THESIS PROPOSAL REPORT 3

DOCTORAL PROGRAM IN COMPUTER SCIENCE
RESEARCH STUDENT MONITORING GROUP

Open Domain Question Answering Over Natural Language Data: A Knowledge Rich Approach[1]

Harish Tayyar Madabushi
H.T.Madabushi@pgr.bham.ac.uk

Supervisors:
John BARNDEN
Mark LEE

Thesis Group:
David PARKER
Manfred KERBER

August 25, 2015

SCHOOL OF COMPUTER SCIENCE
UNIVERSITY OF BIRMINGHAM

[1] Working Title

Abstract

Question Answering (QA) is a task in Natural Language Processing that requires the system to respond, with short one word or phrase answers, to questions posed in Natural Language. Open domain QA, unlike domain specific QA, places no domain restrictions on the question. Question Answering does away with the requirement of reading through large amounts of text and allows users to find the specific piece of information required.

Current state of the art QA systems use shallow reasoning methods, statistical approximation, and supervised learning. Systems that depend on supervised learning, while being able to achieve the best performance, require large amounts of tagged data. All three approaches suffer from an inability to extend to more complex answering systems, such as the ability to answer follow up questions.

My proposed research aims at solving the problem of QA using a knowledge rich approach. Such a method will involve the expansion of a core knowledge base from existing natural language (unstructured data). Additionally, the use of learning to determine specific ontological structures and methods that might be useful in solving a specific problem will be explored.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The indexed Surface Web is estimated to contain around 4.68 billion pages (de Kunder, 2008, 2015). Admittedly the vast majority of this information might have little relevance to a particular individual, however, given that just the popular AI journals publish over five thousand articles a year (SCImago, 2013), it is safe to assume that individuals have access to vastly more information than they can possibly sift through.

Current solutions to this information overload range from search engines to (human) personal assistants. With no indications that this explosive growth in information is likely to diminish, the need for an alternate solution is larger than ever before.

Fortunately, this very information overload is a boon to Computer-based systems. The abundance of information expressed in Natural Languages, combined with the the possibility that language plays a significant role in our consciousness (Minsky, 1986), makes Natural Language Understanding, Representation and Reasoning an ideal starting point for Machine Cognition.

This opportunity for Machine Cognition and difficulties faced by Human users can most easily be addressed through *Question Answering Systems*, which, while providing an intuitive way of sifting through large amounts of information, can simultaneously provide researchers with real-world interaction data that will further efforts towards achieving Machine Cognition.

1.1 Problem Definition

For the purpose of this thesis we limit our research to answering Factoid Questions in *English*. We explicitly avoid addressing descriptive "Why" questions. Not only do such questions require the identification and differentiation between causes and effects (Oh et al., 2013), we believe that addressing such questions using a knowledge rich approach can be made possible only after similarly addressing factoid questions.

Additionally, we focus on answering Natural Language Questions based on information extracted from Natural Language text as opposed to structured Knowledge Bases. Finally, we do not restrict ourselves to document or paragraph level Question Answering wherein the specific document or piece of text contain the Answer is provided.

1.2 Research Questions

In light of our introduction to the problem of Question Answering, our research will aim at addressing the following Research Questions:

Research Question 1:

How can relationships between elements of free text and elements in a Knowledge Bases established?

1. What are the various elements in free text that can be linked to Knowledge Bases?
2. What are the Knowledge Bases that can be used and what are the advantages of each?

Research Question 2:

What is the best structure to represent a combination of free text and information extracted from Knowledge Bases?

1. What structures will enable us to maintain the syntactic structure of free text while enabling us to process text?
2. How can elements of a Knowledge Base be integrated into the structure representing free text?
3. How can this system be used for Question Answering?

Research Question 3:

How can learning algorithms be used to improve the accuracy of the System?

1. Which specific learning algorithm will provide the best results?
2. How can the structures we use to represent the combination of free text and Knowledge Bases be parameterized as input to a learning algorithm?

Chapter 2

Elements of Question Answering

Traditionally, Question Answering systems consist of three elements. Figure 2.1 provides an overview of these elements and the relations between them. In this chapter we briefly explore each of these elements to provide a precursor to the our work. Other than Word Sense Disambiguation, our work does not involve these elements but instead, in some cases, builds on them. Despite our experiments with Word Sense Disambiguation we actively avoid it along with any other task that could be used as a building block for creating a Question Answering system for reasons detailed in Section 7.1.3.

2.1 Natural Language Understanding

Natural Language Understanding (NLU) is by far the most complex element and is required both in understanding the questions posed by a user and in sifting through Natural Language text while collecting information that might be useful in answering the question.

Despite early explorations by Schank (1972) into the several sub-tasks that NLU consists of, the problem remains unsolved.

2.1.1 POS Taggers (State of the Art)

POS tagging systems use a wide variety of methods from Corpus Statistics to some forms of learning. Current State of the Art in POS tagging is capable of achieving accuracies of up to 97.29% (Manning, 2011), making POS tagging a solved problem.

Initial work on POS tagging was based on hidden Markov models (Brants, 2000). Subsequent work has involved the use of maximum-entropy Markov model (MEMM) with external lexical information (Denis and Sagot, 2012), and SVM-based systems such as that described by Giménez and Màrquez (2003). The state of the art system described by Manning (2011) uses Maximum-entropy cyclic dependency networks.

2.1.2 Parsing (State of the Art)

Parts of Speech once identified often have extremely different meanings based on their context. Parsing provides a solution to find the relation between the previously identified Parts of Speech.

Some methods of parsing use Lexicalized probabilistic context-free grammar (PCFG) (Collins, 1996; Bikel, 2004) with variations such as Lexicalized N-Best PCFG with re-ranking (Charniak and Johnson, 2005).

Typed Dependency Parsers, such as the one introduced by de Marneffe et al. (2006) proved to be far more accurate. Subsequently, Compositional Vector Grammar parsers (Socher et al., 2013) and neural-network dependency parser (Chen and Manning, 2014) provide close to 90% accuracy.

2.1.3 Word Sense Disambiguation (State of the Art)

Word-sense disambiguation(WSD), despite being an open problem, has seen significant progress to the extent that current State of the Art systems can achieve significantly high levels of accuracy (for English) in most generic domains. (Agirre and Edmonds, 2007)

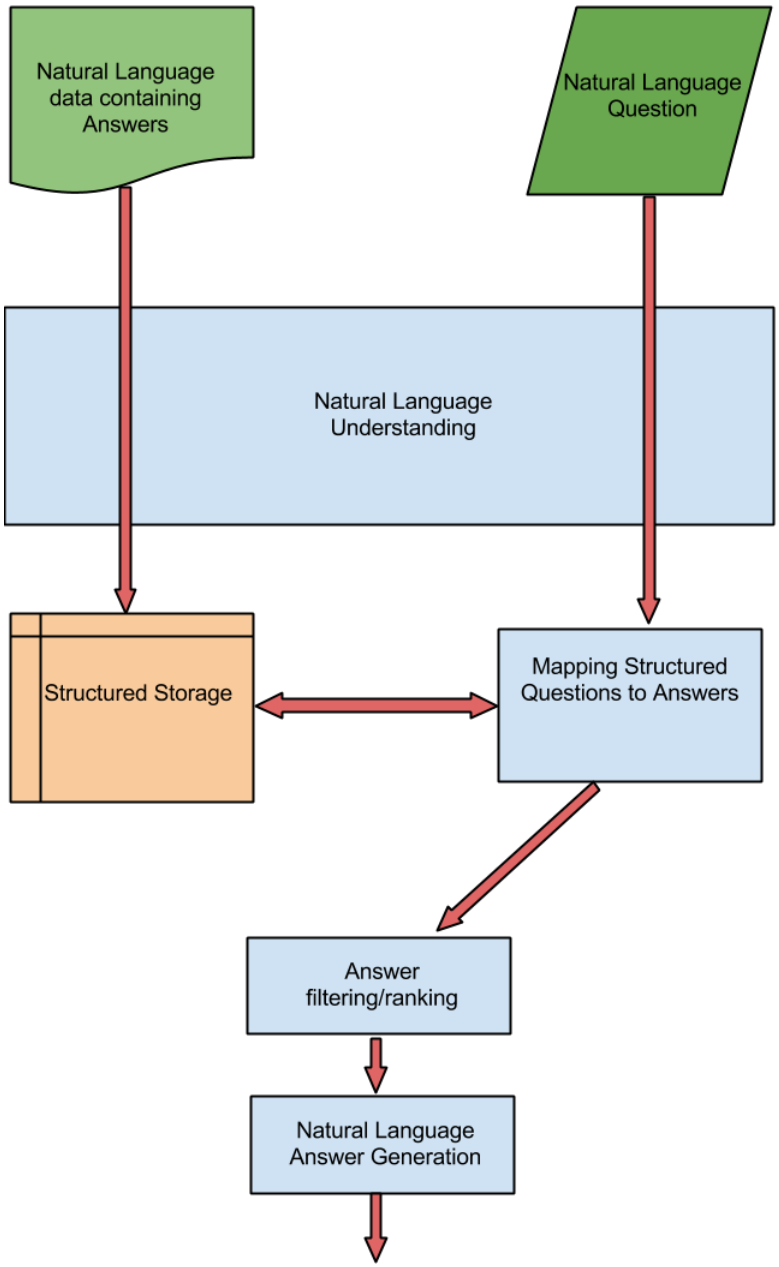


Figure 2.1: An overview of a typical Question Answering System

Dictionary based WSD systems range from the early Lesk algorithm (Lesk, 1986) to those that make use lexical knowledge bases such as WordNet (Miller, 1995). Additionally, graph based Algorithms have had significant success (Agirre et al., 2006), with vastly increased efficiency, when used in conjuncture with a strong lexical dictionary (Navigli and Lapata, 2010). Additionally, the use of Wikipedia has been shown to have a significant effect on improving results (Ponzetto and Navigli, 2010).

By far the most effective algorithms for WSD have been supervised and semi-supervised algorithms (Agirre and Edmonds, 2007).

We discuss our attempts at solving the problem of Word Sense Disambiguation in section 6.4

2.2 Knowledge Representation

Formal Languages have several important applications in a variety of domains in both Computer Science and Mathematics. They are often used for the purpose of storing and processing knowledge because they are easily parsable and machine readable. They fall under the broad category of Ontologies. Most Ontologies that are *Complete* lack the expressive power to fully express the complexities of real world knowledge.

2.2.1 Ontologies

OWL

Web Ontology Language (OWL) (Bechhofer, 2009) and the OWL Family provide an ontology based on XML and due to its work with W3C has had significant adaptation. OWL has subsequently been superseded by OWL2. The primary objective of OWL is to make web data more easily accessible to machines.

CycL

CycL (Lenat and Guha, 1991) is the language that is used by the CYC system to store knowledge. CycL is open-Source.

Dbpedia

Dbpedia (Auer et al., 2007) is a project aimed at extracting and representing as a "database" information created as part of the Wikipedia project.

2.3 Reasoning

A critical element of our work revolves around representation and reasoning systems. The abundance of information makes discovery of information relatively easy, however, it is in "making sense" of this data that the challenge lies.

2.3.1 Description logic and its Extensions

First-order predicate logic, while being more expressive than *Propositional logic* struggles with decidability.

Description logic.

Description logic (Baader et al., 2003) while offering a solution to both decidability and increased expressive power is significantly limited by its inability to work with uncertainty. Given that uncertainty is such a large part of real-world problems, DL is not a system we can work with.

Probabilistic Description Logic

Probabilistic Description Logic (Lukasiewicz, 2008) attempts to solve the problem of uncertainty in Description Logic, however, inferred rules in PDL are progressively less precise than their parents.

Bayesian Network-Ontology combinations

Several Bayesian Network-Ontology combinations (da Costa et al., 2008; Koller et al., 1997) have been employed for the representation of knowledge. The primary problem with such combinations has been in terms of consistency as Bayesian's completeness requirement forces them to be less granular than Ontologies.

Fuzzy and probabilistic semantic networks.

Fuzzy and probabilistic semantic networks (Straccia, 2006) solve the problem of expressing information but fail in the face of information processing as information is lost during reasoning.

2.3.2 Bayesian Knowledge-driven Ontologies

Bayesian Knowledge-driven Ontology (Santos and Jurmain, 2011) is a State of Art framework which is a synthesis of semantic networks and generalised Bayesian networks that accommodate incompleteness. The recency of this framework makes it difficult to gauge its effectiveness although it appears that adding entities to this framework might pose problems in addition to representation of time.

2.3.3 Minsky Frames

Although Frames have been used for representing data in a wide range of domains, Minsky's description of Frames (Minsky, 1975), which refines the definition of frames as data structures that are intended to recognise *instances* of patterns through "attributes" is most appropriate for our purposes. These attributes contain either default values or values specific to an instance or even other attributes.

Despite the adaptation of Minsky Frames for use in Frame Technology, there has been little exploration of Frames for use as knowledge representation and reasoning frameworks.

2.3.4 Knowledge-Line

Knowledge-Line (Minsky, 1980) represents the associations made between mental agents when a particular problem is solved. Subsequently,similar problems can be solved by simply "remembering" the associations made the first time around. Despite being a relatively old concept, K-Line, like Minsky Frames, have not been explored to any reasonable degree.

2.3.5 Natural Language Generation

Once we have extracted the data we require we then have to give it to the user. Most IR systems simply list documents. The creation of Natural Language to give this is NLG, the earliest being FoG (Goldberg et al., 1994).

Current systems are capable of generating reasonably simple Natural Language useful in practice(Reiter and Dale, 2000). There are several NLG techniques(Reiter and Dale, 1997) but we do not dwell on them as this aspect of conversational agents is not our focus.

Chapter 3

Learning Algorithms

Recent trends in Artificial Intelligence have shown that some of the best solutions to a variety of problems are based on machine learning. Machine learning provides ways of extracting patterns from data, without such patterns explicitly being defined. This provides for an extremely powerful way of categorising and sifting through large data sets that are impossible to analyse manually. In this chapter we provide an overview of various learning algorithms while providing additional details on those specific systems (such as ANNs) that we use extensively.

3.1 Learning Paradigms

Primarily there are three learning paradigms. Supervised Learning, Unsupervised Learning and Reinforced Learning. Although the majority of our work will involve supervised and unsupervised learning reinforced learning is powerful in certain tasks such as Synchronous machine translation.

3.2 Unsupervised Learning

Unsupervised learning refers to a set of algorithms that classify data into various sets without prior knowledge of what these classes are. The resultant sets consist of elements that are "close" to each other in N-Space. There are two approaches to Unsupervised learning, Cluster Analysis and and Latent class analysis.

Latent class Analysis allows for the use of a probabilistic model that describes the distribution of data for extraction of classes and extracts information from possible unobserved factors. Clustering on the other hand uses a generic distance measure for calculation of classes. Latent Class Analysis requires that parameters are conditionally independent - a criterion that is often violated in our analysis of Natural Language due to the inherent relation between elements of data such as syntactic and semantic elements. Instead we use clustering, specifically the K-Means algorithm (MacQueen, 1967), which we discuss in Section 3.2.1.

3.2.1 Clustering: K-Means Algorithm

K-Means(Duda and Hart, 1973) is a clustering algorithm that assigns elements to clusters based on the minimum within cluster sum of squares of the vector representation of features in N-Space. Given n elements $x_1, x_2, ...x_n$, the algorithm assigns each of x_i into clusters $c_1, c_2...c_k$ where $k \leq n$. It should be noted that K-Means is prone to local minimisation and requires the number of clusters to be provided.

Picking the number of clusters to use is a difficult problem and there is no clear automated method of doing this. One common technique is to check to see how many clusters are useful in downstream applications if any (such as clustering documents into a reasonable number of topics or T.Shirts into four sizes - S, M, L, XL - as opposed to a much larger number). Our experiments with Clustering using the K-Means Algorithm are detailed in Section 6.11.

X-Means

X-Means (Pelleg and Moore, 2000) gets around both local minima and the requirement of being supplied with the number of clusters. The algorithm works by starting with a single centroid and splitting it into

two by moving it in a direction proportional to the size of the cluster in opposite directions along a randomly chosen vector. Once this is done a Bayesian information criterion (Schwarz, 1978) is used to score the cluster with one and two centroids. This process is repeated until no new centroids are generated.

X-Means uses the Schwarz criterion (Kass and Wasserman, 1995) in the following form:

$$BIC(M_j) = \hat{l}_j(D) - \frac{p_j}{2}.logR$$

Where $\hat{l}_j(D)$ is the log-likelihood of the he data according to the j^th model and taken at the maximum-likelihood point, and p_j is the number of parameters in M_j. The largest drawback with this approach is that it cannot handle data in higher dimensions. Most of the data we handle is often in hundreds of dimensions.

MDL Principle for Robust Vector Quantization

This method of Robust Vector Quantization (Bischof et al., 1999) starts with a large number of clusters and subsequently reducing the number of clusters. The outliers are first identified and subsequently training elements are "encoded" so as to minimise their description length. The advantage of the system is it's ability to identify outliers. Ideally, this algorithm should be initialised with as many clusters as data points. However, such an initialisation would make it extremely inefficient leading to the dependence on initialisation.

G-means

G-means (Hamerly and Elkan, 2004) starts with a one (or a small number of) clusters and increases the number of clusters until each cluster's data comes from a Gaussian distribution. Although this algorithm is effective in finding a reasonable number of clusters, it suffers the same shortcoming as X-means - difficulty in handling data in higher dimensions.

The Elbow Method of identifying the optimal number of Clusters

"The Elbow Method" involves the charting of the residual within cluster sum of squares and finding the point at which the slope reduces (Figure 3.1). It should be noted that there could be several such points or, in the worst case, no such point. If there are multiple points that match our criterion we pick that point at which there is the largest reduction in slope. Our experiments with automating this are detailed in Section 6.11

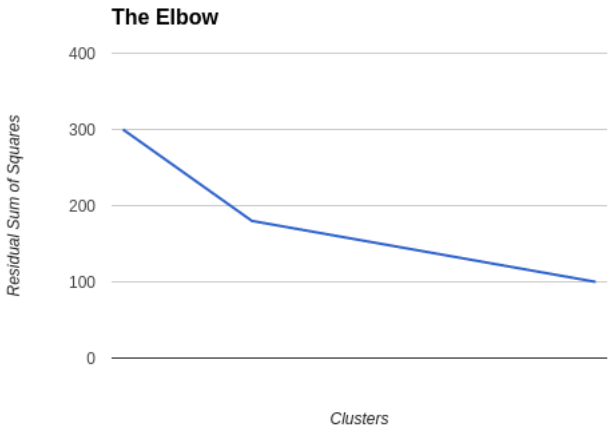


Figure 3.1: The Elbow for Identifying the Ideal Number of Clusters

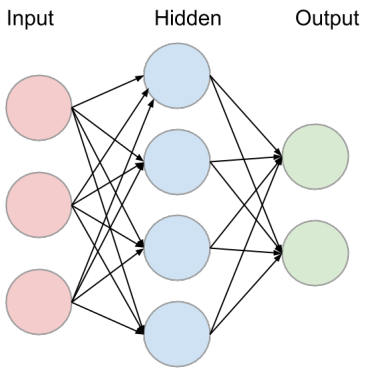


Figure 3.2: Neural Network Architecture

3.3 Supervised Learning

Supervised learning requires training labelled data which is used to create an estimation function. There are several supervised learning algorithms including Linear Regression (Galton, 1889), Logistic Regression (Garnier and Quételet, 1838), Decision Trees (Quinlan, 1986), Random Forests (Breiman, 2001), Support Vector Machines (Cortes and Vapnik, 1995) and Neural Networks (Rosenblatt, 1958).

The specific algorithm picked is based on the expected complexity of the function that is to be estimated, the number of training elements available the number of input variables the number of output variables and other algorithm specific constraints. In this section, we restrict ourselves to Support Vector Machines and Neural Networks as we use these extensively in our work. We discuss Neural Networks, Support Vector Machines and Decision Trees.

3.3.1 Artificial Neural Networks

Neural Networks, despite having been around since 1989 (Funahashi, 1989) have had a resurgence in recent times as computational power has caught up with the requirements of implementing Artificial Neural Networks (ANN).

State of the Art ANNs have been used, with significant success, in areas ranging from spam filtering to inverted helicopter flight (Ng et al., 2006). From our perspective, a significant modification to ANNs is the use of Bayesian Learning (Neal, 1996).

Standard Neural Networks are Directed Acyclic Graphs (DAGS) with input nodes whose activation is based on the current status of a system and whose output is one of several possible classes in a classification problem. Figure 3.2 represents the generic structure of such a Neural Network.

Each node in the input layer is "activated" based on the input parameters. The activation of subsequent layers is based on the input they receive, that nodes parameters and the specific activation function being used (We detail activation functions in Section 3.3.1).

The parameters at each node of the Neural Network are randomly initialised. It is important that none of these values are identical. The output of each input feature set is calculated based on these random values (Forwardpropagation) and the error rate, determined by the cost function (Section 3.3.1), at each node is bubbled back from the known output of the training set using an algorithm called Backpropagation (Section 3.3.1) (Rumelhart et al., 1988). Once a Neural Network is trained, Forwardpropagation is used to predict values of previously unseen input sets.

Cost Function

A Cost Function allows us to estimate how well our hypothesis fits the training data. Given X and Θ to be our parameters and y the output, our objective is to define a cost function ($J(\Theta)$) that captures $h_\Theta(X) - y$ (Rumelhart et al., 1988). This allows us to optimise the hypothesis function by minimising the cost function. In the case of Neural Networks, we additionally define L to be the total number of

layers in the network, s_l to be the number of nodes in layer l and K the number of output nodes or classes.

We pick a cost function that is both convex and can be derived using the principle of maximum likelihood estimation, the generic form of of which is as follows:

$$Cost(h_\Theta(X), y) = \begin{cases} -log(h_\Theta(X)) & \text{if } y = 1 \\ -log(1 - h_\Theta(X)) & \text{if } y = 0 \end{cases} \tag{3.1}$$

We adopt Equation 3.1 to Neural Networks with the notation defined above, which gives us the following cost function:

$$\begin{aligned} J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} log(h_\Theta(x^i))_k + (1 - y_k^{(i)}) log(1 - (h_\Theta(x_{(i)}))_k\right] \\ - \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1} \end{aligned} \tag{3.2}$$

Given this, we minimise the cost function $J(\Theta)$ by use of the following partial derivative:

$$\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta) \tag{3.3}$$

Backpropagation

Backpropagation (Rumelhart et al., 1988) allows us to optimise the parameters of individual nodes (Θ) so as to better fit the training data. The first step in Backpropagation is to calculate the derivative of the cost function (Equation 3.3. The second step is to find the gradient of the parameters of each node in the network and to subtract a fraction of the gradient from the weight. These steps are repeated until the network predicts the output satisfactorily. The specific fraction used in step two represents the learning rate of the network. The higher the learning rate, the faster the training, and the lower it is, the more accurate the training.

