answering (CQA) websites have emerged to meet advanced IR requirements. They are designed as popular social networks, which allow their users to post questions on the CQA website and receive answers from other users. However, the CQA uses keyword-based techniques to retrieval questions, which is far from satisfactory for queries that have the same meaning may use very different wording. When people fail to find any similar questions, they post their own questions. Therefore, many semantically identical questions are included in the question and answer data of CQA. A typical CQA site - *Yahoo! Answers* is shown in 3.1.

For NLP researchers, the large dataset of QA pairs is a valuable natural language resource, including 35 million questions and answers. It also covers a wide range of questions types (e.g., question of factoid, opinions, summaries and so on). Another advantage is that the QA content will be kept expanding and updating by users everyday.

In addition, we observe that such data is extremely useful for measuring semantics of questions, which potentially allows a QA system capable for understanding a wider range of questions. If we could cluster question with similar answers together, then given two considerable different questions, we are able to determine whether they carry same

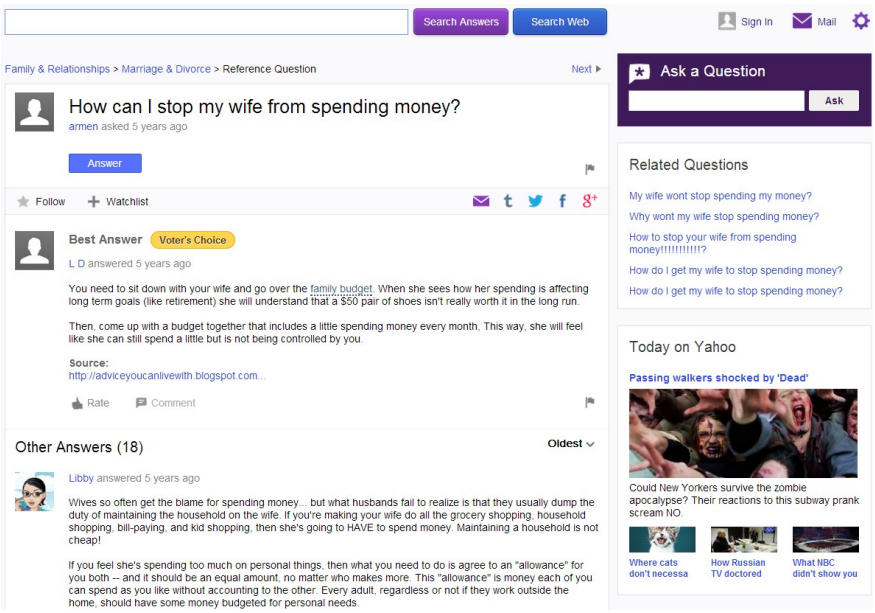

Figure 3.1: A example of the CQA web site (Yahoo! Answers)

semantics by searching their question clusters. However, the challenge is that the textual data is very noisy. Typos and grammar mistakes commonly appear in both questions and answers. For examples, "*Brad Pitt married* ***who****?*", "*Has Brad Pitt* ***marri*** *Angelina yet?*" and "*Is* ***Otawa*** *the capital of Canada?*". Even worse, answers are far from accurate, usually containing dominating irrelevant information. For example, the introduction from Barack Obama's wikipedia page might be posted to answer questions like "*Who is Barack Obama's wife?*".

3.2 Knowledge base

The knowledge base we are using is organized as a directed graph, of which concepts as nodes are connected by their relations. It is stored in the format of RDf(Resource Description Framework) n-triples. Each triple consists of a **Subject**, a **Predicate** and an **Object**. A subject is a concept, which could be a realistic object (e.g., a person or a place)

or a notion (e.g., happiness). A predicate is a special concept representing the relationship from a subject to a object. An object is either a concept or a literal value (e.g., a number or a date). These triples represent a graph that predicates as directed edges link subjects to objects. Figure 3.2 gives an example of the graph database. In practice, it is stored in Linked Data databases or other triple storage systems (e.g., Virtuoso), which supports certain types of graph search query (e.g., SPARQL).

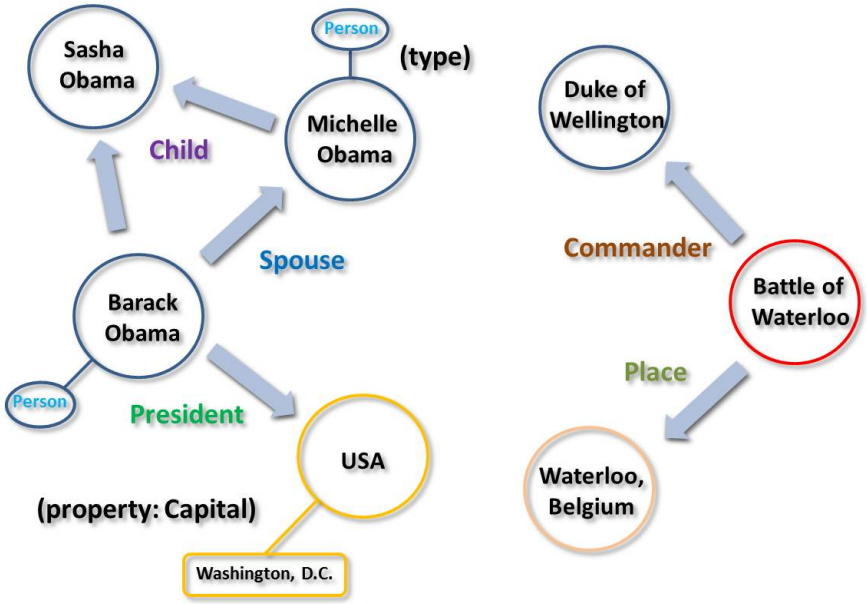

Figure 3.2: A example of the knowledge base

Current knowledge bases are mainly generated by two ways. The first one derives from a unified knowledge resource, such as Wikipedia, which is the largest on-line encyclopedia website, containing over four million articles in English created by domain experts. The DBpedia project [27] extracts structured knowledge from Wikipedia articles and opens it up to programmatic access using Semantic Web technologies. The other type makes use of different machine learning and NLP technologies to identify structured information from reading the entire Internet. They usually use a bootstrapping approach, which keeps discovering concepts and relations iteratively, e.g., Google Freebase and the NELL (Never-Ending Language Learner) project.

To get rid of human efforts, we use the DBpedia knowledge base to supervise our relation extraction of QA pairs.

3.3 Anchor text of Wikipedia

To build connections between questions and answers, the first challenge is to do the schema matching, which is to map words/phrases to database entries. For DBpedia database, the schema-matching information is contained in hyperlinks of anchor text in Wikipedia. A hyperlink usually consists of a text phrase and a Wikipedia page link to which the text refers. Moreover, a Wikipedia page link can be easily rewritten into a DBpedia URI. Such anchor text is used to implement the schema-matching method. An anchor text example was shown as Figure 3.3.

Figure 3.3: Hyperlinks in a Wikipedia article

However, we still need to resolve ambiguities, since only the textual features are considered here. A single concept could have various textual forms. For example, "UW",

"*University of Waterloo*" or "*UWaterloo*" all could be used to refer to the **University of Waterloo**. On the other hand, lacking context, a single text phrase may suggest several different concepts. For example, the word "*Waterloo*" could refer to the **Battle of Waterloo** or the location of **Waterloo in Ontario, Canada**. We need to eliminate such ambiguities from statistical information of the entire dataset.

3.4 Challenges

In Section 2.3, many approaches attempt to give a comprehensive semantic distance between natural language queries. We follow their established methods and introduce new ideas. However, many difficulties and challenges need to be solved.

At the word level, we need our vocabulary rich enough and extensible enough, since new notions and new meaning of old notions appear rapidly in our daily lives. At the sentence level, many approaches are able to measure semantics of two aligned questions by a syntactic edit distance or a synonyms substitution method. However they still do not hold for "*What is the population of Canada*" and "*How many people live in Canada*". Also, computing the semantic distance is apparently language dependent. For example, in Chinese, question indicators, such as "多少" (i.e., *how many*), "什么" (i.e., *what*) and so on, can be put anywhere in sentence. But they usually appear in the head of sentence in English. To come up a language-independent approach, we need additional data to provide nontrivial lingual statistical information.

Recent mobile applications use speech recognition or machine translation to process voice queries and cross-lingual queries. They actually introduce more errors. If we ask Siri, "*What do lobsters eat?*", Siri give answers of seafood restaurants nearby. Also, if we want to translate "狗能活多久" (*How long can a dog live?*) into English, Google translation returns "*Dogs can live long*". Our system needs to tolerate such errors.

QA systems often suffer from typos and grammar mistakes too. Especially, on portable devices, the awkward typing experience leads to more mistakes. In voice input applications, accent of speakers and surrounding noises lead to recognition problems. We need our QA system to overcome these difficulties.

Chapter 4

Semantic Clusters

Thirty-five million question-answer pairs have been crawled from a popular CQA website - *wiki.answers.com*. The dataset contains two features that are significant for understanding questions:

1. It contains extremely abundant questions, and many questions can be aligned on one template. For example, "*Who is Barack Obama's wife*" and "*Who is Brad Pitt's wife*" share the template "*Who is <PlaceHolder>'s wife*".
2. Similar answers of different questions suggest their semantic similarities. For example, "*Who is Barack Obama's wife*" and "*Who married Barack Obama*" are two variations of one question, if we found **Michelle Obama** (Barack Obama's wife) in both of their answers.

