Entity Linking with Effective Acronym Expansion, Instance Selection and Topic Modeling

Wei Zhang[†] **Yan Chuan Sim**[‡] **Jian Su**[‡] **Chew Lim Tan**[†]

[†]School of Computing
National University of Singapore
{z-wei, tancl}@comp.nus.edu.sg

[‡] Institute for Infocomm Research
{ycsim, sujian}@i2r.a-star.edu.sg

Abstract

Entity linking maps name mentions in the documents to entries in a knowledge base through resolving the name variations and ambiguities. In this paper, we propose three advancements for entity linking. Firstly, expanding acronyms can effectively reduce the ambiguity of the acronym mentions. However, only rule-based approaches relying heavily on the presence of text markers have been used for entity linking. In this paper, we propose a supervised learning algorithm to expand more complicated acronyms encountered, which leads to 15.1% accuracy improvement over state-of-the-art acronym expansion methods. Secondly, as entity linking annotation is expensive and labor intensive, to automate the annotation process without compromise of accuracy, we propose an instance selection strategy to effectively utilize the automatically generated annotation. In our selection strategy, an informative and diverse set of instances are selected for effective disambiguation. Lastly, topic modeling is used to model the semantic topics of the articles. These advancements give statistical significant improvement to entity linking individually. Collectively they lead the highest performance on *KBP-2010* task.

1 Introduction

Knowledge bases population (KBP) involves gathering information scattered among the documents to populate a knowledge base (KB) (e.g. Wikipedia). This requires either linking entity mentions in the documents with entries in the KB or highlighting these mentions as new entries to the current KB.

Entity linking [McNamee and Dang, 2009] involves both finding name variants (e.g. both *"George H. W. Bush"* and *"George Bush Senior"* refer to the 41st U.S. president) and name disambiguation (e.g. given *"George Bush"* and its context, we should be able to disambiguate which president in KB it is referring to).

In the name variant finding stage, expanding an acronym (all capitalized short-form word) from its context can effectively reduce the ambiguities of the acronym mention, under the assumption that two variants in the same document refer to the same entity. For example, *TSE* in Wikipedia refers to 33 entries, but with its full name *Tokyo Stock Exchange,* which is unambiguous, we can directly link it to the correct entry without the needs of disambiguation. However, only two simple rule based approaches have been used on entity linking so far. Han and Zhao [2009] only allow expansions adjacent to the acronym in parenthesis (e.g. *...Israeli Air Force (IAF) ...*). The other N-Gram approach [Varma *et al*., 2009] suffers from a low precision, because only one rule is used, that is, *"N"* continuous tokens have the same initials as the acronym.

Beyond entity linking, previous work on finding acronym expansions relies heavily on the presence of text markers such as "*long (short)", "short (long)"* [Pustejovsky *et al*., 2001; Taghva and Gilbreth, 1999; Schwartz and Hearst, 2003; Nadeau and Turney, 2005; Chang *et al*., 2002], the same as what is used by Han *et al.* [2009], or linguistic cues [Larkey *et al*., 2000; Park and Byrd, 2001] which are keywords like: *also called, known as, short for*. These systems are mainly built for biomedical literature, where acronyms are introduced in a more "formal" manner. However, in the newswire domain, acronym-expansion pairs often do not occur in the same sentence, nor do they follow the familiar pattern of being formed from its full form's leading characters. There are acronyms such as *CCP* (*Communist Party of China)*, *MOD/MINDEF/MD* either of which can stand for *Ministry of Defense*. Leading characters may be dropped; multiple letters from a full name could be used; or the order of the letters may be swapped, adding to the complexity of decoding acronyms. Only the following two exceptions use supervised learning approaches. Chang *et al*. [2002] use features that describe the alignments between the acronym and candidate expansion (i.e. whether acronym letters are aligned at the beginning of a word, syllable boundary, *etc*.). Nadeau and Turney [2005] incorporate part of speech information in addition to alignment information. However, the supervised learning approaches have the constraint that acronym and its candidate expansion are located in the same sentence. When extended to the rest of the document, they are greatly affected by noisy data and do not perform as well.

