Extracting Topically Related Synonyms from Twitter using Syntactic and Paraphrase Data

Maria Antoniak

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2014

Reading Committee:

Fei Xia, Chair

Eric Bell

Program Authorized to Offer Degree:
Linguistics

University of Washington

Abstract

Extracting Topically Related Synonyms
from Twitter using
Syntactic and Paraphrase Data

Maria Antoniak

Chair of the Supervisory Committee:
Professor Fei Xia
UW Computational Linguistics

The goal of synonym extraction is to automatically gather synsets (groups of synonyms) from a corpus. This task is related to the tasks of normalization and paraphrase detection. We present a series of approaches for synonym extraction on Twitter, which contains unique synonyms (e.g. slang, acronyms, and colloquialisms) for which no traditional resources exist. Because Twitter contains so much variation, we focus our extraction on certain topics. We show that this focus on topics yields significantly higher coverage on a corpus of paraphrases than previous work which was topic-insensitive.

We demonstrate improvement on the task of paraphrase detection when we substitute our extracted synonyms into the paraphrase training set. The synonyms are learned by using chunks from a shallow parse to create candidate synonyms and their context windows, and the synonyms are incorporated into a paraphrase detection system that uses machine translation metrics as features for a classifier. When we train and test on the paraphrase training set and use synonyms extracted from the same paraphrase training set, we find a 2.29% improvement in F1 and demonstrate better coverage than previous systems. This shows the potential of synonyms that are representative of a specific topic. We also find an improvement in F1 score of 0.81 points when we train on the paraphrase training set and

test on the test set and use synonyms extracted with an unsupervised method on a corpus whose topics match those of the paraphrase test set.

We also demonstrate an approach that uses distant supervision, creating a silver standard training and test set, which we use both to evaluate our synonyms and to demonstrate a supervised approach to synonym extraction.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to the University of Washington and Pacific Northwest National Laboratory, where this work was carried out. Special thanks go to Fei Xia, Eric Bell, Courtney Corley, and Joshua Harrison, without whose help this thesis would not have succeeded. Thank you to Alan Ritter for advice and resources, and thank you to Wei Xu for advice and encouragement.

DEDICATION

to my parents, Zenen and Sherry Antoniak

Chapter 1
INTRODUCTION

Increasing interest in social media has led to research in efficient analytics for large streams of social data such as Twitter posts, blog articles, and YouTube comments. Twitter, a microblogging service, features live streams of tweets (short messages) from its users. Users often tweet about events in real-time, making Twitter an important news source. However, due to the volume of tweets, discovering relevant, new, or interesting tweets can be challenging. Furthermore, the pervasiveness of redundant tweets complicates the task of sifting through the tweets for interesting messages (Zanzotto et al., 2011). Petrović et al. (2012) showed one solution is to organize tweets into semantically related groups, but this requires the ability to automatically detect paraphrases.

Paraphrase detection is the task of determining whether two units (e.g. words, phrases, sentences) have the same semantic meaning. Paraphrase detection can be applied to many NLP tasks, including summarization, machine translation, and determining textual entailment. It can also help to filter large streams of data by revealing which messages contain different information.

Much focus has been given to a connected task, paraphrase generation. Paraphrase generation is the task of creating of a new unit that is semantically equivalent but lexically divergent from the original unit. Unlike paraphrase generation, paraphrase detection does not attempt to create a new sentence but to characterize the paraphrase relationship between two sentences. This judgement could be binary or real-valued. Paraphrase detection also shares many similarities with the machine translation and textual alignment tasks, and it suffers from several difficulties, such as syntactic differences which can obscure the similarity

between two sentences.

One of the greatest difficulties in detecting paraphrases on Twitter is the large number of words and phrases whose meanings are difficult to determine automatically. Traditional synonym resources such as WordNet (Miller, 1995) have limited utility because of the misspellings, acronyms, abbreviations, slang, colloquialisms, and other irregular language usages that are common on Twitter. Some of these phrases share the same meaning but have little or no lexical or semantic (at the word level) overlap. For example, *it would mean the world* and *it would make me so happy* are paraphrases, but the phrases *mean the world* and *make me so happy* are difficult to align; they have no lexical overlap, and the individual words are dissimilar in their standard definitions. These phrases are synonyms only when they are used in this context.

Furthermore, even within Twitter, there is great lexical variation between different tweets about topics. We find that tweets about sporting events use a significantly different vocabulary than tweets about celebrities, and tweets about one celebrity use a distinct vocabulary from tweets about another celebrity. Therefore, if we are interested in a certain topic and wish to extract synonyms that will be useful for this topic, it is important that we search for those synonyms specifically, rather than searching over all of Twitter.

Our intuition is that Twitter contains niches of synonyms which might not be applicable to other niches. Therefore, synonyms extracted for one topic may not be useful for another. Synonyms from this approach include slang and variations on named entities that are only used in context of a certain topic. We demonstrate methods to find synonyms specific to certain topics of interest, showing in the process that these synonyms are in fact more useful that those extracted without regard to topic.

For example, synonyms related to the topic of the Clippers, a basketball team based in Los Angeles, might include [*lebron*, *lbj*, *lebron james*], [*donald sterling*, *mr sterling*], and [*next owner*, *new owner*], while synonyms about the celebrity Ciara (whose son is named Future) might include [*ft ciara*, *feat ciara*], [*child future*, *son future*, *baby future*], and [*photo*, *photos*, *pic*, *snapshot*, *pics*]

Work on Twitter is complicated by a lack of resources such as POS taggers and parsers which exist for traditional resources. Much focus has been given to "translating" Twitter into standard English so that older tools can be used for this new form of social media. However, there are arguments against this approach, including that such normalization is linguistically inappropriate as it tries to impose a a set of rules from one dialect to another (Owoputi et al., 2013). We would take this even further and claim that it is inappropriate to treat Twitter as a single domain; rather, it is a platform for many different subdomains which much be accounted for when analyzing Twitter. In recent years, parsers and POS taggers made specifically for Twitter have allowed more sophisticated processing techniques (Ritter et al., 2011) (Kong et al., 2014) (Owoputi et al., 2013).

We take advantage of these new tools and leverage syntactic information about tweets in order to extract synonyms. Specifically, we use a shallow parse of the tweets to create chunks which are used both as candidate synonyms and as the context windows around those candidates. These context windows, POS tags of the candidates, and a dependency parse of the tweets are used as features for a variety of approaches, which include an unsupervised approach, a supervised approach using an support vector machine (SVM) classifier trained on a silver-standard corpus, and a hybrid approach which relies on a paraphrase corpus.

More importantly, we show that synonyms are most useful for Twitter when they are extracted from tweets on the same topic as the target tweets. For example, synonyms extracted from tweets about Amanda Bynes are more useful for identifying paraphrases about Amanda Bynes than synonyms extracted from tweets about Adidas or synonyms extracted from a large sampling of topics. Our synonyms provide much better coverage than previous synonyms resources such as WordNet (Miller, 1995), the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013), or normalization lexicons (Han et al., 2012) (Xu et al., 2013). Finally, we show that if these topically related synonyms are sufficiently representative of the test set, they can improve the results of a paraphrase detection system which relies on machine translation metrics to perform classification.

The thesis is structured as follows. In Chapter 2 we discuss previous work on paraphrase

detection, synonym extraction, and Twitter normalization, and we demonstrate the relationships between these tasks and the current work. We also discuss the tools currently available for natural language processing on Twitter. In Chapter 3, we present the various data sources used, including previously-built synonym lexicons. In Chapter 4, we explain our methodology, and we present a series of approaches to synonym extraction, some supervised and some unsupervised. We also present methods to supplement the labeled data. In Chapter 5, we demonstrate the results of these various approaches, and in Chapter 6, we discuss our findings and draw conclusions.

Chapter 2

LITERATURE REVIEW

Synonym extraction is the task of automatically creating a thesaurus, a list of n-grams grouped by semantic equivalence. These groups are known as synsets. Synonym extraction is a task closely related to both normalization and paraphrase detection, which also involve semantically grouping n-grams. Normalization is the task of aligning out-of-vocabulary (OOV) words and phrases with in-vocabulary (IV) words and phrases, and paraphrase detection is the task of determining if two semantic units (words, phrases, sentences, etc.) have equivalent meaning. Normalizations are a type of synonym, and synonyms are n-gram paraphrases. According to the Principle of Compositionality, the meaning of a sentence is composed of the meaning of its words (and the syntactic rules governing them), and therefore, paraphrase detection can benefit from synonym extraction.

In this section, we will discuss techniques related to all three of these tasks: synonym extraction, normalization, and paraphrase detection. We will also discuss some of the peculiarities of performing data analysis on Twitter and some of the recent approaches which seek to overcome these difficulties. Finally, we place the various approaches in context of our work and explain the motivation for the present work.

2.1 Paraphrase Detection

Paraphrase detection is the task of determining the semantic similarity of two units (unigrams, n-grams, sentences, etc.). A naive approach of matching n-grams performs poorly because of the many possible syntactic and lexical differences between any two sentences with the same meaning. For example, *I love cookies* and *Cookies are delicious* are paraphrases, but both lexical and syntactic knowledge is required to arrive at the paraphrase relationship.