However, it is not straightforward to extract such morphological information and semantic knowledge from questions. Ideally, we want to learn question templates and their latent intentions from a knowledge base. For instance, a question "*Who is Barack Obama's Wife?*" should be decomposed into three dimensions, as a template "*Who is <PlaceHolder>'s wife*", a question main concept **Babarak Obama** and a relation *dbpedia:Spouse*.

We first extract DBpedia entries from the text of each QA pair, and secondly discover relation paths over the DBpedia graph between question entries and answer entries. Finally,

questions are clustered by their templates and relation paths. We name the constructed dataset as the semantic clusters.

Before introducing detailed techniques, let us unify our terminologies. DBpedia is represented by a set of triples $DB = \{r_1, r_2, ..., r_n\}$ where a triple is $r_i =< Subject, Predicate, Object >$. The SUJ contains all Subjects in DB, $PRED$ contains all Predicates and OBJ contains all Objects. The DBpedia entry set E is $SUJ \cup OBJ$. In the QA pair dataset, questions and answers are both word sequences. Given a question $q = \{w_1, w_2, ..., w_n\}$, the template $t = \{w_1, ..., w_{i-1}, PlaceHolder, w_{j+1}, ..., w_n\}$ is generated by replacing a matched surface text $s = \{w_i, w_{i+1}, ..., w_j\}$ by a place-holder in the sentence.

4.1 Schema matching

Given a database, the schema matching is to find database entries from a word or phrase. The DBpedia schema contains two types of entries, including entity and literal. An entity is a concept, which could be a person, a place, an event or an abstract notion. DBpedia entities are stored in a format of URI (uniform resource identifier), corresponding to a descriptive web page, e.g., *<http://dbpedia.org/resource/Autism>*. The anchor text of Wikipedia pages has been used to map text phrases to DBpedia entities. A literal is a typed value, such as a date, a string, or a number. Rule-based approaches are developed to extract the literal values from sentences.

We exploit a textual search function that finds DBpedia entries from a text phrase.

$$Entry : \{w_1, w_2, ...\} \rightarrow E' \subset E \tag{4.1}$$

Given a word sequence W, schema matching extracts all possible subsequences (s) of W that have entries found by Function 4.1.

$$SMatch(W) = \{Entry(s) | \forall s \sqsubseteq W, Entry(s) \neq \varnothing\} \tag{4.2}$$

For retrieval efficiency, a Trie index has been used to implement the Function 4.1. A Trie, also called digital tree or prefix tree, is an ordered tree data structure that is used to

store a set of strings. For better space efficiency, a compact Trie (also called radix tree) has been used. The compact Trie is a space-optimized data structure where each node with only one child is merged with its child. Thus their edges could be labeled with multiple characters, as shown in Figure 4.1.

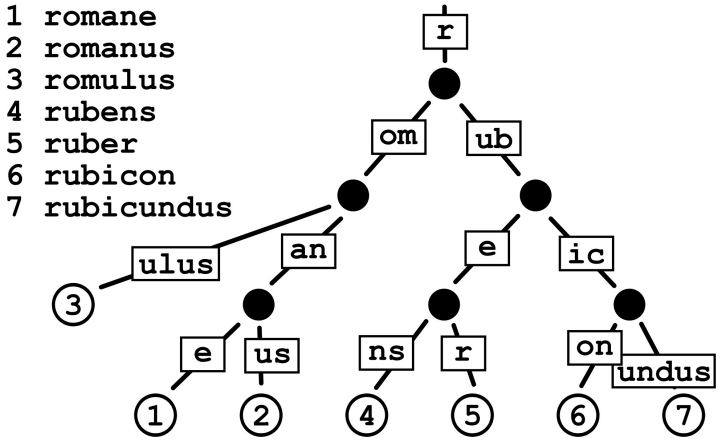

Figure 4.1: A radix tree example

As same as a Trie, the path from root to each node is associated with a string, which can be represented by linking characters of all edges in the path. All the descendants of a node share a common prefix of the string associated with that node, and the root represents the empty string. If a string associated with a node is a key, the index of the key will be stored in the node. It is a more efficient string search solution than a hash table, especially for the subsequence matching task. Given a search string with length m, the Function 4.2 only takes $O(m)$ time on a Trie implementation in the worst case.

In practice, variations of literals have been considered while building index. Given text with different forms as "May, 11th, 1985" and "1985-05-11", we identify them and convert them into a unified date format. To extract entities, over a billion hyperlinks are extracted from Wikipedia page dump (in a xml format). For both entities and literals, the schema matching attempts to find the longest match on the Trie index from text beginning till end recursively. We run the same matching process on question and its answer.

To evaluate the schema-matching confidence for each surface entry pair, we estimate their probability distribution from the hyperlinks of Wikipedia dump. The $Count_h(s, e)$ counts hyperlinks that contain entity e and text s, while $Count_h(s)$ counts hyperlinks that contain text s.

$$P(e|s) = \frac{P(e, s)}{P(s)} = \frac{Count_h(s, e)}{Count_h(s)} \tag{4.3}$$

Then we could give a confidence calculation method based on the conditional popularity of an entry e given a surface text s.

$$C(e, s) = Popularity(e) * P(e|s) \tag{4.4}$$

The score gives us a rough idea about how likely a text is mentioning an entry. But it is still not sufficient to disambiguate, since we only take into account textual features. For example, we can never tell whether the term "*Waterloo*" is referring to the city of **Waterloo, Ontario** or about the city of **Waterloo, Belgium** without additional information. Later on, we introduce methods that eliminate such ambiguity by using contextual information and statistical significance.

4.2 Latent Relation Extraction

In this section, we identify semantic intentions of questions by extracting relationships between questions and their answers. A predicate path (e.g., child -> birthPlace) over the DBpedia graph indicates a type of semantic intention, hence we take all the DBpedia predicate paths to be our relation space. Given an entry pair (e_q, e_a), the relation extraction is to discover possible predicate paths connecting them, where e_q is a DBpedia entry matched by a surface text in a question and e_a is matched in its answer. The predicate path search function is defined as following:

$$path : (e_q, e_a) \rightarrow PRED^{\infty} \tag{4.5}$$

where $path(e_q, e_a)$ return all predicate paths $P = \{p_1, p_2, ...\}(p_i \in PRED^{\infty})$ that connects e_q and e_a in the DBpedia graph. Our program iteratively attempts to discover predicate paths on each entry pair. In addition, we run it on every QA pair.

To solve the path search problem on a graph data, triple storage systems (e.g., Virtuoso) and other graph index techniques are usually applied. They are very extensive supporting general graph search queries (e.g., SPARQL), but in our case, they are not fast enough on the path retrieval for over billions of entry pairs. Thus we implement a much more efficient retrieval system by sacrificing unneeded functionalities (e.g., to support extensive query types). A distributed hash map has been used to index the DBpedia triples. Subjects are stored as keys in the map, values are ranges of triples storage, i.e., start offset and end offset. Figure 4.2 is the storage structure.

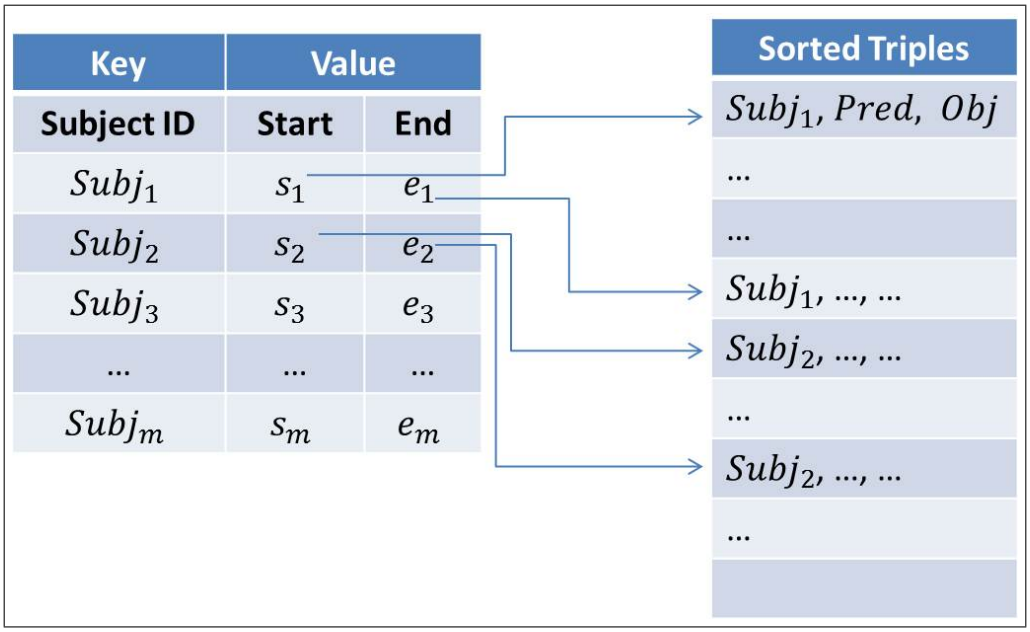

Figure 4.2: The storage structure of DBpedia NTriples

Instead of storing entire URI strings of all entries, only their ids are kept to save memory. Given two DBpedia entries, it is easy to find directed paths by a two direction BFS (Breadth-first search). The statistics of the QA pair dataset shows us there are very few predicate paths longer than three, and most of them are irrelevant or schema-matching mistakes. Actually, in general QA, question posters seldom expect sophisticated answers that are "remotely" related with the question. So we constrain depth of the BFS to no

more than three levels. It makes the relation retrieval extremely fast so that billions of entry pairs can be processed in a few hours.