The calculation of the gradient (or partial derivative in Equation 3.3) can be achieved pragmatically using the following algorithm [1]:

Data: Training Set: $\{(x_1, y_1), ..., (x_m, y_m)\}$
Set $\Delta_{ij}^{(l)} = 0$ (for all l, i, j)
for *i = 1 to m* **do**
  Set $a_1 = x_1$
  Perform forward propagation to compute $a^{(l)}$ for l = 2, 3, ... , L
  Using $y^{(i)}$ compute $\partial^{(l)} = a^{(L)} - y^{(i)}$
  Compute $\partial^{(L-1)}, \partial^{(L-2)}, ..., \partial^{(2)}$
  $\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)} \partial_i^{(l+1)}$
end
Now Calculate:

$$D_{ij}^{(l)} := \begin{cases} \frac{1}{m}\Delta_{ij}^{(l)} + \lambda\Theta_{ij}^{(l)} & \text{if } j \neq 0 \\ \frac{1}{m}\Delta_{ij}^{(l)} & \text{if } j = 0 \end{cases}$$

Algorithm 1: Algorithm for calculating Gradient in Backpropagation

[1] Ng, Andrew. "Machine Learning". Coursera: www.coursera.org

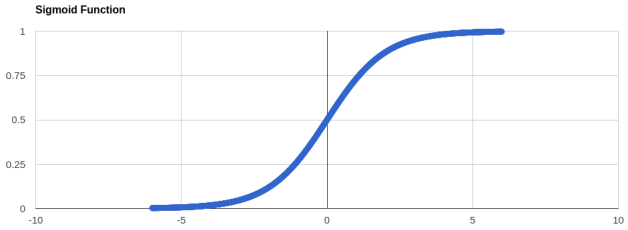


Figure 3.3: The Sigmoid Function

It can be shown that $D_{ij}^{(l)}$ calculated in Algorithm 1 gives the value of the partial derivative in Equation 3.3.

Neural Network Activation Functions

Each individual neuron is activated, with respect to x and Θ, the parameters, based on an activation function. Although different activation functions can be used for this purpose, by far the most common is the Sigmoid Function. The sigmoid function with respect to x and Θ is given in Equation 3.4. The behaviour of the generic Sigmoid function is shown in Figure 3.3.

$$\frac{1}{1 + e^{-\theta^T x}} \tag{3.4}$$

3.3.2 Recurrent Neural Networks

Recurrent Neural Networks are Neural Networks whose connections form directed cycles. They stem from Boltzmann machines (Ackley et al., 1985) which are hard to train due to the exponential number of configurations of hidden variables (Ackley et al., 1985; Welling and Teh, 2003). Although limiting their configurations - such as in the case of the Restricted Boltzmann Machine (Smolensky, 1986) - does get around this problem to some extent, Recurrent Neural Networks have developed to become a super-set of Boltzmann Machines. The cycles within the connections provide a way for the network to store temporal information which has been useful in modelling systems that contain sequential steps, including Natural Language Processing (Mikolov et al., 2013c). Recurrent Neural Networks are also Turing Complete (Siegelmann and Sontag, 1991).

3.3.3 Long Short Term Memory

Long Short Term Memory Network (LSTM Network) (Hochreiter and Schmidhuber, 1997) is a kind of Recurrent Neural Network that contains LSTM nodes in addition to or sometimes instead of the standard Neural Network nodes. These LSTM nodes have the ability to store information and based on inputs they decide whether to hold that value, "forget" that value or to output it. LSTM Networks have been successful in learning grammars (Schmidhuber et al., 2002; Gers and Schmidhuber, 2001).

3.3.4 Structured Prediction

Unlike the machine learning algorithms discussed so far Structured Prediction involves classification of structures as opposed to discrete values. We study Structured Perceptrons which is an extension of the standard leaner Perceptron for structured prediction[2]. It should be noted that Perceptrons are an example of supervised learning with reinforcement.

Structured Perceptron

Structured Perceptron is an extension of the perceptron that provides a method for structured prediction (Collins, 2002). Given the original input $x \in X$ and a hypothesised output $y \in Y$ the value $\Phi(x, y)$ is a vector in Euclidean space that depends on the output. Once we learn the weight vector w, this translates into the "argmax problem" as:

[2]Rai, Piyush. "Machine Learning". The University of Utah: www.cs.utah.edu

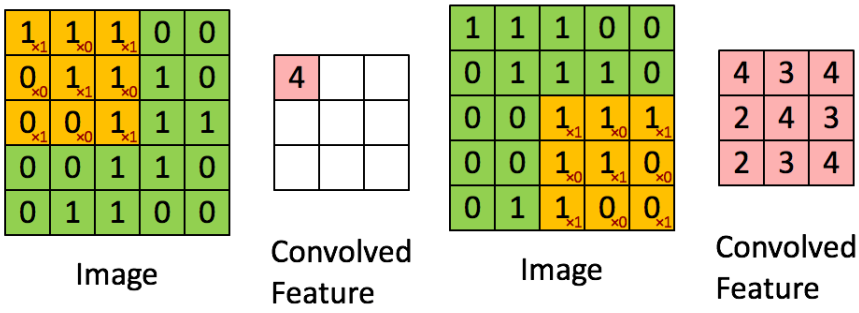


Figure 3.4: Convolution of an Image

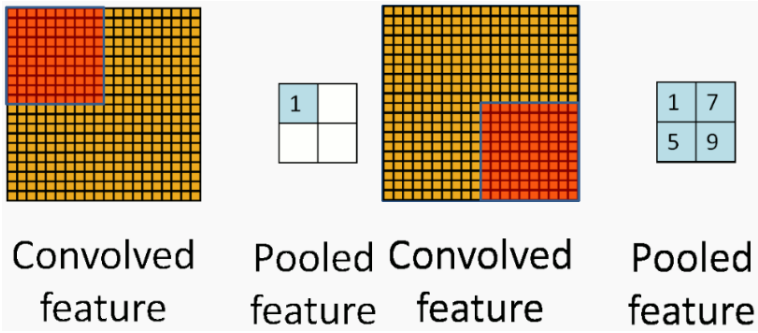


Figure 3.5: Pooling of Convoluted features

$$\hat{y} = arg \max_{y \in Y} w^T \Phi(x, y) \tag{3.5}$$

3.3.5 Convolutional Neural Networks

Convolutional Neural Networks (LeCun and Bengio, 1998) were first used for processing images and speech where input signals that are adjacent to one another are "averaged" to generate a single feature. A CNN consists of a one or more Convolutional layers followed by one or more standard fully connected layers as in a ANN.

Two critical aspects of a CNN are Convolution and Pooling. Convolution involves the "merging" of elements that are in close proximity to one another to form a single feature. Figure 3.4[3] shows how an image can be convoluted.

After Convolution it is possible that the number of features we have are still too many. We make use of the intuition that features that are useful in one part of the input (image) might also be useful in another. This process of either averaging or using the maximum of a certain range is called Pooling. Figure 3.5[3] shows how a convoluted feature set can be Pooled.

Although Convoluted Networks Networks are traditionally used for image and speech processing, they have recently been used to model sentences (Kalchbrenner et al., 2014) and have been shown to achieve high performance without additional features such as POS tags and sentence structure information.

3.3.6 Support Vector Machines

Support Vector Machines (Vapnik and Lerner, 1963) use a model that represents each training example as a point on a hyperplan. The model attempts to maximise the distance between examples of different classes. It should be noted that SVMs, in this form, are used for classification of problems that are linear separable. We do not explore this aspect in depth as most of the problems we encounter are not linear separable.

[3]Ng, Andrew. "Deep Learning Tutorial". Stanford: www.stanford.edu

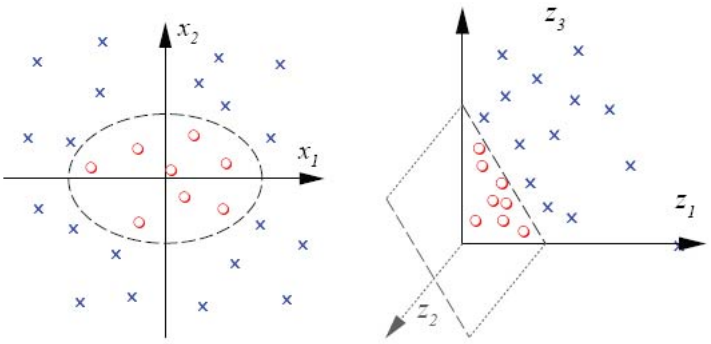


Figure 3.6: Mapping two dimensional data into three dimensions

When input data is not linear separable the SVN model projects the data into higher dimensions wherein it might be linear separable. This is achieved using what is known as the Kernel Trick (Boser et al., 1992). The Kernel Trick allows the model the operate in higher dimensions without actually calculating points in that dimension making it also computationally efficient. Figure 3.6[4] provides an insight into how this mapping is achieved.

We note that SVMs are extremely dependant on the Kernels that are picked and although it is easy to pick a Kernel for lower dimensions this is not necessarily the case for higher dimensions. For this reason we do not use SVMs in our work and instead work with Neural Networks. We additionally note that the "No Free Lunch Theorem" states that there is no way to establish that one classification algorithm is always better than the other and the performance of each is dependant on the data. In our specific case however, we alter Neural Networks to represent sentences making them better suited for our purpose.

3.3.7 Decision Trees

Decision Trees are models that classify data (or predict an output) based on input features. This is achieved by constructing a tree whose leaves are the target classes and branching is determined by the the value of features at each node. Training consists of starting with a root and splitting the training set into subsets based on the values of features. This process is repeated for each subsequent node until we reach a state wherein further splitting does not improve our classification. Figure 3.7[5] is a Decision Tree representing the survivors of RMS Titanic.

3.4 Reinforcement Learning

Reinforcement Learning is based on what is generally considered the acceptable method of learning in psychology (Vapnik, 1995) which, in 1911, Edward Thorndike defined as the The Law of Effect:

> Of several responses made to the same situation, those which are accompanied or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they will be more likely to recur; those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that, when it recurs, they will be less likely to occur. The greater the satisfaction or discomfort, the greater the strengthening or weakening of the bond. (Thorndike, 1911)

This concept of trying different approaches and following that path that provides the "best" result is central to Reinforcement Learning and was introduced by Minsky in his PhD dissertation (Minsky, 1954). In most cases the system is represented as a Markov decision process (Bellman, 1957). A Markov decision processes is formally defined as [6]:

[4]Thornton, Chris. "Machine Learning - Lecture 15 Support Vector Machines". University of Sussex: www.sussex.ac.uk
[5]DTREG. "Titanic Passenger Survival Analysis". DTREG: www.dtreg.com
[6]Ng, Andrew. "Machine Learning (CS229)". Stanford: www.stanford.edu

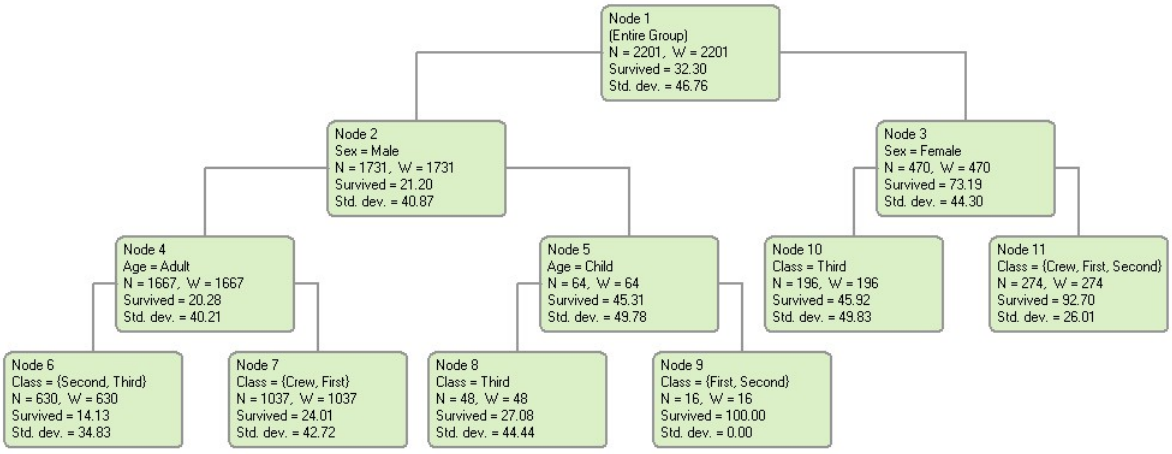


Figure 3.7: Decision Tree representing the survivors of RMS Titanic.

A Markov decision process is a tuple $(S, A, P_{sa}, \gamma, R)$, where:

- S is a set of states. (For example, in autonomous helicopter flight, S might be the set of all possible positions and orientations of the helicopter.)
- A is a set of actions. (For example, the set of all possible directions in which you can push the helicopters control sticks.)
- P_{sa} are the state transition probabilities. For each state $s \in S$ and action $a \in A$, P_{sa} is a distribution over the state space, P_{sa} gives the distribution over the states we will transition to if we take action a in state s.
- $\gamma \in [0, 1)$ called the discount factor.
- $R : S \times A \rightarrowtail \mathbb{R}$ is the reward function.

The aim is to identify a specific policy π, that maps a particular state $s \in S$ to another state s. Once this policy π is established the results of the MDP for any given starting state is fixed. Value Iteration and Policy Iteration are two algorithms that provide a way to calculate π.

3.4.1 Value iteration

Value iteration (Bellman, 1957) works backwards by starting with an arbitrary function π and iteratively updating it by calculating the rewards recursively for a predefined number of steps.

3.4.2 Policy iteration

Policy iteration (Howard, 1960) on the other hand updates the the model once before maximising the the reward function before then updating the model again and so on. This modification provides a stopping condition - when the model does not change over an iteration of the reward function.

Chapter 4

An Overview of the Methods used for Question Answering

Question Answering Systems use a wide verity of methods. In this chapter we provide an overview of these methods based on our exploration of literature in the field.

4.1 Matching Techniques

Most Question Answering systems use a form of matching, either to map questions onto existing structured databases or directory onto text that might potentially contain the answer.

4.1.1 Matching over Existing Structured Databases

The most popular choice of existing structured databases are Freebase (Bollacker et al., 2008) and DBPedia (Auer et al., 2007).

Yao and Van Durme (2014) have shown that using Freebase can often outperform some sophisticated approaches while Berant et al. (2013a) have shown how Question Answer pairs can be used to boost semantic parsing. Fader et al. (2014) have shown how data from Freebase can be combined with automatically extracted data to improve the quality of Question Answering systems. Several other researchers have developed systems based on Freebase (Cai and Yates, 2013b; Berant and Liang, 2014; Kwiatkowski et al., 2013; Bao et al., 2014)

Unger et al. (2012) have proposed a method that extracts information from DBPedia by going beyond the representation of questions as triplets. There have been several other attempts at creating Question Answering Systems using DBPedia (Walter et al., 2012; Yao et al., 2012). Fader et al. (2014) provide an overview of systems that use similar KBs.

Logical Forms over Linked Data

He et al. (2014) in their recent work have explored to use of First-order Logic in finding answers within Linked Data. Other recent work in this area has revolved around an attempt at converting Natural Language to a logical form. (Yang et al., 2014)

4.1.2 Pattern based Direct Mappings

Other techniques of mapping involve directly matching Questions and Answers using patterns. Li et al. (2010) use a semi-supervised method of achieving this while tau Yih et al. (2013) provide a method dependent on word alignment. An important matching technique that several of these methods rely on are Hearst patterns (Hearst, 1992).

4.1.3 Machine Learning

A lot of work including the recent work by tau Yih et al. (2014) focuses on using machine learning techniques to extract similarities between relations and entities. Although such methods have provided significantly improved results, we focus our efforts on transparent methods, while using machine learning as a tool for more fundamental tasks.

4.2 Tuples

Yet another method of representing and reasoning over data for the purpose of question answering is to store information in the form of Tuples. The most recent and influential work in this regard has been by Fader et al. (2011). In their paper, they describe ReVerb[1], a system that provides a method of extracting relevant extractions. Fader et al. (2013) subsequently described a method of mapping Open Domain Questions onto the relations extracted by ReVerb. Fader et al. (2014) have then combined these two methods to implement a Question Answering system that is now near State of the Art.

There have been others who have represented data in the form of tuples for the purpose of Question Answering include tau Yih et al. (2014) who use Convolutional neural network models to find similarities between entities and relations.

4.3 Graphical Representation of Data for Question Answering

A more complex representation of data for the purpose of Question Answering is the use of Graphical Models. Some of these methods involve working over existing structured databases such as those described in Section 4.1.1. (tau Yih et al., 2014).

An interesting line of exploration is that by He et al. (2014) who use Markov Logic Networks (Richardson and Domingos, 2006) for reasoning.

4.4 Question Classification

Question classification is the process of classifying Questions based on the type of Answers that would be expected. Questions such as "Who is the Queen of England?" and "Who is the strongest man in the world?" would be classified as "Person" Questions and those such as "What is the capital of U.K?" and "Where are the Alps located?" would be classified as "Location". It is important to note that the granularity of these classes can be extremely varied.

At first glance this problem seems trivially solved by observing the question words of a question (such as "What", "Where" ...). However, this is not the case as can be seen in the following examples: "What is the name of the person who is considered to be the strongest man in the world?", "What is the location of the Alps?"

Gharehchopogh and Lotfi (2013) provide various machine learning techniques used for Question classification along with the level of accuracy achieved by each of them.

Li and Roth (2002) have achieved an extremely high level of precision (98.80% for coarse-grained and 95% for fine grained classification). Their method however, depends heavily on machine learning and a large number of features. *We have attempted several semantic methods in an attempt to achieve similar results but have failed. These experiments are described in Section 6.3*

4.5 Systems of Note

4.5.1 START

START (Katz and Levin, 1988; Katz and Lin, 2002) is an online Question Answering system developed by MIT CSAIL. We tested this system and found several shortcomings as can be seen in Figure 4.1.

4.5.2 Google Search

Google has recently started including answers to questions as part of the search results. Unfortunately, there is little literature detailing their approach apart from some patents (Masuichi et al., 2010; Todhunter et al., 2010). However, we were able to discern some basic principles behind the workings of this system from the patents and found it to be lacking due to its dependence on statistical methods. Figure 4.2 shows one such inaccuracy.

[1]ReVerb is available at: `http://reverb.cs.washington.edu/`

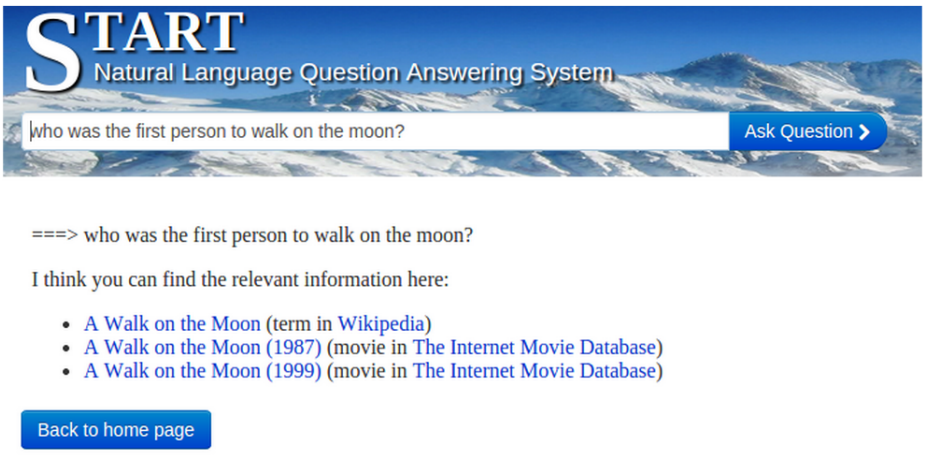


Figure 4.1: Results from the START Question Answering System

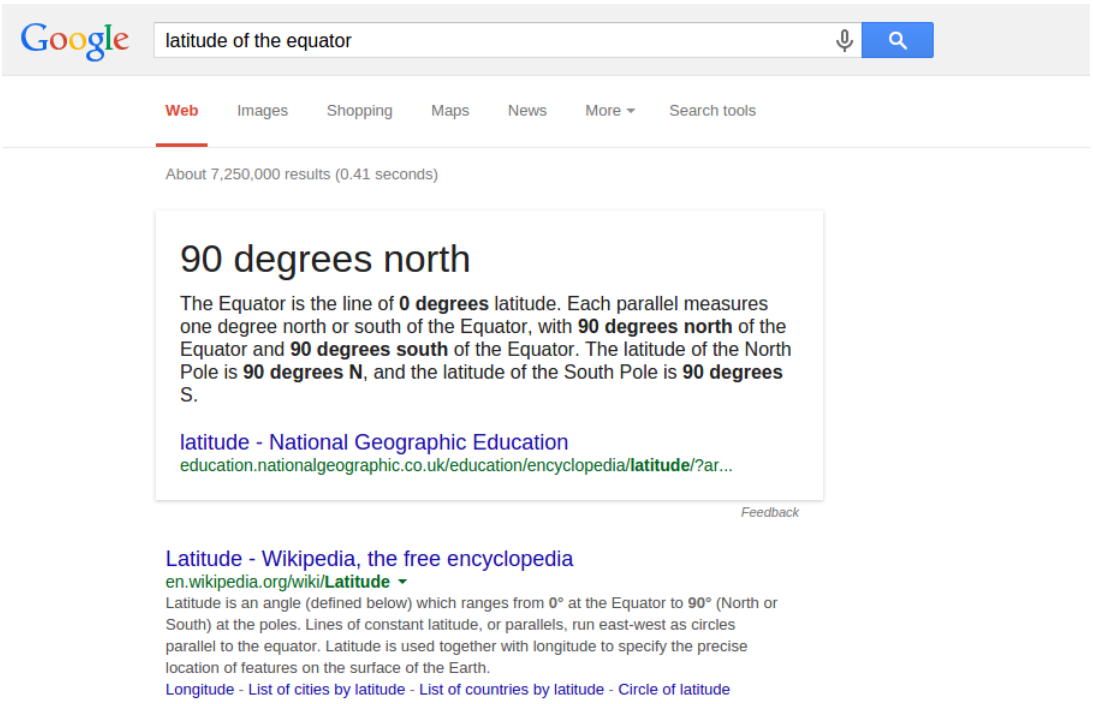


Figure 4.2: Bad Answer Results from Google Search

4.5.3 WolframAlpha

WolframAlpha[2] is is a Computational Knowledge Engine that provides factoid answers to questions in natural language. WolframAlpha has been successful in its ability to provide quality answers. However the number of domains it can answer questions in is limited. This stems from the fact that WolframAlpha extracts answers for questions from a manually created knowledge base.

[2]WolframAlpha, "Computational Knowledge Engine": www.wolframalpha.com

Chapter 5

The State of Art in Question Answering

Research into Question Answering has had a resurgence after the success of IBM Watson. Significant work has gone into Question Answering over the last couple of years and we discuss specific papers that have had a significant impact on the field along with some prior work of extreme importance to Question Answering.

5.1 Memory Networks

Weston et al. (2014) describe a new learning model that uses a combination of inference components (consisting of an existing machine learning model e.g. SVN, Neural Networks) and memory. Memory networks consist of a memory m indexed by m_i and four (potentially learned) components I, G, O and R.

- I: (Input feature map) - Converts the incoming input to the internal feature representation.
- G: (generalisation) - Updates old memories given the new input
- O: (output feature map) - produces a new output given the new input and the current memory state.
- R: (response) - converts the output into the response format desired.

