Guess Me if You Can: Acronym Disambiguation for Enterprises

Yang Li[1]*, **Bo Zhao**[2], **Ariel Fuxman**[1], **Fangbo Tao**[3]
[1]Google, Mountain View, CA, USA
[2]Pinterest, San Francisco, CA, USA
[3]Facebook, Menlo Park, CA, USA
{zheda2006liyang, bo.zhao.uiuc, afuxman, fangbo.tao}@gmail.com

Abstract

Acronyms are abbreviations formed from the initial components of words or phrases. In enterprises, people often use acronyms to make communications more efficient. However, acronyms could be difficult to understand for people who are not familiar with the subject matter (new employees, etc.), thereby affecting productivity. To alleviate such troubles, we study how to automatically resolve the true meanings of acronyms in a given context. Acronym disambiguation for enterprises is challenging for several reasons. First, acronyms may be highly ambiguous since an acronym used in the enterprise could have multiple internal and external meanings. Second, there are usually no comprehensive knowledge bases such as Wikipedia available in enterprises. Finally, the system should be generic to work for any enterprise. In this work we propose an end-to-end framework to tackle all these challenges. The framework takes the enterprise corpus as input and produces a high-quality acronym disambiguation system as output. Our disambiguation models are trained via distant supervised learning, without requiring any manually labeled training examples. Therefore, our proposed framework can be deployed to any enterprise to support high-quality acronym disambiguation. Experimental results on real world data justified the effectiveness of our system.

1 Introduction

Acronyms are abbreviations formed from the initial components of words or phrases (e.g., "AI" from "Artificial Intelligence"). As acronyms can shorten long names and make communications more efficient, they are widely used at almost everywhere in enterprises, including notifications, emails, reports and social network posts. Figure 1 shows a sample enterprise social network post. As we can see, acronyms are frequently used there.

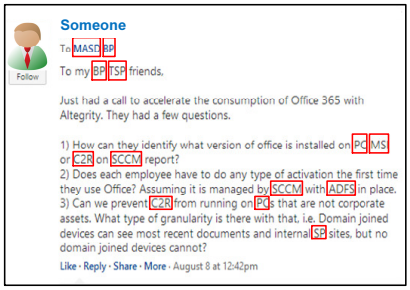


Figure 1: Acronyms in Enterprises

Despite the fact that acronyms can make communications more efficient, sometimes they could be difficult to understand, especially for people who are not familiar with the specific areas, such as new employees and patent lawyers. We randomly sampled 1000 documents from a Microsoft question answering forum and found out that only **7%** of the acronyms co-occur with the corresponding meanings in the same document, which means **93%** of the time when the user does not understand an acronym, she will need to find clues outside of the document. Therefore, it is particularly useful to develop a system that can automatically resolve the true meanings of acronyms in enterprise documents. Such system could be run online as a querying tool to handle any ad-hoc document, or run offline to annotate acronyms with their true meanings in a large corpus. In the offline mode, the true meanings can be further indexed by an enterprise search engine, so that when users search for the true meaning, documents containing the acronym can also be found.

The enterprise acronym disambiguation task is challenging due to the high ambiguity of acronyms, e.g., "SP" could stand for "Service Pack", "SharePoint" or "Surface Pro" in Microsoft. And there is one additional challenge compared with previous disambiguation tasks: in an enterprise document, an acronym could refer

*Work done while authors were at Microsoft Research.

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pages 1308–1317
Melbourne, Australia, July 15 - 20, 2018.

to either an internal meaning (concepts created by the enterprise that may or may not be found outside) or an external meaning (all concepts that are not internal). For example, regarding the acronym "AI", "Asset Intelligence" is an internal meaning mainly used only in Microsoft, while "Artificial Intelligence" is an external meaning widely used in public. A good acronym disambiguation system should be able to handle both internal and external meanings. As we will explain in details, it is important to make such distinction and different strategies are needed for such two cases.

For internal meanings, there are some previous work on word sense disambiguation (Navigli, 2009) and acronym disambiguation (Feng et al., 2009; Pakhomov et al., 2005; Pustejovsky et al., 2001; Stevenson et al., 2009; Yu et al., 2006) on a closed-domain corpus. The main challenge here is that there are rarely any domain-specific knowledge bases available in enterprises, therefore all the signals for disambiguation (including potential meanings, and their popularity scores, context representations, etc.) need to be mined from plain text. Training data should also be automatically generated to make the system easily scale out to all enterprises. Compared with previous work, we developed a more comprehensive and advanced set of features in the disambiguation model, and also used a much less restrictive way to discover meaning candidates and training data, so that both precision and recall can be improved. Moreover, one main limitation of all previous work is that they do not distinguish internal and external meanings. They merely rely on the enterprise corpus to discover information about external meanings, which we observe is quite ineffective. The reason is that for popular external meaning like "Artificial Intelligence", people often directly use its acronym in enterprises without explanation, therefore there is limited information about the connection between the acronym and the external meaning in the enterprise corpus. On the other hand, there are much more such information available in the public domain, which should be leveraged by the system.