Unusual syntax or vocabulary can increase the difficulty of the task, particularly with social media, because traditional semantic resources are of limited utility for this type of data. While dictionaries, POS taggers, and syntactic parsers can be used with high levels of accuracy for traditional data, these tools have so far performed worse on Twitter and similar platforms.

A variety of approaches to paraphrase detection have been attempted with varying degrees of complexity and success. Some take a more linguistically based approach, while others rely on feature weighting, matrix factorization, and machine learning techniques. All the approaches described below, except for that of Xu et al. (2014), were designed for and tested on standard English data (e.g. news articles), usually for the Microsoft Research Paraphrase Corpus (MSRPC) (Quirk et al., 2004), a corpus built from English news articles. We also discuss one recent approach for paraphrase detection on Twitter.

Guo and Diab (2012) determine similarity scores for sentences by modeling the sentences using a variation of latent semantic analysis (LSA). They model missing words rather than observed words, that is, they model what the sentence is *not* about rather than what it is about. This helps to overcome the sparsity problem of vocabularies in short sentences. Weighted matrix factorization is then used to so that the observed and missing missing words can be treated differently. This approach achieves an accuracy of 71.5% on the MSPRC corpus.

Ji and Eisenstein (2013) produce high results for paraphrase detection on the MSRPC corpus. They develop a new discriminative term-weighting metric called TF-KLD, which is found to outperform TF-IDF for paraphrase detection. Non-negative matrix factorization (NMF) is applied to the reweighted vectors, which results in latent representations of sentences that are then converted to vectors representing pairs of sentences (each pair corresponding to a gold label). These vectors can be used by any supervised classification model, unlike previous approaches. The final reported accuracy is 80.41% on the MSRPC, a nearly 4% gain over the previous state of the art (Madnani et al., 2012).

Socher et al. (2011) use unsupervised feature learning and supervised softmax classifica-

tion to achieve high results for the paraphrase detection task. They use a recursive autoencoder with dynamic pooling to achieve an accuracy of 76.8%. Feature vectors are learned for phrases in syntactic trees, and dynamic pooling is used to transform the variably-sized matrices into uniform representations.

Das and Smith (2009) use a combination of systems ("product of experts") to determine the paraphrase relationship between two sentences. The first system includes named entity recognition, dependency parses, POS tags, and features from WordNet. The second system uses a logistic regression model with simple features comparing the precision and recall similarity of two candidate sentences. The combination of these two systems achieves an accuracy of 76.1% on the MSRPC.

Malakasiotis (2011) classifies paraphrases by using a maximum entropy classifier trained using nine similarity and distance metrics: Levenshtein distance, Jaro-Winkler distance, Manhattan distance, Euclidean distance, cosine similarity, n-gram distance, matching coefficient, dice coefficient, and Jaccard coefficient. Three more features are added, representing the presence or absence of negation in the first sentence, the presence or absence of negation in the second sentence, and the length ratio (the minimum of the two sentences divided by the maximum of the two sentences). Including negation as a feature is particularly important because even when two sentences have high string similarity, they may not be paraphrases (e.g. *I will go to school* vs. *I will not go to school*). The original nine features are computed for each of 10 reformulations of the sentences (e.g. replacing the words with POS tags, stems, or soundex codes). When the model is trained on the MSRPC, it achieves an accuracy of 76.17% and F1 score of 82.91. This approach is particularly impressive because it makes no use of WordNet or similar resources, yet it still achieves a high accuracy.

Madnani et al. (2012) follow a similar approach. They use machine translation (MT) metrics as features for a supervised classification model. MT metrics were originally developed to test the similarity between computer-generated and human-produced translations. However, since the metrics measure the similarity of meaning between the two sentences, we can consider paraphrases as English-to-English translations (or monolingual translations)

and use the MT metrics to detect the paraphrase relationship between the sentences.

TER (Translation Edit Rate) is an MT metric that provides an alignment between the hypothesis and reference sentences and calculates the number of edits needed to align the sentences divided by the average number of reference words (Snover et al., 2006). It selects shifts that most reduce the word edit rate (WER) between the reference and hypothesis. It allows block movements of words but requires exact string matches. TERp (Translation Edit Rate Plus) extends TER beyond exact matches in three ways (Snover et al., 2009). First, stem matches are found using the Porter Stemmer. Second, synonym matches are found using WordNet. Finally, phrase substitutions are found using a paraphrase phrase table, which was created using a pivot-based method by Bannard and Callison-Burch (2005).

Madnani et al. (2012) use TER and TERp, in addition to other MT metrics, as features for three classifiers: logistic regression, SVM, and nearest neighbor. The average of the unweighted probability estimates from the classifiers determines the label for a given pair of sentences.

The MT metrics used by Madnani et al. (2012) include BLEU, MAXSIM, BADGER, SEPIA, TER, NIST, METEOR, and TERp. TERp was found to be the single most useful metric; when used alone, it produced an accuracy of 74.3% on the MSRP corpus. It is followed closely by METEOR, then NIST, and then BLEU. The best results were found using a combination of all eight metrics, yielding an accuracy of 77.4% and F1 of 84.1 on the MSRPC.

While the previous approaches focused on data from the MSRPC (Quirk et al., 2004), Xu et al. (2014) develop an paraphrase detection system specific to Twitter. Their method relies on discovering anchor words that connect topically and temporally related paraphrases. These anchor words are predicted using a discriminative model, whose features include POS tags, stems, and topics. This intuitive approach achieves an F1 score of 72.4 on a Twitter corpus created in the same work. Other systems, such as Das and Smith (2009), achieve an F1 score of only 63.0, while Ji and Eisenstein (2013) achieve an F1 score of 64.5 on the Twitter corpus. This corpus is also used for SemEval-15, Task 1, and so we refer to it as the

Reference	Description	Accuracy
Guo and Diab (2012)	LSA, missing words, and factorization	71.5
Das and Smith (2009)	Product of experts	76.1
Malakasiotis (2011)	Text similarity metrics	76.2
Socher et al. (2011)	Recursive autoencoder with dynamic pooling	76.8
Madnani et al. (2012)	Combination of eight machine translation metrics	77.4
Ji and Eisenstein (2013)	Re-weighting of discriminative features followed by matrix factorization	80.4

Table 2.1: Comparison of accuracies produced by different systems for paraphrase detection when training and testing on the MSRPC.

SemEval Paraphrase Corpus.

2.2 Synonym Extraction

Synonyms extraction is the task of automatically producing a thesaurus. Synonyms are learned from a corpus, often in an unsupervised style, and the synonyms can include both unigrams and short phrases.

Synonym extraction traditionally relies on the Distributional Hypothesis (Harris, 1954) and its corollary, which state,

Hypothesis: The meaning of a word is highly correlated to its context.
Corollary: Words that appear in similar contexts have similar meanings.

Figure 2.1: The Distributional Hypothesis.

However, relying exclusively on these theories results in errors such as the grouping together of antonyms or related words (e.g. days of the week) (Agirre et al., 2009). More

sophisticated techniques have either not been available for or performed poorly on Twitter because of the lack of tools to tag, parse, and evaluate tweets, though recent work has given hope that such evaluation might be possible.

Ruiz-Casado et al. (2005) demonstrate a traditional approach to synonym extraction that depends on the Distributional Hypothesis and its corollary (Harris, 1954). By searching for words that appear in similar contexts, they hope to discover synonyms.

Given a pair of words, they collect a set of contexts (words that come before and after a target word) for the first word. Then they search for incidences of the contexts sandwiched around a second target word. The number of hits constitutes a similarity score between the first and second target words. They use a context window of five words, and assign a threshold of two for non-open class words contained in the context window. This approach is unsupervised and requires no special data except for a corpus of text (in this case, the web itself constitutes the corpus, with hits from a search engine used to find the incidences of context-target word combinations).

Unfortunately, while the above approach can yield useful results, it can also result in noise. For example, antonyms are often used in similar contexts, and so the lists of synonyms will be polluted with related but opposite words.

Grigonytė et al. (2010) try to mitigate these errors and focus on precision of the synonyms. They take an unsupervised approach that discovers synonyms by building a paraphrase corpus, aligning segments of paraphrase pairs, and using context windows (as in Ruiz-Casado et al. (2005)) to find similar terms. Because synonyms can only be extracted from the automatically produced paraphrases, there is much less likelihood of noise.

They first remove stop words and identify candidate phrasal terms through IDF filtering. They then use the sumo metric (Cordeiro et al., 2007) to identify paraphrases, thus automatically constructing a corpus of paraphrases. The sumo metric is used because it favors sentences that are lexically divergent, which is preferable for paraphrases (the best paraphrase is one that has no lexical overlap but complete semantic overlap). Next, they align segments of the paraphrase pairs using the Smith-Waterman algorithm (Smith and Water-

man, 1981), and finally, they follow Ruiz-Casado et al. (2005) in using context windows to identify synonyms.

The above method was used successfully on corpora of scientific abstracts, which use standard English spelling and grammar. However, further challenges must be addressed to use a similar approach on Twitter and social media.

Ganitkevitch et al. (2013) created the Paraphrase Database (PPDB)[1], a database of 73 million phrasal paraphrases similar to the synonyms that we would like to extract. The synonyms can be of varying length and include slang, colloquialisms, and syntactic rules. The phrases are extracted by pivoting over bilingual parallel texts. The phrases are then ranked using the cosine distance between vectors of distributional features. These features include counts of n-grams to the left and right, position-aware lexical, lemma-based, POS, and named entity class features, dependency link features, and syntactic features for constituents governing the phrase.

