Question Classification by Approximating Semantics

Guangyu Feng, Kun Xiong, Yang Tang, Anqi Cui
School of Computer Science, University of Waterloo
{gfeng, caq}@uwaterloo.ca, {xiongkun04, tangyang9}@gmail.com

Jing Bai
Microsoft
jbai@microsoft.com

Hang Li
Noah's Ark Lab
Huawei Technologies
hangli.hl@huawei.com

Qiang Yang
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
qyang@cse.ust.hk

Ming Li
School of Computer Science
University of Waterloo
mli@uwaterloo.ca

ABSTRACT

A central task of computational linguistics is to decide if two pieces of texts have similar meanings. Ideally, this depends on an intuitive notion of *semantic distance*. While this semantic distance is most likely undefinable and uncomputable, in practice it is approximated heuristically, consciously or unconsciously. In this paper, we present a theory, and its implementation, to approximate the elusive semantic distance by the well-defined *information distance*. It is mathematically proven that any computable approximation of the intuitive concept of semantic distance is "covered" by our theory. We have implemented our theory to question answering (QA) and performed experiments based on data extracted from over 35 million question-answer pairs. Experiments demonstrate that our initial implementation of the theory produces convincingly fewer errors in classification compared to other academic models and commercial systems.

Categories and Subject Descriptors

F.1.1 [**Models of Computation**]: Foundations of Data Mining; H.3.3 [**Information Search and Retrieval**]: Query Processing

Keywords

Question answering, text classification, information distance, semantic distance

1. INTRODUCTION

In a Question Answering (QA) system, the bottleneck is usually not the lack of knowledge, but how to correctly interpret the queries, as the Internet, social QA communities, Wikipedia, and many other databases usually contain the answers we are looking for.

WWW 2015 Companion, May 18–22, 2015, Florence, Italy.
ACM 978-1-4503-3473-0/15/05.
http://dx.doi.org/10.1145/2740908.2745403.

For example, if we already know the answer to "What is the population of Canada?", then we should also know the answer to "How many people live in Canada?". The answer is the same and the *semantic distance* between these two queries should be approximately zero.

We know what semantics is and we all have an intuition of how to judge the semantic distance between two queries. However, we do not know how to formally define it, letting alone how to compute it. Just like 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 [25], have studied theories of natural language semantics and its relation with syntax. While the 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.

Over the years, many researchers in computational linguistics have tried, consciously or unconsciously, to approximate such a semantic distance. We first give a brief and non-inclusive review on the approaches in the known literatures, and then suggest a few more extensions of computing or approximating the informal concept of *semantic distance* following the conventional paths. Following the notations in [4], 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.

1. At the word level, [11] 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, [16] 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, [28] 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, [14] 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, [19]'s semantic similarity measure is

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

6. Matching in latent space approach by [33] matches documents and images via the semantic space, and goes beyond the above word-level WordNet semantics approaches.
7. At the sentence level, obviously identical sentences or queries mean the same thing. After substituting some words with their synonyms, or with words that are close by word-level similarity measures, they should still carry the same meaning.
8. Generalizing the above, if two queries are similar, say with small syntactic edit distance, then they have similar semantics. However, this does not work for things like "What is the weather like today in Toronto?" and "In Toronto what is the weather like today?"
9. Certainly one can keep on improving the above by allowing permutations of addresses and time blocks, but then we still have trouble with "What is the population of Canada" and "How many people live in Canada". They have large edit or parsing structure distance, but the same semantics.
10. Computing such semantic approximations is also language dependent. For example, the three Chinese sentences in Table 1 all ask about the price of apples. The first two sentences are easily comprehensible through literal translation, however, one can hardly expect a non-Chinese speaker to link "how to sell" with the actual meaning of "how much". Thus we have more problems: how to make a language-independent definition or approximation of semantic distance?

Chinese sentence	Literal translation
苹果什么价钱?	Apple what price?
苹果多少钱一公斤?	Apple how much money one kg?
苹果怎么卖?	Apple how to sell?

Table 1: Three ways of asking the price of apples in Chinese

New applications of natural user interface have created new challenges and revitalized some of the old problems. Current mobile personal assistant systems use practical natural language processing methods, such as template or keyword matches, possibly with added models or probabilities to bias the meaning of a template or keywords, thus, mistakes abound. If we ask Siri, "What do fish eat?", Siri answers with a list of seafood restaurants nearby. When asked "Where can I find hamster food?", Siri also answers with a list of restaurants. The query "What is the temperature on the surface of the Sun?" and "How hot is the Sun's surface?" have the same meaning, but Siri can only answer one of them. One way of explaining this problem is that Siri fails to classify the questions into the correct *vertical domains*. In general, a QA system, or a mobile personal assistant system, naturally consists of a collection of application domains. These domains may include, for example, music, maps, weather, calendars, time, food, hotel, email, news, telephone functions and a general search domain. Once restrained to a certain vertical domain, queries can be processed more efficiently and effectively. However, query classification is no easy job. Questions belonging to different domains may not differ much literally, for example "What is the temperature on the sun?" and "What is the temperature in Tucson?". While the questions in the examples above are expressed in proper English, things become more challenging when the query is translated from another language, for cross language search, or when a non-native speaker *speaks* to a device. Even native speakers suffer from noisy environments. In these cases, the users' questions are often slightly distorted. As a result, current systems may fail to give answers to these questions. For example, the QA website *evi.com* (formerly True Knowledge), accessed on Nov. 10, 2014, at the time of writing this paper, could answer "Who is the mayor of Toronto?", "Who is mayor of Toronto?" (missing "the") and "Who is Toronto's mayor?" correctly, but "Who is the Toronto's mayor?", with an extra "the", still a comprehensible expression to humans, could not be answered.

