Automatic Term Ambiguity Detection

Tyler Baldwin **Yunyao Li** **Bogdan Alexe** **Ioana R. Stanoi**
IBM Research - Almaden
650 Harry Road, San Jose, CA 95120, USA
{tbaldwi,yunyaoli,balexe,irs}@us.ibm.com

Abstract

While the resolution of term ambiguity is important for information extraction (IE) systems, the cost of resolving each instance of an entity can be prohibitively expensive on large datasets. To combat this, this work looks at ambiguity detection at the term, rather than the instance, level. By making a judgment about the general ambiguity of a term, a system is able to handle ambiguous and unambiguous cases differently, improving throughput and quality. To address the term ambiguity detection problem, we employ a model that combines data from language models, ontologies, and topic modeling. Results over a dataset of entities from four product domains show that the proposed approach achieves significantly above baseline F-measure of 0.96.

1 Introduction

Many words, phrases, and referring expressions are semantically ambiguous. This phenomenon, commonly referred to as polysemy, represents a problem for NLP applications, many of which inherently assume a single sense. It can be particularly problematic for information extraction (IE), as IE systems often wish to extract information about only one sense of polysemous terms. If nothing is done to account for this polysemy, frequent mentions of unrelated senses can drastically harm performance.

Several NLP tasks, such as word sense disambiguation, word sense induction, and named entity disambiguation, address this ambiguity problem to varying degrees. While the goals and initial data assumptions vary between these tasks, all of them attempt to map an instance of a term seen in context to an individual sense. While making a judgment for every instance may be appropriate for small or medium sized data sets, the cost of applying these ambiguity resolution procedures becomes prohibitively expensive on large data sets of tens to hundreds of million items. To combat this, this work zooms out to examine the ambiguity problem at a more general level.

To do so, we define an IE-centered ambiguity detection problem, which ties the notion of ambiguity to a given topical domain. For instance, given that the terms *Call of Juarez* and *A New Beginning* can both reference video games, we would like to discover that only the latter case is likely to appear frequently in non-video game contexts. The goal is to make a binary decision as to whether, given a term and a domain, we can expect every instance of that term to reference an entity in that domain. By doing so, we segregate ambiguous terms from their unambiguous counterparts. Using this segregation allows ambiguous and unambiguous instances to be treated differently while saving the processing time that might normally be spent attempting to disambiguate individual instances of unambiguous terms.

Previous approaches to handling word ambiguity employ a variety of disparate methods, variously relying on structured ontologies, gleaming insight from general word usage patterns via language models, or clustering the contexts in which words appear. This work employs an ambiguity detection pipeline that draws inspiration from all of these methods to achieve high performance.

2 Term Ambiguity Detection (TAD)

A term can be ambiguous in many ways. It may have **non-referential** senses in which it shares a name with a common word or phrase, such as in the films *Brave* and *2012*. A term may have referential senses **across topical domains**, such as *The Girl with the Dragon Tattoo*, which may reference either the book or the film adaptation. Terms may

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 804–809,
Sofia, Bulgaria, August 4-9 2013.

also be ambiguous **within a topical domain**. For instance, the term *Final Fantasy* may refer to the video game franchise or one of several individual games within the franchise. In this work we concern ourselves with the first two types of ambiguity, as within topical domain ambiguity tends to pose a less severe problem for IE systems.

IE systems are often asked to perform extraction over a dictionary of terms centered around a single topic. For example, in brand management, customers may give a list of product names and ask for sentiment about each product. With this use case in mind, we define the *term ambiguity detection* (TAD) problem as follows: Given a term and a corresponding topic domain, determine whether the term uniquely references a member of that topic domain. That is, given a term such as *Brave* and a category such as *film*, the task is make a binary decision as to whether all instances of Brave reference a film by that name.

2.1 Framework

Our TAD framework is a hybrid approach consisting of three modules (Figure 1). The first module is primarily designed to detect non-referential ambiguity. This module examines n-gram data from a large text collection. Data from The Corpus of Contemporary American English (Davies, 2008) was used to build our n-grams.