2.3 Twitter Normalization

Because of the lack of resources specifically designed for the style of text found on Twitter, much effort has been spent trying to fit Twitter data back into existing tools. This requires "normalizing" the tweets so that their forms better match those of standard English data. For example, misspellings are corrected and slang is replaced with words from a traditional dictionary. Because the task is very similar to synonym extraction and because this work, unlike synonym extraction, has focused on social media like Twitter, we will describe two of these approaches below.

Han et al. (2012) create a dictionary of one-to-one lexical variants, in which each entry represents a pair of words one in-vocabulary (IV) and one out-of-vocabulary (OOV). Because of the strict one-to-one philosophy, no phrases are included in the synonyms and so, for example, no acronyms are normalized. The basic approach is to extract IV-OOV candidate

[1]http://www.cis.upenn.edu/ ccb/ppdb/

pairs based on their distributional similarity and then re-rank these pairs based on their string similarity. Because of the string similarity re-ranking and the dictionary used to find the IV words, this approach is unlikely to translate well to phrases.

The Aspell dictionary[2] was used to distinguish IV and OOV words and only OOV words which had at least 64 occurrences in the corpus and a character length of at least 4 were included. OOV words with shorter lengths were found to be very ambiguous. Words were represented as feature vectors (features being word co-occurrence frequencies), and similarities were then computed between these vectors. Best results were found when the automatically created dictionary was combined with other, previously published dictionaries.

A second method of Twitter normalization was introduced by Xu et al. (2013). This method overcame some of the shortcomings of the previous work. In particular, Xu et al. (2013) did not confine their normalizations to unigrams, and so they were able to normalize acronyms and other phrases. Like Grigonytė et al. (2010), they first create a corpus of paraphrase pairs and then use these pairs to extract synonyms.

To create the paraphrase corpus, they identify tweets that refer to the same event. This was determined by finding tweets which include the same named entity and date. Insignificant events were discarded after filtering using a significance test. Within the groups of events, tweets were then aligned using Jaccard distance (Jaccard, 1912), which identifies lexically similar strings. GIZA++ (Och and Ney, 2003) and Moses (Koehn et al., 2007) are then used to align the tweets and extract normalizations. They found that the extracted phrases tended towards normalization because correct grammar and spelling are more frequently used, and they strengthened this bias by including data from the New York Times. Xu et al. (2013) found that their lexicon performed best when combined with the normalizations from Han et al. (2012).

---

[2]http://aspell.net/

2.4 Pre-Processing Techniques for Twitter

Twitter presents many problems for traditional POS taggers, parsers, lexicons, named entity extractors, and other natural language processing tools. On Twitter, traditional grammar and spelling are sometimes present but often missing or inconsistent, while the vocabulary includes enough slang and colloquialisms to make only low recall dictionary lookups. This has motivated work in normalization (see Section 2.3), which tries to "translate" the language on social media into language found in traditional corpora. Once translated, existing tools can be used to tag, parse, and analyze the tweets.

However, recent work has focused on tagging, parsing, and performing data extraction on Twitter by using techniques specifically designed for the medium. In particular, two toolsets, Twitter NLP Tools[3] from Alan Ritter and Sam Clark and Tweet NLP[4] from ARK research group at Carnegie Melon, provide a combination of parsers, taggers, tokenizers, named-entity and event extractors, and other resources specialized for Twitter.

Of particular interest are three tools: a POS tagger, a shallow parser, and a dependency parser. We describe these resources below as they relate to the current work.

Ritter et al. (2011) provide a shallow parser for Twitter can be used for named entity and event extraction. The parser is trained on an annotated set of 800 tweets and achieves an accuracy of 87.5%.

Owoputi et al. (2013) provide a state of the art POS tagger for Twitter, achieving an accuracy of 93%. They develop a new tagging scheme for Twitter, since the traditional Penn Treebank (PTB) (Marcus et al., 1993) tags and annotation guidelines were designed for traditional news sources. Cluster-based features and lexical features are used to tag tokenized tweets.

From the same research group, ARK at Carnegie Mellon University, TweeboParser (Kong et al., 2014) is a dependency parser for Twitter that uses the above POS tagger and tokenizer.

---

[3]https://github.com/aritter/twitter_nlp

[4]http://www.ark.cs.cmu.edu/TweetNLP

The parser achieves 80% accuracy and was trained on a new corpus, the Tweebank corpus, released with the parser.

2.5 Summary

The previous approaches have either focused on synonym extraction and paraphrase detection for standard English data such as news articles or focused on normalization for Twitter.

Normalization is a related but distinct task from synonym extraction. While Han et al. (2012) and Xu et al. (2013) seek to reduce noise and align the noisy data with standard spelling and syntax, we are interested in comparing noisy data with other noisy data. We do not need to map each OOV synonym with an IV synonym, and so our synsets are freer than normalization sets. Furthermore, we are focused on lexically divergent synonyms rather than normalizations. Therefore, we cannot use similarity measures to filter or rank our candidate synonyms, as in Han et al. (2012). We are also particularly interested in synonym phrases, such as colloquialisms, which even a traditional dictionary might not be able to interpret. These phrases contain little or no lexical overlap and are a hindrance to paraphrase detection because of the difficulty in aligning them.

Unlike the PPDB (Ganitkevitch et al., 2013), we do not rely on bilingual data, and so we can extract synonyms from any corpus without needing to pivot over translations. Therefore, we are free to extract synonyms that are specific to our topics of interest.

We are interested in discovering synonyms that are representative of the topics in the SemEval Paraphrase Corpus test set. If the synonyms give a higher coverage of the test set, we hypothesize that there will be a corresponding boost in accuracy for the paraphrase detection task. For example, we would like our synonyms to include variations on named entities and slang that are only used in context of a certain topic. When discussing a sports team, *family* and *philadelphia_eagles* are synonyms, while in other contexts, they are not. Similarly, *wizz* and *wiz_kalifa* are synonyms, but unless the corpus includes these variations, they would not be detected without other techniques, such as named entity disambiguation.

In the following chapters, we make a formal comparison between different synonym re-

sources as applied to the machine translation metrics paraphrase detection system (Madnani et al., 2012) (see Section 2.1. We compare the output of variations of our system to synonyms from WordNet (Miller, 1995), the PPDB (Ganitkevitch et al., 2013), and the normalization lexicons of Han et al. (2012) and Xu et al. (2013).

Chapter 3

DATA

There already exist many synonym lexicons, from tradtitional thesaurusi to automatically constructed lists of synsets similar to the one we seek to create. We describe several of these resources below, in Section 3.1, as they relate to our approach. We explain the strengths of each resource and show how our synonyms are unique. We use these resources as points of comparison to measure the utility of our synonyms.

In Section 3.2, We also present two paraphrase corpora which are used to measure performance for paraphrase detection. We use these corpora to evaluate the effect of our synonyms on paraphrase detection.

3.1 Synonym Lexicons

WordNet[1] (Miller, 1995) is a lexicon of English nouns, verbs, adjectives, and adverbs which are grouped into synsets (groups of synonyms). The synsets are arranged in hierarchies, connected through hypernyms or meronyms, among other relationships. WordNet includes a total of 155,287 unique strings and 117,659 synsets.

We also use automatically extracted synonyms/normalizations/phrasal paraphrases from three sources: Han et al. (2012), Xu et al. (2013), and the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013) (see Sections 2.2 and 2.3 for descriptions of these systems).

Han et al. (2012) contains 41,181 normalization pairs, and Xu et al. (2013) contains 9,607,561 normalization pairs. The PPDB (Ganitkevitch et al., 2013) comes in different sizes, with smaller sizes providing higher precision and larger systems providing higher coverage. The smallest size contains 6.8 million paraphrase rules while the largest size contains 169

[1]http://wordnet.princeton.edu/

million tools.

Source	Type of Synonyms	Number of Synonyms
WordNet (Miller, 1995)	unigrams	117,659 synsets
Han et al. (2012)	unigrams	41,181 pairs
Xu et al. (2013)	n-grams	9,607,561 pairs
PPDB S (Ganitkevitch et al., 2013)	n-grams, syntactic rules	6,800,000 rules
PPDB XXXL (Ganitkevitch et al., 2013)	n-grams, syntactic rules	169,000,000 rules

Table 3.1: Comparison of types and sizes of existing synonym lexicons.

3.2 Labeled Paraphrase Corpora

One challenge in gathering synonyms from Twitter is the lack of labeled data for training a statistical classifier. Because our technique searches for words that would not appear in a traditional dictionary or resource such as WordNet, a training set with equivalent synonyms is difficult to construct.

However, labeled corpora of paraphrases do exist. The Microsoft Research Paraphrase Corpus (MSRPC) (Quirk et al., 2004) is a popular choice for training and testing paraphrase detection systems. The corpus includes 5,801 sentence pairs extracted from news articles, manually annotated with binary paraphrase labels. Of the 5,801 pairs, 3,900 (67%) were judged to be paraphrases.