The described method of using memory networks for the task of Question Answering consists of I taking sentences as input (questions and answers during training) and storing them in consecutive memory slots (G). Given a question O produces multiple sentences based on supporting memory elements. R simply returns the particular memory item retrieved although an Recursive Neural Network could be used to generate a textual output. The work also describes methods of keeping track of when a particular memory element was added to ensure that, in cases where information is dependant on time, the appropriate information is picked up.

Memory networks fit into our knowledge representation and reasoning model **link to where**. Unlike Long Short Term Memory networks (Section 3.3.3), Memory Networks do not require individual neurons to be changed thus making them more transparent. This system seems to produce results that are far superior (F1 score of 0.82) compared to (Fader et al., 2014) (F1 score of 0.54), a work we use as a benchmark.

Memory Networks have also been used to incorporate multiple structured Knowledge Bases for the purpose of Question Answering (Bordes et al., 2015). We do not explore this in greater detail as our exploration is focused on extracting Answers from free text **Why link here**.

5.2 Word Alignment

Word alignment, commonly used in machine translation, involves association of words in different sentences, either in different contexts or languages as the case may be. FAQ-based Question Answering involves the mapping of a question into existing questions in FAQs so allowing the reuse of quality

answers. Wang and Ittycheriah (2015) provide a method of achieving State of the Art FAQ matching through word alignment.

Their method involves extracting features from different questions and finding their similarity through a Neural Network. We study this method in detail as the ability to find the similarity between sentences is extremely useful in our exploration of Question Answering.

The questions (which we generalise to sentences) are represented by Q and C where $Q = q_0, q_1...q_m$ and $C = c_0, c_1...c_m$ where q_i and c_i are words in the question. The similarities between words is given by their cosine distance: $sim(q_i, c_i) = max(0, consine(v_{qi}, v_{cj}))$. Additionally the method defines the alignment position for each word (when their order is maintained) to be $align_i$ and the cosine similarity to be sim_i. Unaligned words are represented as $unalign_i$. Finally the importance of each word is taken into account through its inverse document frequency (IDF). With this background the following features are defined:

- **similarity** - $f_0 = \sum_i sim_i * \frac{idf_i}{\sum_i idf_i}$
  This feature measures the similarity based on the aligned words.
- **dispersion** - $f_1 = \sum_i (|align_i - aligni - 1 - 1|)^2$
  This feature prefers candidates that have contiguous aligned words.
- **penalty** - $f_2 = \sum_{unalign_i} \frac{idf_i}{\sum_i idf_i}$
  This feature penalises candidates based on the unaligned words.
- **5 important words** - $f_{i_{th}} = sim_{i_{th}} * idf_{i_{th}}$
  This feature contains 5 features each representing the alignment score of the i^{th} important word of which the top five are picked.
- **reverse** - The above four features are extracted by swapping the questions.

An interesting addition presented in this work is the introduction of "Sparse Features" which improves the accuracy of the system by 5%. This involves the finding of unrelated sentences that the system finds relevant and removing, from our model, features extracted from those sentences.

5.3 Surface Text Patterns

We have discussed Pattern Based Direct Mappings in Section 4.1.2. Text patterns are and important aspect of pattern based mappings. They are patterns - usually regular expressions - that represent the structure of text. Ravichandran and Hovy (2002) described an important method of learning such patterns using bootstrapping. Discovering answers is achieved by treating each sentence as a simple sequence of words and finding repeated word orderings as evidence of useful answer phrases. The following is the algorithm described for learning patterns:

1 Select an example for a given question, for example, "Mozart 1756" for BIRTHYEAR, where "Mozart" is the question term and "1756" is the answer.
2 Submit the question and answer terms as queries to a search engine, extract a large number of results, remove HTML and similar markup before then applying a sentence breaker to the resultant documents.
3 Retain only those sentences that contain both the question and the answer term.
4 Pass each retained sentence through a suffix tree constructor and retain each phrase that contains both the question and answer terms.
5 Generalise the results by replacing the question and answer terms with <NAME>and <ANSWER>.

Algorithm 2: Algorithm for learning patterns

A significant addition to this was the introduction of a precision to each pattern which is achieved through the following algorithm:

1 Select an example for a given question, for example, "Mozart 1756" for BIRTHYEAR, where "Mozart" is the question term and "1756" is the answer.
2 Submit the question and answer terms as queries to a search engine, extract a large number of results, remove HTML and similar markup before then applying a sentence breaker to the resultant documents.
3 Retain only those sentences that contain the question.
4 Calculate $Precision = C_a/C_o$ where C_a is the number of sentences that contain the answer and c_o are those that do not.
5 Additionally, remove those patterns that do not have a minimum number of examples (The authors choose 5)

Algorithm 3: Algorithm for calculating the precision of each pattern

Once the patterns and precision scores are extracted the following algorithm can be used to find answers:

1 Find the Question type of the given question and extract the question term using any existing system.
2 Submit the question term as a query to a search engine, extract a large number of results, remove HTML and similar markup before then applying a sentence breaker to the resultant documents.
3 Replace the question term in each sentence by the question tag and extract words matching the answer tag based on the patterns that we have previous extracted.
4 Sort the answers by the corresponding patterns precision scores.

Algorithm 4: Algorithm for Question Answering using Text Patterns

5.4 Syntax-based Deep Matching

Although not aimed at Question Answering, Wang et al. (2015) describe a method of matching texts using dependency trees and Deep Neural Networks. We explore this method of matching in relation to our interest in finding text similarity. The method stems from the observation that dependency tree matching can provide better correspondence between two sentences than word co-occurrences (Lu and Li, 2013).

It should be noted that we reach the same conclusion from our experiments described in Section 6.5.2.

The authors describe a method that involves the Product of the Dependency trees using the Direct Product of Graphs (PoG) described by Vishwanathan et al. (2010) where the product of two graphs $G_x = V_x, E_x$ and $G_y = V_y, E_y$ is a graph with vertices $V_{X\times Y}$ and edges $E_{X\times Y}$ given by:

$$\begin{aligned} V_{X\times Y} &= \{(v_i^X, v_{i'}^Y)\}, v_i^X \in V_X, v_{i'}^Y \in V_Y \\ E_{X\times Y} &= \{((v_i^X, v_{i'}^Y(v_j^X, v_{j'}^Y)), (v_i^X, v_j^Y) \in E_x \cap (v_{i'}^Y, v_{j'}^Y \in E_Y\} \end{aligned} \tag{5.1}$$

From Equation 5.1 it is immediately obvious that the resultant graph is extremely large. Relations are extracted from this graph - a process that we do not explore - and are used in a Deep Network which allows for the estimation of the relation between two sentences. We are specifically interested in the successful use of dependency trees and less so on the subsequent matching technique primarily because our initial attempts at establishing similarity will be based on Dependency Pattern Extraction described in Section **Insert Section Here**

5.5 Mining Linked Data

We have explored the use of Graphical Representation of Data for Question Answering and Matching over Existing Structured Databases in Sections 4.3 and 4.1.1 respectively. Here we explore some of the papers that are more relevant to our work in greater detail.

There has been a significant amount of work using Freebase (Bollacker et al., 2008) which contains over 2.9 billion triples. Unfortunately, Google is currently in the process of shutting down Freebase (Google+, 2015), a significant blow to research in this area especially because of the number of methods that depend directly on Freebase.

A majority of the work based on Freebase is aimed at converting Natural Language Questions to SPARQL queries. Bast and Haussmann (2015) describe a method of improving Question Answering accuracy by use of Freebase. Significant contributions of this work include the integration of entity recognition and a method of learning to rank pair-wise comparisons of query candidates. The system consists of four core tasks: Entity identification, Template matching, relation matching and Ranking. Entity identification is achieved by matching elements of the Question with elements within the KB which is then used to find relevant templates from an existing list of templates. At this point, the template representing the Query still does not contain the information regarding the specific information mentioned in this question. This is achieved through a combination of methods including word relations and supervised learning. Finally the results are ranked based on relations in the KB. These elements are of relevance to our work as we intend to enrich our Dependency Patterns with information from KBs (Section 7.3.1.

The Conference and Labs of the Evaluation Forum (CLEF) has a evaluation campaign for Question Answering over Linked Data (QALD) and a large number of participants in QALD-5 (2015) showcased methods that focus on creating appropriate SPARQL queries.

5.5.1 ISOFT

The system submitted to QALD-5 by Park et al. (2015) details the system "ISOFT" that consists of three elements: Answer Clue entity identification by use of Explicit Semantic Analysis (Egozi et al., 2011), Answer clue sentence identification and mapping this information to relevant entities on DBpedia. Explicit Semantic Analysis is the vectorial representation of text, wherein each word is represented as a column with individual entities representing its tf-idf value associated with each document. Most commonly, the corpus used is Wikipedia.

The authors also describe a methods of identifying possible Answer Clue entities which is an extension of their earlier work (Park et al., 2014). The method described relies on repeated concatenation of the prepositional or predicate phrases and reduction policies based on the dependency graph. We decide not to move away from our current method of finding Answer Clue entities (Described in Section 7.3.1).

Our experiments on Wikification have shown that simply searching Wikipedia provides performance similar to, if not better than Explicit Semantic Analysis. We discuss the possible reason for this in Section 6.8

5.5.2 SemGraphQA

SemGraphQA (Beaumont et al., 2015) uses a method that consist of Entity identification (Answer Clue entity identification) by use of DBpedia Spotlight (Daiber et al., 2013), class identification for identification of questions that require multiple entities or classes as answers and Relation identification. As we intend to achieve Entity Identification through Wikification (Section 6.8) and we deal only with factoid questions we focus our attention on Relation identification.

Relation identification in SemGraphQA is improved by use of variations of text so as to increase recall. This is achieved by use of a lexicon with variants acquired from WordNet - a method we adopt for extending dependency tree networks (Section 7.3.2). The overall structure of SemGraphQA is similar to our first attempt at using Frames (Section **section:framebasedqa**) and the low F1 score of this system (0.31 as opposed to .73 achieved by Xser - Section 5.5.4) confirms our belief that moving away from

Frames is a move in the right direction (More on why we move towards Dependency Tree Networks in Section 7.1).

5.5.3 HAWK

HAWK (Usbeck et al., 2015)[1], yet another system evaluated at CLEF-5, also uses dependency trees and achieves a much higher F1 Score of 0.61. HAWK performs entity annotation (or Answer Clue Entity Identification) using a combination of DBpedia Spotlight, Wikipedia Miner (Milne and Witten, 2008) [2], TagMe 2 (Ferragina and Scaiella, 2010) [3] and FOX (Speck and Ngonga Ngomo, 2014) [4]. We are currently in the process of testing each of these systems against our Wikification system (Section 6.8) so as to decide on which to use.

HAWK performs Noun phrase identification using the dependency parse of a sentence. The Noun Phrase identification algorithm used in HAWK is superior to the one developed by us and we are in the process of exploring ways in which it can be adapted to our needs. Finally HAWK prunes noisy nodes from the predicate-argument tree so as to narrow in the the elements that are of most interest. This is a method we intend to adapt for pruning our extend graphs as described in Section 7.3.2. We do not discuss other elements of the system that pertain to the generation of SPARQL queries.

5.5.4 Xser

Unfortunately, details of the Xser system used in QALD-5 are yet to be published. We instead, study the QALD-4 (2014) entry of Xser (Xu et al.). Xser uses several methods that are extremely relevant to us. Like in other systems Xser has a phase detection step, however, unlike in other systems, Xser is also assigned a semantic label $l \in \{entity, relation, category, variable\}$. Xser achieves this by use of a structured perceptron (Collins, 2002) which is a perceptron for structured prediction. We have previously discussed Structured prediction in Section 3.3.4.

Xser then creates and parses a phrase DAG based on the entities previously identified. This uses the framework proposed by Sagae and Tsujii (2008) to find the specific phrase DAG that best represents the question. We do not study this aspect of the system in detail as we prune dependency trees using a different approach (described in Section 7.3.2).

Xser also uses PATTY (Nakashole et al., 2012) to construct a lexicon that maps phrases to predicates and categories in DBpedia.

5.6 Learned Inference Rules

Lao et al. (2012) provide a method of using the Path Ranking Algorithm (Lao and Cohen, 2010) to combine the dependency tree of a question with elements of a Knowledge Base. This process involves the linking of elements in the parse tree with corresponding elements in a KB and using the combined graph to generate path types. Given a KB consisting of concepts C and a set R of labels, each label r denotes some binary relation in the KB. The KB is a directed, edge-labelled graph $G = (C, T)$ where $T \subset C \times R \times C$ or the triple (c, r, c'). Each such triple represents a relation $r(c, c')$ where $r \in R$. A path type in G is a sequence $\pi = \langle r_1, ..., r_m \rangle$ and an instance of the path type is a sequence of nodes $c_0, ..., c_m$. Each path type $\pi = \langle r_1, ..., r_m \rangle$ is a real-valued feature and for a a query-answer node pair (s, t), the value of π is $P(s \to t; \pi)$, the probability of reaching t from s by a random walk that includes the type. A path type π is considered active for a pair (s, t) if $P(s \to t; \pi) > 0$.

During training, for each relation r we start with a set of node pairs $S_r = s_i, t_i$ and create the training set $D_r = (x_i, y_i)$ where $x_i = \langle P(s_i \to t_i; \pi) \rangle_{\pi \in B}$, is the vector of path feature values for the pair (s_i, t_i) where y_i indicates whether $r(s_i, t_i)$ holds and B is the subset of path types that are picked based on frequency. This is used to train a Logistic Regression model, where we estimate parameters θ for a training set D by maximising the objective given by Equation 5.2.

[1] Code available at: http://aksw.org/Projects/HAWK.html

[2] Code available at: http://wikipedia-miner.cms.waikato.ac.nz/demos/

[3] API available at: http://tagme.di.unipi.it/

[4] Code available at: https://github.com/AKSW/FOX

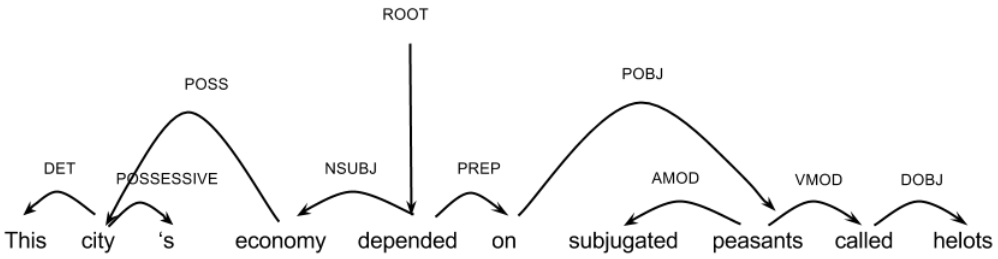


Figure 5.1: Dependency Parse of a sentence used as an example in QANTA

$$\mathcal{F}(\theta) = \frac{1}{|D|} \sum_{(x,y)\in D} f(x, y; \theta) - \lambda_1 ||\theta||_1 - \lambda_2 ||\theta||_2^2 \tag{5.2}$$

where λ_1 and λ_2 control the strength of the L_1-regularisation which helps with structure selection and L_2^2-regularisation which prevents over-fitting. The log-likelihood $f(x, y; \theta)$ of example (x, y) is given by Equation 5.3

$$\begin{aligned} f(x, y, \theta) &= y \ln p(x, \theta) + (1 - y) \ln(1 - p(x, \theta)) \\ p(x, \theta) &= \frac{exp(\theta^T x)}{1 + exp(\theta^T x)} \end{aligned} \tag{5.3}$$

Once this model is trained for each relation r, it can be used to generate new instances of the that relation with high precision. Section 7.4 describes methods we propose for incorporating this method into our work.

5.7 QANTA

Iyyer et al. (2014) describe a Question Answering Neural Network with trans-sentential averaging (QANTA)[5]. Their work makes use of Dependency-Tree Recursive Networks (DT-RNN) to answer Quiz Bowl questions and relies heavily on the fact that there are often several Quiz Bowl questions with the same factoid answer. This redundancy is required because RNNs require many redundant training examples to learning meaningful representations.

5.7.1 DT-RNN Model

DT-RNN is of great significance to our work as it provides a way to represent both the semantic and syntactic features of a sentence into an RNN. This warrants an in-depth analysis of QANTA which we provide in this section.

QANTA takes dependency trees of question sentences and their corresponding answers as input. Each word w in the vocabulary is represented by a vector $x_w \in \mathbb{R}^d$. This method of words mapped to vectors is achieved through the dimentionality reduction of the word co-occurrence matrix. These vectors x_w are stored in a $x \times V$ dimensional matrix W_e where V is the size of our vocabulary. Each node in the parse tree is associated with three elements. First it the word, in the case of a leaf or sub-tree phrase in the case of a node. The second is the vector x_w and finally a hidden vector $h_n \in \mathbb{R}^d$. The DT-RNN recursively combines the current nodes word vector with its childrens hidden vector h_n. A $d \times\ d$ matrix, W_v is introduced to incorporate word vector x_w at a node into the node vector h_n. Finally, a second $d \times d$ matrix W_r is used to store the weights of the relations between words.

The first step is to compute leaf representations of h_n. Given a parse tree (Figure 5.1) the hidden representation of h_{helots} is:

[5]Code available at: https://cs.umd.edu/ miyyer/qblearn/

$$h_{helots} = f(W_v.x_{helots} + b) \tag{5.4}$$

where f is a non-linear activation function such as the sigmoid function (Section **REF**) and b is a bias term. Once the h_n values for the leaves are calculated, each of their parents can then be processed, as an example consider the word "called":

$$h_{called} = f(W_{DOBJ}.h_{helots} + W_v.x_{called} + b) \tag{5.5}$$

It should be noted that W_{DOBJ} is retrieved from W_r and W_v from W_e.

Unlike previous work (Socher et al., 2011), QANTA trains both the question and the answer in a single model. The intuition is to attempt to create vector representations wherein the vectors of question sentences are near their answers and far from the incorrect answers. For training the RNN the gradient and error need to be defined. Given a sentence with its correct answer c, and j randomly selected incorrect answers denoted by the subset Z we note that since both c and z are part of the vocabulary $x_c \in W_e$ and $x_z \in W_e$. S is defined to be the set of all nodes in the sentence's dependency tree, where an individual node $s \in S$ is associated with the hidden vector h. The error for this sentence is:

$$C(S, \theta) = \sum_{s \in S} \sum_{z \in Z} L(rank(c, s, Z)) max(0, (1 - x_c.h_s + x_z.h_z)) \tag{5.6}$$

where the function $rank(c, s, Z)$ provides the rank of the correct answer c with respect to the incorrect answers Z. This rank is transformed into a loss function using: $L(r) = \sum_{i=1}^{r} 1/i$. The final model minimises the sum of error over all sentences T normalized by the number of nodes N in the training set:

$$J(\theta) = \frac{1}{n} \sum_{t \in T} C(t, \theta) \tag{5.7}$$

where the parameters $\theta = (W_{r \in R}, W_v, V_e, b)$ and R represents all dependency relations in the data. The gradient of the objective function, which we calculate using Backpropagation through structure (Section 3.3.4) is:

$$\frac{\partial C}{\partial \theta} = \frac{1}{N} \sum_{t \in T} \frac{\partial J(t)}{\partial \theta} \tag{5.8}$$

5.8 Machine Comprehension

Recent work in the field of machine comprehension by Hermann et al. (2015) provides an insight into one possible method of extracting a large corpus of question answer pairs, a critical requirement for large machine learning based question answering system such as ours. Their method involves the use of the bullet points the CNN and The Daily Mail provide along side each of their articles. Figure 5.2[6] shows a news story from CNN with the associated, manually added, summary.

This provides us with an important way of extracting question answer pairs. The bullet point "North Korea threatens military action if South doesn't turn off loudspeakers at border" can easily be converted to the question "What does North Korea threaten if South doesn't turn off loudspeakers at border?". This provides us with a question and text that contains the answer with the important distinction that the question does not contain sentences from the text.

Their work goes on to describe a method of using Long Short Term Memory Networks (Section 3.3.3) combined with an attention mechanism inspired by work in translation and image detection, which consists of two kinds of text processors: "The Attentive Reader" and "The Impatient Reader". Due to the recency of this work, we are yet to study these methods in detail.

[6]CNN, Retrieved 21st August 2015: www.edition.cnn.com

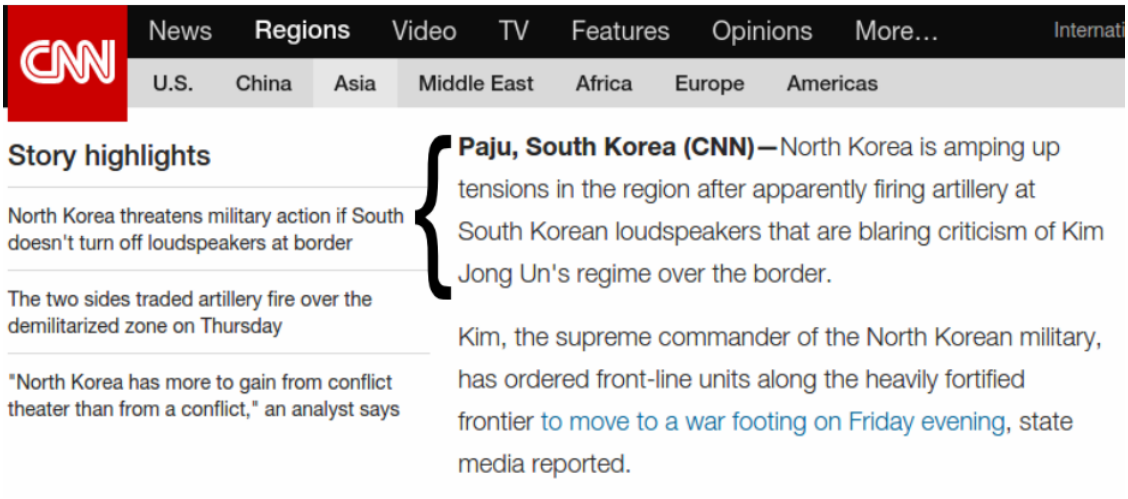


Figure 5.2: Manually added Summary with the corresponding segment from a News Story.

5.9 New avenues for extracting Question Answer pairs

In Section 5.8 we discussed a method of automatically extracting quality question answer pairs proposed by Hermann et al. (2015). In this section we discuss the need to find new avenues of extracting similar data and explore one such possibility.

News, in general, is structured in a way that is commonly called the "Inverted Pyramid". This structure ensures that the most newsworthy information is covered at the start of a news story followed by supporting details and finally any background information - Figure 5.3[7] provides a visualisation of this. It ensures that the most relevant information is at the top while information of diminishing importance is further down the story. From a news reporting perspective, this is extremely useful. It ensures that those of us who choose to read just a little of each article can gather the essence of it and choose to continue to read the article based on how interested we might be.