The rationale behind the n-gram module is based on the understanding that terms appearing in non-named entity contexts are likely to be non-referential, and terms that can be non-referential are ambiguous. Therefore, detecting terms that have non-referential usages can also be used to detect ambiguity. Since we wish for the ambiguity detection determination to be fast, we develop our method to make this judgment solely on the n-gram probability, without the need to examine each individual usage context. To do so, we assume that an all lowercased version of the term is a reasonable proxy for non-named entity usages in formal text. After removing stopwords from the term, we calculate the n-gram probability of the lower-cased form of the remaining words. If the probability is above a certain threshold, the term is labeled as ambiguous. If the term is below the threshold, it is tentatively labeled as unambiguous and passed to the next module. To avoid making judgments of ambiguity based on very infrequent uses, the ambiguous-unambiguous determination threshold is empirically determined by minimizing error over held out data.

The second module employs ontologies to detect across domain ambiguity. Two ontologies were examined. To further handle the common phrase case, Wiktionary[1] was used as a dictionary. Terms that have multiple senses in Wiktionary were labeled as ambiguous. The second ontology used was Wikipedia disambiguation pages. All terms that had a disambiguation page were marked as ambiguous.

The final module attempts to detect both non-referential and across domain ambiguity by clustering the contexts in which words appear. To do so, we utilized the popular Latent Dirichlet Allocation (LDA (Blei et al., 2003)) topic modeling method. LDA represents a document as a distribution of topics, and each topic as a distribution of words. As our domain of interest is Twitter, we performed clustering over a large collection of tweets. For a given term, all tweets that contained the term were used as a document collection. Following standard procedure, stopwords and infrequent words were removed before topic modeling was performed. Since the clustering mechanism was designed to make predictions over the already filtered data of the other modules, it adopts a conservative approach to predicting ambiguity. If the category term (e.g., film) or a synonym from the WordNet synset does not appear in the 10 most heavily weighted words for any cluster, the term is marked as ambiguous.

A term is labeled as ambiguous if any one of the three modules predicts that it is ambiguous, but only labeled as unambiguous if all three modules make this prediction. This design allows each module to be relatively conservative in predicting ambiguity, keeping precision of ambiguity prediction high, under the assumption that other modules will compensate for the corresponding drop in recall.

3 Experimental Evaluation

3.1 Data Set

Initial Term Sets We collected a data set of terms from four topical domains: *books*, *films*, *video games*, and *cameras*. Terms for the first three domains are lists of books, films, and video games respectively from the years 2000-2011 from dbpedia (Auer et al., 2007), while the initial terms

[1] http://www.wiktionary.org/

Tweet	Term	Category	Judgment
Woke up from a nap to find a beautiful mind on. #win	A Beautiful Mind	film	yes
I Love Tyler Perry ; He Has A Beautiful Mind.	A Beautiful Mind	film	no
I might put it in the top 1. RT @CourtesyFlushMo Splice. Top 5 worst movies ever	Splice	film	yes
Splice is a great, free replacement to iMove for your iPhone,	Splice	film	no

Table 1: Example tweet annotations.

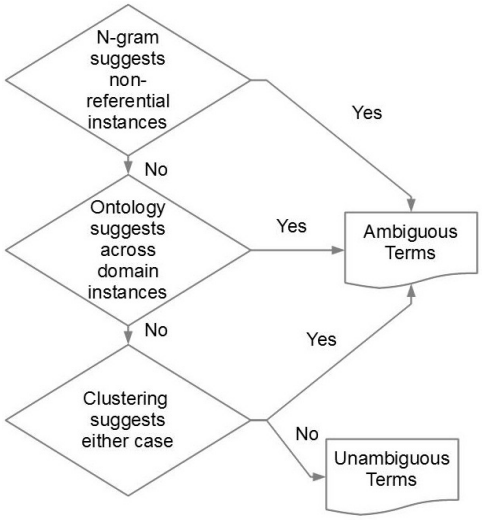

Figure 1: Overview of the ambiguity detection framework.

for *cameras* includes all the cameras from the six most popular brands on flickr[2].

Gold Standard A set of 100 terms per domain were chosen at random from the initial term sets. Rather than annotating each term directly, ambiguity was determined by examining actual usage. Specifically, for each term, usage examples were extracted from large amounts of Twitter data. Tweets for the *video game* and *film* categories were extracted from the TREC Twitter corpus.[3] The less common *book* and *camera* cases were extracted from a subset of all tweets from September 1st-9th, 2012.