For the name disambiguation stage, Varma *et al*. [2009] rank the entries in KB through a Vector Space Model, which cannot combine bag of words with other useful features effectively. In contrast, current state-of-the-art entity linking systems are based on supervised learning approach [Dredze *et al*., 2010; Zheng *et al*., 2010]. However, they require many annotated training examples to achieve good performance. Entity linking annotation is expensive and labor intensive because of the large size of referred KB. Meanwhile, entity linking annotation is highly dependent on the KB. When a new KB comes, entity linking annotation process needs to be repeated. Zhang *et al*. [2010] have tried to automate this entity linking annotation process. However, as discussed in the paper, the distribution of the automatically generated data is not consistent with the real data set, because only some types of training instances can be generated.

In this paper, we propose three advancements for entity linking problem: (1) a supervised learning algorithm for finding flexible acronym's expanded forms in newswire articles, without relying solely on text markers or linguistic cues, (2) using an instance selection strategy to effectively utilize the auto-generated annotation and reduce the effect of distribution problem mentioned above and (3) effectively capturing the semantic information between document and KB entry by a topic model. We conduct empirical evaluation on *KBP-2010* data [Ji *et al*., 2010]. The evaluation results show that our acronym expansion method produces a 15.1% improvement over state-of-the-art methods. Meanwhile, these three advancements achieve statistical significant improvement to entity linking result individually. Collectively they give the highest performance on *KBP-2010* task.

The remainder of this paper is organized as follows. Section 2 introduces the frame work for entity linking. We present our machine learning method for acronym expansion and instance selection strategy for the automatically annotated data in Sections 3 and 4 respectively, and the usage of topic information feature in Section 5. Section 6 shows the experiments and discussions. Section 7 concludes our works.

2 Entity Linking Framework

Name variation resolution finds variants for each entry in KB and then generates the possible KB candidates for the given name mention by string matching. Name disambiguation is to map a name mention to the correct entry in candidate set.

2.1 Name Variation Resolution

Wikipedia contains many name variants of entities like confusable names, spelling variations, nick names *etc*. we extract the name variants of an entity in KB by leveraging the knowledge sources in Wikipedia: "titles of entity pages", "disambiguation pages" "redirect pages" and "anchor texts" [Zhang *et al.,* 2010]. With the acquired name variants for entries in KB, the possible KB candidates for a given name mention can be retrieved by string matching. Particularly, if the given mention is an acronym, we will expand it from the given document, and then use entity linking process. Section 3 will elaborate our proposed approach for acronym expansion.

2.2 Name Disambiguation

First, using a learning to rank method, we rank all the retrieved KB candidates to identify the most likely candidate. In this learning to rank method, each name mention and the associated candidates are formed by a list of feature vectors. The instances linked with *NIL* (no corresponding entry in KB) and the instances with only one KB candidate are removed from the training set. During linking, the score for each candidate entry is given by the ranker. The learning algorithm we used is ranking SVM [Herbrich *et al*., 2000].

Next, the preferred KB candidate is presented to a binary classifier to determine if it is believed as the target entry for a name mention. To determine the likelihood of a correct map from the name mention to the top candidate, we employ a SVM classifier [Vapnik, 1995], which returns a class label. From here, we can decide whether the mention and top candidate are linked. If not, the mention has no corresponding entry in KB (*NIL*).

The base features adopted for both learning to rank and classification include 15 feature groups divided to 3 categories. A summary of the features is listed in Table 1. The feature details can be found in [Zheng *et al*., 2010].

Categories	Feature Names
Surface	Exact Equal Surface, Start With Query, End With Query, Equal Word Num, Miss Word Num
Contextual	TF Sim Context, Similarity Rank, All Words in Text, NE Number Match, Country in Text Match, Country in Text Miss, Country in Title Match, Country in Title Miss, City in Title Match
Others	NE Type

Table 1: Base Feature Set

3 Acronym Expansion

In this section, we describe our algorithm for finding an acronym's expansion from a given document.