Obviously, all these problems would be solved if we knew how to compute the "semantic distance". We will show that even without a formal definition of the elusive "semantic distance", we can still develop a theory to approximate it and mathematically prove that our approximation is the "best" there can be. We want the theory to be natural, universal, and free from *ad hoc* feature selections for each new application domain. This unifying theory naturally leads to a practical system which outperforms existing methods for our QA tasks: (1) classifying a query into a proper domain, and (2) finding a question, closest to the query, with an answer. Our system is then tested using a data set with over 35 million QA pairs.

The paper is organized as follows: Section 2 introduces some related work in the fields of QA question classification and topic classification. Section 3 formalizes our problem. Our theory is introduced in Section 4. In Section 5, we show how to implement our theory. Section 6 describes classification framework. Experimental results are shown in Section 7. Finally, conclusions and future work are addressed in Section 8.

2. RELATED WORK

Question classification in QA systems has long been studied. Traditionally, questions are categorized based on their intents: The popular taxonomy [18] includes Abbreviation, Description, Entity, Human, Location, and Numeric Values. Subsequent work introduces the categories from the prospective of user-intent analysis, including Navigational, Informational, Transactional [20], and Social questions [5], or combining intents with the contents, such as Solution, Reason, Fact, etc. introduced in [2]. Our problem differs from these in the aspect of categorization, that we classify

the questions into more concrete *vertical domains*, such as Weather, Restaurants, and Maps, hence better organizing the knowledge base and providing more accurate answers.

This vertical taxonomy leads our task into a topic classification problem, which is a basic task in text classification. In this field, contents of texts are usually fully exploited, such as their lexical features (e.g. n-grams), syntactic features (e.g. parse trees [10, 27]) and semantic (e.g. WordNet-based) features. Moreover, the contents are usually relevant to users' intents [13]. Therefore, based on these textual features, many models have been developed and have been applied in question classification. For example, specific lexical features are more important to determine the topic and these methods are independent with languages [12, 29]. Syntactic and semantic features combining with machine learning models (e.g. support vector machines) are competent to classification [15, 18, 34]. However, content-based analysis requires well-formed texts with sufficient contexts. For shorter queries, it is difficult to train reliable probabilistic topic models (such as latent semantic indexing / analysis, latent Dirichlet allocation, etc.). What is more, informal texts or errors greatly decrease the accuracy of semantic analysis and bring challenges to topic identification [31], especially in our task, where the questions may be from speech recognition softwares or translated from another language, hence containing errors.

Question-answering on speech transcripts (QAST) has also been studied recently, especially in the Cross Language Evaluation Forum (CLEF) campaigns [32]. These studies usually recognize the named entities (NEs) as the key components of question types [8]. Phonetic codifications are introduced to enrich the possibility of the entities. Then the entities, together with the modifiers are queried within all the answers. These traditional approaches are still limited within the variations of linguistic methods. For more complicated situations, we need a robust understanding of these extemporaneous or imperfect recognized texts. Our solution is to find a question or cluster that is semantically very "close" to the input query, defined by information distance.

The theory of information distance have been widely applied [17], including QA tasks such as named entity extraction by recognizing multiword expressions (MWEs) [3], measuring the "distance" between a question and an answer [35], or measuring the distance from a speech transcript to its real intent (well-formed text) [30]. These preliminary studies illustrate a vast possibility of using information distance to solve natural language processing problems, but they stop short not giving a general theory of approximating semantics and how to do a general implementation. In this paper does the above, and as application we measure the question-question semantic distances rather than question-answer relevance.

3. 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. One question could be asked in hundreds of grammatically correct ways. Furthermore, when a sentence is grammatically wrong, or contains irrelevant words, it is usually still comprehensible. 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 domain with a representative question that has the smallest *distance* from the query. For instance, for the query "What is the weather like in Seattle?", if we have, in our training data, the question "How is the weather in Toronto?" as a positive example, and the question "How is the weather on Mars?" as a negative example, then what we need is a mechanism that determines that the query has a smaller *distance* to the positive than to the negative example.

For the purpose of classification, a number of approaches may be immediately considered, such as support vector machines (SVM), neural networks, decision trees, Bayesian networks, Markov models, etc. Besides the accuracy issues, these methods are often sensitive to noise, thus cannot solve Problem 1. Furthermore, when training for a new domain, typically, one has to choose features *ad hoc*, often resulting in a re-training process, such as changing the kernel or normalizing the probabilities. If we have a proper semantic distance, adding a new domain, even by a user, is just adding data to the database. There are indeed "distances" that try to approximate semantic distance, including all those listed in Section 1, cosine distance (derived from cosine similarity) and Lexical Level Matching (LLM) distance [9]. However, all these distances have problems: The distances listed in Section 1 are word level distances; LLM and cosine are similarity measures, not distances; LLM is not even symmetric; LLM and cosine distance do not respect sentence structures

(hence too aggressive as encoders) hence bound to be defective, as shown in the experiments.