More relevantly for our task, a training and development set of paraphrase-labeled tweets were released in conjunction with SemEval-15[2] (Xu et al., 2014). These tweets were collected for a set of trending topics and labeled using Amazon Mechanical Turk. The labels consist of five votes per pair of tweets, and we follow the recommended voting scheme to determine the binary paraphrase labels. The corpus includes 13,063 pairs of tweets in the training set and

[2]http://alt.qcri.org/semeval2015/task1/

1,902 pairs of tweets in the development set. 34% of the pairs were labeled as paraphrases, and there is an average of 31.8 tweets per topic in the training set.

Source	Training Size	Test Size	Percent Paraphrases
MSRPC	4,076	1,725	67%
SemEval Paraphrase Corpus	13,063	1,902	34%

Table 3.2: Comparison of the MSRPC and the SemEval Paraphrase Corpus.

In this work, we focus on the latter corpus, the SemEval Paraphrase Corpus, which we use to evaluate the utility of our extracted synonyms. The MSRPC is better established, but because the SemEval Paraphrase Corpus is drawn from Twitter, it will provide more representative results for our task of discovering synonyms from Twitter.

Chapter 4

METHODOLOGY

We demonstrate a series of approaches for synonym extraction, including methods that are supervised, unsupervised, and distantly supervised.

Because there is a lack of labeled data for this task and a lack of labeled paraphrases for the topics represented in the SemEval Paraphrase Corpus, we first present methods to automatically extract three different types of data.

We present four different methods for synonym extraction. The first three take a rule-based approach, while the last method trains a statistical classifier for synonym identification. The first approach uses a Twitter paraphrase corpus (Xu et al., 2014). It leverages the information from the paraphrase labels to extract synonyms from semantically equivalent tweets. The second approach is the same as the first except that we use an automatically created paraphrase corpus that can represent any topics of interest. The third approach discards paraphrase labels to increase the search space for the synonyms. The last approach uses WordNet (Miller, 1995) for distant supervision, creating training and development vectors of candidate synonym pairs as input for an support vector machine (SVM) classifier.

4.1 Tools

In the following sections, we rely on a series of tools designed for Twitter. Specifically, we extract POS tags using a system provided by ARK at Carnegie Mellon (Owoputi et al., 2013), dependency parses using TweeboParser, also from ARK (Kong et al., 2014), and shallow parses using the Twitter NLP Tools from Ritter et al. (2011). See Section 2.4 for more information about these resources.

4.2 Data Preparation

One of the central problems in synonym extraction is lack of labeled data. First, we need a larger corpus of tweets than those contained in the SemEval Paraphrase Corpus, but the corpus still needs to be focused on the same topics as in the SemEval Paraphrase Corpus. Essentially, we need to extend the SemEval Paraphrase Corpus to increase our search space for synonyms. This presents two challenges: a) we need to gather tweets on the same topics as the SemEval Paraphrase Corpus test set, and b) we need to automatically assign paraphrase labels to the gathered tweets. In this way, we can extend the paraphrase corpus given in the SemEval Paraphrase Corpus.

Second, there is no gold standard training data; because we are searching for Twitter-specific synonyms that do not already appear in other resources, our task is actually to create the gold standard. Therefore, it is difficult to use statistical classifiers that rely on labeled training data.

In the following sections, we describe how we supplement existing resources with a corpus of topically-related tweets, a silver-standard training corpus for distant supervision, and an automatically constructed paraphrase corpus of tweets.

4.2.1 Topics Corpus

For each topic in the SemEval-15 test set, we extract all the tweets from April, 2014, to October, 2014. As in Xu et al. (2014), the topic of a tweet is determined simply by the occurrence of the topic string (case insensitive) in the tweet. For example, if we are interested in the topic Adidas, we search for all tweets containing the n-gram *adidas*. We extract an average of 66,612 tweets per topic.

Because the SemEval-15 tweets were extracted from April 24, 2014, to May 23, 2014 (Xu et al., 2014), our corpus overlaps with the SemEval-15 corpus. Essentially, we are supplementing the SemEval-15 corpus with more tweets on the same topics, which increases our chances of finding synonyms occurring in the same contexts.

We refer to this corpus of topic-labelled tweets as the Topics Corpus.

4.2.2 Paraphrase Corpus B

Although the SemEval Paraphrase Corpus is valuable, its scope is limited to the topics represented in the training set, which do not overlap with those represented in the test set. These 500+ topics were trending between April 24th and May 3rd, 2014 (Xu et al., 2014), so they are limited both in number and temporal relevance. Therefore, to allow a more flexible approach which can extract synonyms for any topic, we automatically create a paraphrase corpus for the topics contained in the test set.

We follow Grigonytė et al. (2010) and Xu et al. (2013), and we automatically create a corpus of paraphrases which is similar to the test set.

We filter the Topics Corpus by time. We discover the densest hours of tweets and keep only the tweets in those hour. We perform this filtering because we believe that temporally related tweets are usually semantically related. For example, when a certain show plays on TV, many tweets will announce this at the same time. This is a simple way to discover tweets that might be paraphrases.

The processing of the tweets includes removing all mentions, re-tweets, and hashtags from the tweets, converting the tweets to lower case, and removing identical tweets (also removing those that are identical except for numbers). Finally, we determine the paraphrase pairs by iterating through pairs of tweets and keeping those pairs whose Jaccard similarity (Jaccard, 1912) is at least 0.35.

Jaccard similarity (Jaccard, 1912) is defined as the intersection of two sets divided by the union of the two sets.

$$Jaccard(X, Y) = \frac{X \cup Y}{X \cap Y} \tag{4.1}$$

This results in paraphrase pairs that have a high lexical similarity. It would preferable to have lexically divergent paraphrases, but automatically extracting such paraphrases is very

difficult.

We experimented with smaller time periods, for example, using five minutes instead of an hour, to find the dense areas of tweets. We hoped that by doing so, we would further restrict the tweets to those which were closely related semantically. However, we found the best results to be one hour periods; this period restricted the tweets semantically, but allowed a greater number of tweets to be grouped together, allowing for more synonyms to be extracted.

We refer to this corpus as the Paraphrase Corpus B.

4.2.3 Silver-Standard Corpus

One challenge in gathering synonyms from Twitter is the lack of labeled data to use for training a statistical classifier. Because the words we are searching for are exactly the words that would not appear in a traditional dictionary or resource such as WordNet, we do not have any labeled training data. Therefore, our approach is limited to unsupervised methods. A supervised method using a statistical classifier would likely produce higher quality synonyms because it can weight features in a more sophisticated way than a rule-based method.

Therefore, even if our training data is imperfect, it should increase the accuracy of our synonyms. This labeled set can also be used for evaluation. In this way, we can measure the accuracy directly, rather than relying on the paraphrase detection scores to indicate whether the synonyms are useful.

To build our training set, we use WordNet as a source of synonym labels. For the positive training instances, we search for candidate synonyms in our Twitter corpus which also appear in WordNet. We gather synsets (groups of synonyms) for these candidates, and we check if any of these synonyms also appear in our Twitter corpus. If so, we create a feature vector pair of the original candidate and the synonym. Features include counts of all the context windows in which the chunk appears, the POS tags contained in the chunk, and counts of

dependency pairs associated with the chunk, found using TweeboParser[1] (Kong et al., 2014).

We also tried to use the PPDB Ganitkevitch et al. (2013) as a source of synonym labels. We hoped that supplementing this corpus would overcome some of the weaknesses of the WordNet synonyms, since the PPDB includes phrasal and one-to-many synonyms, while WordNet contains almost entirely unigram synonyms. However, the coverage of the phrasal synonyms from the PPDB on our topics was too low, so the extracted synonyms were still mostly unigrams and we did not observe any improvement.

The negative examples require a more sophisticated approach. We require that the words appear in WordNet but the pairs must not be contained in each other's synsets. We further require that these pairs share at least one context window and POS tag, so that they better mirror the actual test set (we make these same requirements of the test set in order to reduce complexity). We also assemble a list of antonyms, again using WordNet, and we add these to the negative examples.

We weight the training set towards negative instances, experimenting with various ratios of negative to positive instances. We settled on a 3:1 ratio of negative to positive instances, which approximates the naturally occurring Twitter data, in which the majority of words are not synonyms.

We use binary labels for the synonym pairs, i.e. the pairs either are synonyms or they are not synonyms. An alternate approach, which we did not attempt, would be to provide a real-valued similarity score for the synonyms (perhaps using the distance between synsets in WordNet).

To combine the vectors representing candidates into vectors representing pairs of candidates, we follow the example of Ji and Eisenstein (2013). Each pair in the training and test sets is represented as a feature vector whose feature values are the sum and absolute difference of the feature values of the individual words.

---

[1] http://www.ark.cs.cmu.edu/TweetNLP/

$$s(\vec{v}_1, \vec{v}_2) = [\vec{v}_1 + \vec{v}_2, |\vec{v}_1 - \vec{v}_2|] \tag{4.2}$$

We perform the above method once for each topic of interest, creating a labeled corpus of feature vectors for each topic. In aggregate, we refer to these corpora of positive and negative synonym pairs as the Silver-Standard Corpus.

4.3 Synonym Extraction

We explore two general methods for synonym extraction. The first follows a rule-based approach that searches for n-grams which share context windows and POS tags. We describe three variations on this approach using the following corpora: the SemEval Paraphrase Corpus, the Paraphrase Corpus B, and the Topics Corpus. The second method trains a statistical classifier for synonym classification using the Silver-Standard Corpus.