3.1 Identifying Candidate Expansions

Given a document D and acronym $A=a_1a_2...a_m$, which are usually capitalized words in the document, we want to find all possible candidate expansions. First, we add all strings found in the pattern "*A (string)*" to our set of candidates, C. Next, we find patterns of "*(A)*", and extract the longest contiguous sequence of tokens, E, before *"(A)"* that do not contain punctuations or more than 2 stop words. Our stop words are from a predefined list of commonly used prepositions, conjunctions, determiners and auxiliary. For example, in the sentence *John received an award from the Association for Computing Machinery (ACM)*, we extract *E=the Association for Computing Machinery*. We add E and all its sub-strings ending with the last token of E to C. Thus, *the Association for Computing Machinery, Association for Computing Machinery, for Computing Machinery, Computing Machinery* and *Machinery*, will be added to C.

To find expansions that are not captured by text markers, we search the document for all the tokens whose first letter matches the first letter of the acronym. When we encounter a token that starts with a_1, we extract the longest contiguous sequence of tokens that do not contain punctuations or more than 2 stop words. From this sentence, *the Association for Computing Machinery has granted the…*, for acronym *"ACM"*, we extract *E=Association for Computing Machinery has*, containing two stop words. Similarly, we consider all substrings of *E* that include the first token of *E*. Thus, *Assoc… Machinery has, Assoc… Machinery, Assoc… Computing, etc.*, will be added to *C*.

Using the above strategies, we would have a very large set of possible candidate expansions for a given acronym, about 8 negative candidates for each positive in our experiments. Hence, we rely on a classifier for selecting the correct candidate expansion.

Feature	Description
Conventional Features	
Acronym length	Length of the acronym *A*
First/last char match	1 if the first/last character of the acronym and the first/last candidate expansion are the same (case insensitive). 0 otherwise.
Consec left/right tokens	The number of tokens on the left/right of the candidate that are lowercase. 0 if the entire string is lowercase.
LCS full	Length of LCS between acronym and expansion string (case insensitive).
LCS lead	Length of LCS between acronym and leading characters of expansion.
Length diff	Absolute difference between acronym length and number of tokens in candidate.
Sentence dist	Number of sentences between the acronym and expansion
POS Features [Park and Byrd, 2001; Nadeau and Turney, 2005]	
PCD start/last	1 if the first/last token of the expansion is a preposition, conjunction or determiner. 0 otherwise.
PCD count	Number of prepositions, conjunctions and determiners in expansion.
Verb count	Number of verbs in the expansion.
New Features	
Leading match 1	Number of common characters (case insensitive) between the acronym and leading characters of the expansion. For example, <*CCP, Communist Party of China*> will be 3 for this feature.
Leading match 2	Same as Leading match 1 but case sensitive.
Sentence dist exp	*Exp(-|k|)* where *k* is the number of sentences between the acronym and expansion

Table 2: Acronym Expansion Feature Set

3.2 Feature Set for Supervised Learning

With the candidate expansion set *C* extracted from document *D* for acronym *A*, we apply a supervised learning algorithm to select expansions that are valid. We represent each of these candidate expansions in the form of feature vector, which can be used as input to supervised learning algorithms. As shown in Table 2, in addition to conventional features used by previous acronym decoding systems to capture alignment information between acronym and expansion, and part of speech features introduced and used in Park and Byrd [2001] and Nadeau and Turney [2005], we add new features that model swapped acronym letters and completely lowercase expansions. The features *Leading match 1* and *2* ignore the order of words in the expansion and measure the overlap between the expansion's leading characters and the acronym. This is to account for variations where the words are swapped, in the case of *Communist Party of China (CCP)*. For *the Sentence dist* feature, we use an exponential scale instead of a linear scale according to our experiments.

Each candidate acronym-expansion pair is presented to a SVM classifier. We return a *NIL* response if there are no positively classified expansions. Otherwise, the candidate with highest confidence score is selected.

4 Instance Selection Strategy

In this section, we explore the method to effectively utilize a large-scale auto-generated data. Zhang *et al.* [2010] propose automatically gathering large-scale training instances for entity linking. The basic idea is to take a document with an unambiguous mention referring to an entity *e1* in KB and replace it with its variation which may refer to *e1*, *e2* or others. For example, an entity mention "*Abbott Laboratories*" in a document only refers to one KB entry *"Abbott Laboratories*". We replace "*Abbott Laboratories*" in the document with its ambiguous synonyms, including "*Abbott*" "*ABT*", *etc*. Follow this approach, from the 1.7 million documents in *KBP-2010* text collection, we generate 45,000 instances.