If we consider utilizing a public knowledge base such as Wikipedia to better handle external meanings of acronyms, the problem becomes very related to the well studied Entity Linking (Ji and Grishman, 2011; Cucerzan, 2007; Dredze et al., 2010; Hoffart et al., 2011; Li et al., 2013, 2016; Ratinov et al., 2011; Shen et al., 2012) problem, which is to map entity mentions in texts to their corresponding entities in a reference knowledge base (e.g. Wikipedia). But our disambiguation task is different from the entity linking task, because the system also needs to handle internal meanings which are not covered by any knowledge bases, and ultimately needs to decide whether an acronym refers to an internal meaning or an external meaning. It is nontrivial to combine the information mined from the enterprise corpus and the public knowledge base so that the system can get the best of both worlds. For instance, we have tried to run an internal disambiguator (leveraging information mined from enterprise corpus) and then resort to a public entity linking system if the internal one's confidence is low, but the performance is very poor. Even for external meanings, it is important to leverage signals from the enterprise corpus since the context surrounding them could be quite different from that in the external world, and context is one of the most important factor for disambiguation. For example, in public world, when people mention "Operating System" they mainly talk about how to install or use it; while within Microsoft, when people mention "Operating System" most of the time they focus on how to design or implement it.

In this work, we design a novel, end-to-end framework to address all the above challenges. Our framework takes the enterprise corpus and certain public knowledge base as input and produces a high-quality acronym disambiguation system as output. The models are all trained via distant supervised learning, therefore our system requires no manually labeled training examples and can be easily deployed to any enterprises.

2 Problem Statement

The *Enterprise Acronym Disambiguation* problem is comprised of two sub-problems. The first one is *Acronym Meaning Mining* (Adar, 2004; Ao and Takagi, 2005; Park and Byrd, 2001; Schwartz and Hearst, 2002; Jain et al., 2007; Larkey et al., 2000; Nadeau and Turney, 2005; Taneva et al., 2013), which aims at mining acronym/meaning pairs from the enterprise corpus. Each meaning m should contain the full name expansion e, popularity score p (indicating how often m is used as the genuine meaning of acronym a) and context words W (i.e. words frequently used in context of the meaning). The popularity score and

context words can provide critical information for making disambiguation decisions. The second one is *Meaning Candidate Ranking*, whose goal is to rank the candidate meanings associated with the target acronym a and select the genuine meaning m based on the given context.

In this paper we assume the acronyms for disambiguation are provided as input to the system, either by the user or by an existing acronym detection module. We do not try to optimize the performance of acronym detection (e.g. identifying acronyms beyond the simple capitalized rule, or distinguishing cases where a capitalized term is not an acronym but a regular English word, such as "OK"). The task of acronym detection is also interesting and important. But due to the space limit, it is beyond the scope of this paper.

3 Framework

We propose a novel end-to-end framework to solve the *Enterprise Acronym Disambiguation* problem. Our framework takes the enterprise corpus as input and produces a high-quality acronym disambiguation system as output. Figure 2 shows the details of our proposed framework. In the mining module, we will sequentially perform Candidates Generation, Popularity Calculation, Candidates Deduplication and Context Harvesting on the input enterprise corpus. The details of these steps will be discussed in Section 4. After mining steps, we will get an acronym/meaning repository storing all the mined acronym/meaning pairs. Feed this repository together with the training data (automatically generated via distant supervision from the enterprise corpus) to the training module, we will get a candidate ranking model, a confidence estimation model and a final selection model. These models form the final acronym disambiguator and will be used in the testing module for actual acronym disambiguation. In the testing module, given the target acronym along with some context as input, the system will output the predicted meaning. Note that the mining and training module run offline once for the entire corpus or periodically when the corpus update, while the testing can be run online repeatedly for processing new documents.

4 Acronym Meaning Mining

4.1 Candidates Generation

As there is no reference dictionary or knowledge base available in enterprise telling us the potential

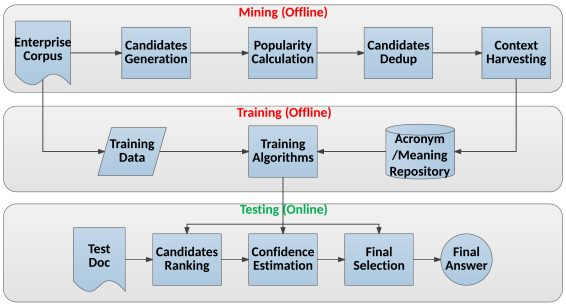


Figure 2: Framework

meanings of acronyms, we have to extract them from plain text. We propose a strategy called *Hybrid Generation* to balance extraction accuracy and coverage. Namely, we treat a phrase as a meaning candidate for an acronym if: (1) the initial letters of the phrase match the acronym **and** the phrase and the acronym co-occur in at least one document in the enterprise corpus; **or** (2) it is a valid candidate for the acronym in public knowledge bases (e.g. Wikipedia). The insight of this strategy is that the valid candidates missed by condition (1) are mainly public meanings which can be found in public knowledge bases. With this strategy we can make our system understand both the internal world and the external world.