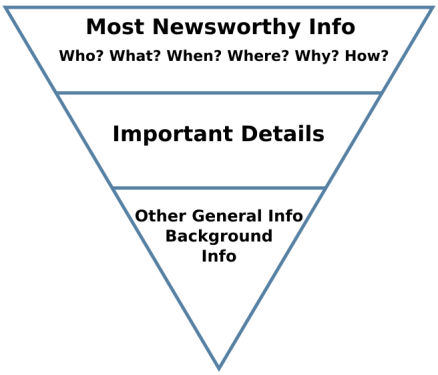


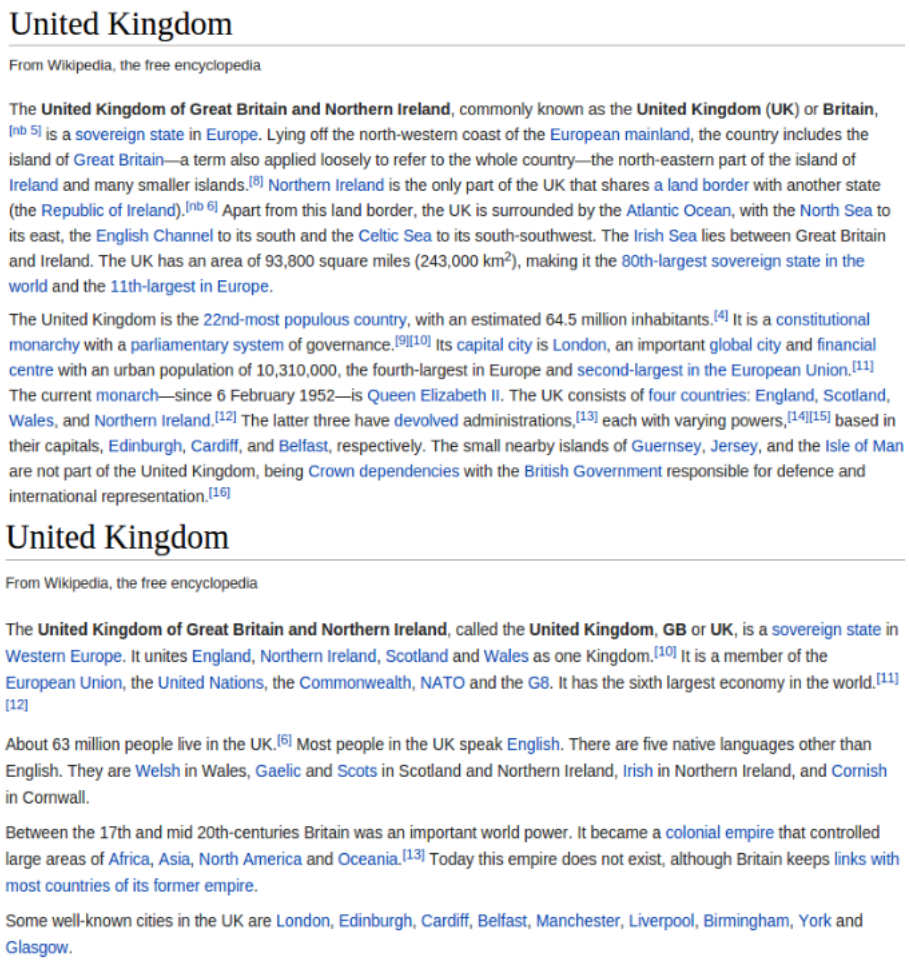
Figure 5.3: The Inverted Pyramid Method of News Reporting.

Despite it's usefulness as a reporting tool, the Inverted Pyramid could potentially skew the way in which our system interprets text within an article. To avoid this, we also use Wikipedia to extract similar information. Simple Wikipedia is an alternative version of Wikipedia wherein the articles are written in Basic English (Ogden, 1932) and Special English which is used by the broadcaster Voice of America.

We employ a method similar to the one described by Hermann et al. (2015) but replace the summary with a sentence from Simple Wikipedia and use the standard Wikipedia as a source to find the answer in.

[7]The Air Force Departmental Publishing Office (AFDPO) derivative work: Makeemlighter. Public domain, via Wikimedia Commons

Figure 5.4[8] provides an comparison of the same article across the two.



United Kingdom

From Wikipedia, the free encyclopedia

The **United Kingdom of Great Britain and Northern Ireland**, commonly known as the **United Kingdom** (**UK**) or **Britain**, [nb 5] is a sovereign state in Europe. Lying off the north-western coast of the European mainland, the country includes the island of Great Britain—a term also applied loosely to refer to the whole country—the north-eastern part of the island of Ireland and many smaller islands.[8] Northern Ireland is the only part of the UK that shares a land border with another state (the Republic of Ireland).[nb 6] Apart from this land border, the UK is surrounded by the Atlantic Ocean, with the North Sea to its east, the English Channel to its south and the Celtic Sea to its south-southwest. The Irish Sea lies between Great Britain and Ireland. The UK has an area of 93,800 square miles (243,000 km^2), making it the 80th-largest sovereign state in the world and the 11th-largest in Europe.

The United Kingdom is the 22nd-most populous country, with an estimated 64.5 million inhabitants.[4] It is a constitutional monarchy with a parliamentary system of governance.[9][10] Its capital city is London, an important global city and financial centre with an urban population of 10,310,000, the fourth-largest in Europe and second-largest in the European Union.[11] The current monarch—since 6 February 1952—is Queen Elizabeth II. The UK consists of four countries: England, Scotland, Wales, and Northern Ireland.[12] The latter three have devolved administrations,[13] each with varying powers,[14][15] based in their capitals, Edinburgh, Cardiff, and Belfast, respectively. The small nearby islands of Guernsey, Jersey, and the Isle of Man are not part of the United Kingdom, being Crown dependencies with the British Government responsible for defence and international representation.[16]

United Kingdom

From Wikipedia, the free encyclopedia

The **United Kingdom of Great Britain and Northern Ireland**, called the **United Kingdom**, **GB** or **UK**, is a sovereign state in Western Europe. It unites England, Northern Ireland, Scotland and Wales as one Kingdom.[10] It is a member of the European Union, the United Nations, the Commonwealth, NATO and the G8. It has the sixth largest economy in the world.[11] [12]

About 63 million people live in the UK.[6] Most people in the UK speak English. There are five native languages other than English. They are Welsh in Wales, Gaelic and Scots in Scotland and Northern Ireland, Irish in Northern Ireland, and Cornish in Cornwall.

Between the 17th and mid 20th-centuries Britain was an important world power. It became a colonial empire that controlled large areas of Africa, Asia, North America and Oceania.[13] Today this empire does not exist, although Britain keeps links with most countries of its former empire.

Some well-known cities in the UK are London, Edinburgh, Cardiff, Belfast, Manchester, Liverpool, Birmingham, York and Glasgow.

Figure 5.4: A comparison of the Article on "United Kingdom" in Simple English Wikipedia and Wikipedia

It should immediately be noted that this method could be more prone to errors than when dealing with CNN as can be seen by the different values of population reported in the two articles (63 million on the Simple Wikipedia as opposed to 64.5 million on Wikipedia). We are in the process of finding ways of getting around this. However, the large number of sentences now available (every sentence in the Simple Wikipedia article as opposed to the summaries provided previously) makes this process easier.

We intend to use work on Generating Questions by Mazidi and Nielsen (2014) for converting summaries and sentences into Questions.

5.9.1 Voice of America

The Voice of America is the office broadcasting institution of the United States and provides news in Special English. As discussed earlier, this provides us with a large amount of text in a format that is potentially easier to parse. Our initial exploration of this has shown us that there might be ways of extracting sentence structures from these news stories, however, we are yet to identify methods of extracting question answer pairs.

5.10 Shortcomings of Existing Systems

We have studied several attempts at Question Answering in this and the previous Chapter. The majority of the systems that perform well do so in the limited setting of either a single domain, over a Knowledge

[8] Wikipedia.org, Retrieved 21st August 2015: en.wikipedia.org and simple.wikipedia.org

Base or within specific kinds of texts. In subsequent Sections (6.5.2, 7.3.2 and 7.1.3) we show the need to include the complete Dependency Trees of sentences and to avoid the use of systems that require sequential use of multiple steps. The significantly better performance of learning algorithms across the domain of Natural Language Processing has shown that it is important to incorporate learning paradigms into Question Answering Systems.

Although recent work in this field, such as that by Iyyer et al. (2014) (Section 5.7), have managed to avoid the use of multiple sequential tasks while include an element of learning they remain limited to a particular kind of text. Our work is aimed at extending these systems so as to move beyond these limitations.

Chapter 6

Experimental Results

We performed some experiments to validate the direction of our work. We also studied several other systems, which have been listed in Appendix B.

6.1 Existing Structured Datasets

Our first experiments involved testing the Recall of existing structured datasets. Although we tested several of these we found that the Recall with respect to the information we required for Answering Questions was very low. The databases we tested include: ConceptNet (Liu and Singh, 2004), DBPedia (Auer et al., 2007) and OpenCyc (Fensel et al., 2008).

6.2 Using Search Engines for Question Answering

Our first attempt at creating a basic Question Answering system was based on AskMSR (Brill et al., 2002). AskMSR extracts keywords from questions, uses a search engine to find pages that might contain results, from which potential answers are extracted.

We attempt to replicate this system with the one significant change: We use a semantic method of extracting search phrases instead of query rewrite rules. We use the Stanford Core NLP Processor (Manning et al., 2014) to find specific sections of the question that are most useful for use in a search phrase. We describe below the algorithm tested using the question "What was the monetary value of the Nobel Peace Prize in 1989?". Figure 6.1 is a representation of the parse tree for this question. We then extract search phrases using the following Grammar:

NP: {<JJ>+<NN|NNP>+}
NP: {<NN|NNP>+}
NP: {<CD>+}

This method extracts the following search phrase: "monetary value Nobel Peace Prize 1989". We use the Bing API to extract results for this particular query. The processes of extracting relevant sections of web pages that contain the given search term is an engineering problem we address through statistical methods. We identify the following as sections of web pages that potentially contain the answer to the given question:

Top Result

On December 10th, 1989, when the Dahai Lama accepted the Peace Prize the rate of exchange was it was $1.00 US to 6.29 Swedish Kronars which means the prize was valued at that time to $476947.
What is the monetary value of the nobel prize when dalai lama .
[How can Dalai Lama get a Nobel peace .
[Why Dalai Lama got the Nobel peace .
What is the monetary value of the nobel prize when dalai lama got .
Best Answer: .The 14th Dalai Lama won the Nobel Prize in 1989.
The Nobel Prize for that year was 3,000,000 Swedish Kronor.

[Chinese, how do you feel that the only Chinese to get a Nobel Prize for promoting Peace is HH Dalai Lama?]

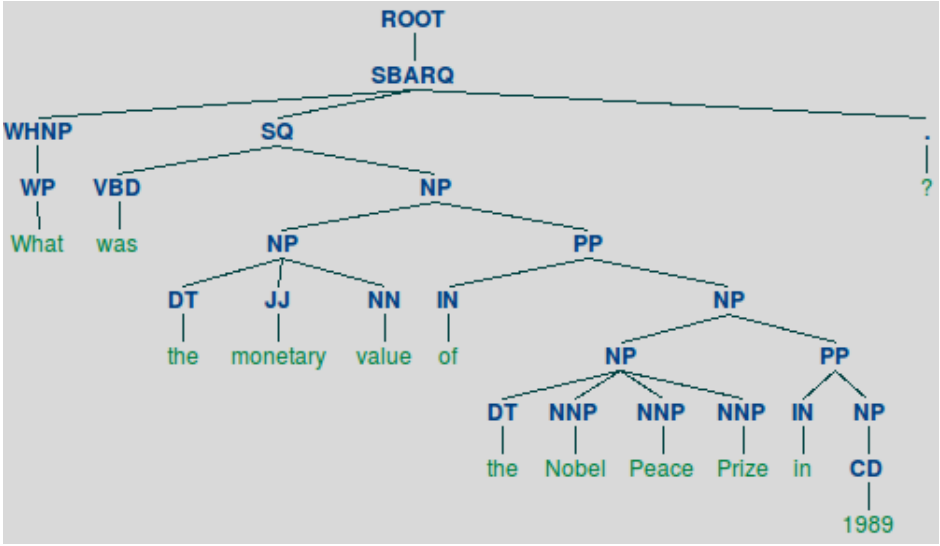


Figure 6.1: Parse tree for : "What was the monetary value of the Nobel Peace Prize in 1989?"

Second Result

UNKNOWN What was the monetary value of the Nobel Peace Prize in 1989?
However, due to the fact that no one sells them and that no one would buy them, the monetary value is quite useless.
UNKNOWN What was the monetary value of the Nobel Peace Prize .

The monetary value of the Nobel Peace Prize is estimated to be $500,000.

6.2.1 Future Work

These results look promising and our subsequent analysis (Section 7.4) shows how we intend to make use of Search Engines as an element of our Question Answering System. Although the first step in developing our system will not require this method as it involves answering questions from within a single document (Section 7.3.2), we intend to eventually extend this method with the ability to discover relevant sentences from multiple documents.

6.3 Question Classification

Our next experiment was an attempt to improve Question Classification (Section 4.4) by use of semantic structures. We use the same parsing techniques developed in the previous section. Our work attempts to extend the work of Li and Roth (2002).

We work with the assumption that the structure of the sentence can be used to better classify questions. We shift our focus from parse trees to dependency relations of a sentence. Based on extensive analysis we make the following empirical observations:

Dependencies related to those words that make a question a question (such as "what", "Where" ...) are of most interest. If there are no such words (such as in the question "Name the largest city in the U.K.", then it is the "Root" of the sentence that is of most interest.

When words are related through one of det, prep_of, prep_as or amod, it is better to consider the subsequent dependency. For example, when parsing the question "What time of day did Emperor Hirohito die ?" we encounter the dependency ['det', 'time', 'What'], which is of less interest than the dependency linked to it, which is: ['prep_of','time','day']. We call this process of finding more interesting dependencies *Dependency Bubbling*.

Despite several attempts we were unable to reach the level of accuracy achieved by Li and Roth (2002). However the techniques developed during these experiments have been invaluable in subsequent explorations.

6.4 Word Sense Disambiguation

Word Sense Disambiguation (WSD) (Previously discussed in Section 2.1.3) was first introduced as a problem in Computer Science in 1949 (Weaver, 1949/1955). Despite this, it is still an unsolved problem. Despite the difficulties in addressing a problem that has been attempted by several researchers over decades, we note that this is the single biggest obstacle to truly "Understanding" Natural Language and attempt to find a reasonable solutions to this problem.

We address the problem of WSD by attempting to link words in a sentence to corresponding senses in WordNet (Miller, 1995) called Synsets. We pick WordNet due to the detailed internal structure it provides - *being able to link words to Synsets will provide researchers with the large amount of data available on WordNet.*

6.4.1 Data sets

We use two large data sets to train and test our WSD system. The first is the data provided at SemEval 2007 (Task 07) for the task of Coarse-Grained WSD (Navigli et al., 2007). The second is the SemCor corpus (Miller et al., 1993) adapted to WordNet 3. Although we use SemCor for training, all our experiments are performed on the SemEval Coarse-Grained data.

6.4.2 Implementation and Evaluation of Existing Semantic Methods of WordNet WSD

We (heavily) modify pywsd (Tan, 2014) to test existing some semantic, but old, methods of WSD. The results observed are listed in Table 6.1.

Sr. No	Method	Accuracy
1	Simple Lesk (Lesk, 1986)	49.69%
2	Simple Lesk with Stemming	58.58%
3	Adapted Lesk (Banerjee and Pedersen, 2002)	49.69%
4	Adapted Lesk with Stemming	57.97%
5	Cosine Lesk (Basile et al., 2014)	52.14%
6	path Similarity (Wu and Palmer, 1994)	65.95%
7	lch Similarity (Leacock and Chodorow, 1998)	54.90%
8	wup Similarity (Wu and Palmer, 1994)	66.25%
9	res Similarity (Resnik, 1995)	56.74%
10	jcn Similarity (Jiang and Conrath, 1997)	54.90%

Table 6.1: Accuracy of Semantic WSD Methods

6.4.3 Semantic Word Sense Disambiguation using Word Vectors

There are two significant contributions of our work. The first is the use of semantic parsing and *Dependency Bubbling* (Section 6.3) to find elements of a sentence that are of more importance than the rest. The second is the use of Word Vectors (Mikolov et al., 2013a,b,c) as a similarity measure.

We use these methods to adapt the Lesk Algorithm (Lesk, 1986) for WSD. We perform Dependency Bubbling using the dependency parse generated by the Stanford Core NLP package[1] and the Google implementation of Word Vectors[2]. We use the pre-trained Google database trained over about *100 billion words* through the package gensim (Řehůřek and Sojka, 2010). We list the results of these experiments in Table 6.2.

[1] Available at: `http://nlp.stanford.edu/software/CRF-NER.shtml`
[2] Available at: `https://code.google.com/p/word2vec/`

Sr. No	Method	Accuracy
1	word2vec	69.5%
2	word2vec + Dependency Bubbling on target sentences	67.8%
3	word2vec + Dependency Bubbling on context and target sentences	72.5%
4	word2vec + Dependency Bubbling and TF-IDF on target sentences	68.8%
5	Ignoring the name of the Synset	67.8%

Table 6.2: Comparison of Results from Various Experiments.

6.4.4 State of the Art in Coarse-Grained Synset Disambiguation

There seems to have been little work in the way of Coarse-grained Synset Disambiguation after 2007. Table 6.3 compares our results of those systems that participated in the SemEval-2007 Task 07 (Navigli et al., 2007).

System	A	P	R	F1
NUS-PT	100.0	82.50	82.50	82.50
NUS-ML	100.0	81.58	81.58	81.58
LCC-WSD	100.0	81.45	81.45	81.45
GPLSI	100.0	79.55	79.55	79.55
BLMF S	100.0	78.89	78.89	78.89
UPV-WSD	100.0	78.63	78.63	78.63
Our System	**100.0**	**72.5**	**72.5**	**72.5**
TKB-UO	100.0	70.21	70.21	70.21
PU-BCD	90.1	69.72	62.80	66.08
RACAI-SYNWSD	100.0	65.71	65.71	65.71
SUSSX-FR	72.8	71.73	52.23	60.44
USYD	95.3	58.79	56.02	57.37
UOFL	92.7	52.59	48.74	50.60
SUSSX-C-WD	72.8	54.54	39.71	45.96
SUSSX-CR	72.8	54.30	39.53	45.75
UOR-SSI	100.0	83.21	83.21	83.21

Table 6.3: Comparison of our system with the participants of SemEval-2007 Task 07. A : Attempted, P : Precision, R : Recall and F1 : the F1 Score.

6.4.5 ANN for improved accuracy

Given these encouraging results, the use of Artificial Neural Networks to improve accuracy seemed to be the obvious next step. The power of ANNs stems from their ability to capture relations between features that would otherwise have to be programmed in through observation. Given their automated nature, they have the additional advantage of capturing extremely complex relations that are often impossible to observe.

Our first attempt is to incorporate all elements described in Section 6.4.3 into the features of an ANN. We use Fast Artificial Neural Network Library (FANN) Nissen (2003) to implement our Neural Network. After generation of features, each Word Sense has, associated with it, a set of features and an output bit that is either 1 or -1 depending on if it is the correct sense or not.

Table 6.4 shows that the best accuracy value achieved is 67.41%, lower than the 72.5% achieved by

Connection Rate	Learning Rate	Input Features	Hidden Layer 1	Hidden Layer 2	Output	Min Error	Accuracy
1.00	0.70	61	100	N/A	1	0.00300	65.16
1.00	1.00	61	100	N/A	1	0.00290	62.92
1.00	1.00	61	100	N/A	1	0.23456	55.05
1.00	0.10	61	100	N/A	1	0.00103	67.41
0.80	0.10	61	100	N/A	1	0.01185	62.92
1.00	0.10	61	10	N/A	1	0.00991	59.55
1.00	0.10	61	500	N/A	1	0.00991	66.29
1.00	0.10	61	100	50	1	0.06431	67.41

Table 6.4: Accuracy values for different types of ANNs tested.

our system without ANNs. To better understand why this is the case we plot the Train Error vs the Cross Validation Error (Figure: 6.2), a common way of testing the Bias and Variance of a model. A high variance shows that our model is over-fitting the train data whereas a high bias shows that our model is not complex enough to represent the function that expresses the training data. It can be seen from Figure 6.2 that neither of this is the case. Additionally we also know that more data will not help because the Train Error and Cross Reference Errors meet.

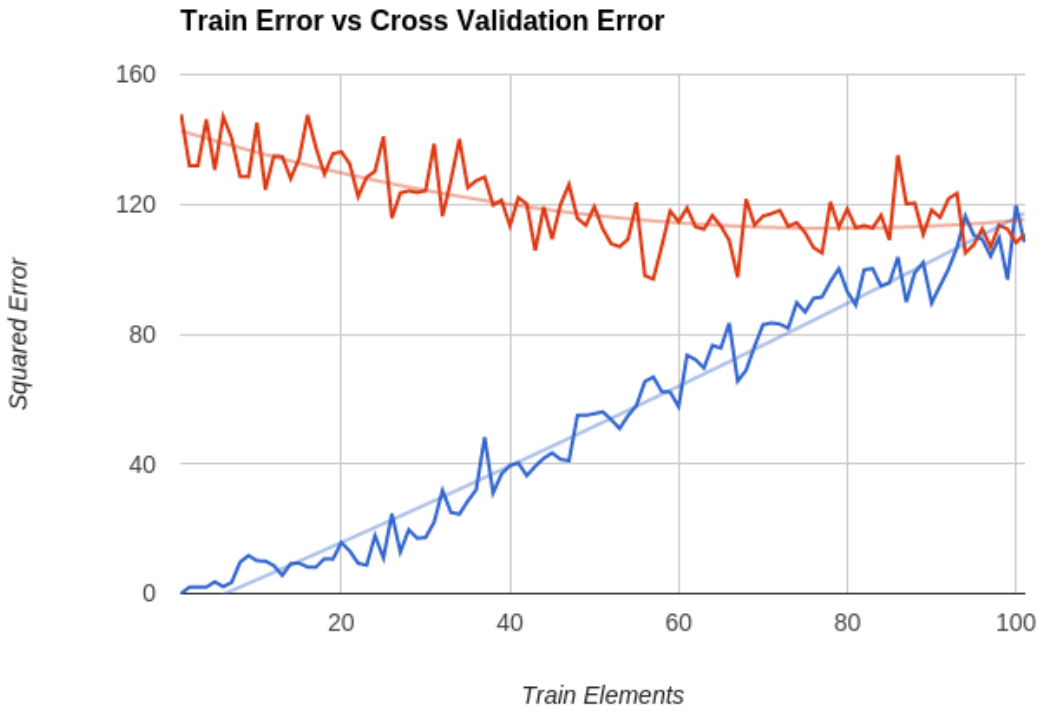


Figure 6.2: Train Error vs Cross Reference Error of ANN based WSD

Possible Causes for failure

In our model, we use a single output Neural Network that classifies each possible sense into a Yes or a No category representing the possibility that that word sense is the correct sense based on the features of the input word. We run the same model on each of the possible Word Senses and pick that Sense that was assigned the highest confidence by the model. However, since we want to classification each word into

multiple senses each of which are independent and binary, we can use multinomial classification which is provided by Artificial Neural Networks with multiple outputs.

6.4.6 Multiple Output Artificial Neural Networks

Multinomial classification has the advantage of being able to utilise cross-entropy error that allows us to simultaneously learn the relations between output classes during training.

Unfortunately, the Multiple output ANN does not do much better than the Single output ANN. As in the case of single output ANNs we tested the Bias and Variance of the system and although these parameters were fine tuned, we reached a similar conclusion.