For each term, two annotators were given the term, the corresponding topic domain, and 10 randomly selected tweets containing the term. They were then asked to make a binary judgment as to whether the usage of the term in the tweet referred to an instance of the given category. The degree of ambiguity is then determined by calculating the percentage of tweets that did not reference a member of the topic domain. Some example judgments are given in Table 1. If all individual tweet judgments for a term were marked as referring to a member of the topic domain, the term was marked as fully unambiguous within the data examined. All other cases were considered ambiguous.[4]

[2] http://www.flickr.com/cameras/

[3] http://trec.nist.gov/data/tweets/

Configuration	Precision	Recall	F-measure
Baseline	0.675	1.0	0.806
NG	0.979	0.848	0.909
ON	0.979	0.704	0.819
CL	0.946	0.848	0.895
NG + ON	0.980	0.919	0.948
NG + CL	0.942	0.963	0.952
ON + CL	0.945	0.956	0.950
NG + ON + CL	0.943	0.978	0.960

Table 2: Performance of various framework configurations on the test data.

Inter-annotator agreement was high, with raw agreement of 94% ($\kappa = 0.81$). Most disagreements on individual tweet judgments had little effect on the final judgment of a term as ambiguous or unambiguous, and those that did were resolved internally.

3.2 Evaluation and Results

Effectiveness To understand the contribution of the n-gram (NG), ontology (ON), and clustering (CL) based modules, we ran each separately, as well as every possible combination. Results are shown in Table 2, where they are compared to a majority class (ambiguous) baseline.

As shown, all configurations outperform the baseline. Of the three individual modules, the n-gram and clustering methods achieve F-measure of around 0.9, while the ontology-based module performs only modestly above baseline. Unsurprisingly, the ontology method is affected heavily by its coverage, so its poor performance is primarily attributable to low recall. As noted, many IE tasks may involve sets of entities that are not found in common ontologies, limiting the ability of the ontology-based method alone. Additionally, ontologies may be apt to list cases of strict ambiguity, rather than practical ambiguity. That is, an ontology may list a term as ambiguous if there are

[4] The annotated data is available at http://researcher.watson.ibm.com/researcher/view_person_subpage.php?id=4757.

several potential named entities it could refer to, even if the vast majority of references were to only a single entity.

Combining any two methods produced substantial performance increases over any of the individual runs. The final system that employed all modules produced an F-measure of 0.960, a significant ($p < 0.01$) absolute increase of 15.4% over the baseline.

Usefulness To establish that term ambiguity detection is actually helpful for IE, we conducted a preliminary study by integrating our pipeline into a commercially available rule-based IE system (Chiticariu et al., 2010; Alexe et al., 2012). The system takes a list of product names as input and outputs tweets associated with each product. It utilizes rules that employ more conservative extraction for ambiguous entities.

Experiments were conducted over several million tweets using the terms from the video game and camera domains. When no ambiguity detection was performed, all terms were treated as unambiguous. The system produced very poor precision of 0.16 when no ambiguity detection was used, due to the extraction of irrelevant instances of ambiguous objects. In contrast, the system produced precision of 0.96 when ambiguity detection was employed. However, the inclusion of disambiguation did reduce the overall recall; the system that employed disambiguation returned only about 57% of the true positives returned by the system that did not employ disambiguation. Although this reduction in recall is significant, the overall impact of disambiguation is clearly positive, due to the stark difference in precision. Nonetheless, this limited study suggests that there is substantial room for improvement in the extraction system, although this is out of the scope of the current work.

4 Related Work

Polysemy is a known problem for many NLP-related applications. Machine translation systems can suffer, as ambiguity in the source language may lead to incorrect translations, and unambiguous sentences in one language may become ambiguous in another (Carpuat and Wu, 2007; Chan et al., 2007). Ambiguity in queries can also hinder the performance of information retrieval systems (Wang and Agichtein, 2010; Zhong and Ng, 2012).

The ambiguity detection problem is similar to the well studied problems of named entity disambiguation (NED) and word sense disambiguation (WSD). However, these tasks assume that the number of senses a word has is given, essentially assuming that the ambiguity detection problem has already been solved. This makes these tasks inapplicable in many IE instances where the amount of ambiguity is not known ahead of time. Both named entity and word sense disambiguation are extensively studied, and surveys on each are available (Nadeau and Sekine, 2007; Navigli, 2009).

Another task that shares similarities with TAD is word sense induction (WSI). Like NED and WSD, WSI frames the ambiguity problem as one of determining the sense of each individual instance, rather than the term as a whole. Unlike those approaches, the word sense induction task attempts to both figure out the number of senses a word has, and what they are. WSI is unsupervised, relying solely on the information that surrounds word mentions in the text.