4.2 Popularity Calculation

As mentioned in Section 2, for each candidate meaning, we need to calculate its popularity score, which reveals how often the candidate meaning is used as the genuine meaning of the acronym. In previous research on *Entity Linking*, popularity is calculated as the fraction of times a candidate being the target page for an anchor text in a reference knowledge base (e.g. Wikipedia). However, in enterprises, we do not have a knowledge base with anchor links. Thus we cannot calculate popularity in the same way. Here we propose to calculate two types of popularity to mimic the effect.

1. *Marginal Popularity*.
$$MP(m_i) = \frac{Count(m_i)}{\sum_{j=1}^{n} Count(m_j)}, \quad (1)$$
where $m_1, m_2, \ldots, m_n$ are the meaning candidates of acronym a and $Count(m_i)$ is the number of occurrences for m_i in the corpus.

2. *Conditional Popularity*.
$$CP(m_i) = \frac{Count(m_i, a)}{\sum_{j=1}^{n} Count(m_j, a)}, \quad (2)$$
where m_1, m_2, $\ldots$, m_n are the meaning candidates of acronym a and $Count(m_i, a)$ is the number of document-level co-occurrences for m_i and a in the corpus.

Conditional Popularity can more reasonably reveal how often the acronym is used to represent each meaning candidate. However, due to the data sparsity issue in enterprises, many valid candidates may get zero value for conditional popularity since they may never co-occur with the acronyms in the enterprise corpus. The Marginal Popularity does not have this problem since it is calculated from the raw counts of the candidates. Yet on the other hand, high marginal popularity score does not necessarily indicate high correlation between the candidate and the acronym. It is unclear how to combine the two scores into one popularity score, so we use both of them as features in the disambiguation model.

4.3 Candidates Deduplication

In enterprises, people often create many variants (including abbreviations, plurals or even misspellings) for the same meaning, therefore many mined meaning candidates are actually equivalent. For example, for the meaning "Certificate Authority" of the acronym "CA", the variants include "Cert Auth", "Certificate Authorities" and many others. It is important to deduplicate these variants before sending them to the disambiguation module. The deduplication helps aggregate disambiguation evidences and reduce noises. We design several heuristic rules[1] to perform the deduplication. Experiments show that the rules can accurately group the variants together. After grouping, we sort the variants within the same group based on their marginal popularity. The candidate with the largest marginal popularity is selected as the canonical candidate for the group. Other variants in the group will be deleted from the candidate list and their popularity scores will be aggregated to the canonical candidate. We maintain a table to record the variants for each canonical candidate.

4.4 Context Harvesting

In this step, we aim to harvest context words for each meaning candidate. These context words could be used to calculate context similarity with the query context. For each meaning candidate m, we put its canonical form and all its variants (from the variants table in Section 4.3) into set S. Then we scan the enterprise corpus, each time we find a match of any $e \in S$, we harvest the words in a width-W word window surrounding e as the context words of m. In our experiments we set window size as 30 after trying to vary the window size from 10 to 50 and finding 30 gives the best result.

[1]Due to space limitations, the detailed rules are omitted. Example rules are "word overlap percentage after stemming > 0.8", "corresponding component words share same prefix".

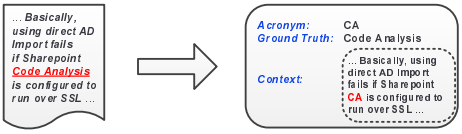


Figure 3: Distant Supervision Example

As mentioned before, some popular public meanings might be mentioned very rarely by their full names in the enterprise corpus since people directly use their acronyms most of the time. Therefore, the above context harvesting process can only get very few context words for those public meanings. To alleviate this, for each public meaning we add its Wikipedia page's content as complementary context. By doing so, we ensure almost all valid candidates get a reasonable amount of context words.

5 Meaning Candidate Ranking

5.1 Candidate Ranking

We first train a candidate ranking model to rank candidates with respect to the likelihood of being the genuine meaning for the target acronym.

5.1.1 Training Data Generation

In order to train a robust ranking model, we need to get adequate amount of labeled training data. Manually labeling is obviously too expensive and it requires a lot of domain knowledge, which severely limits our framework's generalization capability. To tackle this problem, we propose to automatically generate training data via distant supervision. The intuition is that since acronyms and the corresponding meanings are semantically equivalent, people use them interchangeably in enterprises. Therefore we can fetch documents containing the meaning, replace the meaning with the corresponding acronym and treat the meaning as ground truth. Figure 3 shows an example of this automatic training data generation process.