4.3 Clustering

By assuming that each question has a unique semantic intention which is associated with a single concept of the question, we align similar questions to generate templates and select the most possible predicate path for each template.

The first difficulty comes from the diversity and ambiguity of languages. Our schema-matching method spots multiple surface strings in a question, and each string could match many different DBpedia entry candidates. For example, "*Waterloo*" in "*Who is the **mayor** of **Waterloo***" matches DBpedia entities of "**Waterloo, Ontario**", "**Waterloo, Belgium**" and "**Battle of Waterloo**". The "*mayor*" may be matched to some entities too. The relation extraction naturally get rid of some entries that have no related concepts in their answers. Actually, statistical information of the entire QA collection can help us determine the correlation between an entity and a question too. For instance, there are lot many questions with the template "*Who is the mayor of <PlaceHolder>*" but few with the template "*Who is the <PlaceHolder> of Waterloo*". Then the template of "Waterloo" are more likely to be chosen.

Another difficulty we need overcome is distinguishing relevant relations from a noisy answer. In an answer text, many irrelevant concepts may be included, which sometimes lead to unexpected relations. For example, multiple predicate paths including *spouse* and *graduate* are found between question "*Where did **Bill Clinton** graduate from*" and its answer "*Bill Clinton and **Hillary Clinton** received law degrees from **Yale Law School***". Even in an accurate answer, it still confuses us by having multiple paths between two entries. For example, given the question "*Who is the mayor of Chicago?*" and its answer "*Rahm Emanuel*", two predicates of *leaderName* and *birthPlace* are found between "*Chicago*" and "*Rahm Emanuel*". To eliminate such ambiguities, we need to take advantage of statistical significances of the entire dataset.

4.3.1 Question Template

Now we introduce the algorithm that constructs the semantic clusters. The first step is to train templates by aligning questions. After schema matching and relation extraction, we already have a rough template for each question. However, to have better alignment results, several text normalization methods have been applied on the rough question templates.

1. Remove stop words of question sentences, such as prepositions "*the*", "*a*" or "*that*";
2. Remove irrelevant punctuations in the middle of the sentence, such as "," and ";";
3. Stem verbs and nouns. Plural nouns are converted to singular nouns. Variations of verbs are unified to a basic form, e.g., "*is*", "*were*" and "*am*" are unified to "*be*";
4. Replace mentions of location, person and date mentions with NER tags (e.g., LOC, PER and DATE), respectively.

The questions with same templates are grouped together. In order to measure semantic similarity of two questions, the first step is to find their templates. In practice, we build our retrieval method on a traditional inverted index implementation (i.e., Apache Lucene). Each template is stored and indexed as a set of unigrams. Given a question, it finds matched template candidates sorted in a decreased order of the tf-idf(term frequencyâĂŞinverse document frequency) scores. Since we only deal with the questions that has one key concept (a DBpedia entry matched in question), questions with predicate paths associated with multiple words/phrases found would generate multiple templates. For example, two predicate paths found from word $\{w_i\}$ and word $\{w_j\}$ in the question $\{w_1, w_2, w_3, ..., w_n\}$, two templates of $\{w_1, w_2, ..., w_{i-1}, PlaceHolder, w_{i+1}, ..., w_n\}$ and $\{w_1, w_2, ..., w_{j-1}, PlaceHolder, w_{j+1}, ..., w_n\}$ are generated. We decrease the ranking score of such templates, because of unsolved ambiguities.

4.3.2 Predicate Path

For each template, predicate paths and corresponding entry pairs of questions are recorded. Given a question template t, Q_t is the question set of t and $Q_t(p) \subseteq Q_t$ represents a question

subset in which all questions are associated with the predicate path p. The probability distribution of predicate paths over templates are established.

$$P(p|t) = \frac{P(p,t)}{P(t)} = \frac{|Q_t(p)|}{|Q_t|} \tag{4.6}$$

The conditional probability gives a general image how a predicate path is related with a question template. In a group of questions, frequent predicate paths are more likely representing their latent semantic intentions. However, over all question groups, commonly extracted predicate paths are more likely to be irrelevant, comparing to those only belonging to few templates. For example, empty predicate paths that results from question-keyword rephrasing, is meaningless. To take into account such information, we borrow the idea from a very popular IR measurement: idf (inverse document frequency) to boost confidence scores of predicate paths. The idf score of a predicate path p is computed as dividing the number of all templates (T) by the number of templates containing it ($T(p)$). In question template t, the confidence score of a predicate path p is calculated by Equation 4.7:

$$Score_t(p) = log(\frac{|T|}{|T(p)|}) * P(p|t) * \sum_{\substack{q \in Q_t(p) \\ p \Leftarrow (e_q, e_a)}} \frac{C(e_q, s_q) * C(e_a, s_a)}{|Q_t(p)|} \tag{4.7}$$

where s_q, s_a are surface text of e_q, e_a which generates the predicate path p.

In each template, the predicate paths are ranked by the confidence calculation. In practice, we take the top path of each template as the semantic relation. Therefore, question templates with the same predicate paths are clustered together. In Appendix A, we list several examples of our clustering results.

Chapter 5

Semantic Distance

In this chapter, we use the same problem formalization and semantics approximation from our previous work [16], in which we attempt to classify questions by calculating a semantic distance, which is introduced in Section 6.1 of this thesis. This work has been done by my lab partner Guangyu Feng and me(as a co-author). In addition, we show that the same theory can also be applied to solve a machine translation problem.

To interpret queries, the ideal way is to measure their semantics. However, we do not know either how to formally define the semantic distance, or how to compute it. When we talk about "computation", is it the same as "Turing computation"? This particular question relies on the Church-Turing thesis. Logicians, philosophers, and computational linguists, beginning with Richard Montague [36], have studied theories of natural language semantics and its relation with syntax. While Montague semantics is not interested in the real world, but in semantical properties of language, our intuitive notion of semantic distance is really not definable (and undecidable, for example using Gödel's theorems), just like we do not know how to define computability without relying on the Church-Turing thesis.

Similar as approaches in Section 2.3, the purpose of our research is providing a unifying theory and a systematic method to implement the theory. To solve the problems mentioned in Section 3.4, our theory needs to be natural, universal, and free from *ad hoc* feature selections for each new domain. Thus this new theory of approximating semantics should

be of independent interests. It should be noted that in this study, we restrict ourselves to the scenario of single-question QA, instead of conversations.

5.1 Problem Formalization

Intuitively, each query may be thought of as a point, or an information carrying entity, in the information space. We wish to approximate the *semantic distance* between any pair of such information carrying entities using a well-defined "distance". This distance must satisfy basic metric properties, such as the triangle inequality. Thus, given a new query, all we need to do is to measure its "distance" to another question or a domain, hence, properly classifying it. We also wish to make sure that the distance we choose is the only metric. Additionally, when we introduce a new domain and its associate language models for its queries, it will be governed by the same distance metrics so that the new domain language models do not overextend, which might cause conflict with other existing domains.

Let us refine the problems we wish to solve. Specifically, the main problems that we address in this paper are:

1. *Query variation.* Human understanding is robust. In Figure 1.1, we have mined hundred grammatically correct ways to query about "spouse" relationship. Furthermore, when a sentence is grammatically wrong, or contains irrelevant words, we usually still understand. We not only understand "How tall was the world trade center?", or "What is the height of the world trade center?", but also understand "How tall is the **ward** trade center?". Given a question from voice recognition (possibly distorted), or translated from another language, or a proper English question having no answer, we want our system to be able to find a semantically similar question with an answer in the database.
2. *Query classification.* "*What is the temperature outside?*" is asking about weather, while "*What is the temperature of boiling water?*" is not. "*Is the weather suitable to play golf today?*" is a question for the weather domain, while "*What is the climate like on Mars?*" is not. We would like our system to perform classification before finding the answer to a question, i.e., given a (possibly noisy) question, to classify it

into one of the vertical domains which are semantically represented by positive and negative query examples.

These two problems can be seen as related, as they both involve defining a distance between the questions. To solve Problem 1, query variation, we classify questions within a small distance from each other as being the same. To solve Problem 2, query classification, given positive and negative examples of questions for each domain, we choose the do- main with a representative question that has the smallest distance from the query.

Therefore, our solution to the problems stated above relies on our intuitive notion of "*semantic distance*", or a good approximation of it, between two questions. Although a formal "semantic distance" is undefinable and uncomputable, is it possible to find a non-trivial distance metric that would minorize (be no more than) *all* well-defined computable approximation of our intuitive concept of *semantic distance*? Not only we want the theory to be mathematically unique and optimal, thus not replaceable, but also we want a systematic method to implement or approximate this theory for the task of natural language query understanding.