Thus, 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 most likely undefinable and uncomputable, is it possible to find a well-defined distance metric that would minorize (be no more than) *all* other 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 $Q_i \subseteq Q$, and a fixed number of application programming interfaces (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:

$$d_s(q_1, q_2).$$

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

$$d_s(q, V_i) = \min_{q' \in Q_i} d_s(q, q').$$

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

4. APPROXIMATING SEMANTICS

Our intuitive concept of semantic distance cannot be well defined and cannot be precisely computed. In Section 1, 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 even 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* [1]. Although it is still not computable, it directly suggests many natural ways of approximation, including those given in Sections 1 and 5.

We first follow [17] 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 [17] 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, in [1], the authors 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 [1] 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:

$$E(x, y) = \min\{|p| : U(x, p) = y,\ U(y, p) = x\} \qquad (1)$$

Clearly, $E(x, y) \leq K(x|y) + K(y|x)$. In [1] 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:

$$d(x, y) = \max\{K(x|y), K(y|x)\}. \qquad (2)$$

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 [14], [19], [33] 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 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 being un-computable. Our definition directly suggests a natural approximation by compression.

Information distance and its normalized versions have been applied in bioinformatics as well as plagiarism detection [6], clustering [7] and many other applications [17]. In the field of text processing, topics include question and answering systems [35], multiword expression linguistic analysis [3], web page authorship, topic and domain identification, Internet knowledge discovery, multi-document summarization etc.

5. ENCODING ALGORITHMS

In this section, we describe our approximation of information distance in the domain of QA, under the guidance of our general theory. While part of our system, especially

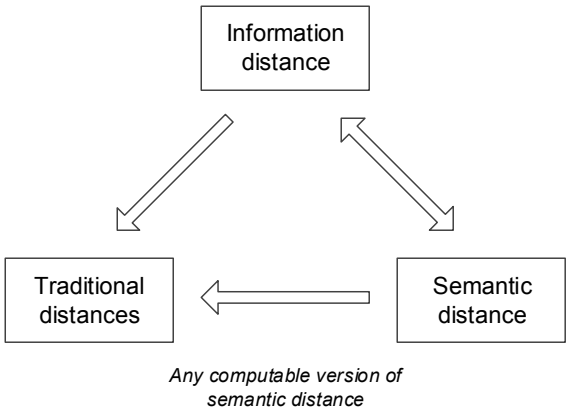

Figure 1: Relationships between distances

at the word level, is based on traditional WordNet-based metrics, the whole system is subtly different and new. On one hand, it avoids the aggressiveness of the unstructured bag-of-words model adopted by methods such as LLM and cosine distance, so that "What do fish eat" and "What eats fish" will not be considered to have the same meaning; on the other hand, it does not rely on structural parsing, which often fails especially on imperfect input sentences. Moreover, the semantic clustering algorithm allows us to discover semantic resemblance regardless of syntactical features.

Our approximation scheme is a natural derivation of the information distance theory, which considered the strings on the semantic level. Given a piece of language, x, which can be a word, a phrase or a sentence, let $\kappa(x)$ be the number of bits required to encode the meaning of x in the semantic space. $\kappa(y|x)$ denotes the number of bits required to encode the semantic meaning of y given input x. Following the definition of information distance, our approximation of semantic distance defines $d_s(x, y) = \max\{\kappa(x|y), \kappa(y|x)\}$.

At word level, we use a WordNet-based approach for the semantic encoding; at sentence level, we use a syntactic encoder as well as a new semantic clustering method. The advantages of our general theory include:

1. It unifies all approaches in Section 1 under one roof, using one measure (encoding bits), so that any measure can be used as long as it is shorter.
2. It avoids pitfalls of some approaches, such as asymmetry (L) of approaches 1 and 2 in Section 1. At sentence level, this problem becomes more prominent. For example, given sentences x and y, should we compress x using y or compress y using x? These two quantities may be different.
3. The new theory is conveniently extendable, allowing other measures of similarity or other databases to be added in, consistently. For example, it has allowed us to introduce a new semantic encoding scheme which incorporates a database of 35 million QA pairs and encodes questions via similar answers.

5.1 Encoding Words

Given a word w, $\kappa(w)$ 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 (or a distributional variation) of the complete vocabulary. Say there is a total of N words in the English vocabulary, then $\log N$ (or $\log P_i$ for i-th word) bits are required to generate any single one of them. This is our baseline of approximating $\kappa(w)$, i.e. the actual $\kappa(w)$ should be no more than this value.

The first useful knowledge for determining $\kappa(w)$ is part-of-speech (POS) tags. 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 $\kappa(w)$. On the other hand, nouns, verbs and adjectives carries relatively more information. Although in many cases it is true that "small words" can completely change the meaning of a sentence, in the context of question answering, such approximations usually suffice.