5.1.2 Training Algorithm

Any learning-to-rank algorithms can be used here. In our system we utilize the LambdaMART algorithm (Burges, 2010) to train the model.

5.1.3 Features

Now we explain the features we developed for the candidate ranking model. First, we have the *Marginal Popularity* score and *Conditional Popularity* score as two context-independent features, which could compensate for each other. However, as discussed in the previous section, some popular public meanings (e.g., "Artificial Intelligence") can be rarely mentioned in enterprise corpus by their full names, therefore both their marginal popularity score and conditional popularity score can be very low. To address this, we add a third feature called *Wiki Popularity*, which is calculated from Wikipedia anchor texts to capture how often an acronym refers to a public meaning in Wikipedia. The fourth feature we adopt is *Context Similarity*. We convert the harvested context for the meaning and the query context of the target acronym into TFIDF vectors and then compute their cosine similarity[2]. We also include two features (i.e. *LeftNeighborScore* and *RightNeighborScore*) to capture the effect of the immediate neighboring words, which are more important than further context words since immediate words could form phrases with the acronym. For example, if we see an acronym "SP" followed by the word "2", then likely it stands for "Service Pack". However, if we see "SP" followed by "2003", then probably its genuine meaning is "SharePoint". The last feature we use is *FullNamePercentage*. This feature is defined as the percentage of the meaning candidate's component words appearing in the context of the target acronym. Table 1 summarizes the features used to train the candidate ranking model.

5.2 Confidence Estimation

After getting the ranking results, we propose to apply a confidence estimation step to decide whether to trust the top ranked answer. There are two motivations behind. First, our candidate generation approach is not perfect, therefore we could encounter cases in which the genuine meaning is not in our candidates. For such cases, the top ranked answer is obviously incorrect. Second, our training data is biased towards the internal meanings since external meanings may rarely appear with full names. As a result, the learned ranking model may lack the capability to properly rank the external meanings. In such cases, we would better have the system return nothing rather than directly provide a wrong answer to mislead the user. In this step, we train a confidence estimation model, which will estimate the top result's confidence.

5.2.1 Training Data Generation

Similar to the ranker training, here the training data is also automatically generated. We run the learned ranker on some distant labeled data (generated from a different corpus), and then check if the top ranked answer is correct or not. If it is correct, we generate a positive training example; otherwise we make a negative training example.

5.2.2 Training Algorithm

Any classification algorithms can be used here. In our system we utilize the MART boosted tree algorithm (Friedman, 2000) to train the model.

5.2.3 Features

We design 7 features (summarized in Table 2) to train the confidence estimation model. There are two intuitions behind: (1) If the top-ranked answer's ranking score is very small, or the top-ranked answer's score is close to the second-ranked answer's score, then the ranking is not very confident; (2) If the acronym has a dominating candidate in the public domain (e.g., "Personal Computer" is the dominating candidate for "PC"), and the candidates' Wiki popularity distribution is significantly different from their marginal/conditional popularity distributions, then the ranker's output is not very confident. The first intuition covers the first 3 features, while the second intuition covers the last 4 features.

5.3 Final Selection

We have discussed that one particular motivation for confidence estimation is that the candidate ranking stage has some bias so it does not always rank public meanings at top when they are correct. Therefore, assuming the confidence estimation model can remove incorrect top-ranked result, we still need one additional step to decide if any public meaning is correct, which we call a final selection model. In this step, we determine whether to return the most popular public meaning (based on Wiki Popularity) as the final answer, and this step is only triggered when the confidence estimator judges that the ranking result is unconfident.

[2]One popular alternative to measure context similarity is using word embeddings (Mikolov et al., 2013; Li et al., 2015). In our system we experimented replacing TFIDF cosine similarity with word embedding similarity, or adding word embedding similarity as an additional feature, but both hurt the disambiguation accuracy. So we only included the TFIDF cosine similarity as the context similarity feature in our system.

Feature	Description
MarginalPopularity	The meaning candidate's marginal popularity score
ConditionalPopularity	The meaning candidate's conditional popularity score
WikiPopularity	The meaning candidate's Wiki popularity score
ContextSimilarity	TFIDF cosine similarity between meaning context and acronym context
LeftNeighborScore	Probability of acronym and meaning sharing the same immediate left word
RightNeighborScore	Probability of acronym and meaning sharing the same immediate right word
FullNamePercentage	Percentage of meaning candidate's component words appearing in acronym context

Table 1: Candidate Ranking Features

Feature	Description
Top1Score	Top 1 ranked meaning candidate's ranking score
Top1&2ScoreDiff	Difference between 1st and 2nd ranked meaning candidates' ranking score
Top1&2CtxSimDiff	Difference between 1st and 2nd ranked meaning candidates' context similarity score
Top1WikiPopularity	Top 1 ranked meaning candidate's Wiki popularity score
MaxWikiPopularity	Max Wiki popularity score among all the meaning candidates
MaxWP&MPGap	Max gap between Wiki and marginal popularity among all the meaning candidates
MaxWP&CPGap	Max gap between Wiki and conditional popularity among all the meaning candidates