To formalize the notations, we have a set of general questions, Q, and vertical domains, $V_1, V_2, \ldots, V_k$. We assume that each domain V_i has a set of comprehensive natural language queries, $Q_i \subseteq Q$, as well as a fixed number of APIs to answer domain-specific questions or to perform domain-specific tasks. The *semantic distance* between two queries, q_1 and q_2, is *denoted* as:

$$d(q_1, q_2).$$

Thus, given a query q, the distance from q to a domain V_i is *defined* as

$$\min_{q' \in Q_i} d(q, q').$$

To reiterate, the semantics distance d cannot and will not be formally defined. In the next section we will give a provably best approximation to it.

5.2 Approximation

Our intuitive concept of semantic distance cannot be well defined and cannot be precisely computed. In Section 2.3, we have seen many authors trying to approximate it. However, can we have an approximation that is indisputably *universal*? The task seems to be impossible: how can we approximate something that is not yet formally defined, and approximate it so well that it covers all reasonable approximation of it?

It turns out that this indeed can be done. The idea is to use *information distance* [2]. Although it is still not computable, it directly suggests many natural ways of approximation. We first follow [28] to give a brief and informal introduction to information distance. The theory of information distance depends on the theory of Kolmogorov complexity, which was invented in the 1960s. Fixing a universal Turing machine U, the Kolmogorov complexity of a binary string x condition to another binary string y, $K_U(x|y)$, is the length of the shortest (prefix-free) program for U that outputs x with input y. Since it can be shown that for a different universal Turing machine U', the metric differs by only a constant, we will write $K(x|y)$, instead of $K_U(x|y)$. We write $K(x|\epsilon)$, where ϵ is the empty string, as $K(x)$. For a casual reader, it is sufficient to understand $K(x)$ simply as the number of bits of the shortest program written in your favorite programming language to output x, given no input. We call a string x *random* if $K(x) \geq |x|$. We refer the readers to [28] for further details of Kolmogorov complexity and its rich applications.

$K(x)$ defines the amount of information in one object x. What would be a good departure point for defining an "information distance" between two information carrying objects? In the early 1990s, the authors of [2] studied the energy cost of conversion between two strings x and y. John von Neumann hypothesized that performing 1 bit of information processing costs $1KT$ of energy, where K is the Boltzmann's constant and T is the room temperature. In the 1960s, observing that reversible computations can be done for free, Rolf Landauer revised von Neumann's proposal to hold only for irreversible computations. Starting from this von Neumann-Landauer principle, it was proposed in [2] to use the minimum number of bits needed to convert between x and y to define their distance. Formally, with respect to a universal Turing machine $U(\cdot, \cdot)$, the cost of conversion between

x and y is defined as:

$$E(x,y) = \min\{|p| : U(x,p) = y,\ U(y,p) = x\} \tag{5.1}$$

Clearly, $E(x,y) \leq K(x|y) + K(y|x)$. In [2] the following optimal result was obtained, modulo $\log(|x| + |y|)$:

Theorem 1 $E(x,y) = \max\{K(x|y), K(y|x)\}$.

Thus, this has enabled the definition of information distance between two sequences x and y as:

$$d(x,y) = \max\{K(x|y), K(y|x)\}.$$

This distance d was shown to satisfy the basic distance requirements, such as non-negativity, symmetry, and triangle inequality. Furthermore, d is *universal* in the sense that d always minorizes any other reasonable computable distance metrics, such as [22], [31], [44] and so on. In particular,

Theorem 2 *If $d'(x,y)$ is any reasonable (satisfying some basic density constraints) computable distance approximating the semantic distance, then there is a constant c, for all x, y,*

$$d(x,y) \leq d'(x,y) + c.$$

Now we are ready to make a bold proposal: let's equate information distance, which is well-defined, with our intuitive concept of "semantic distance". Figure 5.1 explains the consequences: for any computable traditional distance $d'(x,y)$ that tries to approximate "semantic distance", it is minorized by $d(x,y)$ in the sense that there is a c, for all x, y, we have $d(x,y) \leq d'(x,y) + c$. That is, if under d', x, y are close in "semantics", then so do they under d.

Thus, we have replaced an undefined concept *semantic distance* by a well-defined *information distance*, although both are un-computable. Our definition directly suggests a natural approximation (by compression). In the next section, we will demonstrate that

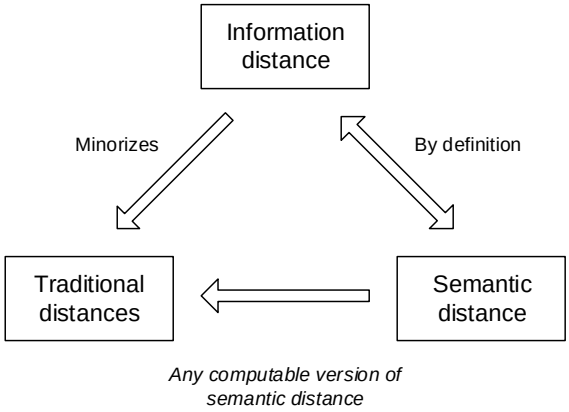

Figure 5.1: Relationships between distances

small semantic distance implies small information distance. On the other hand, does small information distance imply small semantic distance? In the QA context, from the data we have seen (our 35 million QA pairs and [38]), we believe this is usually the case.

Information distance and its normalized versions have been applied in bioinformatics as well as plagiarism detection [11], clustering [12] and many other applications [28]. In the field of text processing, topics include question and answering systems [47], multiword expression linguistic analysis [8], web page authorship, topic and domain identification, Internet knowledge discovery, multi-document summarization etc.

5.3 Encoding

In this section, we describe our approximation of information distance. Although information distance requires the non-computable Kolmogorov complexity, we demonstrate that we can effectively approximate such a distance in the domain of question answering. The advantages of our general theory include:

1. It unifies all semantic distance approaches under one roof, using one measure (en-

coding bits), so that any measure can be used as long as it is shorter.
2. The new theory is additive, so that we could naturally extend our encoding scheme from words to whole sentences.
3. The new theory is extendable, allowing other reasonable measures.

Applying additivity (2), we first encode words, phrases, then sentences recursively. Applying extendability (3), we introduce a new semantic measure, which is encoding sentences based on their templates in the semantic clusters.

5.3.1 Encoding Words

Given a word w, $K(w)$ should be a measure of how much information w carries. Without any information about the language, we can only treat each word as a uniformly random occurrence of the complete vocabulary. Say there is a total of N words in the English vocabulary, then we need $\log N$ bits as an average cost to generate any single one of them. This is our baseline of approximating $K(w)$, but the actual $K(w)$ could be bigger or smaller.

A useful tool for determining $K(w)$ is the POS tag. Words such as "the" (article), "into" (preposition) and "might"(auxiliary verb), are considered "irrelevant" words, carrying very little information, therefore can be assigned very small values of $K(w)$. On the other hand, nouns, verbs and adjectives carries relatively more information. Another useful piece of knowledge is that variations of a word should not change the value of $K(w)$ very much, since they carry the similar information. This holds for concepts as well. For example, "National Basketball Association" and "NBA" stand for the same organization. They have been take care of while normalization in chapter 4 and later in this chapter.

In order to further assign reasonable values of $K(w)$ and $K(w_1|w_2)$, a unified definition of semantic relations between them is needed. Here we use WordNet [34] and Yago [20]. The WordNet not only groups synonyms together, but also contains hierarchical semantic relations between words. In addition, the Yago expands its coverage by mapping DBpedia entities to WordNet synsets. For example, the database containing all nouns has the following useful properties:

1. Words that denote the same concept are within the same node (synset), for example, "sofa" and "couch" is contained in the synset *seating*. For words are not included in WordNet, DBpedia is used as a complementary method to identify them. Also, Yago categorizes the identified DBpedia entries into WordNet synsets. Hence, if w_1 and w_2 are within the same synset or mapped to the same synset, then $K(w_1|w_2) \approx 0$;
2. The synsets are organized in a tree structure. A parent-child relation represents hypernym-hyponym between them, for example, the synset {seating} is a parent of the synset {sofa, couch}. Therefore, if w_1 is a hypernym of w_2, then $K(w_1|w_2) = 1$, meaning given w_2, we need one step of generalization to produce the semantic meaning of w_1;
3. The root node, "entity", is the ancestor of all entities, thus every w that appears in the tree corresponds to a path originating from the root node. We let $K(w)$ to be the length of this path, i.e., the number of steps of specialization required to produce a specific entity.

Additionally, we also use the similarity and hypernym relations in WordNet's verb and adjective databases. Given two words w_1 and w_2, we first find the synsets s_1 and s_2 by looking up from WordNet dictionary or from Yago mapping database, and then calculate their information distance $d_w(w_1, w_2) = d_{wordnet}(s_1, s_2)$: (An example is shown in Figure 5.2)

- The POS tag of w_1 falls into the "irrelevant" category described above. Then $d_{wordnet}(s_1, s_2) = K(w_2)$. The same rule applies when w_2 is an "irrelevant" word.
- s_1 and s_2 are the same synset. Then $d_{wordnet}(s_1, s_2) = 0$;
- s_1 and s_2 have a least common ancestor (LCA) in the tree. Then $d_{wordnet}(s_1, s_2) = steps(LCA, s_1) + steps(LCA, s_2)$ where $steps(a, b)$ return the number of steps from a to b.
- s_1 and s_2 do not exist in the same tree. Then $d_{wordnet}(s_1, s_2) = \max\{K(w_1), K(w_2)\}$.

It should be noted that the process above applies not only to words, but short phrases as well.