Possible Causes for failure

So as to capture all information we also include the "position" of each word sense. This position reflects the frequency with which each sense is used.

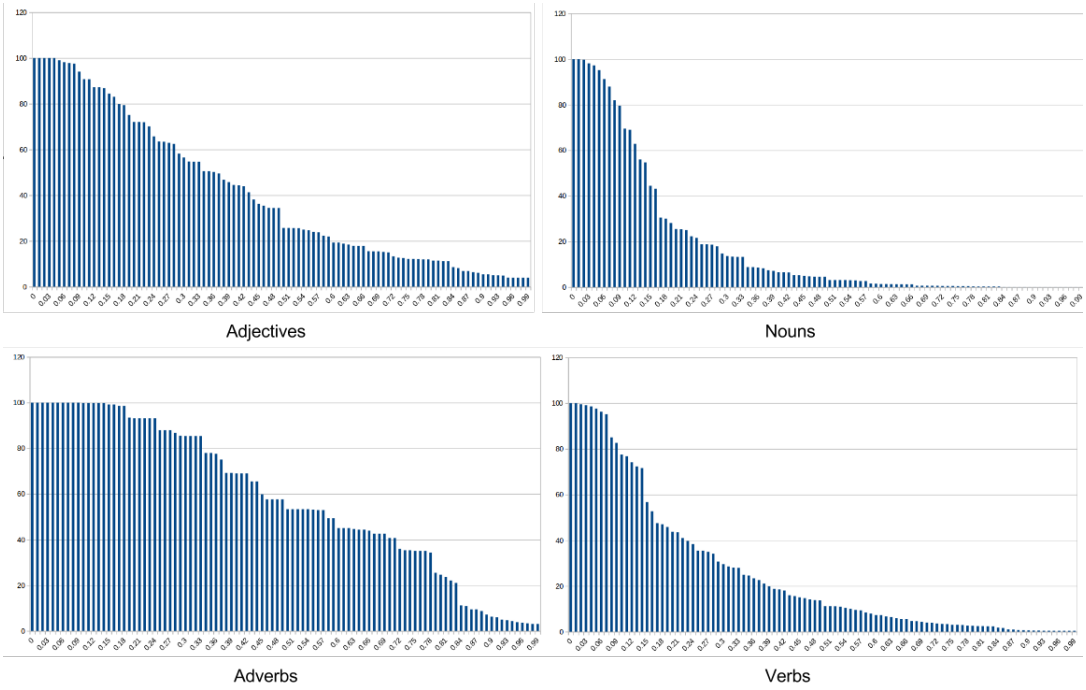


Figure 6.3: The Frequency of use of a Word Sense plotted against the Frequency Position of a Word Sense

Figure 6.3 represent the frequency with which each position is the correct sense in the SemCor corpus (Fellbaum et al., 1997). It is immediately obvious that the first couple of frequently occurring senses do so with an extremely high frequency. This is further exemplified by the fact that picking the most common sense provides an accuracy of over 57%. This skewing of data causes the Neural Network to vastly increase the importance of the position variable thereby limiting the extent to which our model can improve.

6.5 Verb Sense Disambiguation

In an attempt to improve the accuracy of our Word Sense Disambiguation system, we first analyse the accuracy of our system on each of the different Senses. Table 6.5 shows the accuracy achieved by our system broken up by different Word Senses along with the frequency with which each of those word senses occur in SemCor.

An analysis of Table 6.5 shows that an increase in accuracy of Verb Senses will have the largest positive impact on our system.

Word Sense	Accuracy (%)	% of Total Words
Adverb	86.66	12.20
Adjective	47.05	13.82
Noun	80.00	44.72
Verb	75.00	29.27

Table 6.5: Accuracy achieved by our method for each Word Sense along with that Sense's frequency.

We also note that, so far, we have used vector similarities to measure similarity between a word in a particular context and each of the possible senses of that word - a method that, although extremely powerful for nouns, might not be useful in the case of Verb Sense Disambiguation.

6.5.1 Verb Frames

WordNet provides standard structures that each verb, when used in a particular sense, can be used in. For example, WordNet lists the following to be the verb frames for the word "run" when used in the form "to move fast using one's feet": *['Something run', 'Somebody run', 'Somebody run PP']*. It should be noted that frames do not uniquely identify the sense of that word. The same word "run", when used in the sense "the story or argument runs as follows" also has the frame *['Something run']*. However, we start with the hypothesis that the additional information that can be extracted from frames can be used, in conjuncture, with other information, to improve Verb Sense Disambiguation.

To achieve this we attempt to find ways in which we can find the frame in which a particular word is being used, given the sentence that it occurs in.

It is important to note the distinction between Verb Frames and Frames used for Natural Language representation such as Minsky Frames (Section 2.3.3).

6.5.2 Dependency Bubbling

Our first attempt at discovering frames is based on Dependency Bubbling (Section 6.3). Although Dependency Bubbling works well for simple sentences, we find that it fails for more complex sentence structures and the algorithmic complexity of improving Dependency Bubbling is prohibitive.

As an example, consider the sentence "The jury said it did find that many of Georgia's registration and election laws 'are outmoded or inadequate and often ambiguous' ". The Dependency Parse of this sentence is as follows is shown in Figure 6.4.

```
[['root', 'ROOT', 'said'],
  ['det', 'jury', 'The'],
  ['nsubj', 'said', 'jury'],
  ['nsubj', 'did', 'it'],
  ['ccomp', 'said', 'did'],
  ['csubj', 'outmoded', 'find'],
  ['det', 'many', 'that'],
  ['dobj', 'find', 'many'],
  ['poss', 'registration', 'Georgia'],
  ['prep_of', 'many', 'registration'],
  ['nn', 'laws', 'election'],
  ['conj_and', 'registration', 'laws'],
  ['cop', 'outmoded', 'are'],
  ['ccomp', 'did', 'outmoded'],
  ['conj_or', 'outmoded', 'inadequate'],
  ['advmod', 'ambiguous', 'often'],
  ['conj_and', 'outmoded', 'ambiguous']]
```

Figure 6.4: Dependencies of the sentence (Details in text)

As a first step towards finding Frames we attempt to discover what we call the left and right connectors. Left and right connects are noun phrases on the left and right of the verb of interest. We expect the left

and right connectors of the verb "find" in the above sentence to be "Jury" and "Law", providing us with the frame template "Jury find Law".

This however, is not the result we achieve use the Dependency Bubbling method described earlier. Dependency Bubbling fails to identify a left connector and identifies "that many" to be the right connector, resulting in "find that many" as the frame template.

6.5.3 Dependency Tree Parsing

From our exploration of Dependency Bubbling (Section 6.5.2), we have seen that dependencies alone do not contain all the information we require in extracting the left and right connects (also described in Section 6.5.2). Discovering connectors requires us to find, potentially distant, words that are connected verb of interest. We achieve this by using the entire dependency tree. We note that this is the same conclusion reached by Wang et al. (2015), in their work described in Section 5.4.

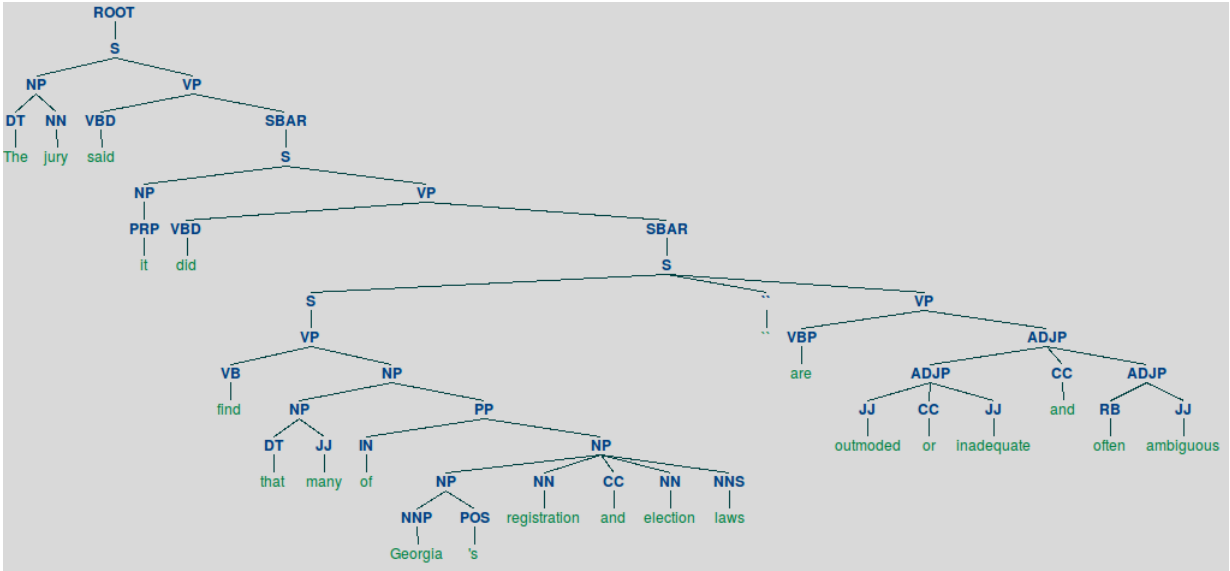


Figure 6.5: Dependency Tree of the sentence (Details in text)

Consider the dependency tree for the same sentence detailed in Figure 6.5. As in the case of Dependency Bubbling, we attempt to find the connectors for verb "find". Before we attempt to find connectors we first find the location of the Verb of interest in the Parse Tree, which we achieve through Depth First Search. Once this is done we address the discovery of each of the two connectors independently. We present the algorithm we use to discover the left connector in Algorithm 5.

The intuition behind the algorithm is that the connector on the left is part of the first noun phrase that appears on any left branch of the lowest parent of the node associated with the verb of interest. As can be seen from the example we discuss, this simultaneously eliminates the need for anaphora resolution within a sentence as we ignore elements that are not nouns.

We note that it is possible for a node in the dependency tree to have more than two children. In such cases we define a "left" sub-tree to be a sub-tree starting from every child that is to the left of path we took to reach this position and the "right" sub-tree to be every sub-tree starting from every child to the right of the path we took to reach this position.

Data: Parse Tree for a sentence and the verb (V) for which the left connector is required
Result: The Left Connector of the verb V
Stack $S = \emptyset$
Node $N_v =$ Location of Verb.
Current Node $N_c = N_v$
while *Connection is yet to be found* **do**
  $N_p = Parent(N_c)$
  if $N_P = Root$ **then**
    Set the Left Connector to be None
    Set Connection is Found to be True.
  else
    if $Type(N_p) =$ *(S or SBAR)* **then**
      if *We have not processed the left branch of* N_p **then**
        Push N_p on to stack S
        Perform a Depth First Search on the left branch of N_p to find a Noun Phrase
        if *We find a Noun Phrase* **then**
          Find the Head of the Noun Phrase
            (Which we currently do by picking the left most word in the Noun Phrase)
          Set the Left Connector to be the Head of the Noun Phrase
          Set Connection is Found to be True.
        else
          Restore N_p from the Stack S and set it as the Current Node being Processed
          Mark the left tree of N_p as processed
        end
      else
        Set $N_c = N_p$
      end
    end
  end
end

Algorithm 5: Algorithm for Discovering the Left Connector of a Verb.

We use a similar method to discover the right connect of a verb. The difference however is that we now attempt to find a noun phrase in the highest right sub-tree of the parent of the node representing the verb of interest. We present our method of discovering the right connector in Algorithm 6.

Our algorithm starts with the node representing the Verb Phrase of the Verb of interest and performs a right depth first search to find a noun phrase. We define a right depth first search to be a depth first search where sub-trees are always searched starting with the right most possible branch at every stage. Once we discover a noun phrase we pick the head of that noun phrase as the right connector.

Both of these algorithms have been extended to handle phrases. This allows us to consider phrases such as “speak up”, which are verb phrases have meanings that are different from the verb they contain (in this case “speak”).

Data: Parse Tree for a sentence and the verb (V) for which the right connector is required
Result: The Right Connector of the verb V
Stack $S = \emptyset$
Node N_v = Location of Verb.
Current Node $N_c = N_{vp}$ (Node representing the Verb Phrase of V)
while *Connection is yet to be found* **do**
  Push all Right Children of N_c on to Stack S moving left to right.
  for *Node N_i on Stack S* **do**
    if $Type(N_i) = NP$ **then**
      Set the Right Connector to be the Head of the Noun Phrase N_i
      Set Connection is Found to be True.
      Pop all Elements in Stach S
    else
      Push all Right Children of N_i on to Stack S moving left to right.
    end
  end
  if *Connection is yet to be found* **then**
    Set the Right Connector to be None
    Set Connection is Found to be True.
  end
end

Algorithm 6: Algorithm for Discovering the Right Connector of a Verb.

We present our results in the Table 6.6. It should be noted that the last column is an instance wherein we have failed to find the appropriate Right Connector. This is because of errors in POS tagging, a shortcoming that we are forced to work with. Despite our best efforts, the complexity structures of sentences and the errors in pre-processing methods (POS tagging and Dependency Parsing) result in an system that has errors. We tested the system by manually sampling verbs from a hundred sentences for which we find right and left connectors.

Engineering Difficulties

These algorithms were implemented in Python 3 by use of NLTK (Bird et al., 2009) and the Stanford CoreNLP (Manning et al., 2014). We tried to ensure that our implementation is compatible with NLTK so as to ensure that we can make use of the rich text processing features of NLTK. This required us to convert the CoreNLP representation of dependency trees to NLTK and to then to work with that representation. NLTK, to the best of our knowledge, does not provide inbuilt methods for Depth First Search, especially, Right and Left first Depth First Search. This forced us to implement all of these elements while ensuring structural compatibility with NLTK. This flexible implementation enabled us to easily extending these algorithms to be able to handle phrases.

6.5.4 Converting Frame Connectors to Frame Strings

Now that we have a method of extracting left and right connectors we can build what we call a Frame Connector. A Frame Connector is a triple (n_l, v, n_r) where n_l represents the left connector, n_r the right connector and v the verb. So as to eventually match the frame that a particular verb occurs into the valid frames of each sense of that verb we convert Frame Connectors to Frame Strings.

This process requires us to identify if nouns n_l and n_r are "Someone" or "Something". We make use of the hierarchical structure of WordNet. At this juncture it should be noted that this, in essence, requires Noun Sense disambiguation as we need to find the appropriate sense of the noun so as to find its "type" (Something or Someone). However, we observe that several different senses of the same word often have the same type. Table 6.7 shows the Hypernym Closures of each of the Nouns Senses of the word "Article". We note that all of them resolve to "Something" (as opposed to "Someone").

Sentence	Verb	Left Connector	Right Connector
Longer Your Oct. 6 editorial "The Ill Homeless" referred to research by us and six of our colleagues that was reported in the Sept. 8 issue of the Journal of the American Medical Association.	referred	editorial	research
	reported	research	issue
There is no sign that you bothered to consider the inverse of your logic : namely , that mental illness and substance abuse might be to some degree consequences rather than causes of homelessness.	consider	you	inverse
	bothered	you	None

Table 6.6: Results of the Left and Right Connector Algorithms.

Definition	Hypernym Closure
nonfictional prose forming an independent part of a publication	article.n.01 → nonfiction.n.01 → piece.n.06 → prose.n.01 → creation.n.02 → writing_style.n.01 → artifact.n.01
one of a class of artifacts	article.n.01 → artifact.n.01 → whole.n.02 → object.n.01article.n.01 → physical_entity.n.01 → entity.n.01
a separate section of a legal document (as a statute or contract or will	article.n.02 → section.n.01 → music.n.01 → writing.n.02 → auditory_communication.n.01 → written_communication.n.01 → communication.n.02
(grammar) a determiner that may indicate the specificity of reference of a noun phrase	article.n.04 → determiner.n.02 → function_word.n.01 → word.n.01 → language_unit.n.01 → part.n.01 → relation.n.01 → abstraction.n.06 → entity.n.01

Table 6.7: Hypernym Closures of each Noun Sense of the Word "article", showing that they all resolve to "Something".

Based on this observation we hypothesise that the type that most Noun Senses resolve to is the type of the noun we are interested in. In addition, we need to establish a method of translating top concepts in WordNet to a particular type. We do this by mapping the WordNet concepts *'person', 'people', 'living_thing', 'animal' and 'social_group'* to be of the type *"Someone"* and *'information', 'event', 'written_communication', 'physical_property', 'possession', 'abstraction' , 'psychological_feature', 'physical_entity'* to be of the type *"Something"*.

6.5.5 Verb Senses Disambiguation using Verb Frames

Once we find the Frame for a verb in a given sentence we attempt Verb Sense Disambiguation. As we noted in Section 6.5.1, Verb Frames do not uniquely determine the Sense of a verb because multiple senses of a verb can have the same frame. Table 6.6 further shows how our method (Described in Algorithms 5 and 6) of finding Frame Connectors is not perfect. We also make assumptions when converting nouns in Frame Connectors (Described in Section 6.5.4). Errors in each of these steps tend to be amplified by the next step resulting in poor Verb Sense Disambiguation.

Verb Sense Disambiguation using Verb Frames and Statistical Methods

As a final attempt we use multiple statistical methods to try to boost the performance of our Verb Sense Disambiguation system. We use Frames in conjuncture with the positional frequency analysis we performed in Section 6.5, and Word Vectors discussed in Section 6.4.3 as parameters to various statistical model to see how the performance might improve.

We first attempt to combine these results using Bayes' Theorem. We test the accuracy of each method individually and define $P(f), P(p)$ and $P(v)$ to be the probability with which each of the methods Frames, Positional and Word Vectors succeed at Verb Sense Disambiguation. We then run all three methods for each instance of Verb Sense Disambiguation and use conditional probability to disambiguate the sense of that verb.

This method, as it turns out, is theoretically unsound as the probabilities of success of each method are calculated across multiple attempts at VSD whereas we use them to calculate the probabilities in a single instance across multiple senses of the same word resulting in poor results.

Despite this shortcoming we attempt to repeat this by replacing conditional probabilities with ANN. Unfortunately, this provides us with an accuracy of 69%, well below the 75% we achieved by use of Word Vectors.

6.6 Extending Verb Connectors

In this section we describe our experiments with Verb Connectors that we initially developed for the purpose of Verb Frame identification (Section 6.5.3. Consider the sentence we analysed earlier "Your Oct. 6 editorial 'The Ill Homeless' referred to research by us and six of our colleagues that was reported in the Sept. 8 issue of the Journal of the American Medical Association.". We have previously seen how the Verb Connectors for the Verb "reported" in this sentence are "research" and "Issue". We observe that these words alone, while providing a lot of context miss important information regarding how these elements interact with each other.

We observe that a phrase that better captures the context of the verb "reported" is: "research by" "reported in the" "Issue of". In order to capture this we extend our Connector Identifications Algorithms (5 and 6) to also capture prepositions, a set of different tenses of "be" such as "is", "are" and the determiner "the". We call these Extended Verb Connectors.

Extended Verb Connectors provide us a way of better contextualising verbs. We combine the Left and Right Connectors (without extensions) with the Extended Verb to form two independent Extended Connectors. More concretely, in the example above, this would give us "research reported in the" and "reported in the issue". We extract sentence containing these phrases and observe that all of them contain the word "reported" in the same sense as the original sentence. Table 6.8 is a list of sentences extracted from the Internet for each of the Extended Connectors.

From the results in Table 6.8 we notice that this method provides several sentences all of which use the word "reported" in the same sense. Extended Connectors also have the advantage of discovering sentences in different contexts as is demonstrated by the highlighted sentences in the same table.

However, there are several instances wherein the verb might not be extendable as is the case of the verb "had" in the sentence: "The jury further said in term end presentments that the City Executive Committee, which had over-all charge of the election , 'deserves the praise and thanks of the City of Atlanta' for the manner in which the election was conducted.". The verb "had" has not Connectors Extendable and so we create the left and right extended connectors using the the left and right extended

Left Extended Connector ("research reported in the")	Right Extended Connector ("reported in the issue")
In addition to our regular posts, we launched a How to "Research the Headlines" series. Intended as a '10 top tips' to "help [our readers] to get closer to the truth of any *research reported in the* media".	You might need to install the module or theme on your test site in order to follow the steps *reported in the issue*.
Dr Chris Stevens' 'research reported in the Guardian	That Question was put to all 650 big-league performers by the New York Times over ten days in June 1983, with the results *reported in the issue* of July 4, a neatly chosen date.
In this activity students will make use of a website which provides commentaries on health research *reported in the* mainstream media.	**In case an issue is detected in automation script, the defect is *reported in the issue* tracker with Immediate priority and will be addressed within the day in order to provide a successful execution next day.**
Through our How to "Research the Headlines" series, we've provided some simple suggestions to assist with your critical consumption of *research reported in the* media.	**@webron , actually I found this issue because the default value is csv as *reported in the issue* #1160 .**
Psychologists often find their *research reported in the* popular press.	**In Visual Studio, unnecessary or missing transitive includes are *reported in the issue* list as one of the following issues: HA.DUPLICATE, HA.OPTIMIZE or HA.UNUSED.**

Table 6.8: Sentences Extracted from the Internet for Each of the Extended Connectors, with a significantly different context highlighted in bold.

connectors concatenated with the verb, giving us: "Committee had" and "had charge of". Regardless of the method we use, we always have two connectors to work with. We try to get around this by use of Normalized Google distance, proposed by Cilibrasi and Vitányi (2004) for finding the similarity of words using Google. The Normalized Google Distance between two search terms x and y is given by Equation 6.1.

$$NGD(x,y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x,y)}{\log M - \min\{\log f(x), \log f(y)\}} \tag{6.1}$$

where M is the total number of web pages searched by Google, f(x) and f(y) are the number of hits for search terms x and y, respectively and f(x, y) is the number of web pages on which both x and y occur.

We adapt NGD, to identify the Extended Connector that is better suited to capture the context of the verb of interest. We do this by combining results from Microsoft Bing and Bing News. We note that we cannot directly use NGD as the number of results reported are more representative of the actual number rather than the exact number of results, a change possibly incorporated by search engines recently. We do not detail the equations used for this purpose as this is still a work in progress.

6.7 Verb Walker

We are in the process of experimenting with replacing the verb in Extended Verb Connectors (Section 6.6) with other verbs to find verbs that can be used used in place of the original verb. This method allows us to find verbs that are related to the original verb while not necessarily being a synonym. An example of such a pair is "won" and "awarded". We would like to infer that a person winning something is identical to that same person being awarded the same thing, despite the fact that "won" and "awarded" are not synonyms.

This method also enables us to narrow down synonyms of a verb in a particular context. As an example, the synonyms of the word "run" include "escape, operate and extend" depending on the particular sense it is being used in. In the sentence "When he saw that he was in danger, he ran", the word "ran" can be replaced with the word "escaped" whereas in the sentence "The trains are run on time", we cannot replace the word "run" with "escaped" but can only do so with the word "operated".

We call this method of replacing a particular verb with another "Verb Walking". We compare the similarity of two Extended Verb Connectors by using a method similar to that of Normalised Google Distance proposed by Cilibrasi and Vitányi (2004) and described in Section 6.6. We significantly introduce the counts of each of the verbs and connectors so as to ensure that we are not dealing with different contexts as in the examples that are in bold in Table 6.8.