4.3.1 Rule-Based Methods

For the following three approaches (Sections 4.3.1.a, 4.3.1.b, and 4.3.1.c), we follow a similar approach to Ruiz-Casado et al. (2005), in that we extract pairs of synonyms based on the distributional similarity of the pairs within a corpus. We search for n-grams that appear in the same contexts, and we use POS tags to filter the matches.

The first two approaches use shallow parse chunks as candidate synonyms and context windows, while the third approach uses n-grams. The Paraphrase Approach A (Section 4.3.1.a) leverages the paraphrase labels in the SemEval-15 corpus to extract synonyms. It only searches for synonyms and context windows within groups of paraphrases. The Paraphrase Approach B (Section 4.3.1.b) is the same as the first except that it uses the Paraphrase Corpus B instead of the SemEval-15 corpus. In this way, it can take advantage of paraphrase data that represents the test topics instead of only the training topics. The Unsupervised N-gram Approach (Section 4.3.1.b) discards paraphrase labels to increase the search space for synonyms, searching for synonyms and context windows across all the tweets from the

Topics Corpus, without filtering for paraphrases.

4.3.1.a Paraphrase Approach A

We use the training data from Task 1 of SemEval-15 (Xu et al., 2014) to extract candidate synonyms and context windows. This approach is based on that of Grigonytė et al. (2010), in that we leverage the information given by the paraphrase labels. The SemEval-15 training corpus is structured so that a series of paraphrase pairs all share the same reference sentence; we make the assumption that the hypothesis sentences are in fact all paraphrases of each other as well as the reference sentence (the paraphrase relationship is transitive). Rather than search for synonyms across the entire corpus, we only search within these groups of paraphrases.

We score the resulting synonym candidates (which can be single words or phrases of up to four words) based on the number of named entities (extracted using the Twitter Tools from Ritter et al. (2011)) and stop words contained in the candidates. From the resulting synonyms, we extract shorter synonyms, using sliding windows of n-grams within the chunks.

The results from this method show improvement, but there is still considerable noise. Therefore, we make two simple alterations to the context window system.

Our methods rely on two intuitions. First, to counteract our lack of training data, we leverage the information given by the paraphrase labels in the SemEval-15 Task 1 corpus. The corpus is structured such that a series of hypothesis tweets all share the same reference tweet. We make the assumption that the hypothesis tweets are all paraphrases of each other as well as of the reference tweet (the paraphrase labels are transitive). Because synonyms are likely to appear in these groups of paraphrases while antonyms and related but non-synonymous words are not likely to appear, we only search within these groups of paraphrases for our synonyms, rather than across the entire corpus.

Our second intuition is to use chunks from a shallow parse to form our candidate synonyms and context windows. By using these chunks, we hope to align synonyms of varying length and improve performance over simple n-grams, which split the tweet arbitrarily. We use the

Twitter Tools[2] from Ritter et al. (2011) to chunk the tweets.

We remove stop words from each chunk and then remove chunks which fail any of a series of tests. If the length of the chunk is less than three characters, if the chunk contains more than four unigrams, or if the length of the chunk is greater than seven characters but contains less than two vowels, we discard this chunk as a candidate synonym. We also discard chunks that are identical except for a number, e.g. *pls_follow* and *pls_follow89*. This eliminates some noise, such as nonsense words that contain only consonants and spam containing numbered tweets. We do not apply a frequency threshold because we are particularly interested in rare and unusual synonyms that are specific to the corpus.

Each chunk is surrounded by a left-hand chunk and right-hand chunk. These form the context window which is used to align the candidate synonym with other synonyms. If a chunk occurs at least once in the same context window as another chunk, we label them as synonyms (see Figure 4.1). This loose approach is possible because of our strictness in only searching within groups of paraphrases.

Tweet A: *omg i* ***am*** *so excited* ***to hear*** *ciara s new album*

Tweet B: *i* ***am*** *pumped* ***to hear*** *the new album from ciara*

Synonyms: excited, pumped

Figure 4.1: Using context windows to find synonyms.

From the resulting synonym chunks, we extract shorter synonyms, using sliding windows of n-grams within the synonym chunks. We follow the same approach as above except that we use n-grams instead of shallow parse chunks. For example, if we have two chunk synonyms *lizzie_mcguire* and *lizy_mcguire*, we would extract *lizzie* and *lizy* as synonyms.

From the pairs of synonyms, we create synsets (groups of synonyms). For each synonym, we create a list of all the synonyms with which it has been paired. For example, the synonym

[2]https://github.com/aritter/twitter_nlp

artist might be represented by the list [*singer*, *painter*, *creative person*] and the synonym *singer* might be represented by the list [*artist*, *pop star*, *vocalist*].

The challenge is to increase the precision as much as possible, since any error introduced into the synonyms can propagate in the paraphrase detection task.

We could filter the synonyms based on the number of occurrences, either the number of occurrences in certain contexts or the simply the count of each n-gram, as in Han et al. (2012). However, we are particularly searching for rare words and phrases, words and phrases that are specific to certain topics or cultural niches within Twitter. Filtering based on the number of occurrences would eliminate many of our most valuable synonyms.

We could also filter the synonyms using lexical similarity, as Han et al. (2012), but this would defeat our purpose of finding lexically divergent synonyms and synonyms phrases, rather than normalizations like misspellings. For the paraphrase detection task, the synonyms are self-filtering in that we only substitute a synonym into a sentence when it appears in the paired sentence. If a synonym is sufficiently unusual (for example, nonsense typing), it will probably never appear in the paraphrase corpus and so will never be used.

We refer to this approach as the Paraphrase Approach A.

4.3.1.b Paraphrase Approach B

Instead of using our manually labeled corpus, we use the Paraphrase Corpus B, which better represents the topics from the test set (see Section 4.2.2). In this way, we hope to retain the value of the paraphrase labels while expanding the coverage of our synonyms to new topics.

We follow the same techniques as above (see Section 4.3.1.a). The only difference is that we use the automatically created paraphrase corpus instead of the corpus from SemEval-15 (Xu et al., 2014). We find the top ten densest hours of tweets for each topic, we filter these tweets using Jaccard similarity, and finally, we use the resulting groups in the same way that we used groups of paraphrases in the first approach. We search only within the groups for synonyms.

We refer to this approach as the Paraphrase Approach B.

4.3.1.c Unsupervised N-gram Approach

Our third approach discards the paraphrase labels and shallow parse chunks. Rather than search for synonyms within a subset of paraphrases, we search across all the tweets for a given topic, using the Topics Corpus (see Section 4.2.1). Furthermore, rather than using chunks as candidate synonyms, we use n-grams of length up to four. While we lose the valuable semantic information from the paraphrase labels, this approach allows us to use more data and produce synonyms with a higher coverage of the test set. By discarding the shallow parse chunks, our non-unigram candidates might represent useless slices of the tweets, but we do use the POS tags to filter the candidate pairs. Each pair must share the same POS tags.

We follow the same method as above (see Sections 4.3.1.a and 4.3.1.b) except that we use n-grams instead of chunks and we search across all the tweets for a topic rather than within groups of paraphrases.

We refer to this approach as the Unsupervised N-gram Approach.

4.3.2 Statistical Classifier Approach

In this section, we present an alternative to the rule-based approaches described above (see Section 4.3.1). By bootstrapping a silver-standard corpus, which contains labeled pairs of synonyms, we are able to train a statistical classifier and classify new pairs of synonyms.

4.3.2.a Silver-Standard Approach

For our final approach, we address the problem of lack of training data. Because we are searching for synonyms that do not appear in an existing thesaurus, we do not have gold labels to use for training a statistical classifier. This severely limits our approach because we must manually add, subtract, and re-weight features, while a statistical model could learn optimal weights automatically.

To counteract the lack of training data, we use a form of distant supervision by training a

statistical classifier using the Silver-Standard Corpus (see Section 4.2.3 for more information about the creation of this corpus). We discover positive and negative examples of synonym pairs, extract POS, shallow parse, dependency parse, and context window features for these pairs from the Topics Corpus, and combine the vectors so that each pair of synonyms is represented by a single vector. We use this set to train a support vector machine (SVM) classifier for synonym detection.

Features include counts of all the context windows in which the chunk appears, the POS tags contained in the chunk, and counts of dependency pairs associated with the chunk (pairs in which at least one word appears in the chunk), found using TweeboParser[3] (Kong et al., 2014). We also experimented with features such as nearby words, POS tags of the context windows, the left and right contexts, and named entities, but these features were not found to be useful.

For the unlabeled test set, we gather candidate synonyms from the Topics Corpus which do not already appear in the Silver-Standard Corpus. As in the
firstApproach, we search for shallow parse chunks, and we require that each pair of candidate synonyms share at least one context window and at least one POS tag. This reduces the complexity of the comparison between every unique pair of chunks.

We train an SVM classifier with a linear kernel, and we train and test the classifier for each topic. We vary the penalty parameter C equal by calibrating on a portion of the training set.

For the test set, we construct candidate synonyms using the chunks from Alan Ritter's Twitter Tools[4] (Ritter et al., 2011). We require these candidate pairs to share at least one context and at least one POS tag. This filters out some noise and also reduces the complexity of testing all the unique combinations of candidates.

We refer to this approach as the Silver-Standard Approach.