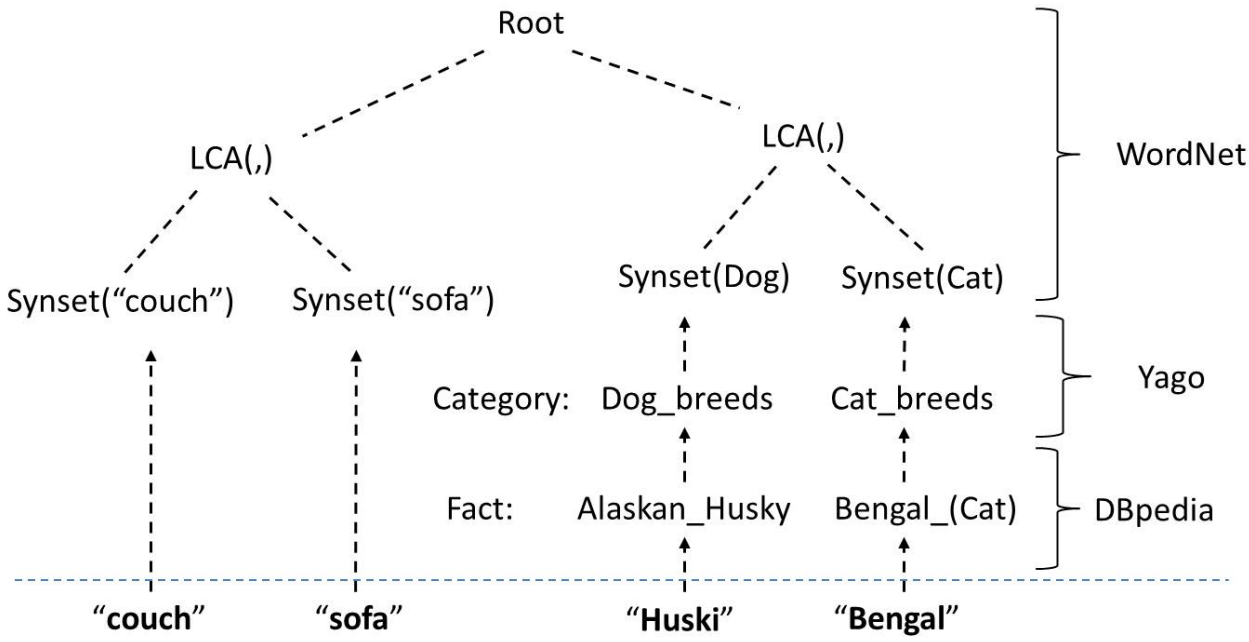

Figure 5.2: Examples of encoding words

5.3.2 Encoding Sentences

Two different approaches are considered to compute the semantic distance between two questions.

I Using additivity, we treat the sentences syntactically, and encode them by encoding words and phrases recursively.

II Using extendability, the semantic clusters are used to measure semantics of sentences. Given two questions, we find the closest templates of them among the templates, and measure the similarities between their predicate paths and main concepts.

The shortest of the two is taken as the final encoding.

Standard techniques of processing natural languages, such as normalization, named entity recognition, and POS tagging, are applied first, so that a sentence becomes a sequence of tagged words and phrases.

Approach **I** deals with the cases that templates cannot be found by questions. Then similarity here comes from alignments. An alignment of the two sentences is performed using the standard dynamic programming with respect to the word-level distances. Thus for sentences within small edit distances, for example, sentences differing only by one insertion/deletion or substitution, their information distance depends on the information distances between the words by which they differ, as in the case "What's the weather like in Beijing" versus "What's the weather like in New York".

Before performing the alignment, a few adjustments are conducted:

- Redundant words. Sometimes people add phrases such as "Can you tell me" and "I would like to know" before their questions. In our QA context we consider this to be irrelevant information and remove it during pre-processing. Stop words would be removed too.
- Permutations of phrases. "How is the weather in London" and "In London, how is the weather" are essentially the same question. Some re-ordering does occur in natural conversations. Therefore, we set a few rules to re-arranges the phrases. For instance, a time phrase at the end of a sentence is placed at the beginning of the sentence, so that all time phrases can be aligned.

In approach **II**, we deal with questions that cannot be aligned, For example, "*What is the population of Canada*" and "*How many people live in Canada*". These questions are using different templates but have the same meaning. Basically, a question is either explicitly or implicitly asking about relationships between entities. Questions asking about the same kind of relationship should have shorter relative encoding. The semantic cluster dataset introduced in Section 4.3 is used for this purpose.

Given two questions q_1 and q_2, we first find their best matching templates $t(q_1)$ and $t(q_2)$. While aligning a question with a template, the matched words/phrases, which are text in questions aligned with the place-holder of templates, are marked at the same time,

as w_1 and w_2. The s_1 and s_2 are predicate paths of the templates. If we can measure the similarity d_r between two predicate paths, then the semantic distance $d_s(q_1, q_2) = d_r(s_1, s_2) + d_w(w_1, w_2)$.

A predicate path is a sequence of DBpedia predicates, hence the basic edit distance is used to compute the d_r. The Figure 5.3 shows an example of encoding two questions.

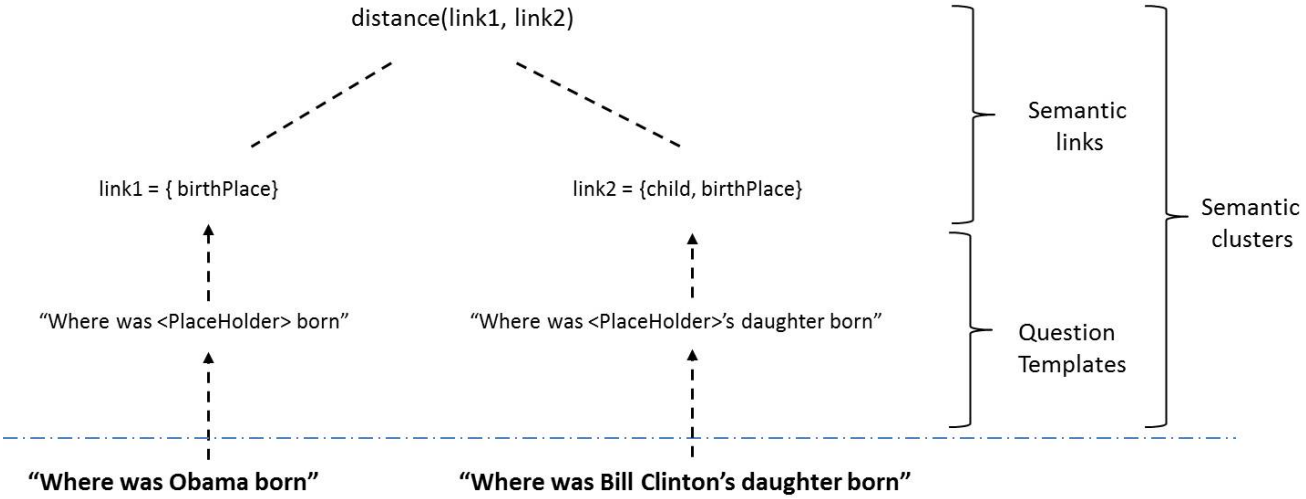

Figure 5.3: Examples of encoding words

In the next chapter, we apply the encoding methods on practical systems and conduct a series of experiments on them.

Chapter 6

Applications and Experiments

To test our theory, we implement two applications in different research areas, including question classification and question translation. They both rely on the semantic distance approximation and the semantic clusters dataset.

6.1 Question Classification

The question classification is our first attempt to apply the semantic distance in our previous experiments [16]. We briefly introduce our classification system and experiments here.

Given a noisy query, our QA system (namely RSVP) needs to classify it into one of the vertical domains (e.g., Weather, Map). Usually, a domain denotes a specific semantical class of questions. In practice, domains are defined by certain APIs or databases. For example, in a current voice input map application, the map domain here only includes questions that they are capable to answer, such as "*How to get to Toronto*" and "*Where is the nearest Starbucks*". The geographical question "*Which country is on the border of Canada*" is not included. Lacking clear definitions, domains are sometimes overlapped with each other. To solve such practical classification problems, our system should be able to categorize questions on complicated domain settings.

6.1.1 System Implementation

We train our classifier by simply providing question examples. To each domain V_k, the questions from question set Q that are labeled as positive examples are collected as Q_k^+. All the rest of the questions in Q form Q_k^-.

In the classifying process, given a query q, the semantic distance between q and each vertical domain V_k $(k = 1, 2, \ldots, n)$ is calculated by following steps:

1. Iterate through questions in Q_k^+ and Q_k^-, and find question q_k^+ and question q_k^- that has the smallest semantic distance from q.
2. If the distance $d_s(q, q_k^+) \leq d_s(q, q_k^-)$, then return $d_s(q, q_k^+)$ as $d_s(q, V_k)$; otherwise, return $+\infty$.

Finally, each domain V with $d_s(q, V) < +\infty$ are the true results of the classification. The Figure 6.1 shows an example of it.