However, the distribution of the auto-generated data is not consistent with the real data set, as the data generation process can only create some types of training instances. In the case of *"Abbott Laboratories"*, more than ten "*Abbott*" mentions are linked to "*Abbott Laboratories*" entry in KB, but no "*Abbott*" example is linked to other entries like "*Bud Abbott*" "*Abbott Texas*", *etc*. Thus, we use an instance selection strategy to select a more balanced subset from the auto-annotated instances and reduce the effect of the distribution problem. The approach of this instance selection strategy is similar to active learning [Brinker, 2003; Shen *et al*., 2004] for reducing the manual annotation effort on training instances through proposing only the useful candidates to annotators. As we already have a large set of automatic generated training instances, the selection here is a fully automatic process to get a useful and more balanced subset.

We use the SVM classifier mentioned in Section 2.2 to select the instances from the auto-generated data set. The initial classifier can be trained on a set of initial training instances, which can be a small part of the whole automatic generated data. We want to select an informative and diverse batch of instances and add them to the current training instance set at each iteration to further adjust current hyperplane for more accurate classification in an iterative process.

We use the distance of instances to the hyperplane as the measure to select the informative instances. The distance of an instance's feature vector to the hyperplane is computed as follows:

$$Dist(w) = |\sum_{i=1}^{N} \alpha_i y_i k(s_i, w) + b| \qquad (1)$$

Where $\boldsymbol{w}$ is the feature vector of the instance, α_i, y_i, s_i corresponds to the weight, the class and the feature vector of the i^{th} support vector respectively. $\boldsymbol{N}$ is the number of the support vectors in current model.

Clearly, the current classifier is not able to reliably classify the instance closed to the hyperplane. Thus, the instance with the smallest *Dist(w)* to the current hyperplane will be selected first to the batch. The most informative instances in the remaining set will be compared individually with the selected instances in the batch to make sure their similarity is less than a threshold β. This is to diverse the training instance in the batch to maximize the contribution of each instance, we set β to the average similarity between the instances in the original data set.

When a batch of 80 instances in our experiment is selected, we add them to the training instance set and retrain the classifier. Such a batch learning process will iterate until the entity linking performance of the current classifier and the classifier in the last iteration converge on a development set.

5 Incorporating Semantic Feature

The previous approaches treat the context of the mention as a bag of words, n-grams, noun phrases or/and co-occurring named entities, and measures context similarity by the comparison of the weighted literal term vectors [Varma et al., 2009; Zhang et al., 2010; Zheng et al., 2010; Dredze et al., 2010]. Such literal matching suffers from the problems: lack of semantic information and sparsity issues. Thus, we introduce a topic model – latent Dirichlet allocation (LDA) [Blei *et al*, 2003] to entity linking to discover the underlying topics of documents and KB entries.

LDA is a three-level hierarchical Bayesian model. Each document is represented as a set of *N* words, and the collection has *M* documents. Each word *w* in a document is generated from a topic distribution, which is a multinomial distribution over words. The topic indicator *Z* of the word *w* is assumed to have a multinomial distribution $\vec{\theta}$ over topics, which in turn has a Dirichlet prior with hyperparameter $\vec{\alpha}$.

For our task, we trained LDA models on KB, where the text of each entry is treated as a document. Once the model is trained, we map the document where the name mention appears, to the hidden topic space by calculating the topic proportions $\vec{\theta}$. Then, the topic probability of each KB entry and the mention document is learned. Thus, we can calculate the context similarity in the *K*-dimensional topic space by their Hellinger distance as below:

$$Similarity_{d,e} = \sum_{k=1}^{K} (\sqrt{\vec{\theta}_{d,k}} - \sqrt{\vec{\theta}_{e,k}})^2 \qquad (2)$$

Finally, such semantic similarity can be combined with other term matching features to SVM ranker and classifier for entity linking.