Table 2: Confidence Estimation Features

The goal of the final selection model is similar to that of the confidence estimation model. In confidence estimation, we judge whether the top-ranked answer is correct; while in final selection, we check whether the most popular external meaning is correct. Thanks to this similarity, we can reuse the data, features and training algorithm in confidence estimation model. We take the same training data in Section 5.2.1 and update the labels correspondingly: if the genuine answer is the most popular external meaning, we generate a positive example; otherwise we make a negative one.

6 Experiments

6.1 Data

6.1.1 Mining and Training Corpus

We use both the Microsoft Answer Corpus (MAC) and the Microsoft Yammer Corpus (MYC) as the mining corpus. These corpus are kindly shared to us by Microsoft for research purpose. MAC contains 0.3 million web pages from a Microsoft internal question answering forum. MYC is consisted of 6.8 million posts from Microsoft's Yammer social network. In total, our mining module harvested 5287 acronyms and 17258 meaning candidates from this joint corpus.

For model training, the confidence estimation model and final selection model need to be trained on a different corpus than the candidate ranking model. So we train the candidate ranking model on MAC, with 12500 training examples being automatically generated; and train the confidence estimation and final selection model on MYC, with 40000 training instances being generated.

6.1.2 Evaluation Datasets

We prepared four datasets[3] for evaluation purposes. The first one ***Manual*** is obtained from the recent pages of Microsoft answer forum. Note these pages are disjoint from those used as mining/training corpus. We randomly sampled 300 pages and filtered out pages which do not contain ambiguous acronyms. After filtering, 240 test cases were left and we manually labeled them.

The second one ***Distant*** is generated via distant labeling on Microsoft Office365 documents. We sampled 2000 documents which contain at least one occurrence of a meaning candidate. Then we replaced the meanings with the corresponding acronyms and treat the meanings as ground truths. We manually checked through this dataset to remove some bad cases (e.g., "AS" for "App Store"). This resulted in a test set of 1949 test cases.

Comparing the ***Manual*** dataset with the ***Distant*** dataset, the ***Manual*** one, though in smaller size, can more accurately evaluate the system performance, since the target acronyms in it are sampled from the real distribution, while in the ***Distant*** dataset acronyms are artificially generated from

[3]Due to data confidentiality issue, we were unable to directly release these datasets.

randomly sampled meanings.

We also want to compare our method with the state-of-the-art Entity Linking (EL) systems based on public knowledge bases such as Wikipedia. However, it is unfair to directly compare as most enterprise specific meanings are unknown to them. Therefore, we need to only consider cases where the true meaning is a public meaning covered by both our system and the compared system. By filtering the distant dataset from Office365, we get the third dataset ***JoinW*** (1659 test cases) for comparing with the Wikifier (Ratinov et al., 2011), and the fourth dataset ***JoinA*** (237 test cases) for comparing with AIDA (Hoffart et al., 2011).

6.2 Compared Methods

6.2.1 Ablations of Our System

We compare the following ablations of our system, to illustrate the effectiveness of the features and components.

- **Internal Popularity (IP)**: Only the internal popularity features (i.e., marginal popularity and conditional popularity).
- **Popularity (P)**: The internal popularity features plus Wiki popularity features.
- **Popularity+Context (P+C)**: The popularity features plus context similarity feature.
- **Popularity+Context+Neighbbor (P+C+N)**: The popularity features, context similarity feature and immediate neighbor features.
- **Popularity+ Context+ Neighbbor+ Fullname (a.k.a. Candidate Ranker, or CR)**: Using all the features in candidate ranking module.
- **Candidate Ranker + Confidence Estimator (CR+ CE)**: Using the candidate ranking model plus the confidence estimation model.
- **Candidate Ranker + Confidence Estimator + Final Selector (a.k.a. Acronym Disambiguator, or AD)**: Using the candidate ranking model, the confidence estimation model and the final selection model. Full version of our system.

6.2.2 State-of-the-art EL Systems

We also compare our method with two state-of-the-art Entity Linking (EL) systems.

- **Wikifier**: a popular EL system using machine learning to combine various features together.
- **AIDA**: a robust EL system using mention-entity graph to find the best mention-entity mapping.