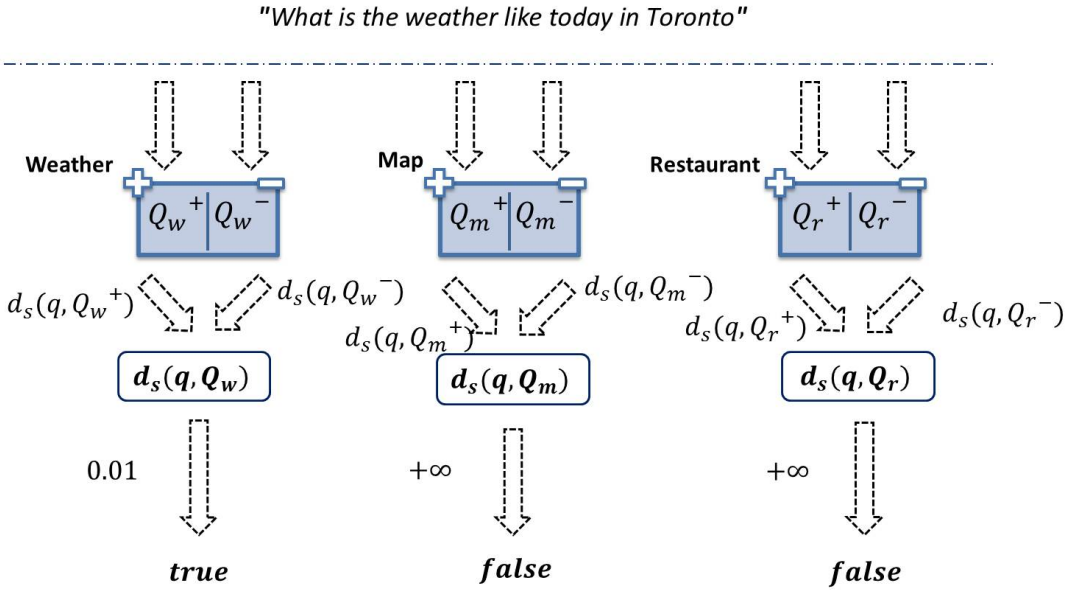

Figure 6.1: Question classification flow chat

6.1.2 Experiment Setup

The training set is extracted from the question set of 35 questions Q. Since they are all hand-written text, many of them contain typos and grammatical mistakes. These errors are kept still in training. To some extent, it is a good way to simulate the real QA scenarios, including queries from device typing, speech recognition, or language translation.

In order to compare with both academic and commercial systems, the domains that we test on should be widely covered by the question dataset Q, and they should be well-support by the target commercial systems. Finally, there popular domains have been chosen: *Weather*, *Maps*, and *Restaurants*. Three binary classifiers are developed for all three domains individually.

To build a training data set, we first use an initial domain-related keyword set to retrieve questions from Q. Among the question subset, we generate n-gram (unigram, bigram and trigram) dictionary sorted by their frequency. High frequency and related phrases are selected manually to expand the domain related keyword set. This work has been done recursively until there is no new questions retrieved. Then we deploy clustering on the candidates to narrow down the question subset further. It leaves a relatively small number of possible candidates on each domain, respectively. This reduces many obvious negative questions, hence avoiding unnecessary annotations for question labeling. The last step is the human annotation. It has been done by three linguistic experts, with agreements of Cohen's $\kappa = 0.95, 1.00, 0.93$ of the Weather, Maps, and Restaurants domains. Two annotators annotated the entire question sets separately. Moreover, the third annotator is involved when there are conflicts. Questions of each domain are selected separately. For each question to a domain, we simply label whether it belongs to the domain.

Table 6.1 lists the total number of selected questions, as well as the number of positive and negative samples for each domain. In each domain, 100 questions are randomly selected as the testing set, and the rests are treated as the training set.

Table 6.1: Experiment dataset statistics

	Weather	Maps	Restaurants
# Positive	1,509	1,275	644
# Negative	2,513	4,885	2,146
# Uncertain	420	156	286
Total # Questions	4,442	6,316	3,076

6.1.3 Empirical Results

Practical statistical models, such as naïve Bayesian model, logistic regression model, and SVM (support vector machine), are commonly applied to solve text classification problems and achieved great successes. Recent studies show that the SVM has the best performance on the text classification. Additionally, it is consistent with our previous experiments that a linear SVM classifier outperforms a naïve Bayesian model and a logistic regression model, all with same features that are fully developed from textual, syntactic and semantic information.

In experiments, we compare our classifier to a linear SVM on the same test setting. In addition, a tree-kernel SVM [13] is implemented. The tree kernel method measures similarity between two syntactic trees in terms of their sub-structures.

The results of comparison experiments are shown in Table 6.2. As revealed, the RSVP system is close to the several best results from tree-kernel SVM and even outperforms it in most measurements.

6.2 Question Translation Correction

Machine translation techniques are required by practical QA systems to transform cross-lingual queries into an identifiable language (e.g., English). Then they are able to have a unified query parser and a unified knowledge base for all cross-lingual queries. However,

Domain	Measurement	SVM-linear	SVM-tree	RSVP
Weather	Accuracy	0.880	0.920	**0.950**
	Precision	0.862	0.838	**0.912**
	Recall	0.758	**0.939**	**0.939**
	F_1-score	0.806	0.886	**0.925**
Maps	Accuracy	0.900	**0.950**	**0.950**
	Precision	0.800	**1.000**	0.885
	Recall	0.800	0.800	**0.920**
	F_1-score	0.800	0.889	**0.902**
Restaurants	Accuracy	0.850	0.910	**0.940**
	Precision	0.750	0.846	**0.862**
	Recall	0.667	0.815	**0.926**
	F_1-score	0.706	0.830	**0.893**

Table 6.2: Comparison of SVM and RSVP on the three domains

machine translation technologies are not fully practical, due to fundamental difficulties, i.e., language ambiguity and statistical model errors. The results from commercial translation systems(e.g., Google, Mircosoft) are usually disordered and even misleading. The machine generated errors are vital for a QA system, since a problematic translation result possibly leads to an irrelevant answer. Thus, given a translation result, we select morphologically close question templates from the semantic clusters, rewrite the question based on the templates, and select the most likely questions according to their semantic distance to the translation result.

6.2.1 System Implementation

We use the same approach in [42] to correct translation candidates, but a extended approximation algorithm of the semantic distance. The translation correction aims to correct translated questions that have errors according to a large question dataset Q, containing proper English questions. Given a question q^* output from a machine translation system,

we want to generate an accurate question q that has the minimal distance with both q^* and a question of Q. The result could be either q^* itself (when translation is correct) or a question of Q. In our RSVP QA system, the CQA questions are used as the question set Q. Instead of searching for a single question, we find the most likely question template t for semantic clusters, which is a group of questions. The implementation of our translation corrector includes several steps (the input query has been translated into English already):

1. Pre-process input query (in English):

   First, a NER tool is used to tag locations and dates in sentences. Secondly, the same DBpedia schema matching is applied on the question. Also, we generate templates of the question for every matched surface text.

   Finally, the templates are normalized by a stemming method based on POS tagging and WordNet. For example, verbs like "*is*" and "*were*" are normalized to "*be*", and plural nouns are normalized to single nouns.

   The normalized templates are used as search queries to match templates in the semantic clusters.

2. Retrieve relevant query templates:

   In Section 4.3, the question templates have been clustered and indexed for search. For all search queries, we retrieve syntactically similar templates. A search query are treated as a set of words and NER tags.

   For example, a search query "<PlaceHolder> can live long" finds several template candidates: 1) "*How long can <PlaceHolder> live*", 2) "*Can <PlaceHolder> live for long*" 3) "*How long can a <PlaceHolder> live*".

   The search results are ranked by the sum of term tf-idf scores. We bias the score by the template credits in Section 4.3.

3. Rewrite queries on templates:

   For each template found, we generate a new query by simply replacing its place-holder by the word/phrase aligned in original query.

4. Rank queries using semantic distance:

   We applied the encoding algorithm to compute the semantic distance between the original query and every candidate query. The query with the lowest semantic distance is select as the final translation result.

Figure 6.2 shows a demonstration process of our translation system.

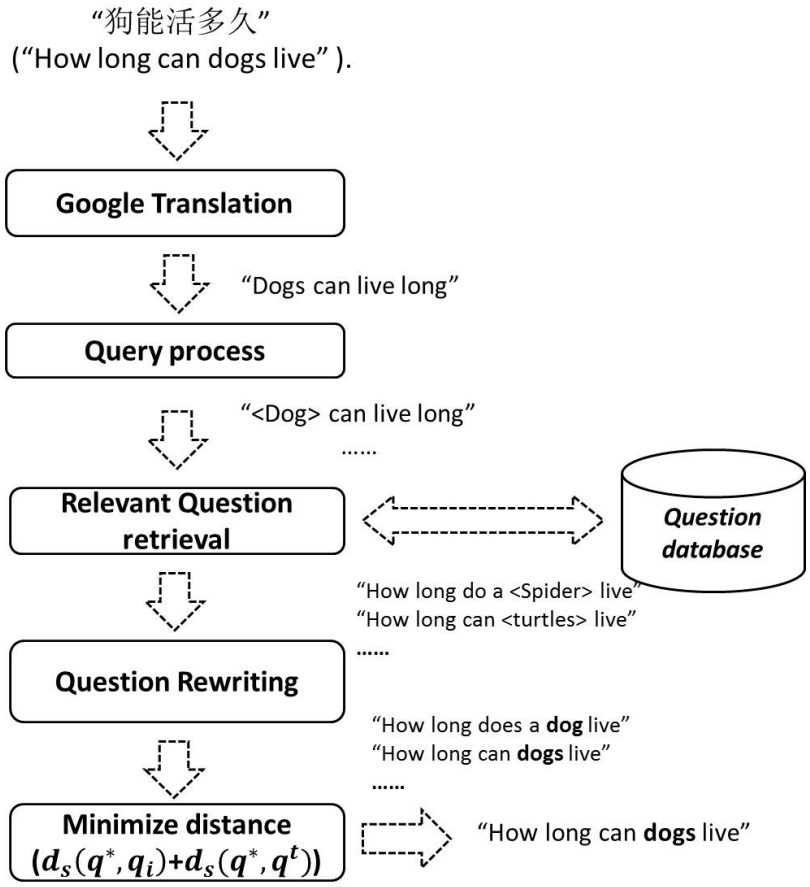

Figure 6.2: RSVP Translation flow chat

6.2.2 Experiment Setup

Our experiments have been only conducted on the translation from Chinese to English, although the methodology applies on other languages as well. The dataset, including 300 English questions, is randomly selected from an independent Microsoft QA set [1]. We recruit translation experts to translate the questions back to Chinese. The translated question set is used as the testing set, and the original English question set is the benchmark of our experiments.