---

[3]http://www.ark.cs.cmu.edu/TweetNLP/

[4]https://github.com/aritter/twitter_nlp

4.4 Summary

We have presented two basic methods to synonym extraction. The first includes three variations that follow a rule-based approach, and the second uses an SVM classifier trained using distant supervision. These methods rely on data which we gathered automatically, as described in Section 4.2.

Chapter 5

EVALUATION

Evaluating the output of our synonym extraction systems is a challenge in itself because we cannot compare our synonyms to any gold standard. The synonyms we extract are exactly those synonyms which are not labeled as synonyms in other resources (slang, colloquialisms, phrases, acronyms, etc.), and so we must search for alternate ways to evaluate our synonyms.

We take two approaches to evaluation. First, we substitute our synonyms into a paraphrase detection system, and we show that our synonyms improve the outcome of the paraphrase detection when the synonyms and the test set are drawn from the same set of topics. Second, we use distant supervision in the form of silver standard training and test sets of synonyms, created automatically using WordNet (see Section 4.2.3).

5.1 Evaluation Setup

We test our synonyms by substituting them back into the paraphrase corpus and running a baseline paraphrase detection system. We use a simplified implementation of Madnani et al. (2012) as a baseline. This system uses machine translation metrics as features for a supervised classification model. Machine translation metrics were originally developed to test the similarity between computer-generated and human-produced translations. However, we can consider paraphrases as English-to-English translations (or monolingual translations) and use the machine translation metrics to detect the semantic similarity and paraphrase relationship between the sentences. Madnani et al. (2012) use a variety of machine translation metrics as features for a statistical classifier. We limit these metrics to TER and TERp and use these metrics as features for an SVM classifier.

TERp uses WordNet and a paraphrase phrase table, which was created using a pivot-

based method by Bannard and Callison-Burch (2005) to align synonyms. Therefore, when we add our synonyms to the MT system, we are demonstrating the effect of supplementing WordNet and the phrase table with our synonyms.

We chose to test our synonyms on the system by Madnani et al. (2012) because while it performs well on standardized data such as the Microsoft Research Paraphrase Corpus (MSRPC) (Quirk et al., 2004), it performs very poorly on Twitter (see Table 5.3). This low performance is at least partially due to TERps reliance on WordNet and a paraphrase phrase table created by Bannard and Callison-Burch (2005) which are not representative of the vocabulary on Twitter. Therefore, the system is a good candidate for improvement through Twitter-specific synonyms.

For comparison, we also include results using the logistic regression system by Das and Smith (2009), which is a popular baseline for paraphrase detection. This system uses simple features comparing the precision and recall similarity of two candidate sentences. We use the implementation released with SemEval-15 Task 1 by Wei Xu[1].

We run a series of experiments on the SemEval Paraphrase Corpus, alternately training on the training set and testing on the testing set, training on the training set and testing on the training set, and training on the test set and testing on the test set. More information about this corpus can be found in Section 3.2 The synonyms are never extracted from the test set (the paraphrase labels on the test set are never used), even when we train the classifier on the test set.

We run the above tests for each set of synonyms from our different systems. We also compare our synonyms to the output of normalisation systems by Han et al. (2012) and Xu et al. (2013) and the PPBD (Ganitkevitch et al., 2013)

For each set of synonyms, we substitute the synonyms into the paraphrase training and test sets. For each pair of tweets, *Tweet A* and *Tweet B*, we search within *Tweet A* for n-grams contained in our synonym list. If an n-gram *a* is found, we search *Tweet B* for

[1]http://alt.qcri.org/semeval2015/task1/

synonyms of *a*. If we find a synonym *b* in *Tweet B* and *a* is not already present in *Tweet B*, we replace *b* with *a*.

Synonyms:	**concert**, **show**
Tweet A:	*omg I cant wait for the jayz* ***concert***
Tweet B:	*jayz s* ***show*** *tonight is gonna be great*
New Tweet B:	*jayz s* ***concert*** *tonight is gonna be great*

Figure 5.1: Substitution of synonyms into the paraphrase corpus.

The resulting tweets are used to extract the machine translation metrics, TER (Snover et al., 2006) and TERp (Snover et al., 2009). These vectors are then used to train the SVM model for the paraphrase detection task.

5.2 Results

In the following sections, we provide results for each of the different approaches described in Chapter 4. We also compare our results to existing synonym resources. We include examples of the extracted synonyms as well as results from the paraphrase detection task.

We show the results from several different approaches in Tables 5.1 and 5.2, which show results for the paraphrase systems by Madnani et al. (2012) and Das and Smith (2009). These results will be discussed in more detail in the following sections. It is important to note again that the results shown for the system by Madnani et al. (2012) in Table 5.1 are supplements to a WordNet baseline. That is, we use our synonyms in addition to the synonyms from WordNet in each test.

5.2.1 Results from Paraphrase Approach A

Table 5.3 displays the results of our two-part system: synonyms are extracted from the training set, and then an SVM classifier is used to label paraphrases. Variations in train

Synonyms	Accuracy	F1
WordNet Baseline	74.09	52.33
Extracted by Paraphrase Approach A	**74.24**	52.68
Extracted by Paraphrase Approach B	73.83	52.46
Extracted by Unsupervised N-gram Approach	74.07	**53.14**
Extracted by Silver-Standard Approach	74.11	52.40

Table 5.1: Accuracies and F1 scores for the paraphrase detection task using the system by Madnani et al. (2012) when training on the training set and testing on the test set.

Synonyms	F1	Precision	Recall
None	60.0	55.1	**65.7**
Extracted by Paraphrase Approach A	60.1	**57.2**	63.2
Extracted by Unsupervised N-gram Approach	59.9	57.0	63.1
Extracted by Silver-Standard Approach	59.8	**57.2**	62.6

Table 5.2: Precision, recall, and F1 scores for the paraphrase detection task using the logistic regression system by Das and Smith (2009) when training on the training set and testing on the test set.

and test sets for the SVM classifier are shown, but the synonyms substituted into these sets are always extracted from the training set. Our results in Table 3 show an improvement of 0.81% in accuracy and 2.29% improvement in F1 when we substitute our synonyms into the training set and run the paraphrase detection system on the same training set. When we run the paraphrase detection system on the test set, we see a smaller improvement: 0.15% in accuracy and 0.35% in F1.

Because we used the paraphrase data, we are able to avoid classic errors such as grouping antonyms or related words, such as the days of the week. The synonyms and synonym phrases produced through this method do include some noise, partly due to errors in the shallow

SVM Train	SVM Test	Synonyms Substituted	Accuracy	F1
train	test	no	74.09	52.33
train	test	yes	**74.24**	**52.68**
train	train	no	76.01	58.89
train	train	yes	**76.82**	**61.18**
test	test	no	74.07	52.18
test	test	yes	**74.14**	**52.42**

Table 5.3: Accuracy and F1 scores for paraphrase detection with different combinations of train and test sets with synonyms extracted from the training set.

Example Synonyms
two minutes, 180 seconds, 2 minutes, 180 secs, 2 mins, minute, record 2 minutes, approximately 2 minutes
nets and bulls game, netsbulls game, netsbulls series, nets bulls game
classic, hilarious, fucking crazy, live, good game, priceless, just friggin amazing
sac, 916, sac bitch, sac town
nice, piff, aite
fam david amerson, team redskinsnation, family, redskins, washington redskins
dallas, cowboys

Table 5.4: Example synonyms extracted from SemEval-15 Task 1 training set.

parse, but the majority are accurate and specific to the given topics (see Table 5.4). For example, by examining our synonyms, one can correctly conclude that our corpus includes many tweets about sporting events. Such specific synonyms do not appear in other synonym resources, such as WordNet (Miller, 1995) or the PPDB (Ganitkevitch et al., 2013).

For example, when discussing a sports team, *family* and *philadelphia_eagles* are synonyms

Example Substitutions
i can see tyler wilson getting the starting job in **oakland**
i can see tyler wilson getting the starting job in **raiders**
damn why the jets **released** tebow
damn why the jets **canned** tebow
jones got a tko at the end of the **1st**
jones got a tko at the end of the **first round**
hey stella **pls follow** me xx
hey stella **please follow** me xx
that **pacific rim con footage** is a big pile of incredible
that **pacific rim trailer** is a big pile of incredible
coutinho is a **super** player
coutinho is a **proper** player

Table 5.5: Example substitutions of synonyms back into the training set.

which refer to the same sports team, while in other contexts, they have different meanings. Similarly, *wizz* and *wiz_kalifa* are synonyms, but unless the corpus used for synonyms extraction includes these variations, they would likely not be detected.

From these results, it can been seen that our method is effective at extracting synonyms for the topics present in the training set. Because the test set covers different topics than the training set, it is not surprising that synonyms extracted from the training set do not boost the test sets scores. This is demonstrated in that while there were 1,296 substitutions in the training set, there were only 50 substitutions in the test set.

Table 5.10 displays results from when our SVM classifier is trained and tested on the training set with substitutions from different synonym lexicons. The first system, which uses WordNet, is the baseline on top of which the other systems add their scores. Our synonyms

Synonyms	Accuracy	F1
Madnani et al., 2012 (WordNet baseline)	76.01	58.89
Han et al., 2012	76.02	58.93
Ganitkevitch et al., 2012 (PPDB)	76.10	59.10
Xu et al., 2013	76.15	59.25
Current work	**76.82**	**61.18**

Table 5.6: Comparison of improvements from substituting synonym lexicons over the baseline system by Madnani et al. (2012). Training and testing only on the training set.

produce the best accuracies and F1 scores. Our accuracy is 0.67% higher and our F1 is 1.93% higher than the next best system by Xu et al. (2013).