In order to further assign reasonable values of $\kappa(w)$ and $d_s(w_1, w_2)$, we need to have knowledge of the semantic relations between them. Here we use WordNet [24], because it not only groups synonyms together, but also contains hierarchical semantic relations between words. For example, the database containing all nouns has the following useful properties:

- 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 $\kappa(w)$ to be the length of this path, i.e. the number of steps of specialization required to produce a specific entity.
- Words that denote the same concept are within the same node (synset), for example, "sofa" and "couch". Hence, if w_1 and w_2 are within the same synset, then $\kappa(w_1|w_2) \approx 0$;
- The parent-child relation in its tree structure corresponds to hypernym-hyponym relations between entities, for example, the synset {seating} is the parent node of the synset {sofa, couch}. Therefore, if w_1 is a hypernym of w_2, then $\kappa(w_2|w_1) = 1$, meaning given w_1, we need one step of specialization to produce the semantic meaning of w_2;

Additionally, we also use the similarity and hypernym relations in WordNet's verb and adjective databases.

Therefore, given two words w_1 and w_2, we first find the synsets containing them, $synset(w_1)$ and $synset(w_2)$, and then calculate $d_s(w_1, w_2)$:

- The POS tag of w_1 falls into the "irrelevant" category described above. Then $d_s(w_1, w_2) = \kappa(w_2)$. The same rule applies when w_2 is an "irrelevant" word.
- $synset(w_1)$ and $synset(w_2)$ are the same synset. Then $d_s(w_1, w_2) = 0$;
- $synset(w_1)$ and $synset(w_2)$ both exist in the WordNet tree of entities. Then, $d_s(w_1, w_2)$ is computed by finding the least common ancestor (LCA) of $synset(w_1)$ and $synset(w_2)$, and then taking the greater of the number of steps from LCA to $synset(w_1)$, and from LCA to $synset(w_2)$, in accordance with the definition $d_s(x, y) = \max\{\kappa(x|y), \kappa(y|x)\}$;
- $synset(w_1)$ and $synset(w_2)$ do not exist in the same tree. Then $d_s(w_1, w_2) = \max\{\kappa(w_1), \kappa(w_2)\}$.

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

5.2 Encoding Sentences

Given query sentences q_1 and q_2, to compute $d_s(q_1, q_2)$, three scenarios are considered. (a) No semantic relationships exist, hence converting one sentence to the other requires word-by-word substitutions, resulting in a relatively large value; (b) When the two sentences are syntactically similar, converting one sentence to the other can be done with only small local modifications, thus the encoding of the whole sentence is the encoding of these local modifications; (c) Using our 35 million QA database to encode questions via their answers. The shortest of the three is taken as the final encoding.

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

In case (b), the source of similarity comes from alignments. An alignment of the two sentences is performed using the standard dynamic programming method with respect to the semantic distances between words and phrases. Thus for sentences within small edit distances, for example, sentences differ only by one insertion / deletion or substitution, their semantic distance depends on the semantic 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, however, a few adjustments are necessary:

- 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 preprocessing.
- 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 case (c), we deal with questions that cannot be aligned, such as "What is the population of Canada" and "How many people live in Canada". They should have a relatively small distance since they ask about the same thing and have the same answers. We hereby introduce a semantic clustering method that discovers question similarity by processing relationships between the entities in the QA pairs.

5.3 Semantic Clustering

Intuitively, each question is either explicitly or implicitly about relationships between entities. Questions asking about the same kind of relationship hence have shorter relative encoding. For the purpose of discovering relationships of two questions, we used our data set of over 35 million pairs collected from community question answering websites. For each $\{q, a\}$ pair, we extract the named entities that q and a contains, namely e_q and e_a. The named entity set is limited to the fact set of an external graphic knowledge base, such as DBpedia. Given surface text s, the corresponding named entity e is retrieved by searching through the index of the hyperlinks of Wikipedia pages, with a confidence score

$$C(e, s) = Popularity(e) * P(e|s), \tag{3}$$

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

Figure 2: Selected templates representing the relationship of "spouse"

where $P(e|s)$ is the probability of the entity e given text s and $Popularity(e)$ is the number of occurrence of entity e in the hyperlink set.

Then we use the same knowledge base to determine relationships between e_q and e_a, that is, to find predicate paths connecting them. We assume each question has only one entity that is related with its answers. Thus, for each relation r that is found over the entity pair e_q and e_a, we generate a question template t, with e_q extracted from q. Note that multiple relations may result in multiple tuples of (t, e_q, e_a, r) for a single $\{q, a\}$ pair.

Next, we group tuples with the same template together. For a template t, we want to select the most relevant relation r by calculating an ranking score:

$$Score(r|t) = P(r|t) \cdot \log \frac{|T|}{|T_r|} \cdot \sum_{\substack{q \in Tuple(t,r) \\ r \Leftarrow (e_q, e_a)}} \frac{C(e_q, s_q) \cdot C(e_a, s_a)}{|Tuple(t, r)|}. \tag{4}$$

The probability $P(r|t)$ denotes the frequency of the relation r in the tuples that contains template t. The item $log \frac{|T|}{|T_r|}$ is the inverse template frequency, where $|T|$ is the total number of templates and $|T_r|$ is the number of templates that are associated with relation r. The third part of the product is averaging the confidence of named entity recognitions that produce relation r over tuples contains both t and r. Finally, the tuples are clustered such that those that share the same relation r are put into the same cluster.