Initially, each question in testing set is translated into English by the Google translation API. Our corrector takes the translation results as input queries and returns corrected queries. In the comparison experiment, we compare these queries with the original translation results.

6.2.3 Empirical Results

The initial experimental results show that Google translation does not have very good accuracy on questions. However, we are able to correct most of them. Table 6.3 shows several translation examples.

Many google translation results are returned in a messed order, and our system are able to correct them. The comprehensive comparison experiments will be conducted on Google translation, Microsoft Bing translation, Baidu tanslation, and Youdao translation. And also we will evaluate our correction on several different practical QA systems. For each QA system, only the correctly understood questions could be counted as the correct translation results.

[1] http://research.microsoft.com/en-us/downloads/88c0021c-328a-4148-a158-a42d7331c6cf

Table 6.3: Comparison of Google Translation and RSVP

Chinese Question	鱼会睡觉吗
Google Translation	Fish do sleep?
RSVP Translation	Does fish sleep?
Chinese Question	庞贝还有什么信息吗?
Google Translation	There is no information about Pompeii?
RSVP Translation	Are there any other information about?
Chinese Question	塑料可以降解吗?
Google Translation	Biodegradable plastic is something?
RSVP Translation	Are plastic bags biodegradable?
Chinese Question	马有多少块骨头?
Google Translation	How many horse hones?
RSVP Translation	How many bones are in a horse?
Chinese Question	在月球上能点燃火柴吗?
Google Translation	Lighting a match on the moon can do?
RSVP Translation	Can you light a match on the moon?

Chapter 7

Conclusion and Future Works

In this thesis, we formulate QA-related tasks by measuring semantics of natural language queries. By computing the semantic distance between two queries, we are able to solve the problems of question classification and translation correction. However the semantic distance is uncomputable and undecidable. Hence we approximate it on a dataset of semantic clusters using information distance theory. Our main contribution is constructing such semantic dataset from a great number of question-answer pairs automatically. In each semantic cluster, various question templates have one semantic relationship, which is a predicate path of the DBpedia. Based on the semantic distance, we implement a question classifier and a translation corrector. Moreover, we demonstrate that they are very useful for our practical QA system.

The question classifier is compared with state-of-the-art statistical studies, including naïve Bayesian, logistic regression, and SVM. From Table **??**, our results are close to the some of the best results given by a tree kernel SVM, but have significant improvements on most other measurements of all baseline approaches. We have tried our best to implement these learning models by developing explicit linguistic features, including textual, syntactic, and semantic information. Hence the comparison experiments are conducted properly.

Actually the SVM classifiers have a pretty high accuracy on average, but they sometimes fail to estimate the probability for confusing cases. For example, the question "*What is the temperature of the Sun?*" is wrongly classified into Weather domain. However, given a

question, we explicitly examine the distance to each example in training set. For positive examples of Weather domain, the minimal distance is given by the example "*What is the temperature of the Toronto?*", and this results in a relatively large distance because "Sun" is very far from "Toronto" in WordNet. For negative examples, the minimal distance might be given by the example "*How hot is the Moon?*", and this results in a very small distance because we know the two templates are in a single semantic cluster and "Sun" is close to "Moon" in WordNet.

The time cost is a concern of our approach, since the processing time is in direct proportion to the number of training examples. Currently a thousand semantic distance calculations take about a hundred milliseconds, so our system could be very slow if a domain has too many examples. In our future work, we will attempt to select the minimal training set that only contains representative examples, in order to decrease the calculation time.

The question corrector is implemented based on the template set of the semantic clusters. It is another usage of the semantic dataset. Since our question corrector is built on the top of the Google translation service, we naturally compare the translation results with Google for the Chinese to English translation. Some examples reveal that statistical machine translators have high accuracy on word or phrase translation, but fail to assemble them in a proper order for sentence translation, especially for questions. We take advantage of a database of proper English question templates to reassemble translated words according to a semantically similar template. It is a language-independent method that works for many languages, since we do not require any features from the original questions in the original language.

Initial experimental results of the translation show that we are able to improve the Google translation results. The possible explanation is that the generative models Google uses give a relatively poor estimation to the probability distribution for low frequent samples, i.e., questions. On the other hand, our correction method is a question-template based approach. Using the source of 35 million human-understandable questions, it is easier for us to establish the probability distribution of question templates and retrieve semantically similar templates.

Our dataset contains very rich question templates that cover most questions of the translation testing set, but there are still three of them lack strictly similar templates:

"*Why is some sand white, some brown, and some black?*"

"*Do flying squirrels fly or do they just glide?*"

"*And was there ever a movement to abolish the electoral college?*"

Obviously, our translation corrector fails when no question templates are found. It is because we only take into account the questions with one main concept. Involving templates with multiple slots definitely expands the templates, but it is extremely hard to detect their semantic intentions. This would be another future investigation.

The dataset of semantic clusters is the basis of our studies, especially for computing the semantic distance. We believe it is useful for other natural language query understanding tasks too. However, the dataset construction depends on many different NLP tools. When a named entity fails to be identified, it will be treated as separate words, thus increasing the semantic distance from the true answer. Although DBpedia provides a rich named entities collection, it is slow on updating and lack of recently emerging concepts from twitter or other social media. In our future work, we will attempt to use Google Freebase, which is a much bigger (over 2 billion facts) and up-to-date knowledge base, to expand our NER coverage.

Finally, the semantic distance measurement has additional value for QA systems. Although traditional statistical methods yeild decent results, they hardly provide any clue about how to answer the questions. In our system, given a query, by finding a semantically close (with a small semantic distance) question that we already know how to answer, we are one step closer towards answering the query.

APPENDICES

Appendix A

Sample templates of semantic clusters

Question template samples with the predicate path

"militaryRank" - *http://dbpedia.org/ontology/militaryRank*:

```
what was <PlaceHolder> 's position in the tennessee militia
what millitary service did <PlaceHolder> serve in
what was <PlaceHolder> role in world war 1
what did john f <PlaceHolder> do in the navy
what was <PlaceHolder> 's rank in the revolutionary war
what did <PlaceHolder> work as
what could be considered <PlaceHolder> 's defining achievement as president
what countries did <PlaceHolder> liberate
what did <PlaceHolder> do
what war other than the civil awr did <PlaceHolder> fight
what role did <PlaceHolder> play in the american revolution
what was <PlaceHolder> 's rank in the military
what war did <PlaceHolder> serve in
what did <PlaceHolder> serve as in the civil war
what was <PlaceHolder> military rank
what carrers did <PlaceHolder> have before the cival war
```

```
what was <PlaceHolder> 's role in cuba
what did <PlaceHolder> do to become famous
what did <PlaceHolder> do in his wars
which military did <PlaceHolder> serve in
what are the jobs <PlaceHolder> had in thwe past
what did <PlaceHolder> do before he became a congressman
what was <PlaceHolder> 's military service record
what military rank is <PlaceHolder> in
where did <PlaceHolder> used to work
what jobs did <PlaceHolder> do
what was <PlaceHolder> 's rank when he retired from the navy
```