Many different clustering-based WSI methods have been examined. Pantel and Lin (2002) employ a clustering by committee method that iteratively adds words to clusters based on their similarities. Topic model-based methods have been attempted using variations of Latent Dirichlet Allocation (Brody and Lapata, 2009) and Hierarchical Dirichlet Processes (Lau et al., 2012). Several graph-based methods have also been examined (Klapaftis and Manandhar, 2010; Navigli and Crisafulli, 2010). Although the words that surround the target word are the primary source of contextual information in most cases, additional feature sources such as syntax (Van de Cruys, 2008) and semantic relations (Chen and Palmer, 2004) have also been explored.

5 Conclusion

This paper introduced the term ambiguity detection task, which detects whether a term is ambiguous relative to a topical domain. Unlike other ambiguity resolution tasks, the ambiguity detection problem makes general ambiguity judgments about terms, rather than resolving individual instances. By doing so, it eliminates the need for ambiguity resolution on unambiguous objects, allowing for increased throughput of IE systems on large data sets.

Our solution for the term ambiguity detection

task is based on a combined model with three distinct modules based on n-grams, ontologies, and clustering. Our initial study suggests that the combination of different modules designed for different types of ambiguity used in our solution is effective in determining whether a term is ambiguous for a given domain. Additionally, an examination of a typical use case confirms that the proposed solution is likely to be useful in improving the performance of an IE system that does not employ any disambiguation.

Although the task as presented here was motivated with information extraction in mind, it is possible that term ambiguity detection could be useful for other tasks. For instance, TAD could be used to aid word sense induction more generally, or could be applied as part of other tasks such as coreference resolution. We leave this avenue of examination to future work.

Acknowledgments

We would like to thank the anonymous reviewers of ACL for helpful comments and suggestions. We also thank Howard Ho and Rajasekar Krishnamurthy for help with data annotation and Shivakumar Vaithyanathan for his comments on a preliminary version of this work.

References

Bogdan Alexe, Mauricio A. Hernández, Kirsten Hildrum, Rajasekar Krishnamurthy, Georgia Koutrika, Meenakshi Nagarajan, Haggai Roitman, Michal Shmueli-Scheuer, Ioana Roxana Stanoi, Chitra Venkatramani, and Rohit Wagle. 2012. Surfacing time-critical insights from social media. In *SIGMOD Conference*, pages 657–660.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. 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. Springer-Verlag.

David Blei, Andrew Ng, and Micheal I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January.

Samuel Brody and Mirella Lapata. 2009. Bayesian word sense induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 103–111, Stroudsburg, PA, USA. Association for Computational Linguistics.

Marine Carpuat and Dekai Wu. 2007. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72.

Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 33–40, Prague, Czech Republic, June. Association for Computational Linguistics.

Jinying Chen and Martha Palmer. 2004. Chinese verb sense discrimination using an em clustering model with rich linguistic features. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Sriram Raghavan, Frederick Reiss, and Shivakumar Vaithyanathan. 2010. SystemT: An algebraic approach to declarative information extraction. In *ACL*, pages 128–137.

Mark Davies. 2008-. The corpus of contemporary american english: 450 million words, 1990-present. Avialable online at: `http://corpus.byu.edu/coca/`.

Ioannis P. Klapaftis and Suresh Manandhar. 2010. Word sense induction & disambiguation using hierarchical random graphs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 745–755, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jey Han Lau, Paul Cook, Diana McCarthy, David Newman, and Timothy Baldwin. 2012. Word sense induction for novel sense detection. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 591–601, Stroudsburg, PA, USA. Association for Computational Linguistics.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1):3–26.

Roberto Navigli and Giuseppe Crisafulli. 2010. Inducing word senses to improve web search result clustering. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 116–126, Stroudsburg, PA, USA. Association for Computational Linguistics.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):10:1–10:69, February.

Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In *Proceedings of the eighth*

ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '02, pages 613–619, New York, NY, USA. ACM.

Tim Van de Cruys. 2008. Using three way data for word sense discrimination. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 929–936, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yu Wang and Eugene Agichtein. 2010. Query ambiguity revisited: Clickthrough measures for distinguishing informational and ambiguous queries. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 361–364, Los Angeles, California, June. Association for Computational Linguistics.

Zhi Zhong and Hwee Tou Ng. 2012. Word sense disambiguation improves information retrieval. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–282, Jeju Island, Korea, July. Association for Computational Linguistics.