To illustrate the result of the clustering, in Figure 2, we list the templates that represent the "spouse" relationship, where "<E>" denotes a named entity.

With these data, given a question, we can determine the relationships it asks about by finding the matching templates, and their corresponding relationships. If two questions share the same relationship, then the information distance between them is defined to be the information distance between the named entities in the questions or the answers.

5.4 Building Domains

The encoding steps above are reasonably accurate to approximate the uncomputable information distance in this natural language setting. We now briefly describe how to obtain Q_i, the set of questions for a vertical domain V_i.

Depending on the scenario of application, the question set Q_i can be constructed in multiple ways. In our system, it is done semi-automatically, by clustering on a human anno-

tated corpus. Details of our implementation are described in Section 6. Q_i can also be constructed fully automatically, when less manpower is available. First we start with a few representative queries. Then we use a compiler that generates queries that are similar to the representative queries, i.e. queries that are within a small semantic distance from the representatives. More formally, starting from the representative set R_i for each $q \in R_i$, the compiler generates the set $q^* = \{q' | d_s(q', q) \leq \epsilon\}$, where ϵ is a small constant. Finally we get $Q_i = \bigcup_{q \in R_i} q^*$.

Building domains requires the most human effort in the process of developing our QA system. Therefore we are hopeful that the completion of the compiler will greatly reduce the need of human intervention by automating the task. Generally speaking, to build a domain is to expand the set of representative questions. The user only needs to provide a few representative questions, and leaves the task of generalizing similar questions to the compiler.

Figure 3 illustrates the vertical domain Time. The core of the domain is the key concepts that defines it, or, more concretely, the APIs to answer domain-specific questions, which is to be implemented by the QA system. To instantiate these concepts, the procedure described above is performed, which goes from keywords to questions and then to clusters.

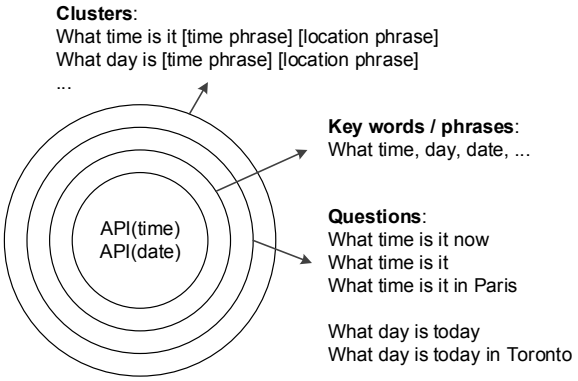

Figure 3: Vertical domain Time

6. CLASSIFICATION FRAMEWORK

In this section, we briefly describe the framework of the classification process.

Given a set of labeled data, i.e. questions Q and the corresponding domains they belong to, we first prepare the training sets for classification. Concretely, for each domain V_k, all the questions from Q that are labelled as positive examples in V_k are collected as Q_k^+. All the rest questions in Q form Q_k^-.

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

1. Iterate through all the 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 results of classification. This distance also serves as a likelihood measure, i.e. the domain V_k with the smallest $d_s(q, V)$ is considered to be the most likely domain that q belongs to.

In practice, however, a few optimizations are necessary:

1. For each vertical domain V_k, we use a pre-determined list of keywords to filter the questions. Questions that do not contain any of these keywords are excluded from Q_k^-, and are considered impossible to be a positive example for V_k. The keyword list is supposed to be exhaustive, so as not to create false negatives. This does not violate the principle of automatic processing, as the human effort of picking keywords for each domain is a vital part in creating the vertical search domain.
2. To further speed up the computation, we perform clustering on the questions in Q_k^+, using plain edit distance with a low threshold. This step will avoid repeated comparisons with highly similar questions in Q_k^+.

7. EXPERIMENTS AND RESULTS

7.1 Data

To sufficiently test our theory, we have decided to obtain the largest QA dataset available to academic research. We have crawled the Internet and downloaded more than 35 million English question-answer pairs. These questions and answers are all posted by users of QA websites, such as WikiAnswers and Yahoo Answers. Although many of these questions contain typos and grammatical errors, we decided not to correct them to be honest to the Internet data. Also, by keeping the original data, we hope to simulate, to some degree, the erroneous speech recognition results and the translated questions in cross language search.

Removing domains (1) for which we do not have data (e.g. phone functions), (2) that are not available on commercial systems, or (3) that are not specific enough (e.g. news), we are left with three popular domains: *Weather*, *Maps* and *Restaurants*. The experiments are conducted individually on the domains, i.e. considering the problem as three binary classification problems.

In order to do the testing, we have first selected over 50,000 relevant questions by clustering methods. Then we manually annotated these 50,000 relevant questions, classifying them into different domains. To our knowledge, this is the first large scale effort and dataset for testing question classification and it is also the largest dataset available for our purposes. The data is available on request.

To build a training data set, we first use some keywords to filter the questions, leaving a relatively small number of questions as possible candidates on each domain, respectively. This reduces many obvious negative questions, hence avoid unnecessary annotations for question labelling. We also deploy clustering on the candidates to sample the most representative questions within a domain.