6 Experiments and Discussions

6.1 Experimental Setup

In our study, we use *KBP-2010*[1] KB and document collection to evaluate our approach. The KB has been auto-generated from Wikipedia. Each KB entry consists of the Wikipedia Infobox[2] and the corresponding Wikipedia text. The KB contains 818,741 entries and the document collection contains 1.7 million documents. The *KBP-2010* name mentions have 5,404 training instances and 2,250 test instances, across three named entity types: Person, Geo-Political Entity and Organization. The documents containing these mentions are from newswire and blog text. We randomly select 500 examples from the training set as our development data set for instance selection. For pre-processing, we perform Named Entity Recognition using a SVM based system trained and tested on ACE 2005 with 92.5(P) 84.3(R) 88.2(F). In our implementation, we use the binary SVM$^{\text{Light}}$ by Joachims [1999] and Stanford Topic Modeling Toolbox[3] with default learning parameters. In selection process, we select the KB entries with more than 15 linked documents in the auto-generated data as our initial training set (1,800 instances) to train the initial classifier. We adopt micro-averaged accuracy used in *KBP-2010* to evaluate our Entity Linker, i.e. the number of correct links divided by the total number of mentions.

6.2 System with Acronym Expansion

We semi-automatically collect training instances (170 instances) for our acronyms classifier from *KBP* document collection. We mine name mentions which are acronyms, and for each of these mentions, we find in the same document for its expansion. If we do not find the expansion in the document, we label it as *NIL*. Our test instances (1,042 instances) for the experiment are the acronyms in the set of *KBP-10* name mentions. Their expansions are the names of linked entries in KB.

To benchmark the performance of our approach, we implement Schwartz and Hearst's [2003] and Nadeau and Turney's [2005] algorithms, and evaluate it against our test data. Schwartz and Hearst's algorithm is the approach used by previous entity linking systems. On the other hand, Nadeau and Turney [2005] represent the recent advancement of acronym expansion. In their paper, they search for candidate expansions within the same sentence as the acronym. We expand the search space for their algorithm to include expansions from the rest of the text.

Table 3 shows the results on our test instances containing 306 *NIL*, 368 *(A)* and 368 *FREE* acronyms. Using the *Leading match* features significantly improves the accuracy

[1] http://nlp.cs.qc.cuny.edu/kbp/2010/
[2] http://en.wikipedia.org/wiki/Template:Infobox
[3] http://nlp.stanford.edu/software/tmt/tmt-0.3/

of our classifier for all types of acronyms. The *sent dist exp* feature is especially useful for *FREE* acronyms, where it's relevant. We see that our approach for decoding acronyms achieves 15.1% or more significant improvement over Nadeau's and Schwartz's algorithms. Our classifier is able to do better for *(A)* acronyms because it can better capture the variability of expansions in newswire and web domain. Similarly, for *FREE* acronyms, our classifier is also significantly better.

Method	All	NIL	(A)	FREE
Conventional + POS	82.4	95.8	78.5	75.0
+leading match	88.9	97.1	87.0	84.0
+sent dist exp	**92.9**	98.8	90.0	91.6
Nadeau$^+$	77.8	98.0	71.6	67.0
Schwartz	54.3	98.0	72.8	0

Table 3: Results of Acronym Expansion. *NIL* are acronyms whose expansions are not found in the document. (A) refers to acronyms and expansions that are adjacent. FREE refers to expansions that can be found anywhere in the document and are not adjacent to the acronym.

Methods	ALL	NIL	Non-NIL	ORG	GPE	PER
No Algo	76.1	94.0	50.9	76.5	73.5	40.0
Schwartz	77.6	95.2	52.9	78.1	73.5	40.0
Nadeau$^+$	79.4	95.8	56.4	80.0	73.5	40.0
Ours	**82.8**	96.8	63.2	83.5	76.5	40.0

Table 4: Results of Entity Linking for Acronym Mentions

We further conduct experiments on our 1,042 test instances to test the effectiveness of our acronym expansion method for entity linking. As mentioned in Section 1, previous work only uses some rule-based methods for entity linking. Table 4 lists the performance of entity linking with overall accuracy as well as accuracy on subsets of the data. It compares our method with no acronym process (*No Algo*), the rule based method [Schwartz and Hearst, 2003], and the extension of a machine learning method [Nadeau and Turney, 2005] (*Nadeau*$^+$). This table shows that our method achieves significant improvements over the other three methods ($\rho < 0.05$, χ^2 testing). Table 5 shows the reason of the improvements. Using our expansion method, the number of retrieved candidates for each name mention is only 4.2 on average, which means less ambiguity ($\rho < 0.05$, χ^2 testing). Meanwhile, this smaller candidate set does not compromise the recall. Our acronym expansion method will be used in the following experiments for instance selection and semantic features.