Method	Number of Synsets
Extracted by Paraphrase Approach A	1,307
Extracted by Paraphrase Approach B	73.83
Extracted by Unsupervised N-gram Approach	103,277
Extracted by Silver-Standard Approach	11,486

Table 5.7: Number of synonym sets extracted using different methods.

Table 5.7 shows the number of synsets (groups of synonyms) that we extracted using different methods. This table can be compared to Table 3.1, which shows the number of synonym pairs/synsets/rules for previously created resources. In particular, it can be seen that even our largest number of synsets, 10,277 extracted using an unsupervised n-gram method, is much smaller than the PPDB and the normalization lexicon from Xu et al. (2013), which run in the millions.

However, despite the lower number of synsets compared to previous work, our coverage on

Synonyms Source	Number of Substitutions in Training Set	Percentage of Pairs in Training Set with Substitutions
Han et al., 2012	15	0.1%
Xu et al., 2013	123	1.1%
PPDB XXXL Phrasal	133	1.1%
PPDB S Phrasal	158	1.4%
Extracted by Paraphrase Approach A	**1,262**	**10.9%**
Extracted by Paraphrase Approach B	363	3.1%
Extracted by Unsupervised N-gram Approach	784	6.8%
Extracted by Silver-Standard Approach	11	0.1%

Table 5.8: Numbers of substitutions on the SemEval-15 training corpus using different synonyms sets.

the training set is significantly better than other synonym resources. While we obtain 1,262 substitutions in the training set, the S Phrasal PPDB (Ganitkevitch et al., 2013) achieves 158 substitutions, the XXXL Phrasal PPDB achieves 133 substitutions, Xu et al. (2013) achieve 123 substitutions, and Han et al. (2012) achieve only 15 substitutions. This demonstrates the importance of extracting synonyms for a particular topic, rather than number of synonyms (see Tables 3.1 and 5.7 for sizes of synonym lexicons).

It is particularly interesting that the size of the PPDB did not have much effect; in fact, when we moved from the smallest size to the largest size, there were fewer substitutions found in the test corpus. The provides strong evidence that to achieve good coverage, the source of the synonyms is more important than the number of synonyms.

Out of 11,530 total pairs of tweets in the SemEval-15 training set, 10.9% underwent

Synonyms Source	Number of Substitutions in Test Set	Percentage of Pairs in Test Set with Substitutions
Han et al., 2012	19	0.5%
Xu et al., 2013	34	0.8%
PPDB XXXL Phrasal	55	1.3%
PPDB S Phrasal	39	0.9%
Extracted by Paraphrase Approach A	45	1.1%
Extracted by Paraphrase Approach B	177	4.3%
Extracted by Unsupervised N-gram Approach	**317**	**7.6%**
Extracted by Silver-Standard Approach	2	0.0%

Table 5.9: Numbers of substitutions on the SemEval-15 test corpus using different synonyms sets.

substitutions when we used synonyms extracted using the Paraphrase Approach A (see Table 5.8). We also achieved 6.8% substitutions when we used synonyms extracted using the Unsupervised N-gram Approach. Our coverage on the training set was over 9 points higher than existing resources such as the PPDB.

Out of 4,142 total pairs of tweets in the SemEval-15 test set, 7.6% underwent substitutions when we used synonyms extracted using the Unsupervised N-gram Approach, a greater than 6 point gain over existing resources such as the PPDB (see Table 5.9). While the coverage is not as high as the highest coverage we achieved on the training set, it was enough to boost the F1 score (see Table 5.1). Using the Das and Smith (2009) system, we find a 2 point increase in precision, though a corresponding fall in recall causes the F1 score to remain constant.

5.2.2 Results from Paraphrase Approach B

363 substitutions on training set using synonyms extracted from the automatically created paraphrase corpus (created from test topics). 173 substitutions on the test set. While the coverage is better than previous systems (see Table 5.8), it is not sufficient to have a great effect on the accuracy and F1. The lack of coverage might be because of the filtering to produce the paraphrase corpus, which results in a limited set from which to draw synonyms. This, coupled with the lower confidence paraphrase labels, led to an accuracy of 73.83 and F1 score of 52.46, a slight decrease in accuracy and slight increase in F1 from the baseline (see Table 5.1).

5.2.3 Results form Unsupervised N-gram Approach

We find 784 substitutions on the training set and 317 substitutions on the test set. 74.07% and F1 of 53.14% when training on the training set and testing on the test set. While the accuracy is slightly lower than the baseline system, which only uses WordNet synonyms, the F1 score is 0.81 points above the baseline (see Table 5.1). This is the highest F1 score found when training on the training set and testing on the test set, but it not surprising because this method also achieves the greatest number of substitutions on the test set (see Table 5.8).

This demonstrates that it is possible to extract useful synonyms that are representative of the test set without using a pre-made paraphrase corpus.

5.2.4 Results form Silver-Standard Approach

Cross-validation on the silver standard corpus is not completely representative of the accuracy of our extracted synonyms. All of the synonyms in the silver corpus are drawn from WordNet, so they have significant differences from many of the synonyms we extract. For example, all of the synonyms from WordNet are unigrams, while many of the synonyms we extract are bigrams, trigrams, or fourgrams. Therefore, even though our cross-validation accuracies,

these do not necessarily translate into high accuracies for all of our synonyms.

Synonyms	Accuracy	F1
Madnani et al., 2012 (WordNet baseline)	76.01	58.89
Han et al., 2012	76.02	58.93
Ganitkevitch et al., 2012 (PPDB)	76.10	59.10
Xu et al., 2013	76.15	59.25
Current work	**76.82**	**61.18**

Table 5.10: Comparison of improvements from substituting synonym lexicons over the baseline system by Madnani et al. (2012) Madnani et al. (2012). Training and testing only on the training set.

Topic	Accuracy	F1
The Great Gatsby	94.53	88.79
A Walk To Remember	92.84	86.81
Ciara	92.84	85.94
TNT	92.92	86.97
The Nets	95.97	92.05

Table 5.11: Accuracy and F1 for synonym classification (cross-validation results on the silver-standard corpus).

For each topic in the Silver-Standard Corpus, we split the tweets so that 70% make up a training set and 30% make up a test set. We train the SVM classifier on the training set and report a random selection of our test results in Table 5.11. The accuracies are nearly all in the 90-95% range, and the F1 scores are in the 85-95% range.

5.3 Error Analysis

The Paraphrase Approach A mostly avoids classic errors such as eauating antonyms. However, some errors can be observed in Table 5.12

The Unsupervised N-gram Approach, which does not have the advantage of using the paraphrase labels, makes classic errors such as equating antonyms or related words (see Table 5.13). These words commonly appear in the same contexts, even though their meanings are different (Agirre et al., 2009).

Error Type	Example Synonyms
Unequal amounts of info	[welldeserved tony, tony award], [next weekend, weekend]
Unequal specificity	[white guy, andrew bogut], [saints, steve gleason]
Related phrases	[boston suspect, new york s mayor], [2 minutes, 3 minutes]
Nonsense	[sorry, soccer six bolton], [yaya, front camera]

Table 5.12: Erroneous synonyms extracted using the Paraphrase Approach A.

Error Type	Example Synonyms
Antonyms	[good, wrong], [best dunker, biggest bitch]
Related phrases	[week, year], [july 17, april 3]

Table 5.13: Erroneous synonyms extracted using the Unsupervised N-gram Approach.

5.4 Topic Vocabularies

Our results show that different sets of synonyms are used for different topics on Twitter. This implies that the vocabularies used for different topics are different. To test this hypothesis, we calculate the overlap in vocabulary between every two topics. Some topics are more

similar than others. For example, the topic *Miley Cyrus* (a pop singer) has low lexical overlap with hockey players *Josh Harding* or *Tanner Glass* but higher lexical overlap with *iTunes* and *Pandora*.

To measure this variation, we calculate the Jaccard similarity between two vocabularies by dividing the intersection of the vocabularies by their union, as shown in Figure 4.1.

Table 5.14 shows the topics whose vocabularies overlap the most with the vocabulary for the topic *Zach Randolph*. Zach Randolph is an American basketball player, and as expected, the top five most similar vocabularies are those of other basketball players and basketball coaches. The five least similar include businesses, a song, and unrelated phrases (see Table 5.15). Similar trends are observed for other topics.

Even very similar topics, such as *Great Gatsby* and *The Great Gatsby* only have a lexical overlap of only 48.58%. Most have much lower overlap, many below 1%. This indicates that searching for synonyms within one topic and applying them to another topic will not be as successful as searching within the same topic.

Topic	Overlap with *Zack Randolph*
Z-Bo	18.88%
Deandre Jordan	17.29%
Scott Brooks	16.71%
Reggie Miller	16.50%
Derek Fisher	15.67%

Table 5.14: Lexical overlap between topic *Zach Randolph* and **most** similar topics.