We have selected questions from each domain individually, thus questions in different domains do not overlap. For each question we simply give it a label whether or not it belongs to the domain. Table 2 lists the total number of questions (selected from over 50,000 questions which in turn were selected from 35 million) as well as the number of positive and negative samples for each domain.

Table 2: Experiment dataset statistics

	Weather	Maps	Restaurants
# Positive	1,520	1,246	610
# Negative	2,517	4,805	2,068
# Uncertain	420	156	286
Total # Questions	4,457	6,207	2,964

100 questions from each domain were randomly chosen for the purpose of testing. Table 3 shows the statistics of these test data.

Table 3: Test data statistics

	Weather	Maps	Restaurants
# Positive	32	25	27
# Negative	68	75	73
Total # Questions	100	100	100

The annotations were conducted by two human experts, with agreements of Cohen's $\kappa = 0.95, 1.00, 0.93$ of Weather, Maps and Restaurants domains, respectively. A third annotator is introduced to resolve the conflicts. Generally speaking, human annotators had little difficulty. But to computers, the questions can be misleading in various ways, as illustrated in Figure 4. This usually happens when the specific keywords exist but the question is actually asking for some other facts. For instance, keywords "hot", "cold", "temperature", "today", etc. are to some extent revealing the topic of Weather, but apparently there can be general domain questions that contain these words.

7.2 Baseline Algorithms

We performed classification with our algorithm (*InfoDist*) in comparison with other methods. Our baseline methods include:

- LLM: Introduced in [9], the original LLM similarity is defined as:

$$LLM_{orig}(s_1, s_2) = \frac{\sum_{v \in s_2} \max_{u \in s_1} sim(u, v)}{|s_2|},$$

where $sim(u, v)$ is a similarity metric defined over semantic units u and v in two sentences s_1 and s_2. Here we use $WNSim$, a similarity measure based on WordNet. Note it is not symmetric. Hence we take the arithmetic mean instead:

$$LLM(s_1, s_2) = \frac{LLM_{orig}(s_1, s_2) + LLM_{orig}(s_2, s_1)}{2}.$$

- Word2Vec cosine similarity: The original cosine similarity deploys a simple vector space model with TD-IDF weights. We improved it by using the popular Word2Vec [22, 23] to vectorize text. The vector for a sentence is obtained by adding up vectors of its words. Thus

$$Word2Vec(s_1, s_2) = cossim(\sum_{u \in s_1} vec(u), \sum_{v \in s_2} vec(v)),$$

where u and v are words in the sentences s_1 and s_2, respectively. $vec(u)$ is the vector of that word, containing 300 dimensions.

- Traditional classification approaches. We have implemented a naïve Bayesian classifier, a logistic regression model, and SVMs, with linear, polynomial, radial, and Sigmoid kernels. Among these methods, SVM performs the best. Therefore we only included SVM in the question classification experiments. Details of these methods are discussed below.

- In addition, we also tested two commercial systems: Apple's Siri and Samsung's S-Voice. These tests were performed manually. The weather domain experiment was done in July, 2012 using Siri on iPhone 4S and S-Voice on Samsung Galaxy III, both with the latest version at the time. The tests for the restaurant and map domains were performed in January, 2013, using Siri on iPhone 5 and S-Voice on Samsung Galaxy III, also with the latest versions. At the time of our experiment, S-Voice did not contain the functionality of finding restaurants, hence, we did not include its results for the restaurant domain.

Note that all of these tests are for classifying purposes only, not to see the correctness of the answer. Hence, when Siri replied "Sorry, I can't look for restaurants in Canada", apparently the question was classified into the restaurant domain. Our statistics were then done accordingly.

For the Bayesian classifier, by using the n-gram language model and assuming Markov properties, the probability of observing that the sentence W would consist of the words $w_1, w_2, \ldots, w_m$ is approximated as:

$$P(w_1, w_2, \ldots, w_m) \approx \prod_{i=n+1,\ldots m} P(w_i | w_{i-n}, \ldots, w_{i-1}).$$

The naïve Bayesian classifier simply tries to find the question set c_i that maximizes the probability $P(W)$:

$$C = argmax_{c_i \in C} P(c_i) \prod_{w \in W} P(W | c_i).$$

In the binary classifying context, $C = \{c_0, c_1\}$, where c_1 denotes the set of questions belonging to the domain, and c_0 denotes the set of questions not belonging to the domain.

The following features are used for both logistic regression and the SVM models:

(1) Number of words in the question;
(2) Difference of unigram, bigram and trigram probabilities: $P(W|c_0) - P(W|c_1)$;
(3) Sentence tense: past, present or future;
(4) Having location expressions: yes or no;
(5) Having time expressions: yes or no;
(6) Question type: who, what, how, and so on.

The hypothesis of the logistic regression classifier is represented as a logistic function of a linear combination of inputs: $h(x) = \sigma(w^T x)$. $h(x)$ can then be interpreted as $P(x \in c_1)$.