6.3 Quality of Mined Acronyms/Meanings

We first conduct experiments to evaluate the quality of the acronym/meaning pairs harvested through our offline mining module. Out of the 17258 mined pairs, we randomly sampled 2000 of them and asked 5 domain experts to manually check their validness. An acronym/meaning pair is considered as valid if the majority of the experts think the acronym is indeed used to abbreviate the meaning. For example, (*AS*, *Analysis Service*) is a valid pair, but (*AS*, *App Store*) is considered as invalid because people will not actually use *AS* to represent *App Store*. Among the sampled 2000 pairs, **94.5%** are labeled as valid, indicating our offline mining module can accurately extract acronym/meaning pairs from enterprise corpus. It is hard to precisely evaluate the coverage/recall of our mining method, since it is very difficult to obtain the complete meaning list for a given acronym. To get a rough idea, we randomly picked up 100 acronyms and asked the 5 domain experts to enumerate the valid meanings for these acronyms. In total we got 230 valid meanings and **all** of them are covered by the mined pairs.

6.4 Disambiguation Performance

We first conduct experiments to evaluate the disambiguation performance of our ranking model, and compare the helpfulness of the features used in the model. Figure 4 shows the precision (i.e., percentage of correctly disambiguated cases among all predicted cases), recall (i.e., percentage of correctly disambiguated cases among all test cases) and F_1 (i.e., harmonic mean of precision and recall) of the compared methods on the *Manual* dataset and the *Distant* dataset. In terms of the helpfulness of the features, the context similarity feature and the immediate neighbor features contribute most to the performance gain. Other features are less helpful, yet still bring improvements to the overall performance.

Next we conduct experiments to illustrate the effectiveness of the confidence estimation module and the final selection module in our system. Figure 5 shows the precision, recall and F_1 of the compared system configurations on the *Manual* and *Distant* dataset. As can be seen, the confidence estimation module can improve precision at the cost of hurting recall. Fortunately, the final selection module can recover some recall losses without sacrificing too much on precision. In

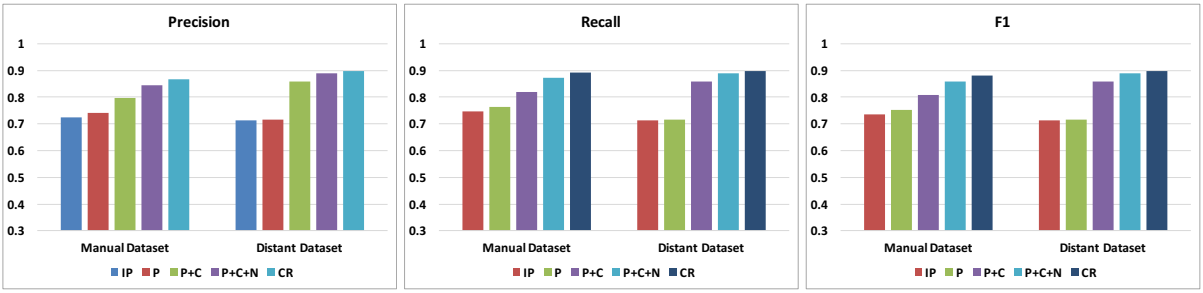


Figure 4: Ranking Performance

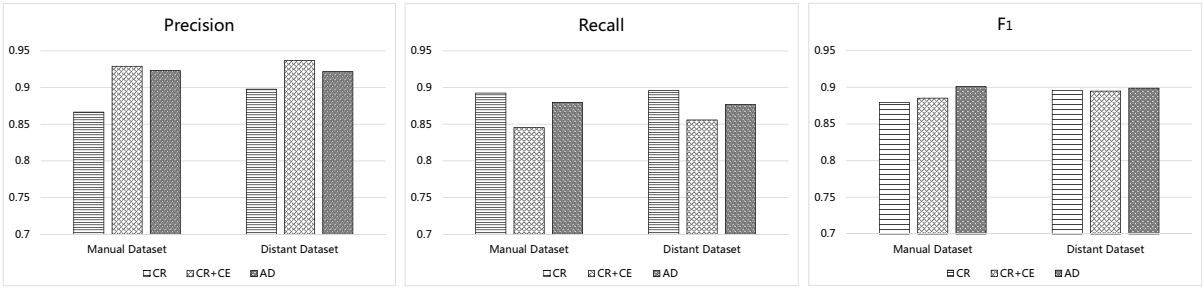
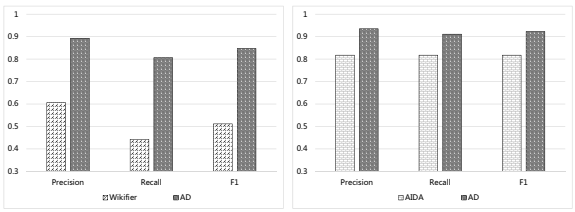


Figure 5: Effectiveness of Confidence Estimator and Final Selector

terms of the F_1 measure, the final system achieves the best performance.