Methods	Recall	Avg. # of Candidates
No Algo	92.9	9.40
Schwartz	94.0	7.16
Nadeau$^+$	94.1	6.68
Ours	**94.1**	**4.20**

Table 5: Results of Candidates Retrieve

6.3 System with Instance Selection

Table 6 compares the performance of different training sets on *KBP-10* testing data. Row 1 (*All*) uses the data set auto-generated by [Zhang *et al.*, 2010]. Row 2 is trained on *KBP-10* training set which is manually annotated. Row 3 (*SubSet*) refers to the subset selected from the auto-generated set by instance selection strategy in Section 4.

Methods	ALL	NIL	Non-NIL	ORG	GPE	PER
All	81.2	81.8	80.5	80.8	72.5	90.3
KBP	82.8	84.3	80.9	82.3	76.1	90.0
SubSet	**83.6**	85.3	81.6	82.7	76.2	91.9

Table 6: Results of Entity Linking for Instance Selection

Comparing Row 3 with Row 1, our instance selection process gives significant improvements ($\rho < 0.05$ by χ^2 testing) to entity linking. This proves that the selection process makes the training set more balanced. By comparing Row 3 with Row 2, our method achieves better performance without hard intensive work on annotating 5,404 articles. In Figure 1, we train the system on different number of instances from full *KBP10* training set to 10%. With the decrease in training data, the accuracy is also decreasing. The performance would be significantly different from our method at 50%. This means that systems using manually annotated data need more than 2702 training examples to perform as well as our fully automatic instance generation.

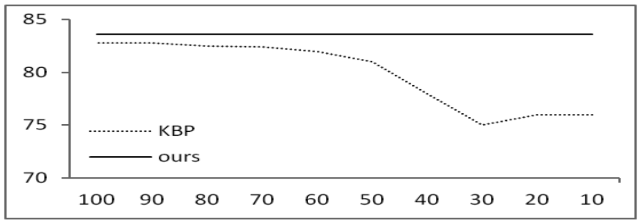

Figure 1: Learning curves of systems

6.4 System with Semantic Features

Table 7 shows the effectiveness of semantic features on *KBP-10* test data. The first row only uses the base features in Section 2.2 which treats the context as literal term vectors. The second row reports the results incorporating semantic features. Both of them are trained on the selected subset of auto-generated data.

Features	ALL	NIL	Non-NIL	ORG	GPE	PER
Baseline	83.6	85.3	81.6	82.7	76.2	91.9
+ Semantic	**86.1**	89.2	82.3	85.2	79.4	93.6

Table 7: Results of Entity Linking for Semantic Features

We can see that using topic model outperforms the baseline system significantly, which models the context similarity as literal term matching ($\rho < 0.05$ by χ^2 testing). This proves that the underlying topic of documents has good disambiguation power for entity linking. As discussed in Section 5, literal matching suffers from sparsity issue. Topic model can help to solve this problem.

Finally, we compare our method with the top 7 systems[4] in *KBP-2010* shared task [Ji *et al*., 2010]. As shown in Figure 2, the first column shows the performance of our system for entity linking, which outperforms the best solution.