Among these methods, the linear kernel SVM has achieved the highest accuracy by a 10-fold cross validation. This SVM has accuracies ranged from 85.1% to 91.1% among the three domains, about 1 percentage point and 10 percentage points higher than the logistic regression and Naïve Bayesian classifiers, respectively. This linear kernel also performs the best among the other traditional kernels (approx. 1 point, 5 points and 10 points higher than the Sigmoid, polynomial

Domain	Questions	Relevancy
Weather	What is the weather like for the Superbowl?	Relevant
	Is silk good to wear in hot weather?	Irrelevant
	How cold weather can cats deal with outside?	Irrelevant
Maps	Distance from the origin to a point?	Irrelevant
	How long is emirates plane?	Irrelevant
	How do you reach Blue Mountain by train?	Relevant
Restaurants	Where is ice cream popular?	Irrelevant
	What's the best Chinese food in town?	Relevant
	Food that begins with New York?	Irrelevant

Figure 4: Examples of misleading questions.

and radial kernels, respectively). Therefore, SVM with linear kernel (denoted as *SVM-linear*) will be used in the next section as the baseline standard approach.

Besides the traditional kernels, we have also implemented a tree-kernel SVM [26] for an advanced baseline, denoted as *SVM-tree*. This SVM exploits the syntactic parse tree information, hence is more suitable in NLP problems.

7.3 Main Results

Table 4 lists the result of different services (models) on the domains. The best result of each measurement is in **bold**. As revealed, InfoDist outperforms the baseline approaches as well as some state-of-art services.

7.4 Question Variations

Our proposed method is able to process variants of the questions. Syntactical and text edit level variations are more straightforward to see, here we give one example on our semantic encoding. While large scale experiments require much larger datasets for the purpose of sufficient clustering, here is a preliminary test. A set of questions with respect to the "military service" relationship, i.e. questions asking about people's military roles, was extracted by clustering from our database of 35 million QA pairs. The cluster is similar to those in Figure 2. With the algorithm mentioned in Section 5, we are able to recognize all these variations, and give short semantic encoding via the "military service" cluster. In contrast, Siri and Wolfram Alpha failed to answer such kinds of questions and always referred to web search. *evi.com*, on the other hand, did slightly better. Although it did not have all the knowledge, in some cases it did recognize the named entities, in which case we consider it being able to understand the questions.

The questions *evi.com* was able to understand include:

- What military service did John F. Kennedy serve in?
- What military rank is George Washington in?
- What did John F Kennedy do in the navy?
- What role did George Washington play in the American revolution?

The questions *evi.com* was *not* able to understand are:

- What was Andrew Jackson's position in the Tennessee militia?
- What was George Washington rank in the revolutionary war?
- What was Abraham Lincoln's military rank?
- What war did Elvis serve in?
- What did Robert Todd Lincoln serve as in the civil war?

7.5 Discussions

Not only our system is implemented based on a solid theory, but also it outperforms commercial systems as well as baseline academic systems such as SVM, cosine distance and LLM. Not only producing inferior results, the traditional classification methods (such as SVM) hardly provide any clue on how to answer the questions even after correct classification. By contrast, given a query, by finding a question (with answer) within a small information distance to it, we can answer the query using the same answer.

To conclude, our seemingly simple method is robust. It uses all text information and has the potential of deeper comprehension of the semantic intent.

8. CONCLUSIONS AND FUTURE WORK

While we have provided a unifying theory to capture the essence of the undefined semantic distance, our implementation is deceptively simple. Simplicity implies robustness, as shown in our experimental results. Although we have used traditional building blocks such as WordNet for word similarities, there is indeed a subtle difference between our approach and known ones. In our work, we aimed for simplicity (for shortest encoding), but we kept the bottom line of decodability. In comparison, other methods such as cosine distance, LLM, and SVM (most kernels) do not respect decodability hence unable to distinguish questions like "what eat fish" and "what fish eat". We have also introduced new algorithms of encoding via semantic space by clustering questions with similar answers.

It is important to point out that our theory has established an absolute distance. While the encoding algorithm we have provided is only an initial effort to approximate, it is a once-and-for-all deal. The advantage of a universal approximation using information distance is that it enables us to use any useful encoding scheme, including everything listed in Sections 1 and 5, in a consistent way. Currently our encoding method treats each word or phrase as one isolated object, and does not take the context or probability into consideration. Thus, it does not handle ambiguity well. In fact, this is exactly the reason why our classifier failed in some test cases. Solving ambiguity is difficult, but in some cases it does help to infer the exact meaning of a word from the context.

We are also implementing a compiler that, given the examples of a particular domain, generates similar questions within small information/semantic distance away. With such a compiler, users can easily create new vertical domains,

Table 4: Comparison of Siri, S-Voice, LLM, Word2Vec, SVM and InfoDist on the three domains

Domain	Measurements	Siri	S-Voice	LLM	Word2Vec	SVM-linear	SVM-tree	InfoDist
Weather	Accuracy	0.470	0.230	0.850	0.820	0.880	0.920	**0.950**
	Precision	0.344	0.176	0.750	0.727	0.862	0.838	**0.912**
	Recall	0.667	0.364	0.818	0.727	0.758	**0.939**	**0.939**
	F_1-score	0.454	0.238	0.783	0.727	0.806	0.886	**0.925**
Maps	Accuracy	0.770	0.870	0.860	0.920	0.900	**0.950**	**0.950**
	Precision	0.528	0.875	0.762	0.870	0.800	**1.000**	0.885
	Recall	0.760	0.560	0.640	0.800	0.800	0.800	**0.920**
	F_1-score	0.623	0.683	0.696	0.833	0.800	0.889	**0.902**
Restaurants	Accuracy	0.670	–	0.830	0.880	0.850	0.910	**0.940**
	Precision	0.438	–	0.679	0.759	0.750	0.846	**0.862**
	Recall	0.778	–	0.704	0.815	0.667	0.815	**0.926**
	F_1-score	0.560	–	0.691	0.786	0.706	0.830	**0.893**

thus significantly speeding up the development of natural user interface systems.