Note that the ablation **P+C** naturally corresponds to the existing acronym disambiguation approaches (Feng et al., 2009; Pakhomov et al., 2005; Pustejovsky et al., 2001; Stevenson et al., 2009; Yu et al., 2006) mainly relying on context words and domain specific resources. These approaches do not specifically distinguish internal and external meanings. They merely rely on the internal corpus to discover information about external meanings, which is quite ineffective in the scenario of enterprise acronym disambiguation (as discussed in Section 1). In comparison, our system (**AD**) is able to leverage public resources together with the internal corpus to better handle the problem and therefore significantly outperforms them.



(a) *Wikifier vs. AD* (b) *AIDA vs. AD*

Figure 6: Comparison with EL Systems

6.5 Comparison with EL Systems

We also compare our system (**AD**) with two state-of-the-art Entity Linking (EL) systems: **Wikifier** and **AIDA**. As explained in Section 6.1.2, we made two datasets (i.e., *JoinW* and *JoinA*) for fair comparisons. Figure 6(a) and Figure 6(b) present the comparison of our AD system against Wikifer and AIDA, respectively. As we can see from the figures, AD significantly outperforms both Wikifier and AIDA on all three measures. The reason is that even for public meanings (e.g., Operating System) indexed by Wikifier and AIDA, the usage of them could be quite different in enterprises (e.g., inside Microsoft people talk more about designing Operating System rather than how to install it). Wikifier and AIDA utilize information from public knowledge bases (e.g., Wikipedia) to generate features, therefore can hardly capture such enterprise-specific signals. In contrast, our AD system mines disambiguation features directly from the enterprise corpus and utilizes them together with the public signals. As a result, it can more accurately represent the characteristics of the enterprise and lead to much better disambiguation performances.

7 Related Work

Acronym meaning discovery has received a lot of attentions in vertical domains (mainly in biomedical). Most of the proposed approaches (Adar, 2004; Ao and Takagi, 2005; Park and Byrd, 2001; Schwartz and Hearst, 2002; Wren et al., 2002) utilized generic rules or text patterns (e.g. brackets, colons) to discover acronym meanings. These methods are usually based on the assumption that

acronyms are co-mentioned with the corresponding meanings in the same document. However, in enterprises, this assumption rarely holds. Enterprises themselves are closed ecosystems, so it is very common for people to define the acronyms somewhere and use them elsewhere. As a result, such methods cannot be used for acronym meaning discovery in enterprises.

Recently, there have been a few works (Jain et al., 2007; Larkey et al., 2000; Nadeau and Turney, 2005; Taneva et al., 2013) on automatically mining acronym meanings by leveraging Web data (e.g., query sessions, click logs). However, it is hard to apply them directly to enterprises, since most data in enterprises are raw text and therefore the query sessions/click logs are rarely available.

Acronym disambiguation can be seen as a special case for the Entity Linking (EL) (Ji and Grishman, 2011; Dredze et al., 2010) problem. Approaches that link entity mentions to Wikipedia date back to Bunescu et. al's work (Bunescu and Paşca, 2006). They computed the cosine similarity between the text around the mention and the entity candidate's Wikipedia page. The referent entity with the maximum similarity score is selected as the disambiguation result. Cucerzan's work (Cucerzan, 2007) is the first one to realize the effectiveness of using topical coherence to globally perform EL. In that work, the topical coherence between the referent entity candidate and other entities within the same context is calculated based on their overlaps in categories and incoming links in Wikipedia. Recently, several methods (Hoffart et al., 2011; Li et al., 2013, 2016; Ratinov et al., 2011; Shen et al., 2012; Cheng and Roth, 2013) also tried to enrich "context similarity" and "topical coherence" using hybrid strategies. Shen et. al (Shen et al., 2015) provided a comprehensive survey for the techniques used in EL. However, these EL techniques cannot be used for acronym disambiguation in enterprises, since most enterprise meanings are not covered by public knowledge bases, and there are rarely any domain-specific knowledge bases available in enterprises. Automatic knowledge base construction (Suchanek et al., 2013) is promising, but the quality is far from applicable. Moreover, the structural information (e.g. entity taxonomy, cross-document hyperlinks) within Wikipedia, is rarely available in enterprises.

Most of the previous work (Feng et al., 2009; Pakhomov et al., 2005; Pustejovsky et al., 2001; Stevenson et al., 2009; Yu et al., 2006) on acronym disambiguation heavily rely on context words and domain specific resources. In comparison, our method explored a more comprehensive set of domain-independent features. Moreover, our method used a much less restrictive way to discover meaning candidates and training data, which is far more general than the methods relying on strict definition patterns (Schwartz and Hearst, 2002). Another particular limitation of all these previous work is that they do not distinguish internal and external meanings. They merely rely on the internal corpus to discover information about external meanings, which is quite ineffective.