Initial experiments have been promising, however, we do not list the details of the equations defining the similarity between connectors here because, as of writing this report, it remains a work in progress.

We were tempted to revisit the problem of Verb Sense Disambiguation and attempt to solve it using Verb Walking. Unfortunately, not all senses of a verb have unique synonym making it impossible to use this method to disambiguate senses.

6.7.1 Verb Tense Converter

The examples provided in Section 6.7 show that it is essential to have synonyms of verbs in the same tense so as to use our method of Verb Walking. We develop a system that converts a verb to the same tense as that of another verb. Our work extends the programs available on the Nodebox English Linguistics library[3]

6.8 Wikification

Wikification is the processes of linking elements in free text to the corresponding Wikipedia Pages. Wikify! (Mihalcea and Csomai, 2007) is one of several such system (Milne and Witten, 2008; Ferragina and Scaiella, 2010; Speck and Ngonga Ngomo, 2014) that achieves impressive results. Figure 6.6 provides an example of this process. More formally, the task requires one to find the Wikipedia Page W that is associated with an element e in the sentence s.

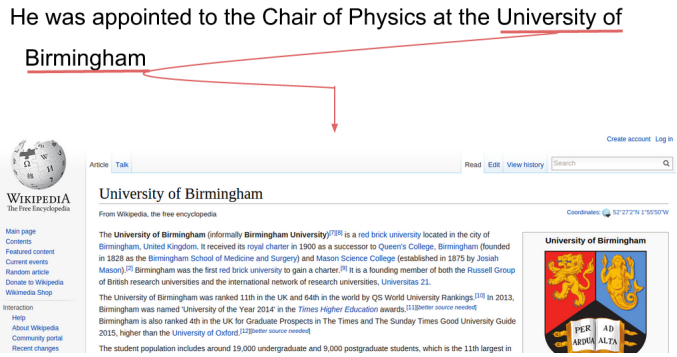


Figure 6.6: An Example of Wikification

[3]NodeBox. "NodeBox Linguistics Tools". Code available at: www.nodebox.net

We address this problem by simply searching for the element e on Wikipedia, extracting the results R and measuring the similarity between e and each $r \in R$. Our experiments show that the most effective similarity measure is word overlap, failing which we use a vector similarity measure **SECTION**. Should we fail to find a result r that either has a word overlap or a vector similarity higher than λ (Which we determine empirically), we return no associated page E.

Significantly, this method ensures that we have access to the latest version of Wikipedia and that we implicitly incorporate click through statistics that Wikipedia incorporates into it's search results.

6.9 Temporal and Regional Contextualisation

Words and phrases often mean very different things in different countries and at different times. For example the word "War" might conjure up very different ideas depending on where and when it was mentioned. In the 40s for example, most people would associate the word "war", with no additional contextualisation, to refer to the Second World War whereas that is not the case today. Similarly the word "President" refers to different individuals in different countries. There are, however, certain elements of knowledge that are agnostic to the time and place of utterance. We attempt to get around this by use of what we call Top-of-mind Awareness.

6.9.1 Top-of-mind Awareness.

Top-of-mind Awareness (TOMA), is a marketing term, that refers to a brand or specific product coming first in customers' minds when thinking of a particular industry and is considered an important metric in measuring brand awareness (Farris et al., 2010).

We use this idea to establish temporal and regional contextualisation by using the frequency with which a particular term is used on Twitter, a popular social networking platform. Intuitively, we expect phrases that are closely associated to appear in the same "Tweet" and so when attempting to identify what people in a particular region associate a phrase with we analyse tweets containing that phrase from that region. More concretely, to identify who people in India and Athens refer to when talking about "Prime Minister", we search for tweets containing the phrase "Prime Minister" from both those locations and analyse these results. Figure 6.7[4] shows search results for the phrase "Prime Minister" limited to results from Athens and Delhi.

Figure 6.7: Twitter Results for "Prime Minister" from Athens (left) and Delhi (right)

Given a phrase P and a region r we define Twitter search results for P from r to be $T_r(P)$. Let E represent an entity that appears with a frequency greater than λ (which we determine empirically) in a set of search results, we define E_{T_r} to be such an entity in the search results $T_r(P)$, $|E_{T_r}|$ as the number of results from region r containing the element E, and $E_{T_r^1}, ..., E_{T_r^i}$ and $|E_{T_r^1}|, ..., |E_{T_r^i}|$ to be all such entities and their counts respectively. We note here that an "Entity" is a phrase that contains an associated Wikipedia entry. This use of Wikification (Section **SECTION**) allows us to standardise entity names. To identify entities related to the phrase P in regions r_1 amongst regions of interest r_1 and r_2 we use the equation:

[4]Twitter. Search Results, Retrieved 22nd Aug 2015: www.twitter.com

$$\max_{j \in |E|} \left(\frac{|E_{T^j_{r_1}}|}{max(1, |E_{T^j_{r_2}}|) \times max(1, |E_{T^j_{r_g}}|)} \right) \tag{6.2}$$

where r_g refers to global results with no regional restrictions.

We have also used TOMA to find the more widely accepted uses of phrases ("The big brown fox" as opposed to "The brown big fox"(Minsky, 1986)). Figure 6.8[4] illustrates how this can be achieved. It must be noted that we do not consider the absolute number of results but the number of results over a fixed duration - "The big brown fox" has several results in the couple of months prior to the day the search results were extracted whereas "The brown big fox" has three results over several years.

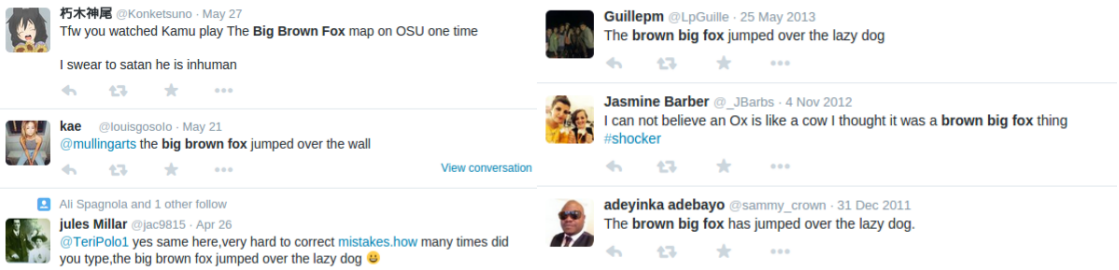


Figure 6.8: Train Error vs Cross Reference Error of Connect Four Training Data

TOMA provides us a way with contextualising phrases an important element when working with fast changing information such as news. We use TOMA sparingly as we focus on creating a model for a more general Question Answering System that is not limited to News.

6.10 Showing the Need for More Complex Neural Networks

Despite the vast amounts of literature detailing instances wherein different kinds of Neural Networks are more powerful than standard Backpropagation Neural Networks, we felt the need to convince ourselves of this fact so as to have an intuitive understanding of the kind of problems that cannot be solved by use of standard ANNs.

For this purpose we pick the popular game "Connect Four" which consists of which is a two player connection game played on a 6 high and 7 wide grid. Each player, picks a different colour and drops his disks along a column which falls on the next empty row along that column. The aim of the game is to Connect four disks either vertically, horizontally or diagonally.

We parametrise the board for an ANN by representing disks belonging to one player by 1, the other by -1 and empty slots by 0 and concatenating all rows. We intentionally do not attempt to optimise these parameters so as to allow our model to learn all possible interactions between various elements.

Training examples are obtained by having the system play against itself repeatedly. At the end of each game, the winning move is given a score of 1, the move by the losing player a -1 and all previous moves are given training weights of 1×7^{-n} for the winning player and -1×7^{-n} for the losing player where n represents the number of moves that player is from completing the game. So for example, when the game is won, the value of n will be 0. We use divide the score by 7 to represent the probability of a particular move being the optimal move or not based on the 7 possible moves available.

We left our program running for several days and found no improvement in its ability to play connect four. We show the changes in the Train Error and Cross reference error in Figure 6.9. We note that we expect a graph that is similar to that in Figure 6.2 (Train and Cross reference errors for WSD). This bizarre graph can only be explained by the fact that standard Neural Networks cannot adequately learn the intricacies of the game which requires planning over time and so Recurrent Neural Networks.

6.11 K-Means

We explore methods of automatically finding the number of optimal classes in K-means (described in Section 3.2.1) using the Elbow method (Detailed in Section 3.2.1).

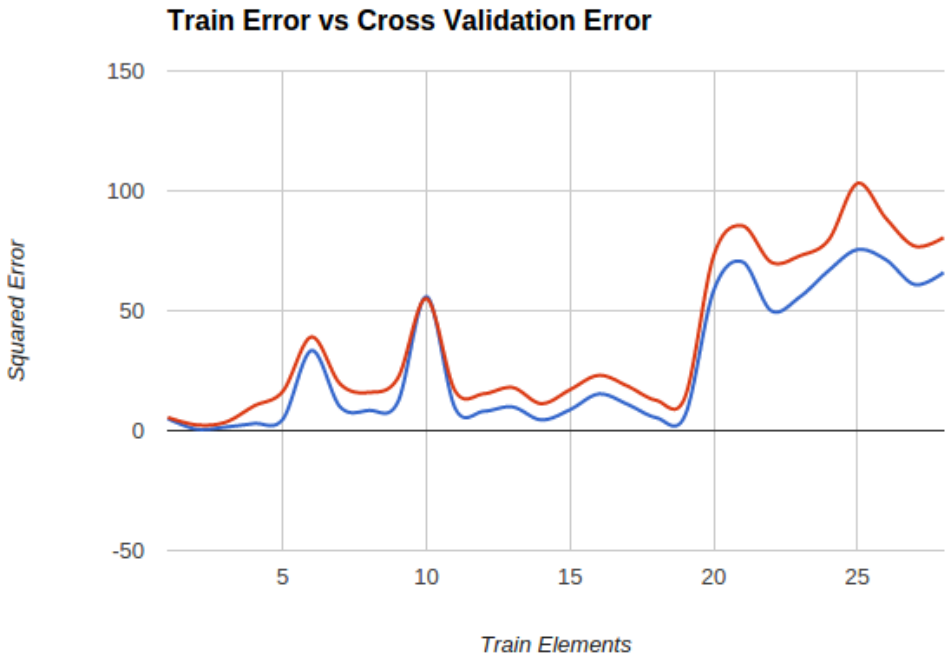


Figure 6.9: Train Error vs Cross Reference Error of Connect Four Training Data

Our approach involved running K-Means repeatedly to classify elements into classes of sizes between 1 and the number of elements in the data. We then calculate the change in the residual within cluster sum of squares to find the point at which the slope changes significantly, indicating the elbow.

Repeated runs of this method do not produce the same number of clusters as the optimal number of clusters. To get around this, we run this several times, use the clusters provided by our system as parameters and treat the task of finding the optimal number of clusters as a clustering problem from which we pick the cluster with the largest size. We note that, should the number of elements be extremely large, this process might have to be repeated multiple times inductively.

Unfortunately, our experiments showed that this method does not produce the same number of cluster as the optimal number of clusters.

Chapter 7

Research Objectives, Methods and Evaluation

In Chapters 6 we discussed the various experiments we performed for better understanding the intricacies of Natural Language Processing and more specifically the identification of elements of sentences that might be useful for Question Answering. In this Chapter we start with our observations and conclusions based on these experiments, link them to the current State of the Art in Question Answering described in Chapter 5, before then detailing the direction of research we intend to explore so as to achieve Open Domain Question Answering.

7.1 Observations based on Experiments

As our focus is the creation of a Question Answering System based on Natural Language Understanding and not statistical methods (Problem Definition, Section 1.1) we started off by exploring existing Knowledge Bases (KB). Our experiments with such KBs proved that the data contained within them, despite decades of data collection, is insufficient for our purpose (Section 6.1).

This led us to move to methods that involve searching the web for answers instead of searching through existing KBs (Section 6.2). Despite encouraging results, our exploration of such a method showed that we require more fundamental methods of Understanding Natural Language. With this in mind, we address the problem of Synset Disambiguation (Section 6.4).

Despite various attempts at Synset Disambiguation we have failed to achieve significant improvement on current State of the Art in that field. Although this might be possible by extending some of our methods, we have chosen not to pursue these avenues as our primary research objective is not Synset Disambiguation.

7.1.1 Question Answering based on Frames

Our initial vision for achieving Open Domain Question Answering was based on the idea that the model for Question Answering that is closest to what we intend to develop should be based on the Object Oriented Programming paradigm (Cox, 1985), a method similar to Minsky Frames (Section 2.3.3). *We note that Frames and Classes are not interchangeable. Our intention is to point out that Classes are potentially a convenient way of expressing Frames.* Figure 7.1 is a simplistic representation of this approach based on the example provided in Section 6.2.

7.1.2 Problems with Frame based Question Answering

Fame based Question Answering requires several elements of NLP such as POS tagging, Dependency Parsing, Named Entity Resolution and Word Sense Disambiguation. The current State of the Art in some of these tasks is too low to be used in this proposed method. Our experiments with Word Sense Disambiguation, a critical requirement for Frame based Question Answering has shown that the task is potentially as difficult as the task at hand. These complexities and inaccuracies in tasks that we depend on make Frame based Question Answering infeasible.

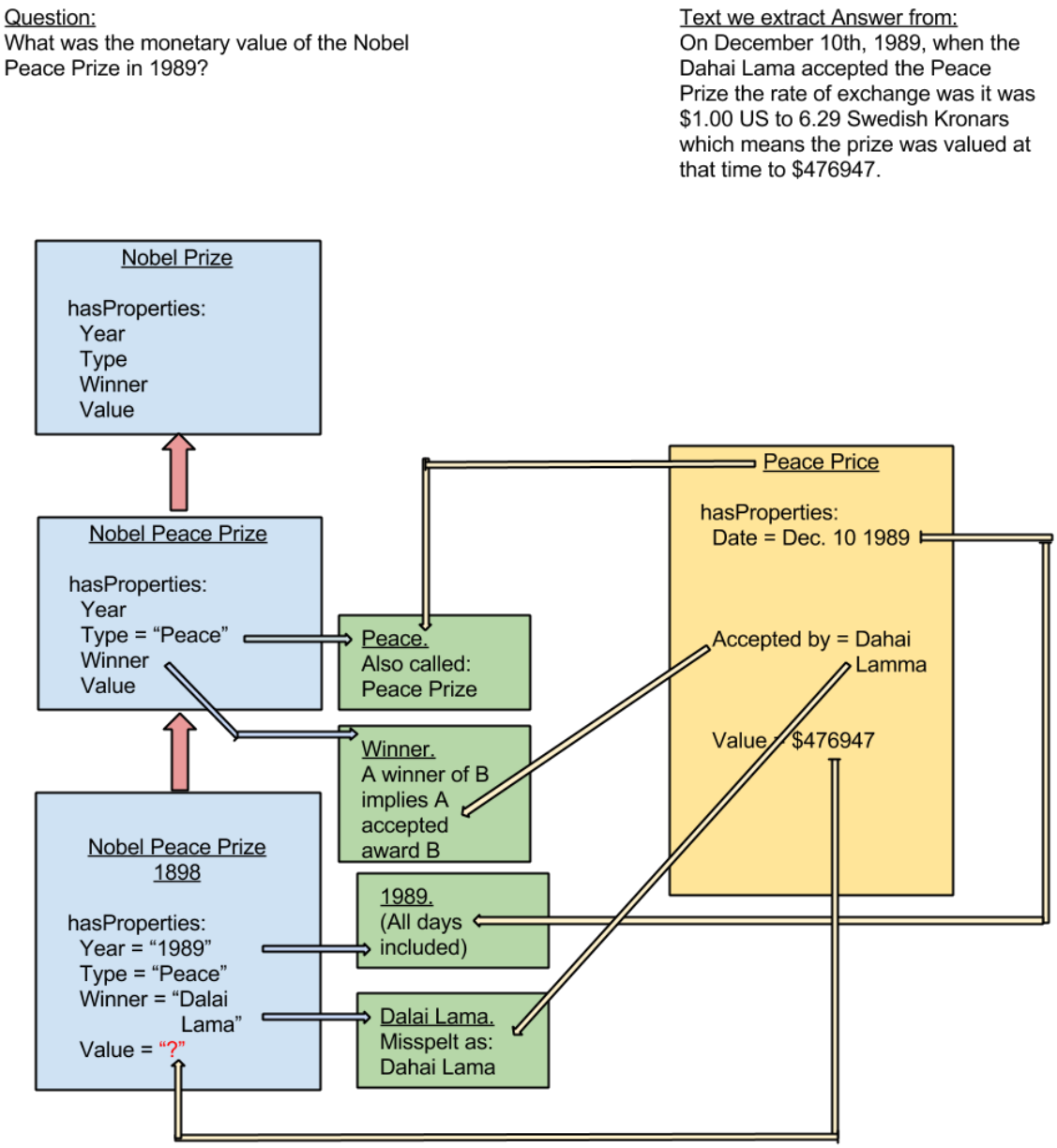


Figure 7.1: A Simplistic Representation of a Frame based Question Answering System, which we Abandon.

7.1.3 The problem with Stringing together Tasks

Frame based Question Answering required us to string together multiple tasks. It's tempting to use this approach in any new direction of exploration we might choose to use for achieving Question Answering. This, however, has one significant drawback that our experiments have brought out.

Any given method will nearly always have errors and these errors tend to propagate through our a system that uses multiple methods sequentially, resulting in the decay of accuracy in all successive steps.

This seemingly trivial observation, when ignored, as we did when attempting Verb Sense Disambiguation (Section 6.5), can have a profound impact on our ability to solve that problem. We conclude that an approach that minimises sequential steps will potentially have the best results.

This observation also leads us to discard exploration of methods of "understanding" text using techniques that incorporate various elements of Natural Language Processing, such as Named Entity Recognition, Wikification, Word Sense Disambiguation simultaneously so as to boost the performance of each other (Finkel and Manning, 2010). Such methods, illustrated in Figure 7.2 suffer from the possibility of

accuracy decay that propagates from one task to the other.

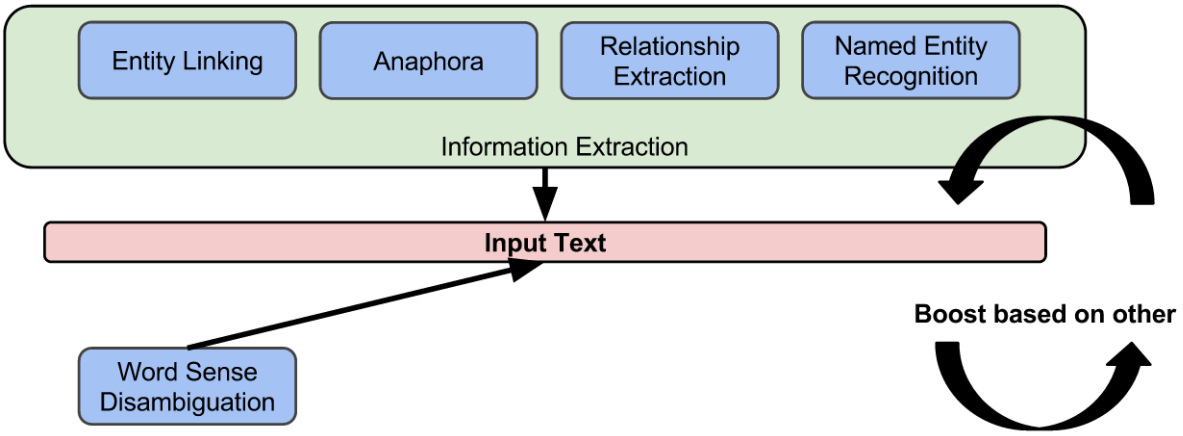


Figure 7.2: Boosted Learning - A Method that can Suffer from Error Propagation *Reducing* Accuracy.

7.2 Revising the Research Questions

Based on our review of the current State of the Art in Question Answering, our experiments and the conclusions we have drawn based on those experiments we discuss how we intend to address the Research Questions that we defined in Section 1.2, which, to ensure completeness, we restate here:

Research Question 1:
How can relationships between elements of free text, elements in a Knowledge Bases and other information pertaining to the text be established?

1. What are the various elements in free text that can be linked to Knowledge Bases?
2. What are the Knowledge Bases that can be used and what are the advantages of each?
3. What are the other elements of free text that might be useful?

Research Question 2:
What is the best structure to represent a combination of free text and information extracted from Knowledge Bases?

1. What structures will enable us to maintain the syntactic structure of free text while enabling us to process text?
2. How can elements of a Knowledge Base be integrated into the structure representing free text?
3. How can this system be used for Question Answering?

Research Question 3:
How can learning algorithms be used to improve the accuracy of the System?

1. Which specific learning algorithm will provide the best results?
2. How can the structures we use to represent the combination of free text and Knowledge Bases be parameterized as input to a learning algorithm?

In addition to this we reiterate our requirement of a system that does not consist of sub-tasks. With this background we make the following Hypotheses:

Hypothesis 1:
DBPedia and WordNet seem to be ideal KBs to link free text with using a combination of Wikification, TOMA, Type identification and Extended Connectors. POS tags of the sentence being

processed along with details of the patterns that it contains and details of Word Alignment distances could be important parameters.

Hypothesis 2:

A Dependency Tree with nodes linked to nodes of each of the Knowledge Bases we intend to integrate seems to be a good representation of Question Answer pairs.

Hypothesis 2:

Recurrent Neural Networks, with their ability to represent variable length input along with the fact that they are Turing Complete seem to be ideally suited for our purpose, although we choose Long Short Term Networks due to the Vanishing Gradient problem that limits Recurrent Neural Networks.

We provide a more complete and detailed analysis of how we intend to achieve Open Domain Question Answering using these elements in Section 7.3.

7.3 Current Vision for the Proposed Question Answering System

In this section we provide a detailed description of the method we intend to develop for Question Answering along with rational behind our hypotheses presented in the previous section.

7.3.1 Linking Knowledge Bases

Freebase is often the first choice for most researchers when picking a Knowledge Base for integration into research work. However, as discussed in Section 5.5, Google is currently in the process of shutting down that system. DBPedia, although containing significantly less entities, is not only active but also has the advantage of being directly linked to Wikipedia. We believe that this additional relation can be exploited for extracting text patterns related to each of these entities.

DBpedia also benefits from the large number of methods (Milne and Witten, 2008; Ferragina and Scaiella, 2010; Speck and Ngonga Ngomo, 2014) available for linking elements of free text to elements contained in the Knowledge base. In addition, we have access to the method of Wikification that we have developed, enabling us to customise the links we establish. For these reasons we pick DBPedia as the specific Knowledge Base that we exploit for our purpose.

DBpedia provides us with a source of external knowledge. However, we also need a source that provides us with a hierarchical classification of entities in free text. Unfortunately, a fine grained hierarchical classification system will require Word Sense Disambiguation. To get around this we use the method we have developed for Type Identification, described in Section 6.5.4, for which we depend on WordNet.

7.3.2 Representing Free Text and Related Information in a Single Structure

Our experiments, described in Section 6.5.2 have demonstrated the need for the entire Dependency Tree to be able to represent the structure of a sentence without any loss of information. With this in mind we use Dependency Trees to represent sentences.