This approach depends on the hypothesis that, in most cases, semantic distance and information distance coincide. It is clear that small (computable approximation of) 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 [21]), we believe this is the case. However, in other applications domains, this might not hold. It is important to study the limitations of our approach, as all good theories have boundaries.

9. ACKNOWLEDGEMENTS

We appreciate the annotators for testing and labeling the compared services and the anonymous reviewers for their comments. This work was supported in part by NSERC Grant OGP0046506, Canada Research Chair program and a CFI infrastructure grant, an NSERC Collaborative Grant, Killam Prize, and an IDRC grant. We are also supported by National Grant Fundamental Research (973 Program) of China under Project 2014CB340304.

10. REFERENCES

[1] C. H. Bennett, P. Gács, M. Li, P. M. Vitányi, and W. H. Zurek. Information distance. *IEEE Transactions on Information Theory*, 44(4):1407–1423, 1998.

[2] F. Bu, X. Zhu, Y. Hao, and X. Zhu. Function-based question classification for general QA. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1119–1128, 2010.

[3] F. Bu, X.-Y. Zhu, and M. Li. A new multiword expression metric and its applications. *Journal of Computer Science and Technology*, 26(1):3–13, 2011.

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

[5] L. Chen, D. Zhang, and L. Mark. Understanding user intent in community question answering. In *Proceedings of the 21st International Conference Companion on World Wide Web*, pages 823–828, 2012.

[6] X. Chen, B. Francia, M. Li, B. Mckinnon, and A. Seker. Shared information and program plagiarism detection. *IEEE Transactions on Information Theory*, 50(7):1545–1551, 2004.

[7] R. Cilibrasi and P. M. Vitányi. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545, 2005.

[8] P. R. Comas and J. Turmo. Robust question answering for speech transcripts: UPC experience in QAst 2009. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, pages 297–304. Springer, 2010.

[9] Q. Do, D. Roth, M. Sammons, Y. Tu, and V. Vydiswaran. Robust, light-weight approaches to compute lexical similarity. Technical report, University of Illinois, 2009.

[10] U. Hermjakob. Parsing and question classification for question answering. In *Proceedings of the Workshop on Open-domain Question Answering - Volume 12*, pages 1–6, 2001.

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

[12] Z. Huang, M. Thint, and Z. Qin. Question classification using head words and their hypernyms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 927–936, 2008.

[13] B. J. Jansen and D. Booth. Classifying web queries by topic and user intent. In *CHI'10 Extended Abstracts on Human Factors in Computing Systems*, pages 4285–4290, 2010.

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

[15] M. Le Nguyen, T. T. Nguyen, and A. Shimazu. Subtree mining for question classification problem. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, pages 1695–1700, 2007.

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

[17] M. Li and P. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer, 1997.

[18] X. Li and D. Roth. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, 2002.

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

[20] Y. Liu, S. Li, Y. Cao, C.-Y. Lin, D. Han, and Y. Yu. Understanding and summarizing answers in community-based question answering services. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 497–504, 2008.

[21] Microsoft Research. Microsoft research question-answering corpus, 2008. http://research.microsoft.com/en-us/downloads/88c0021c-328a-4148-a158-a42d7331c6cf/.

[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[23] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[24] G. A. Miller. WordNet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

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

[26] A. Moschitti. A study on convolution kernels for shallow semantic parsing. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 335–342, 2004.

[27] A. Moschitti, S. Quarteroni, R. Basili, and S. Manandhar. Exploiting syntactic and shallow semantic kernels for question answer classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 776–783, 2007.

[28] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence-Volume 1*, pages 448–453. Morgan Kaufmann Publishers Inc., 1995.

[29] T. Solorio, M. Pérez-Coutiño, M. Montes-y Gémez, L. Villasenor-Pineda, and A. López-López. A language independent method for question classification. In *Proceedings of the 20th International Conference on Computational Linguistics*, page 1374, 2004.

[30] Y. Tang, D. Wang, J. Bai, X. Zhu, and M. Li. Information distance between what I said and what it heard. *Communications of the ACM*, 56(7):70–77, 2013.

[31] G. Tur and R. De Mori. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*. Wiley, 2011.

[32] J. Turmo, P. R. Comas, S. Rosset, O. Galibert, N. Moreau, D. Mostefa, P. Rosso, and D. Buscaldi. Overview of QAst 2009. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, pages 197–211. Springer, 2010.

[33] W. Wu, Z. Lu, and H. Li. Learning bilinear model for matching queries and documents. *The Journal of Machine Learning Research*, 14(1):2519–2548, 2013.

[34] D. Zhang and W. S. Lee. Question classification using support vector machines. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, pages 26–32, 2003.

[35] X. Zhang, Y. Hao, X. Zhu, and M. Li. 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, 2007.