Question template samples with the predicate path "spouse" - *http://dbpedia.org/ontology/spouse*:

```
who is <PlaceHolder> married to
who was <PlaceHolder> married to
who has <PlaceHolder> dated
who is <PlaceHolder> 's wife
who was <PlaceHolder> 's wife
what was <PlaceHolder> 's wife 's name
who is the wife of <PlaceHolder>
who is <PlaceHolder> 's husband
who was <PlaceHolder> wife
who was <PlaceHolder> 's husband
what is <PlaceHolder> 's wife name
who did <PlaceHolder> marry
what is <PlaceHolder> 's wife 's name
who is <PlaceHolder> dating
who is <PlaceHolder> wife
who was the wife of <PlaceHolder>
```

who was <PlaceHolder> 's first wife
who married <PlaceHolder>
who was married to <PlaceHolder>
who was <PlaceHolder> 's first lady
what is <PlaceHolder> 's husband 's name
what is <PlaceHolder> wife name
what was <PlaceHolder> wife name
who was <PlaceHolder> first wife
who is <PlaceHolder> husband
who was <PlaceHolder> husband
who was <PlaceHolder> 's second wife
who was <PlaceHolder> first lady
what was <PlaceHolder> wife 's name
what is <PlaceHolder> wifes name
who is <PlaceHolder> spouse
what was <PlaceHolder> 's wifes name
what is <PlaceHolder> 's husbands name
what was <PlaceHolder> 's wife name
who was <PlaceHolder> 's first husband
who were <PlaceHolder> 's wives
who is married to <PlaceHolder>
where did <PlaceHolder> get married
who is <PlaceHolder> 's boyfriend
who is <PlaceHolder> engaged to
who is <PlaceHolder> 's first wife
who is the first lady of <PlaceHolder>
who did <PlaceHolder> married
what is <PlaceHolder> 's wife called
who was <PlaceHolder> famous wife
who was the famous lady that was married to <PlaceHolder>
who got marrey to <PlaceHolder>

```
who was <PlaceHolder> married to in 2001
what was <PlaceHolder> 's husbands name
who was <PlaceHolder> 1st lady for
who was <PlaceHolder> spouse
what was <PlaceHolder> 's first wife 's name
who did <PlaceHolder> get married to
who where <PlaceHolder> 's wifes
what was president <PlaceHolder> 's wife like
what was <PlaceHolder> 's wife called
what was name of <PlaceHolder> wife
who is <PlaceHolder> a lover of
what <PlaceHolder> wife 's name
```

Question template samples with the predicate path "partner" - *http://dbpedia.org/ontology/partner*:

```
who is <PlaceHolder> dating
who has <PlaceHolder> dated
who does <PlaceHolder> go out with
who is <PlaceHolder> 's girlfriend
who is <PlaceHolder> dating now
who doe <PlaceHolder> like
who is <PlaceHolder> girlfriend
who is <PlaceHolder> currently dating
who is <PlaceHolder> 's wife
who is <PlaceHolder> married to
who was <PlaceHolder> married to
who is <PlaceHolder> boyfriend
who is <PlaceHolder> in a relationship with
who has gone out with <PlaceHolder>
who is <PlaceHolder> engaged to
who is dating <PlaceHolder>
```

who is going out with <PlaceHolder>
who got <PlaceHolder> pregnant
who married <PlaceHolder>
who is current husband of <PlaceHolder>
who was engaged with <PlaceHolder>
who is <PlaceHolder> wife
who does <PlaceHolder> love
whoo is <PlaceHolder> dating
who is helena <PlaceHolder> married to
where are <PlaceHolder> children from
who is <PlaceHolder> 's partner
who is <PlaceHolder> husband
who <PlaceHolder> dating
who is <PlaceHolder> real life girlfriend
who is real girlfriend of <PlaceHolder>
who was engaged to <PlaceHolder>
who does <PlaceHolder> later marry in the great ziegfeld
who was <PlaceHolder> 's first husband
where do <PlaceHolder> 's children come from

References

[1] Dave Beckett and Art Barstow. N-triples. *W3C RDF Core WG Internal Working Draft*, 2001.

[2] Charles H Bennett, Péter Gács, Ming Li, Paul MB Vitányi, and Wojciech H Zurek. Information distance. *IEEE Transactions on Information Theory*, 44(4):1407–1423, 1998.

[3] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. Dbpedia-a crystallization point for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154–165, 2009.

[4] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of Machine Learning Research*, 3:993–1022, 2003.

[5] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of data*, pages 1247–1250, Vancouver, Canada, 2008. ACM.

[6] Sergey Brin. Extracting patterns and relations from the world wide web. In *The World Wide Web and Databases*, pages 172–183. Springer, 1999.

[7] Fan Bu, Xingwei Zhu, Yu Hao, and Xiaoyan Zhu. Function-based question classification for general QA. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 1119–1128, MIT Stata Center, Massachusetts, USA, 2010.

[8] Fan Bu, Xiao-Yan Zhu, and Ming Li. A new multiword expression metric and its applications. *Journal of Computer Science and Technology*, 26(1):3–13, 2011.

[9] Alexander Budanitsky and Graeme Hirst. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and Other Lexical Resources*, volume 2, 2001.

[10] Razvan C Bunescu and Raymond Mooney. Learning to extract relations from the web using minimal supervision. In *Proceedings of the 45th Annual Meeting on Association for Computational Linguistics*, volume 45, page 576, Prague, Czech Republic, 2007.

[11] Xin Chen, Brent Francia, Ming Li, Brian Mckinnon, and Amit Seker. Shared information and program plagiarism detection. *IEEE Transactions on Information Theory*, 50(7):1545–1551, 2004.

[12] Rudi Cilibrasi and Paul MB Vitányi. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545, 2005.

[13] Aron Culotta and Jeffrey Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423, Barcelona, Spain, 2004. Association for Computational Linguistics.

[14] Wang Di. Learning automatic question answering from community data. pages 28–36, 2012.

[15] S Dumais, G Furnas, T Landauer, S Deerwester, S Deerwester, et al. Latent semantic indexing. In *Proceedings of the Text Retrieval Conference*, Gaithersburg, Maryland, 1995.

[16] Guangyu Feng, Kun Xiong, and et al. Understanding a question by approximating its semantics. Technical report, Waterloo, Ontario, Canada.

[17] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1994.

[18] Ulf Hermjakob. Parsing and question classification for question answering. In *Proc. workshop on Open-Domain Question Answering*, pages 1–6, Stroudsburg, PA, USA, 2001.

[19] Graeme Hirst and David St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet: An electronic lexical database*, pages 305–332, 1998.

[20] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. Yago2: a spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61, 2013.

[21] Zhiheng Huang, Marcus Thint, and Zengchang Qin. Question classification using head words and their hypernyms. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 927–936, Waikiki, Honolulu, Hawaii, 2008.

[22] Jay J Jiang and David W Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *arXiv preprint cmp-lg/9709008*, 1997.

[23] Nanda Kambhatla. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 22, Barcelona, Spain, 2004. Association for Computational Linguistics.

[24] Donald Knuth. *The TEXbook*. Addison-Wesley, Reading, Massachusetts, 1986.

[25] Leslie Lamport. *LATEX — A Document Preparation System*. Addison-Wesley, Reading, Massachusetts, second edition, 1994.

[26] Claudia Leacock and Martin Chodorow. Combining local context and WordNet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2): 265–283, 1998.

[27] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören

Auer, and Christian Bizer. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 2014.

[28] Ming Li and Paul Vitányi. *An introduction to Kolmogorov complexity and its applications*. Springer, 1997.

[29] Wei Li. Question classification using language modeling. *Center of Intelligent Information Retrieval (CIIR). Technical Report*, 2002.

[30] Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, Taipei, Taiwan, 2002.

[31] Dekang Lin. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, pages 296–304, Madison, Wisconsin, USA, 1998.

[32] Dekang Lin and Patrick Pantel. Dirt@ sbt@ discovery of inference rules from text. In *Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 323–328, San Francisco, CA, USA, 2001. ACM.

[33] Frank Manola, Eric Miller, Brian McBride, et al. Rdf primer. *W3C recommendation*, 10:1–107, 2004.

[34] George A Miller. WordNet: a lexical database for English. *Comm. of the ACM*, 38 (11):39–41, 1995.

[35] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Baltimore, MD, USA, 2009. Association for Computational Linguistics.

[36] Richard Montague. English as a formal language. *Linguaggi nella societae nella tecnica*, pages 189–224, 1970.

[37] Alessandro Moschitti, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. Exploiting syntactic and shallow semantic kernels for question answer classification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 776–783, Prague, Czech Republic, 2007.

[38] Microsoft Research. Microsoft Research Question-Answering Corpus, 2008. http://research. microsoft.com/en-us/downloads/88c0021c-328a-4148-a158-a42d7331c6cf/.

[39] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*, 1995.

[40] Sebastian Riedel, Limin Yao, and Andrew McCallum. Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer, 2010.

[41] Thamar Solorio, Manuel Pérez-Coutiño, Manuel Montes-y Gémez, Luis Villasenor-Pineda, and Aurelio López-López. A language independent method for question classification. In *Proceedings of the 20th International Conference on Computational Linguistics*, page 1374, University of Geneva, Switzerland, 2004.

[42] Yang Tang, Di Wang, Jing Bai, Xiaoyan Zhu, and Ming Li. Information distance between what i said and what it heard. *Communications of the ACM*, 56(7):70–77, 2013.

[43] G. Tur and R. De Mori. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*. Wiley, 2011. ISBN 9781119993940. URL http://books.google.ca/books?id=RDLyT2FythgC.

[44] Wei Wu, Hang Li, and Jun Xu. Learning query and document similarities from click-through bipartite graph with metadata. In *Proceedings of the International ACM Conference on Web Search and Data Mining*, pages 687–696, Rome, Italy, 2013.

[45] Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. Structured relation discovery using generative models. In *Proceedings of the Conference on Empirical*

Methods in Natural Language Processing, pages 1456–1466, John McIntyre Conference Centre, Edinburgh, UK, 2011. Association for Computational Linguistics.

[46] Dell Zhang and Wee Sun Lee. Question classification using support vector machines. In *Proceedings of the 26th Annual International ACM SIGIR Conference*, pages 26–32, Toronto, Canada, 2003.

[47] Xian Zhang, Yu Hao, Xiaoyan Zhu, Ming Li, and David R Cheriton. Information distance from a question to an answer. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 874–883, San Jose, CA, USA, 2007. ACM.