We still need a method of linking this information to elements of a Knowledge Base. To achieve this we use the method introduced by Lao et al. (2012), discussed earlier in Section 5.6. This consists of representing both the sentence and the Knowledge Base as graphs and drawing edges between nodes representing the same element. Establishing these connections is achieved through a Entity Resolution mechanism, which in our case is Wikification. It should be noted that we use this method to link both DBpedia elements and Types that we extract from WordNet.

This Graph could potentially be extremely large. We prune this graph using the method proposed by Usbeck et al. (2015) which we have previously detailed in Section 5.5.3. Figure 7.3 is an example of a *part* of such a graph.

The graph in Figure 7.3 represents a part of the Dependency Tree for the sentence "The City's economy depended on subjugated peasants called helots" (Nodes in Blue), along with relevant elements from DBpedia (Nodes in Green) and Types Identification information (Nodes in Red). The blue edges represent edges added by our system through Wikification and Type Identification. It should be noted that we

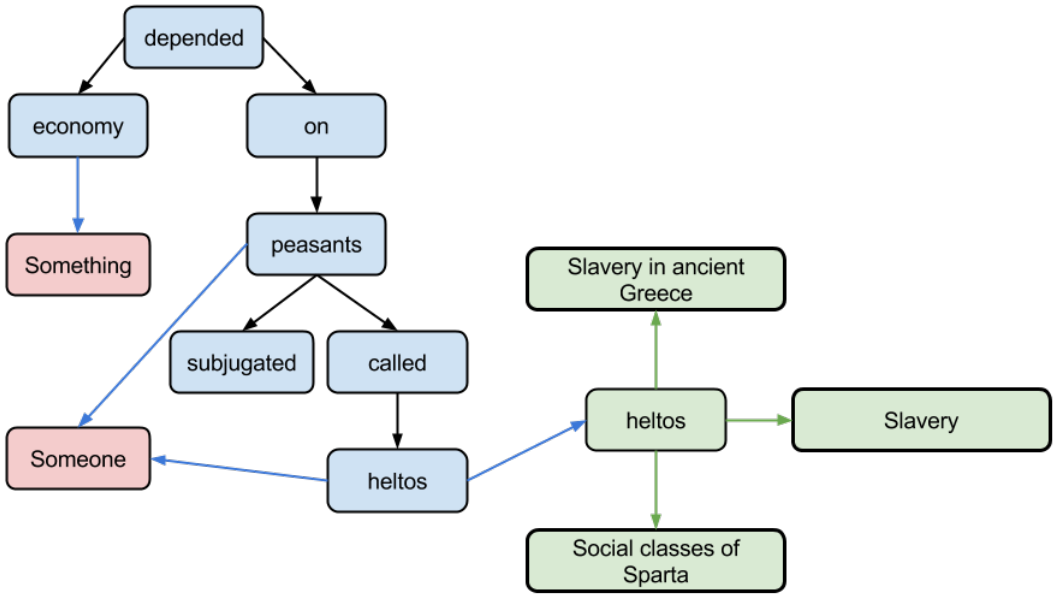


Figure 7.3: The Proposed Combined Graphical Representation of Free Text and Associated KBs

have left out some information from the Graph for readability. This includes the edge types of each edge in the Dependency Tree, the weight of each of the edges that we add which represents the confidence with which we add that edge and several nodes in DBpedia. It should also be noted that the DBpedia nodes with darker edges (Slavery in ancient Greece, Slavery) represent categories as opposed to those with light edges that represent entities (Helots).

7.3.3 Incorporating Learning Mechanisms

Our exhaustive study of Machine Learning Algorithms presented in Chapter 3 show that Recurrent Neural Networks, which are Turing complete (Siegelmann and Sontag, 1991), provide a powerful way of capturing relations across long distances. Unfortunately, Recurrent Neural Networks suffer from the Vanishing gradient problem (Hochreiter et al., 2001). To get around this we use Long Short Term Memory Networks.

We are still left with the formidable task of representing the graphs we have created (Section 7.3.2) into features for a LSTM Network. We achieve this by making use of the method proposed by Iyyer et al. (2014) which we have detailed in Section 5.7.

We note that there are differences between the graph used by Iyyer et al. (2014) in their work and that we intend to use. Additionally they use RNN instead of LSTM Networks. These adaptations will be one of the core elements of our work.

7.4 Putting it all Together

In this section we describe how we intend to combine the various elements discussed in the previous sections to create an Open Domain Question Answering system. We retain the use of Search Engines for Question Answering as described in our experiments in Section 6.2 which is similar to some of the first attempts at Open Domain Question Answering (Unger et al., 2012) and more recent attempts in languages other than English (Tufi et al., 2008). We intend to modify the search terms based on Connectors that we introduce in Section 6.5.3 so as to improve our ability to extract documents that potentially contain answers.

We note that the tasks described in the previous sections are best suited for working with sentences, one of which contains the answer. We intend to start our research work by focusing on single documents

that contain the answer to the given question. We use the method described by Hermann et al. (2015) and detailed in Sections 5.8 and 5.9 to extract such data.

Once this has been achieved we intend to extend our work to include document extraction by picking sentences from multiple search results by using Word Alignment Scores between sentences as described in Section 5.2, TOMA (Section 6.9.1) and Dependency Walk **SECTION**.

7.5 Evaluation

Due to the requirement of having questions that can be answered based on text from a single document we start our exploration by making use of the method proposed by Hermann et al. (2015) for extracting Question Answer pairs collected from CNN and The Daily Mail (Detailed in Sections 5.8 and 5.9).

In addition, the following are some of the standard datasets used for evaluating the performance of Question Answering Systems:

- TREC Question Answering track (Voorhees, 2001)
- SimpleQuestions(Bordes et al., 2015)
- WebQuestions(Berant et al., 2013b)
- Free917 (Cai and Yates, 2013a)

Chapter 8

Proposed Timetable

We present a tentative timetable for our research below. We leave it flexible due to possible unforeseen eventualities and any changes will be reported in subsequent RSMG meetings. During the course of our research, we will publish our work at each any stage results become available.

September - December 2015:

1. Explore Recursively Neural Networks and Long Short Term Memory Networks.
2. Dwell into the mathematical foundations of these networks to able to better adapt them to our work.
3. Ensure that they can be used for our purposes.

January - March 2016:

1. Explore the current implementation of DT-RNNs and see how they can be modified into LSTM Networks.
2. Create a system for extracting Question Answer pairs from CNN and The Daily Mail.
3. Experiment with DT-LSTM Networks

April - June 2016:

1. Course correction based on experimental results.
2. RSMG Report 4

July - September 2016:

1. Make changes to DT-LSTM Networks based on any changes required.
2. Evaluate results of the updated DT-LSTM Network

Beyond the next year, the timetable will be based on the results of the research during the year and might change. However, we provide a brief outline based on what we currently believe the status of our research will be.

October 2015 - March 2016:

1. Find methods of extracting and linking sentences across multiple documents.
2. Work on methods of extending DT-LSTM Networks to integrate this information.

3. Experiment with cross document DT-LSTM Networks
4. RSMG Report 4

October 2015 - March 2016:

1. Make changes to cross document DT-LSTM Networks based on experimental results.
2. Perform additional experiments with DT-LSTM Networks
3. Start writing Thesis
4. RSMG Report 5

April 2016 - September 2016:

1. Find methods of integrating document discovery, cross document linking and answer discovery with DT-LSTM Networks.
2. Experiment with these networks
3. Continue writing Thesis
4. RSMG Report 6

October 2016 - January 2017:

1. Complete writing Thesis
2. Defend Thesis
3. Contingency time

Bibliography

David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147 – 169, 1985. ISSN 0364-0213. doi: http://dx.doi.org/10.1016/S0364-0213(85)80012-4. URL http://www.sciencedirect.com/science/article/pii/S0364021385800124.

Eneko Agirre and Philip Edmonds. *Word Sense Disambiguation: Algorithms and Applications*. Springer Publishing Company, Incorporated, 1st edition, 2007. ISBN 1402068700, 9781402068706.

Eneko Agirre, David Martínez, Oier López de Lacalle, and Aitor Soroa. Two graph-based algorithms for state-of-the-art wsd. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 585–593, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. ISBN 1-932432-73-6. URL http://dl.acm.org/citation.cfm?id=1610075.1610157.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *Proceedings of the 6th International The Semantic Web and 2Nd Asian Conference on Asian Semantic Web Conference*, ISWC'07/ASWC'07, pages 722–735, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 3-540-76297-3, 978-3-540-76297-3. URL http://dl.acm.org/citation.cfm?id=1785162.1785216.

Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, New York, NY, USA, 2003. ISBN 0-521-78176-0.

Satanjeev Banerjee and Ted Pedersen. An adapted lesk algorithm for word sense disambiguation using wordnet. In *Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing '02, pages 136–145, London, UK, UK, 2002. Springer-Verlag. ISBN 3-540-43219-1. URL http://dl.acm.org/citation.cfm?id=647344.724142.

Junwei Bao, Nan Duan, Ming Zhou, and Tiejun Zhao. Knowledge-based question answering as machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 967–976. Association for Computational Linguistics, 2014. URL http://aclweb.org/anthology/P14-1091.

Pierpaolo Basile, Annalina Caputo, and Giovanni Semeraro. An enhanced lesk word sense disambiguation algorithm through a distributional semantic model. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1591–1600. Dublin City University and Association for Computational Linguistics, 2014. URL http://aclweb.org/anthology/C14-1151.

Hannah Bast and Elmar Haussmann. More accurate question answering on freebase. In *Proceedings of the 24th ACM International Conference on Conference on Information and Knowledge Management*, CIKM '15, New York, NY, USA, 2015. ACM.

Romain Beaumont, Brigitte Grau, and Anne-Laure Ligozat. Semgraphqa@ qald-5: Limsi participation at qald-5@ clef. CLEF, 2015.

Sean Bechhofer. Owl: Web ontology language. In LING LIU and M.TAMER ZSU, editors, *Encyclopedia of Database Systems*, pages 2008–2009. Springer US, 2009. ISBN 978-0-387-

35544-3. doi: 10.1007/978-0-387-39940-9_1073. URL http://dx.doi.org/10.1007/978-0-387-39940-9_1073.

Richard Bellman. A Markovian Decision Process. *Indiana Univ. Math. J.*, 6:679–684, 1957.

Jonathan Berant and Percy Liang. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425. Association for Computational Linguistics, 2014. URL http://aclweb.org/anthology/P14-1133.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of EMNLP*, 2013a.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1533–1544, 2013b. URL http://aclweb.org/anthology/D/D13/D13-1160.pdf.

Daniel M. Bikel. Intricacies of collins' parsing model. *Comput. Linguist.*, 30(4):479–511, December 2004. ISSN 0891-2017. doi: 10.1162/0891201042544929. URL http://dx.doi.org/10.1162/0891201042544929.

Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., 1st edition, 2009. ISBN 0596516495, 9780596516499.

Horst Bischof, Ale Leonardis, and Alexander Selb. Mdl principle for robust vector quantisation. *Pattern Analysis & Applications*, 2(1):59–72, 1999. ISSN 1433-7541. doi: 10.1007/s100440050015. URL http://dx.doi.org/10.1007/s100440050015.

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. ACM, 2008.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *CoRR*, abs/1506.02075, 2015. URL http://arxiv.org/abs/1506.02075.

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM. ISBN 0-89791-497-X. doi: 10.1145/130385.130401. URL http://doi.acm.org/10.1145/130385.130401.

Thorsten Brants. Tnt: A statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLC '00, pages 224–231, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics. doi: 10.3115/974147.974178. URL http://dx.doi.org/10.3115/974147.974178.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. ISSN 0885-6125. doi: 10.1023/A:1010933404324. URL http://dx.doi.org/10.1023/A%3A1010933404324.

Eric Brill, Susan Dumais, and Michele Banko. An analysis of the askmsr question-answering system. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 257–264. Association for Computational Linguistics, July 2002. doi: 10.3115/1118693.1118726. URL http://www.aclweb.org/anthology/W02-1033.

Qingqing Cai and Alexander Yates. *Large-scale semantic parsing via schema matching and lexicon extension*, volume 1, pages 423–433. Association for Computational Linguistics (ACL), 2013a. ISBN 9781937284503.

Qingqing Cai and Alexander Yates. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*

(Volume 1: Long Papers), pages 423–433. Association for Computational Linguistics, 2013b. URL http://aclweb.org/anthology/P13-1042.

A. Carlson, C. Cumby, J. Rosen, and D. Roth. The snow learning architecture, 5 1999. URL http://cogcomp.cs.illinois.edu/papers/CCRR99.pdf.

Eugene Charniak and Mark Johnson. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 173–180, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219862. URL http://dx.doi.org/10.3115/1219840.1219862.

Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar, October 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D14-1082.

Rudi Cilibrasi and Paul M. B. Vitányi. The google similarity distance. *CoRR*, abs/cs/0412098, 2004. URL http://arxiv.org/abs/cs/0412098.

Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, pages 1–8, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1118693.1118694. URL http://dx.doi.org/10.3115/1118693.1118694.

Michael John Collins. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, ACL '96, pages 184–191, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics. doi: 10.3115/981863.981888. URL http://dx.doi.org/10.3115/981863.981888.

Ronan Collobert. Deep learning for efficient discriminative parsing. In Geoffrey J. Gordon and David B. Dunson, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS-11)*, volume 15, pages 224–232. Journal of Machine Learning Research - Workshop and Conference Proceedings, 2011. URL http://www.jmlr.org/proceedings/papers/v15/collobert11a/collobert11a.pdf.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. Natural language processing (almost) from scratch. *CoRR*, abs/1103.0398, 2011. URL http://arxiv.org/abs/1103.0398.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. ISSN 0885-6125. doi: 10.1007/BF00994018. URL http://dx.doi.org/10.1007/BF00994018.

B.J. Cox. *Object oriented programming*. Addison-Wesley,Reading, MA, Jan 1985.

PauloCesarG. da Costa, KathrynB. Laskey, and KennethJ. Laskey. Pr-owl: A bayesian ontology language for the semantic web. In PauloCesarG. da Costa, Claudia dAmato, Nicola Fanizzi, KathrynB. Laskey, KennethJ. Laskey, Thomas Lukasiewicz, Matthias Nickles, and Michael Pool, editors, *Uncertainty Reasoning for the Semantic Web I*, volume 5327 of *Lecture Notes in Computer Science*, pages 88–107. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-89764-4. doi: 10.1007/978-3-540-89765-1_6. URL http://dx.doi.org/10.1007/978-3-540-89765-1_6.

Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems (I-Semantics)*, 2013.

Maurice de Kunder. Geschatte grootte van het geïndexeerde world wide web. *Master's thesis, Universiteit van Tilburg*, 2008.

Maurice de Kunder. The size of the World Wide Web. http://www.worldwidewebsize.com/, 2015. [Retrieved: 1st April 2015].

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure trees. In *LREC*, 2006. URL http://nlp.stanford.edu/pubs/LREC06_dependencies.pdf.

Pascal Denis and Benoît Sagot. Coupling an annotated corpus and a lexicon for state-of-the-art pos tagging. *Lang. Resour. Eval.*, 46(4):721–736, December 2012. ISSN 1574-020X. doi: 10.1007/s10579-012-9193-0. URL http://dx.doi.org/10.1007/s10579-012-9193-0.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Willey & Sons, New Yotk, 1973.

Ofer Egozi, Shaul Markovitch, and Evgeniy Gabrilovich. Concept-based information retrieval using explicit semantic analysis. *ACM Trans. Inf. Syst.*, 29(2):8:1–8:34, April 2011. ISSN 1046-8188. doi: 10.1145/1961209.1961211. URL http://doi.acm.org/10.1145/1961209.1961211.

Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1535–1545, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-937284-11-4. URL http://dl.acm.org/citation.cfm?id=2145432.2145596.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Paraphrase-driven learning for open question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1608–1618, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P13-1158.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 1156–1165, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2956-9. doi: 10.1145/2623330.2623677. URL http://doi.acm.org/10.1145/2623330.2623677.

P.W. Farris, N.T. Bendle, P.E. Pfeifer, and D.J. Reibstein. *Marketing Metrics: The Definitive Guide to Measuring Marketing Performance*. Pearson Education, 2010. ISBN 9780137053131. URL https://books.google.co.uk/books?id=7Ptw4nBoGmkC.

Christiane Fellbaum, Joachim Grabowski, and Shari Land. Analysis of a hand-tagging task. In *Tagging Text with Lexical Semantics: Why, What, and How?*, pages 34–40, 1997. URL http://www.aclweb.org/anthology/W97-0206.

D. Fensel, F. van Harmelen, B. Andersson, P. Brennan, H. Cunningham, E. Della Valle, F. Fischer, Zhisheng Huang, A. Kiryakov, T.K.-i. Lee, L. Schooler, V. Tresp, S. Wesner, M. Witbrock, and Ning Zhong. Towards larkc: A platform for web-scale reasoning. In *Semantic Computing, 2008 IEEE International Conference on*, pages 524–529, Aug 2008. doi: 10.1109/ICSC.2008.41.

Paolo Ferragina and Ugo Scaiella. TAGME: on-the-fly annotation of short text fragments (by wikipedia entities). *CoRR*, abs/1006.3498, 2010. URL http://arxiv.org/abs/1006.3498.

Jenny Rose Finkel and Christopher D. Manning. Hierarchical joint learning: Improving joint parsing and named entity recognition with non-jointly labeled data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 720–728, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1858681.1858755.

Ken-Ichi Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183 – 192, 1989. ISSN 0893-6080. doi: http://dx.doi.org/10.1016/0893-6080(89)90003-8. URL http://www.sciencedirect.com/science/article/pii/0893608089900038.

Francis Galton. *Natural Inheritance*. Number v. 42; v. 590 in Natural Inheritance. Macmillan, 1889. URL https://books.google.co.uk/books?id=GwLeuZGfIB0C.

J.G. Garnier and A. Quételet. *Correspondance mathématique et physique*. Impr. d'H. Vandekerckhove, 1838. URL https://books.google.co.uk/books?id=8GsEAAAAYAAJ.

F.A. Gers and J. Schmidhuber. Lstm recurrent networks learn simple context-free and context-sensitive languages. *Neural Networks, IEEE Transactions on*, 12(6):1333–1340, Nov 2001. ISSN 1045-9227. doi: 10.1109/72.963769.

Farhad Soleimanian Gharehchopogh and Yaghoub Lotfi. Machine Learning based Question Classification Methods in the Question Answering Systems. *International Journal of Innovation and Applied Studies*, 4(2):264–273, 2013. ISSN 2028-9324. URL http://www.ijias.issr-journals.org/abstract.php?article=IJIAS-13-218-18.

Jesús Giménez and Lluís Màrquez. Fast and accurate part-of-speech tagging: The SVM approach revisited. In *Recent Advances in Natural Language Processing III, Selected Papers from RANLP 2003, Borovets, Bulgaria*, pages 153–163, 2003.

Eli Goldberg, Norbert Driedger, and Richard I. Kittredge. Using natural-language processing to produce weather forecasts. *IEEE Expert: Intelligent Systems and Their Applications*, 9(2):45–53, April 1994. ISSN 0885-9000. doi: 10.1109/64.294135. URL http://dx.doi.org/10.1109/64.294135.

Freebase Google+. Freebase. https://plus.google.com/109936836907132434202/posts/bu3z2wVqcQc, 2015. [Retrieved: 17th August 2015].

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *CoRR*, abs/1410.5401, 2014. URL http://arxiv.org/abs/1410.5401.

Greg Hamerly and Charles Elkan. Learning the k in k-means. In S. Thrun, L.K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 281–288. MIT Press, 2004. URL http://papers.nips.cc/paper/2526-learning-the-k-in-k-means.pdf.

Shizhu He, Kang Liu, Yuanzhe Zhang, Liheng Xu, and Jun Zhao. Question answering over linked data using first-order logic. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1092–1103, Doha, Qatar, October 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D14-1116.

Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 2*, COLING '92, pages 539–545, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics. doi: 10.3115/992133.992154. URL http://dx.doi.org/10.3115/992133.992154.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *CoRR*, abs/1506.03340, 2015. URL http://arxiv.org/abs/1506.03340.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artif. Intell.*, 194:28–61, January 2013. ISSN 0004-3702. doi: 10.1016/j.artint.2012.06.001. URL http://dx.doi.org/10.1016/j.artint.2012.06.001.

Ronald A. Howard. *Dynamic Programming and Markov Processes*. The MIT press, New York London, Cambridge, MA, 1960. URL https://books.google.co.uk/books?id=fXJEAAAAIAAJ.

Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. A neural network for factoid question answering over paragraphs. In *Empirical Methods in Natural Language Processing*, 2014.

Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *CoRR*, cmp-lg/9709008, 1997. URL http://arxiv.org/abs/cmp-lg/9709008.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *CoRR*, abs/1404.2188, 2014. URL http://arxiv.org/abs/1404.2188.

Robert E. Kass and Larry Wasserman. A Reference Bayesian Test for Nested Hypotheses and its Relationship to the Schwarz Criterion. *Journal of the American Statistical Association*, 90(431):928–934, 1995. ISSN 01621459. doi: 10.2307/2291327. URL http://dx.doi.org/10.2307/2291327.

Boris Katz and Beth Levin. Exploiting lexical regularities in designing natural language systems. In *Proceedings of the 12th Conference on Computational Linguistics - Volume 1*, COLING '88, pages 316–323, Stroudsburg, PA, USA, 1988. Association for Computational Linguistics. ISBN 963 8431 56 3. doi: 10.3115/991635.991700. URL http://dx.doi.org/10.3115/991635.991700.

Boris Katz and Jimmy Lin. Annotating the semantic web using natural language. In *Proceedings of the 2Nd Workshop on NLP and XML - Volume 17*, NLPXML '02, pages 1–8, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1118808.1118816. URL http://dx.doi.org/10.3115/1118808.1118816.

Daphne Koller, Alon Levy, and Avi Pfeffer. P-classic: A tractable probablistic description logic. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Conference on Innovative Applications of Artificial Intelligence*, AAAI'97/IAAI'97, pages 390–397. AAAI Press, 1997. ISBN 0-262-51095-2. URL http://dl.acm.org/citation.cfm?id=1867406.1867466.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1545–1556. Association for Computational Linguistics, 2013. URL http://aclweb.org/anthology/D13-1161.

Ni Lao and William W. Cohen. Relational retrieval using a combination of path-constrained random walks. *Mach. Learn.*, 81(1):53–67, October 2010. ISSN 0885-6125. doi: 10.1007/s10994-010-5205-8. URL http://dx.doi.org/10.1007/s10994-010-5205-8.

Ni Lao, Amarnag Subramanya, Fernando Pereira, and William W. Cohen. Reading the web with learned syntactic-semantic inference rules. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1017–1026, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390948.2391061.

C. Leacock and M. Chodorow. Combining local context and wordnet similarity for word sense identification. In Christiane Fellfaum, editor, *MIT Press*, pages 265–283, Cambridge, Massachusetts, 1998.

Yann LeCun and Yoshua Bengio. The handbook of brain theory and neural networks. chapter Convolutional Networks for Images, Speech, and Time Series, pages 255–258. MIT Press, Cambridge, MA, USA, 1998. ISBN 0-262-51102-9. URL http://dl.acm.org/citation.cfm?id=303568.303704.

Douglas B. Lenat and R. V. Guha. The evolution of cycl, the cyc representation language. *SIGART Bull.*, 2(3):84–87, June 1991. ISSN 0163-5719. doi: 10.1145/122296.122308. URL http://doi.acm.org/10.1145/122296.122308.

Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems*

Documentation, SIGDOC '86, pages 24–26, New York, NY, USA, 1986. ACM. ISBN 0-89791-224-1. doi: 10.1145/318723.318728. URL http://doi.acm.org/10.1145/318723.318728.

Shasha Li, Chin-Yew Lin, Young-In Song, and Zhoujun Li. Comparable entity mining from comparative questions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 650–658, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1858681.1858748.

Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1072228.1072378. URL http://dx.doi.org/10.3115/1072228.1072378.

Yuri Lin, Jean-Baptiste Michel, Erez Lieberman Aiden, Jon Orwant, Will Brockman, and Slav Petrov. Syntactic annotations for the google books ngram corpus. In *Proceedings of the ACL 2012 System Demonstrations*, ACL '12, pages 169–174, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390470.2390499.

H Liu and P Singh. Conceptnet a practical commonsense reasoning tool-kit. *BT Technology Journal*, 22(4):211–226, 2004. ISSN 1358-3948. doi: 10.1023/B:BTTJ.0000047600.45421.6d. URL http://dx.doi.org/10.1023/B%3ABTTJ.0000047600.45421.6d.

Zhengdong Lu and Hang Li. A deep architecture for matching short texts. In C.j.c. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 1367–1375. 2013. URL http://media.nips.cc/nipsbooks/nipspapers/paper_files/nips26/697.pdf.

Thomas Lukasiewicz. Expressive probabilistic description logics. *Artificial Intelligence*, 172(67):852 – 883, 2008. ISSN 0004-3702. doi: http://dx.doi.org/10.1016/j.artint.2007.10.017. URL http://www.sciencedirect.com/science/article/pii/S0004370207001877.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press. URL http://projecteuclid.org/euclid.bsmsp/1200512992.

Christopher D. Manning. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In *Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing - Volume Part I*, CICLing'11, pages 171–189, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-19399-6. URL http://dl.acm.org/citation.cfm?id=1964799.1964816.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P/P14/P14-5010.

H. Masuichi, T. Ohkuma, H. Yoshimura, and D. Sugihara. Question answering system, data search method, and computer program, November 30 2010. URL http://www.google.com/patents/US7844598. US Patent 7,844,598.

Karen Mazidi and D. Rodney Nielsen. Linguistic considerations in automatic question generation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 321–326. Association for Computational Linguistics, 2014. URL http://aclweb.org/anthology/P14-2053.

Rada Mihalcea and Andras Csomai. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-803-9. doi: 10.1145/1321440.1321475. URL http://doi.acm.org/10.1145/1321440.1321475.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013a. URL http://arxiv.org/abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013b. URL http://arxiv.org/abs/1310.4546.

Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT-2013)*. Association for Computational Linguistics, May 2013c. URL http://research.microsoft.com/apps/pubs/default.aspx?id=189726.

George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995. ISSN 0001-0782. doi: 10.1145/219717.219748. URL http://doi.acm.org/10.1145/219717.219748.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. A semantic concordance. In *Proceedings of the Workshop on Human Language Technology*, HLT '93, pages 303–308, Stroudsburg, PA, USA, 1993. Association for Computational Linguistics. ISBN 1-55860-324-7. doi: 10.3115/1075671.1075742. URL http://dx.doi.org/10.3115/1075671.1075742.

David Milne and Ian H. Witten. Learning to link with wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 509–518, New York, NY, USA, 2008. ACM. ISBN 978-1-59593-991-3. doi: 10.1145/1458082.1458150. URL http://doi.acm.org/10.1145/1458082.1458150.

Marvin Minsky. Neural nets and the brain-model problem. *Unpublished doctoral dissertation, Princeton University, NJ*, 1954.

Marvin Minsky. Minsky's frame system theory. In *Proceedings of the 1975 Workshop on Theoretical Issues in Natural Language Processing*, TINLAP '75, pages 104–116, Stroudsburg, PA, USA, 1975. Association for Computational Linguistics. doi: 10.3115/980190.980222. URL http://dx.doi.org/10.3115/980190.980222.

Marvin Minsky. K-lines: A theory of memory. *Cognitive Science*, 4(2):117–133, 1980. ISSN 1551-6709. doi: 10.1207/s15516709cog0402_1. URL http://dx.doi.org/10.1207/s15516709cog0402_1.

Marvin Minsky. *The Society of Mind*. Simon & Schuster, Inc., New York, NY, USA, 1986. ISBN 0-671-60740-5.

Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. Patty: A taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1135–1145, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390948.2391076.

Roberto Navigli and Mirella Lapata. An experimental study of graph connectivity for unsupervised word sense disambiguation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(4):678–692, April 2010. ISSN 0162-8828. doi: 10.1109/TPAMI.2009.36. URL http://dx.doi.org/10.1109/TPAMI.2009.36.

Roberto Navigli, Kenneth C. Litkowski, and Orin Hargraves. Semeval-2007 task 07: Coarse-grained english all-words task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 30–35, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1621474.1621480.

Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1996. ISBN 0387947248.

AndrewY. Ng, Adam Coates, Mark Diel, Varun Ganapathi, Jamie Schulte, Ben Tse, Eric Berger, and Eric Liang. Autonomous inverted helicopter flight via reinforcement learning. In Jr. Ang, MarceloH. and Oussama Khatib, editors, *Experimental Robotics IX*, volume 21 of *Springer Tracts in Advanced Robotics*, pages 363–372. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-28816-9. doi: 10.1007/11552246_35. URL http://dx.doi.org/10.1007/11552246_35.

S. Nissen. Implementation of a fast artificial neural network library (fann). Technical report, Department of Computer Science University of Copenhagen (DIKU), 2003. http://fann.sf.net.

C.K. Ogden. *Basic English: a general introduction with rules and grammar*. Psyche miniatures: General series. K. Paul, Trench, Trubner & Co., Ltd., 1932. URL https://books.google.co.uk/books?id=g9AtAAAAIAAJ.

Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Motoki Sano, Stijn De Saeger, and Kiyonori Ohtake. Why-question answering using intra- and inter-sentential causal relations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1733–1743, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P13-1170.

Seonyeong Park, Hyosup Shim, and Gary Geunbae Lee. Isoft at qald-4: Semantic similarity-based question answering system over linked data. In *CLEF*, 2014.

Seonyeong Park, Soonchoul Kwon, Byungsoo Kim, and Gary Geunbae Lee. Isoft at qald-5: Hybrid question answering system over linked data and text data. CLEF, 2015.

Dan Pelleg and Andrew W. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 727–734, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-707-2. URL http://dl.acm.org/citation.cfm?id=645529.657808.

Simone Paolo Ponzetto and Roberto Navigli. Knowledge-rich word sense disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1522–1531, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1858681.1858835.

J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986. ISSN 0885-6125. doi: 10.1023/A:1022643204877. URL http://dx.doi.org/10.1023/A%3A1022643204877.

Deepak Ravichandran and Eduard Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 41–47, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073092. URL http://dx.doi.org/10.3115/1073083.1073092.

Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

Ehud Reiter and Robert Dale. Building applied natural language generation systems. *Nat. Lang. Eng.*, 3 (1):57–87, March 1997. ISSN 1351-3249. doi: 10.1017/S1351324997001502. URL http://dx.doi.org/10.1017/S1351324997001502.

Ehud Reiter and Robert Dale. *Building Natural Language Generation Systems*. Cambridge University Press, New York, NY, USA, 2000. ISBN 0-521-62036-8.

Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'95, pages 448–453, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-363-8, 978-1-558-60363-9. URL http://dl.acm.org/citation.cfm?id=1625855.1625914.

Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006. ISSN 0885-6125. doi: 10.1007/s10994-006-5833-1. URL http://dx.doi.org/10.1007/s10994-006-5833-1.

F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958. URL /bib/rosenblatt/Rosenblatt1958/frosenblatt.pdf.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Neurocomputing: Foundations of research. chapter Learning Representations by Back-propagating Errors, pages 696–699. MIT Press, Cambridge, MA, USA, 1988. ISBN 0-262-01097-6. URL http://dl.acm.org/citation.cfm?id=65669.104451.

Kenji Sagae and Jun'ichi Tsujii. Shift-reduce dependency dag parsing. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 753–760, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-44-6. URL http://dl.acm.org/citation.cfm?id=1599081.1599176.

E. Santos and J.C. Jurmain. Bayesian knowledge-driven ontologies: Intuitive uncertainty reasoning for semantic networks. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, pages 856–863, Oct 2011. doi: 10.1109/ICSMC.2011.6083717.

Roger C. Schank. Conceptual dependency: A theory of natural language understanding. *Cognitive Psychology*, 3(4):552 – 631, 1972. ISSN 0010-0285. doi: http://dx.doi.org/10.1016/0010-0285(72)90022-9. URL http://www.sciencedirect.com/science/article/pii/0010028572900229.

J. Schmidhuber, F. Gers, and D. Eck. Learning nonregular languages: a comparison of simple recurrent networks and LSTM. *Neural computation*, 14(9):2039–2041, September 2002. ISSN 0899-7667. doi: 10.1162/089976602320263980. URL http://dx.doi.org/10.1162/089976602320263980.

Gideon Schwarz. Estimating the dimension of a model. *Ann. Statist.*, 6(2):461–464, 03 1978. doi: 10.1214/aos/1176344136. URL http://dx.doi.org/10.1214/aos/1176344136.

SCImago. SJR - SCImago Journal & Country Rank. http://www.scimagojr.com, 2013. [Retrieved: 1st April 2015].

Hava T. Siegelmann and Eduardo D. Sontag. Turing computability with neural nets. *Applied Mathematics Letters*, 4(6):77 – 80, 1991. ISSN 0893-9659. doi: http://dx.doi.org/10.1016/0893-9659(91)90080-F. URL http://www.sciencedirect.com/science/article/pii/089396599190080F.

Sameer Singh, Sebastian Riedel, Brian Martin, Jiaping Zheng, and Andrew McCallum. Joint inference of entities, relations, and coreference. In *Proceedings of the 2013 Workshop on Automated Knowledge Base Construction*, AKBC '13, pages 1–6, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2411-3. doi: 10.1145/2509558.2509559. URL http://doi.acm.org/10.1145/2509558.2509559.

P. Smolensky. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Information Processing in Dynamical Systems: Foundations of Harmony Theory, pages 194–281. MIT Press, Cambridge, MA, USA, 1986. ISBN 0-262-68053-X. URL http://dl.acm.org/citation.cfm?id=104279.104290.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 151–161, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-937284-11-4. URL http://dl.acm.org/citation.cfm?id=2145432.2145450.

Richard Socher, John Bauer, D. Christopher Manning, and Ng Andrew Y. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465. Association for Computational Linguistics, 2013. URL http://aclweb.org/anthology/P13-1045.

Ren Speck and Axel-Cyrille Ngonga Ngomo. Ensemble learning for named entity recognition. In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig Knoblock, Denny Vrandei, Paul Groth, Natasha Noy, Krzysztof Janowicz, and Carole Goble, editors, *The Semantic Web ISWC 2014*, volume 8796 of *Lecture Notes in Computer Science*, pages 519–534. Springer International Publishing, 2014. ISBN 978-3-319-11963-2. doi: 10.1007/978-3-319-11964-9_33. URL http://dx.doi.org/10.1007/978-3-319-11964-9_33.

Umberto Straccia. Chapter 4 a fuzzy description logic for the semantic web. In Elie Sanchez, editor, *Fuzzy Logic and the Semantic Web*, volume 1 of *Capturing Intelligence*, pages 73 – 90. Elsevier, 2006. doi: http://dx.doi.org/10.1016/S1574-9576(06)80006-7. URL http://www.sciencedirect.com/science/article/pii/S1574957606800067.

Liling Tan. Pywsd: Python implementations of word sense disambiguation (wsd) technologies [software]. https://github.com/alvations/pywsd, 2014.

Wen tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. ACL Association for Computational Linguistics, August 2013. URL http://research.microsoft.com/apps/pubs/default.aspx?id=192357.

Wen tau Yih, Xiaodong He, and Christopher Meek. Semantic parsing for single-relation question answering. In *Proceedings of ACL*. Association for Computational Linguistics, June 2014. URL http://research.microsoft.com/apps/pubs/default.aspx?id=214353.

E.L. Thorndike. *Animal Intelligence: Experimental Studies*. Macmillan, New York, NY, USA, 1911. ISBN 9781295416264. URL https://books.google.co.uk/books?id=UgmToAEACAAJ.

J. Todhunter, I. Sovpel, and D. Pastanohau. Question-answering system and method based on semantic labeling of text documents and user questions, September 16 2010. URL http://www.google.com/patents/US20100235164. US Patent App. 12/723,449.

Dan Tufi, Dan tefnescu, Radu Ion, and Alexandru Ceauu. Racais question answering system at qa@clef2007. In Carol Peters, Valentin Jijkoun, Thomas Mandl, Henning Mller, DouglasW. Oard, Anselmo Peas, Vivien Petras, and Diana Santos, editors, *Advances in Multilingual and Multimodal Information Retrieval*, volume 5152 of *Lecture Notes in Computer Science*, pages 284–291. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-85759-4. doi: 10.1007/978-3-540-85760-0_34. URL http://dx.doi.org/10.1007/978-3-540-85760-0_34.

Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. Template-based question answering over rdf data. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 639–648, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1229-5. doi: 10.1145/2187836.2187923. URL http://doi.acm.org/10.1145/2187836.2187923.

Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Lorenz Bhmann, and Christina Unger. Hawk - hybrid question answering using linked data. In *The 12th Extented Semantic Web Conference (ESWC2015)*, May 2015. URL http://data.semanticweb.org/conference/eswc/2015/paper/research/144.

Lucy Vanderwende, Gary Kacmarcik, Hisami Suzuki, and Arul Menezes. Mindnet: An automatically-created lexical resource. In *Proceedings of HLT/EMNLP on Interactive Demonstrations*, HLT-Demo '05, pages 8–9, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1225733.1225738. URL http://dx.doi.org/10.3115/1225733.1225738.

V Vapnik and A Lerner. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24, 1963.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995. ISBN 0-387-94559-8.

S. V. N. Vishwanathan, Nicol N. Schraudolph, Risi Kondor, and Karsten M. Borgwardt. Graph kernels. *J. Mach. Learn. Res.*, 11:1201–1242, August 2010. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1756006.1859891.

Ellen M. Voorhees. The trec question answering track. *Nat. Lang. Eng.*, 7(4):361–378, December 2001. ISSN 1351-3249. doi: 10.1017/S1351324901002789. URL http://dx.doi.org/10.1017/S1351324901002789.

Sebastian Walter, Christina Unger, Philipp Cimiano, and Daniel Br. Evaluation of a layered approach to question answering over linked data. In Philippe Cudr-Mauroux, Jeff Heflin, Evren Sirin, Tania Tudorache, Jrme Euzenat, Manfred Hauswirth, JosianeXavier Parreira, Jim Hendler, Guus Schreiber, Abraham Bernstein, and Eva Blomqvist, editors, *The Semantic Web ISWC 2012*, volume 7650 of *Lecture Notes in Computer Science*, pages 362–374. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-35172-3. doi: 10.1007/978-3-642-35173-0_25. URL http://dx.doi.org/10.1007/978-3-642-35173-0_25.

Mingxuan Wang, Zhengdong Lu, Hang Li, and Qun Liu. Syntax-based deep matching of short texts. *CoRR*, abs/1503.02427, 2015. URL http://arxiv.org/abs/1503.02427.

Z. Wang and A. Ittycheriah. FAQ-based Question Answering via Word Alignment. *ArXiv e-prints*, July 2015.

Warren Weaver. Translation. In William N. Locke and A. Donald Boothe, editors, *Machine Translation of Languages*, pages 15–23. MIT Press, Cambridge, MA, 1949/1955. Reprinted from a memorandum written by Weaver in 1949.

Max Welling and Yee Whye Teh. Approximate inference in boltzmann machines. *Artificial Intelligence*, 143(1):19 – 50, 2003. ISSN 0004-3702. doi: http://dx.doi.org/10.1016/S0004-3702(02)00361-2. URL http://www.sciencedirect.com/science/article/pii/S0004370202003612.

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *CoRR*, abs/1410.3916, 2014. URL http://arxiv.org/abs/1410.3916.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q. Zhu. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pages 481–492, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1247-9. doi: 10.1145/2213836.2213891. URL http://doi.acm.org/10.1145/2213836.2213891.

Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, pages 133–138, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics. doi: 10.3115/981732.981751. URL http://dx.doi.org/10.3115/981732.981751.

Kun Xu, Yansong Feng, and Dongyan Zhao. Xser@ qald-4: Answering natural language questions via phrasal semantic parsing.

Min-Chul Yang, Nan Duan, Ming Zhou, and Hae-Chang Rim. Joint relational embeddings for knowledge-based question answering. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 645–650, Doha, Qatar, October 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D14-1071.

Limin Yao, Sebastian Riedel, and Andrew McCallum. Unsupervised relation discovery with sense disambiguation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 712–720, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390524.2390625.

Xuchen Yao and Benjamin Van Durme. Information extraction over structured data: Question answering with freebase. In *Proceedings of the 52nd Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 956–966. Association for Computational Linguistics, 2014. URL http://aclweb.org/anthology/P14-1090.

Chengxiang Zhai, Kuansan Wang, David Yarowsky, Stephan Vogel, and Evelyne Viegas. Web n-gram workshop 2010. *SIGIR Forum*, 44(2):59–63, January 2011. ISSN 0163-5840. doi: 10.1145/1924475.1924487. URL http://doi.acm.org/10.1145/1924475.1924487.

Appendices

Appendix A

Results: Using Search Engines for Question Answering

In this section we detail further results in our attempt to create a Question Answering system described in Search 6.2.

Question: "What does the Peugeot company manufacture?"
Search Terms Extracted: "Peugeot company"

Result 1

Our useful calculator makes it easy to work out the tax implications of your fleet, company or leasing vehicles.
Read about the latest technological innovations from Peugeot.

Use our configurator to price up your ideal Peugeot car or van.

Result 2

PSA Peugeot Citroen says it is ready to ink a contract with Iran Khodro to produce vehicles in the country, should nuclear talks between Tehran and .
In the second and final part of an interview with Maxim Picat, the head of Peugeot explains why the brand cannot launch a low-cost model, and details plans for new energy cars.
Plimsolls UK Peugeot Car Dealers analysis is the most definitive and accurate study of the UK Peugeot Car Dealers sector in 2013.
Iran Khodro says as well as parallel negotiations with PSA to set up a new joint venture, it is also in talks with a "popular" Western company as it looks to rapidly capitalise on what appears to be a dramatically improving .
PSA Peugeot Citroen - Financial and Strategic SWOT Anal .
Faurecia says it has inked a joint venture with Dongfeng Hongtai, which will serve the Chinese company and its automotive partners for passenger and .
Car sales in France continued to recover in March with domestic automakers PSA Peugeot Citroen and Renault both showing strong gains as mid market brands .
PSA Peugeot Citroen - Financial and Strategic SWOT Analysis Review provides you an in-depth strategic SWOT analysis of the companys businesses and operations.
Can Peugeot become a near-premium brand, in time?
Peugeot Car Dealers - Industry Report .

Automotive industry company news .

Result 3

Our useful calculator makes it easy to work out the tax implications of your fleet, company or leasing vehicles.
Read about the latest technological innovations from Peugeot.

Use our configurator to price up your ideal Peugeot car or van.

Question: "Who is the author of the book, The Iron Lady: A Biography of Margaret Thatcher"?
Search Terms Extracted: "author book Iron Lady A Biography Margaret Thatcher"
No Results identified as relevant

Question: "What is the name of the rare neurological disease with symptoms such as: involuntary movements (tics), swearing, and incoherent vocalizations (grunts, shouts, etc.)?" **Search Terms Extracted:** "name rare neurological disease swearing"

Result 1

Later stages of the disease may include further loss of physical and intellectual functions, a state of unconsciousness , and increased susceptibility to repeated infections of the respiratory tract .
Enter a disease name or synonym to search NORD's database of reports.
Permission is hereby granted to print one hard copy of the information on an individual disease for your personal use, provided that such content is in no way modified, and the credit for the source and NORD's copyright notice are included on the printed copy.
As the disease progresses, there may be rapidly progressive deterioration of cognitive processes and memory , resulting in confusion and disorientation, impairment of memory control, personality disintegration, agitation, restlessness, and other symptoms and findings.
NORD's full collection of reports on over 1200 rare diseases is available to subscribers (click here for details).
We are now also offering two full rare disease reports per day to visitors on our Web site.
In addition, in some extremely rare cases, CJD may take an infectious form.
NORD's reports provide a brief overview of rare diseases.
Creutzfeldt-Jakob disease is an extremely rare degenerative brain disorder characterized by sudden development of rapidly progressive neurological and neuromuscular symptoms.
Alone we are rare.

In early December 2000, European Union agriculture ministers agreed upon new measures to combat the spread of mad cow disease, including incinerating any cow over 30 months of age that had not tested negative for BSE.

Result 2

Hi my cousin has just been diagnosed with a rare neurological degenerative disorder, but no one in my family seems to be able to tell me the name.
Symptoms include loss of mobility... show more Hi my cousin has just been diagnosed with a rare neurological degenerative disorder, but no one in my family seems to be able to tell me the name.
Name of a rare degenerative neurological disease that is diagnosed in .
[What was your best friend name from .
[Newly disabled by neurological disease that's degenerative & want to die before I become more of a burden .
[Lou Gehrig may not have died from the disease named after him?
Name of a rare degenerative neurological disease that is diagnosed in .

If anyone has heard of a disorder like this I would appreciate a name - some .

Appendix B

Other Systems Explored

We list here the various systems that were explored in deciding on which existing systems to adapt to our needs. We note that although none of the below systems have currently been used, we might be required to use some of them in the future.

B.1 Systems Explored

- Yago2 (Hoffart et al., 2013)
- Microsoft N-Gram dataset (Zhai et al., 2011)
- The ClueWeb12 Dataset (www.lemurproject.org)
- ReVerb (Fader et al., 2011) (Also see: reverb.cs.washington.edu)
- Berkeley Entity Resolution System (Singh et al., 2013)
- Stanford Named Entity Recogniser (Finkel and Manning, 2010)
- SENNA (Collobert et al., 2011; Collobert, 2011): NLP predictions including POS tagging, chunking, name entity recognition, semantic role labeling and syntactic parsing.
- Probase (Wu et al., 2012): Large Common Sense database from Microsoft Research.
- MindNet (Vanderwende et al., 2005): Knowledge Representation project at Microsoft Research.
- Google Books N-Gram data (Lin et al., 2012)
- SEMPRE (Fader et al., 2014): Used to train semantic parsers that map Natural Language Text to certain logical forms.

B.2 Methods to be Studied in Greater Detail

We list here some methods we believe will be of importance and require further study.

- Minimum Bayes Risk
- Markov Logic Networks
- Bootstrapped Learning (Recursive self-improvement) (Bootstrap aggregating) (bagging).
- Categorial grammar
- Sparse Network of Winnows
- Restricted Boltzmann machines
- Monte Carlo Tree Search
- Neural Turing Machines (Graves et al., 2014)
- Sparse Network of Winnows (Carlson et al., 1999)