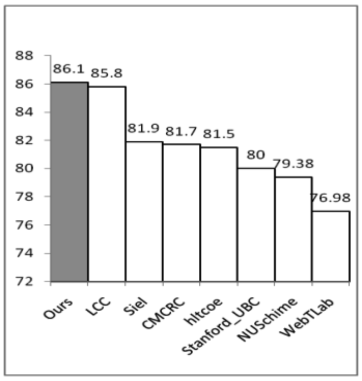

Figure 2: A comparison with *KBP-10* systems

7 Conclusion

In our paper, we propose a supervised learning algorithm for acronym expansion, which is able to give statistical significant improvement over state-of-the-art acronym expansion methods. Furthermore, we propose using an instance selection strategy to reduce the effect of distribution problem in auto-generated data. Thus, the entity linking system achieves better performance without hard labor. Finally, a topic model is introduced to entity linking, which can discover the semantic knowledge between words.

Acknowledgment

This work is partially supported by Microsoft Research Asia eHealth Theme Program.

References

[Blei *et al*., 2003] D. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research 3:993-1022,* 2003.

[Brinker, 2003] K. Brinker. Incorporating Diversity in Active Learning with Support Vector Machines. In *Proceeding of ICML*. 2003

[Chang *et al*., 2002] J.T. Chang *et al*. Creating an online dictionary of abbreviations from MEDLINE. *Journal of the American Medical Informatics Association*, 9(6):612.

[Dredze *et al*., 2010] M. Dredze, *et al*. Entity Disambiguation for Knowledge Base Population. *23rd International Conference on Computational Linguistics, 2010, China*

[Herbrich *et al.,* 2000] R. Herbrich, T. Graepel, and K. Obermayer. Large Margin Rank Boundaries for Ordinal Regression. *Advances in Large Margin Classifiers, 2000*

[Han and Zhao, 2009] X. Han and J. Zhao. NLPR_KBP in TAC 2009 KBP Track: A Two-Stage Method to Entity Linking. *Test Analysis Conference , 2009.*

[Ji *et al., 2010*] H. Ji, *et al*. Overview of the TAC 2010 Knowledge Base Population Track. *Test Analysis Conference. 2010.*

[Joachims, 1999] T. Joachims. Making large-Scale SVM Learning Practical. *Advances in Kernel Methods - Support Vector Learning,* MIT Press, 1999.

[Larkey *et al*., 2000] L.S. Larkey *et al.* Acrophile: an automated acronym extractor and server. In *Proceedings of the fifth ACM conference on Digital libraries*.

[McNamee and Dang, 2009] P. McNamee and H. T. Dang. Overview of the TAC 2009 Knowledge Base Population Track. In *Proceeding of Text Analysis Conference 2009.*

[Nadeau and Turney, 2005] D. Nadeau and P.D. Turney. A supervised learning approach to acronym identification. *Advances in Artificial Intelligence*, (May):319–329.

[Park and Byrd, 2001] Youngja Park and R.J. Byrd. Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of EMNLP*, 2001.

[Pustejovsky *et al*., 2001] J Pustejovsky *et al*. Automatic extraction of acronym meaning pairs from MEDLINE databases. *Studies in health technology and informatics.*

[Schwartz and Hearst, 2003] A. Schwartz and M. Hearst. A simple algorithm for identifying abbreviation definitions in biomedical text. *Pacific Symposium on Biocomputing*.

[Shen *et al*., 2004] D. Shen, *et al*. Multi-Criteria-based Active Learning for Named Entity Recognition. In *Proceedings of the ACL 2004.*

[Taghva and Gilbreth. 1999] Kazem Taghva and Je Gilbreth. Recognizing acronyms and their definitions. *International Journal on Document Analysis and Recognition*.

[Vapnik, 1995] V. Vapnik. The Nature of Statistical Leaning Theory. *Springer-Verlag*, *New York. 1995*

[Varma *et al*., 2009] V. Varma *et al*. IIIT Hyderabad at TAC 2009. In *Proceedings of Test Analysis Conference 2009*.

[Zhang *et al*., 2010] W. Zhang, *et al*. Entity Linking Leveraging Automatically Generated Annotation. *International Conference on Computational Linguistics. China.*

[*Zheng et al.,* 2010] Z. Zheng, *et al*. Learning to Link Entities with Knowledge Base. *The North American Chapter of the Association for Computational Linguistics. 2010.*

[4] Another system submission shows 86.8%. However, it accesses web which is not allowed in KBP benchmark as the purpose is to develop a standalone system, which is our focus here as well.