8 Conclusions

In this paper, we studied the *Acronym Disambiguation for Enterprises* problem. We proposed a novel, end-to-end framework to solve this problem. Our framework takes the enterprise corpus as input and produces a high-quality acronym disambiguation system as output. The disambiguation models are trained via distant supervised learning, without requiring any manually labeled training examples. Different from all the previous acronym disambiguation approaches, our system is capable of accurately resolving acronyms to both enterprise-specific meanings and public meanings. Experimental results on Microsoft enterprise data demonstrated that our system can effectively construct acronym/meaning repositories from scratch and accurately disambiguate acronyms to their meanings with over 90% precision. Furthermore, our proposed framework can be easily deployed to any enterprises without requiring any domain knowledge.

References

Eytan Adar. 2004. Sarad: A simple and robust abbreviation dictionary. *Bioinformatics*, 20(4):527–533.

Hiroko Ao and Toshihisa Takagi. 2005. Alice: an algorithm to extract abbreviations from medline. *Journal of the American Medical Informatics Association*, 12(5):576–586.

Razvan Bunescu and Marius Paşca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of EACL*, pages 9–16.

Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11:23–581.

Xiao Cheng and Dan Roth. 2013. Relational inference for wikification. In *Proceedings of EMNLP*, pages 1787–1796.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of EMNLP-CoNLL*, pages 708–716.

Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity disambiguation for knowledge base population. In *Proceedings of COLING*, pages 277–285.

Shicong Feng, Yuhong Xiong, Conglei Yao, Liwei Zheng, and Wei Liu. 2009. Acronym extraction and disambiguation in large-scale organizational web pages. In *Proceedings of CIKM*, pages 1693–1696.

Jerome H Friedman. 2000. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of EMNLP*, pages 782–792.

Alpa Jain, Silviu Cucerzan, and Saliha Azzam. 2007. Acronym-expansion recognition and ranking on the web. In *Information Reuse and Integration*, pages 209–214.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of ACL*, pages 1148–1158.

Leah S Larkey, Paul Ogilvie, M Andrew Price, and Brenden Tamilio. 2000. Acrophile: an automated acronym extractor and server. In *Proceedings of ACM conference on Digital libraries*, pages 205–214.

Chao Li, Lei Ji, and Jun Yan. 2015. Acronym disambiguation using word embedding. In *Proceedings of AAAI*, pages 4178–4179.

Yang Li, Shulong Tan, Huan Sun, Jiawei Han, Dan Roth, and Xifeng Yan. 2016. Entity disambiguation with linkless knowledge bases. In *Proceedings of WWW*, pages 1261–1270.

Yang Li, Chi Wang, Fangqiu Han, Jiawei Han, Dan Roth, and Xifeng Yan. 2013. Mining evidences for named entity disambiguation. In *Proceedings of SIGKDD*, pages 1070–1078.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in NIPS*, pages 3111–3119.

David Nadeau and Peter D Turney. 2005. A supervised learning approach to acronym identification. In *Proceedings of CSCSI*, pages 319–329.

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

Serguei Pakhomov, Ted Pedersen, and Christopher G Chute. 2005. Abbreviation and acronym disambiguation in clinical discourse. In *AMIA Annual Symposium Proceedings*, pages 589–593.

Youngja Park and Roy J Byrd. 2001. Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of EMNLP*, pages 126–133.

James Pustejovsky, Jose Castano, Brent Cochran, Maciej Kotecki, Michael Morrell, and Anna Rumshisky. 2001. Extraction and disambiguation of acronym-meaning pairs in medline. *Medinfo*, 10(2001):371–375.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of ACL*, pages 1375–1384.

Ariel S Schwartz and Marti A Hearst. 2002. A simple algorithm for identifying abbreviation definitions in biomedical text. In *Biocomputing*, pages 451–462.

Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity linking with a knowledge base: Issues, techniques, and solutions. *Knowledge and Data Engineering, IEEE Transactions on*, 27(2):443–460.

Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. 2012. Linden: linking named entities with knowledge base via semantic knowledge. In *Proceedings of WWW*, pages 449–458.

Mark Stevenson, Yikun Guo, Abdulaziz Al Amri, and Robert Gaizauskas. 2009. Disambiguation of biomedical abbreviations. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, pages 71–79.

Fabian Suchanek, James Fan, Raphael Hoffmann, Sebastian Riedel, and Partha Pratim Talukdar. 2013. Advances in automated knowledge base construction. *SIGMOD Records*.

Bilyana Taneva, Tao Cheng, Kaushik Chakrabarti, and Yeye He. 2013. Mining acronym expansions and their meanings using query click log. In *Proceedings of WWW*, pages 1261–1272.

Jonathan D Wren, Harold R Garner, et al. 2002. Heuristics for identification of acronym-definition patterns within text: towards an automated construction of comprehensive acronym-definition dictionaries. *Methods of information in medicine*, 41(5):426–434.

Hong Yu, Won Kim, Vasileios Hatzivassiloglou, and John Wilbur. 2006. A large scale, corpus-based approach for automatically disambiguating biomedical abbreviations. *ACM Transactions on Information Systems*, 24(3):380–404.