Topic	Overlap with *Zack Randolph*
Facebook	0.45%
iPad	0.66%
Sounds	0.98%
This Saturday	0.99%
Truly Yours 2	1.09%

Table 5.15: Lexical overlap between topic *Zach Randolph* and **least** similar topics.

Chapter 6

CONCLUSION

6.1 Discussion

When training on the training set and testing on the test set, the highest accuracy for the paraphrase detection task was found using the Paraphrase Approach A, in which synonyms were extracted from the SemEval-15 training set. The highest F1 score was found using the Paraphrase Approach B, in which synonyms were extracted using an unsupervised method with n-grams instead of shallow parse chunks. This indicates that while some syntactic information is useful, the errors associated with the parsers must be taken into account. While the POS tags have an accuracy of 93%, the parsers achieve only about 80% accuracy. Therefore, it makes sense to rely on the POS tags more heavily than the parsers, and this is demonstrated in the strong performance of the unsupervised n-gram method (see Section 5.1), which uses POS tags but not parses.

Our coverage of both the training set and test set is much better than previous methods, and that might be why our accuracy and F1 are also higher. Coverage is very important, and to achieve high coverage, you need to search for synonyms within the same subdomain. It is not sufficient to look at Twitter as a single domain; you must search within the distinct topics represented on Twitter.

We achieve higher coverage even though we extract fewer synonyms total than the other systems. This is because our synonyms are targeted towards the test set. This also improves the efficiency of the substitution of the synonyms back into the test set. The PPDB (Ganitkevitch et al., 2013) contains many more synonym pairs that our lists (the XXXL size contains 169 million paraphrase rules), yet we achieve better coverage, accuracy, and F1.

Our analysis of the overlap between the vocabularies of different topics supports our

conclusion that topic must be taken into account when extracting synonyms. The overlap between even similar topics is no higher than 50%, and between dissimilar topics, it is often below 1%. Since our methods of synonym extraction rely on finding matching context windows, dependency pairs, and/or POS tags, and because we are interested in synonyms that will provide high coverage for a specific topic, this lexical variation means that it is better to search within in a single topic's tweets rather than across other topics' tweets, even if more data (or more useful data, such as pivoting translations) is available.

Our lists of example synonyms show that we have extracted many synonyms that are specific to the context in which they are used. For example, *family* is not always a synonym of *washington redskins*, but they are synonyms when used in the context of a particular sporting event. This supports our theory, once again, that topics are important sources of information when extracting synonyms.

6.2 Summary

We have shown that automatically extracted synonyms can be used to improve the results of paraphrase detection on Twitter, if the synonyms' topics are sufficiently representative of the paraphrase test set. Our approach outperforms previous synonym resources both in accuracy and F1 for paraphrase detection and coverage on the training set, and it relies on a few simple intuitions that require no bilingual data or labeled synonyms. We have further shown that given a corpus of paraphrases, we can extract high-quality synonyms that avoid common errors such as equating antonyms. This technique could be applied to other domains outside of Twitter, so long as a corpus of labeled paraphrases is available.

Significantly, we have shown that for synonyms to be most effective on Twitter, they must be extracted from a corpus whose topics are similar to those in the test set. The language used on Twitter is organic and varied, so it is important to target the topics of interest; otherwise, the coverage will be too low for the synonyms to be useful.

We have also shown the utility of using syntactic information on Twitter, a medium which has traditionally been considered very difficult to parse or perform other evaluation.

High-accuracy resources for these tasks did not previously exist, but the advent of parsers such as TweeboParser and Alan Ritter's Twitter NLP Tools have allowed more sophisticated analysis, as described in this thesis. Even though these tools do introduce some errors, their accuracies are high enough that they are still useful. In particular, we show the importance of high accuracy POS tags on this task.

Finally, we have demonstrated two methods to bootstrap labeled paraphrase and synonym corpora for Twitter. We automatically created a paraphrase corpus and we created a silver standard synonym corpus using WordNet. In this way, we were able to take a distantly supervised approach and use a statistical classifier to extract the synonyms, and we were able to approximate the semantic information given by the paraphrase labels.

6.3 Future Work

To achieve more substantial results on the paraphrase detection, much work is still needed. While we have shown improvement on the task, we believe much greater improvements could be achieved if the synonyms achieved higher precision and recall. The more substitutions that can be made on the test set, the better the results, so long as the substitutions are accurate.

In particular, we would like to improve the silver standard approach and unsupervised n-gram approach, since these have the most potential to extract synonyms for an arbitrary set of topics. A more sophisticated technique to extract negative instances for the silver-standard training set could be very useful. We would also like to extend the positive instances beyond unigrams from WordNet, since these are not representative of the real candidate synonyms. Specifically, we would like to experiment with UrbanDictionary.com, a user-created dictionary of modern slang. This dictionary also includes phrasal synonyms and named entities, which would be useful supplements for WordNet. This dictionary could be used to seed the silver standard training set, or it could be used independently as a source of synonyms.

We would also like to explore more fully the possibilities of the dependency parse (for

example, we would like to experiment with using the dependency parse in the place of the shallow parse when constructing context windows), and we would like to experiment with looser restrictions on context window matching (for example, by stemming or allowing fuzzy matches).

This work shows potential for other tasks, such as named entity disambiguation. Many of the extracted synonyms are variations on the same named entity. For example, *philly*, *eagles*, and *birdgang* are synonyms all referring to the same sports team, the Philadelphia Eagles. Our system groups these n-grams and so could be used disambiguate the words and phrases used to refer to the entity. Our improvements on paraphrase detection could also yield improvements on tasks such as summarization, which relies on paraphrase detection to align the content of multiple texts.

BIBLIOGRAPHY

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27. Association for Computational Linguistics, 2009.

Colin Bannard and Chris Callison-Burch. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 597–604. Association for Computational Linguistics, 2005.

Joao Cordeiro, Gael Dias, and Pavel Brazdil. A metric for paraphrase detection. In *Computing in the Global Information Technology, 2007. ICCGI 2007. International Multi-Conference on*, pages 7–7. IEEE, 2007.

Dipanjan Das and Noah A Smith. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 468–476. Association for Computational Linguistics, 2009.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. PPDB: The paraphrase database. In *Proceedings of NAACL-HLT*, pages 758–764, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

Gintarė Grigonytė, Joao Cordeiro, Gaël Dias, Rumen Moraliyski, and Pavel Brazdil. Paraphrase alignment for synonym evidence discovery. In *Proceedings of the 23rd International*

Conference on Computational Linguistics, pages 403–411. Association for Computational Linguistics, 2010.

Weiwei Guo and Mona Diab. Modeling sentences in the latent space. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 864–872. Association for Computational Linguistics, 2012.

Bo Han, Paul Cook, and Timothy Baldwin. Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 421–432. Association for Computational Linguistics, 2012.

Zellig S Harris. Distributional structure. *Word*, 1954.

Paul Jaccard. The distribution of the flora in the alpine zone. *New phytologist*, 11(2):37–50, 1912.

Yangfeng Ji and Jacob Eisenstein. Discriminative improvements to distributional sentence similarity. In *EMNLP*, pages 891–896, 2013.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180. Association for Computational Linguistics, 2007.

Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A Smith. A dependency parser for tweets. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, to appear*, 2014.

Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady*, volume 10, page 707, 1966.

Nitin Madnani, Joel Tetreault, and Martin Chodorow. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 182–190. Association for Computational Linguistics, 2012.

Prodromos Malakasiotis. *Paraphrase and Textual Entailment Recognition and Generation*. PhD thesis, Ph. D. thesis, Department of Informatics, Athens University of Economics and Business, Greece, 2011.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38 (11):39–41, 1995.

Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51, 2003.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. Improved part-of-speech tagging for online conversational text with word clusters. In *HLT-NAACL*, pages 380–390, 2013.

Saša Petrović, Miles Osborne, and Victor Lavrenko. Using paraphrases for improving first story detection in news and twitter. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 338–346. Association for Computational Linguistics, 2012.

Chris Quirk, Chris Brockett, and William B Dolan. Monolingual machine translation for paraphrase generation. In *EMNLP*, pages 142–149, 2004.

Alan Ritter, Sam Clark, Oren Etzioni, et al. Named entity recognition in tweets: an experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534. Association for Computational Linguistics, 2011.

Maria Ruiz-Casado, Enrique Alfonseca, and Pablo Castells. Using context-window overlapping in synonym discovery and ontology extension. *Proceedings of RANLP-2005, Borovets, Bulgaria*, 39:43, 2005.

Temple F Smith and Michael S Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, pages 223–231, 2006.

Matthew G Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. Ter-plus: paraphrase, semantic, and alignment enhancements to translation edit rate. *Machine Translation*, 23(2-3):117–127, 2009.

Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y Ng. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*, pages 801–809, 2011.

Wei Xu, Alan Ritter, and Ralph Grishman. Gathering and generating paraphrases from twitter with application to normalization. In *Proceedings of the Sixth Workshop on Building and Using Comparable Corpora (BUCC)*, pages 121–128, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

Wei Xu, Alan Ritter, Chris Callison-Burch, William B. Dolan, and Yangfeng Ji. Extracting lexically divergent paraphrases from Twitter. *Transactions of the Association for Computational Linguistics*, 2014.

Fabio Massimo Zanzotto, Marco Pennacchiotti, and Kostas Tsioutsiouliklis. Linguistic redundancy in twitter. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 659–669. Association for Computational Linguistics, 2011.