JHU1 : An Unsupervised Approach to Person Name Disambiguation using Web Snippets

Delip Rao **Nikesh Garera** **David Yarowsky**
Dept. of Computer Science
Johns Hopkins University
Baltimore, MD 21218
{delip, ngarera, yarowsky}@cs.jhu.edu

Abstract

This paper presents an approach to person name disambiguation using K-means clustering on rich-feature-enhanced document vectors, augmented with additional web-extracted snippets surrounding the polysemous names to facilitate term bridging. This yields a significant F-measure improvement on the shared task training data set. The paper also illustrates the significant divergence between the properties of the training and test data in this shared task, substantially skewing results. Our system optimized on $F_{0.2}$ rather than $F_{0.5}$ would have achieved top performance in the shared task.

1 Introduction

Being able to automatically distinguish between John Doe, the musician, and John Doe, the actor, on the Web is a task of significant importance with applications in IR and other information management tasks. Mann and Yarowsky (2004) used bigographical data annotated with named entitities and perform fusion of extracted information across multiple documents. Bekkerman and McCallum (2005) studied the problem in a social network setting exploiting link topology to disambiguate namesakes. Al-Kamha and Embley (2004) used a combination of attributes (like zipcodes, state, etc.), links, and page similarity to derive the name clusters while Wan et. al. (2005) used lexical features and named entities.

2 Approaches

Our framework focuses on the K-means clustering model using both bag of words as features and various augumented feature sets. We experimented with several similarity functions and chose Pearson's correlation coefficient[1] as the distance measure for clustering. The weights for the features were set to the term frequency of their respective words in the document.[2]

2.1 Submitted system: Clustering using Web Snippets

We queried the Google search engine with the target person names and extracted up to the top one thousand results. For each result we also extracted the snippet associated with it. An example is shown below in Figure 2.1. As can be seen the

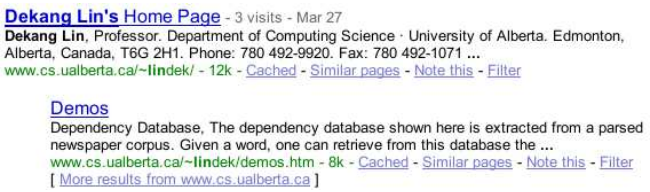

Figure 1: Google snippet for "Dekang Lin"

snippets contain high quality, low noise features that could be used to improve the performance of the system. Each snippet was treated as a document and

[1] This performs better than the standard measures like Euclidean and Cosine with K-means clustering on this data.

[2] We found that using TF weights instead of TF-IDF weights gives a better performance on this task.

Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007), pages 199–202,
Prague, June 2007. ©2007 Association for Computational Linguistics

clustered along with the supplied documents. This process is illustrated in Figure 2. The following example illustrates how these web snippets can improve performance by lexical transitivity. In this hypothetical example, a short test document contains a Canadian postal code (T6G 2H1) not found in any of the training documents. However, there may exist an additional web page not in the training or test data which contains both this term and also overlap with other terms in the training data (e.g. 492-9920), serving as an effective transitive bridge between the two.

Training Document 1	492-9920, not(T6G 2H1)
Web Snippet 2	both 492-9920, T6G 2H1
Test Document 3	T6G 2H1, not(492-9920)

Thus K-means clustering is likely to cluster the three documents above together while without this transitive bridge the association between training and test documents is much less strong. The final clustering of the test data is simply a projection with the training documents and web snippets removed.

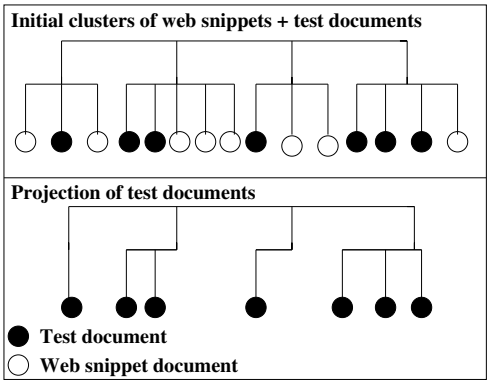

Figure 2: Clustering using Web Snippets

2.2 Baselines

In this section we describe several trivial baselines:

1. **Singletons:** A clustering where each cluster has only one document hence number of clusters is same as the number of documents.

2. **One Cluster:** A clustering with only one cluster containing all documents.

3. **Random:** A clustering scheme which partitions the documents uniformly at random into K clusters, where the value of K were the optimal K on the training and test data.

These results are summarized in Table 1. Note that all average F-scores mentioned in this table and the rest of the paper are microaverages obtained by averaging the purity and invese purity over all names and then calculating the F-score.

	Train		Test	
Baseline	$F_{0.2}$	$F_{0.5}$	$F_{0.2}$	$F_{0.5}$
Singletons	.676	.511	.843	.730
One Cluster	.688	.638	.378	.327
Random	.556	.493	.801	.668

Table 1: Baseline performance

2.3 K-means on Bag of Words model

The standard unaugumented Bag of Words model achieves $F_{0.5}$ of 0.666 on training data, as shown in Table 2.

2.4 Part of speech tag features

We then consider only terms that are nouns (NN, NNP) and adjectives (JJ) with the intuition that most of the content bearing words and descriptive words that disambiguate a person would fall in these classes. The result then improves to 0.67 on the training data.

2.5 Rich features

Another variant of this system, that we call RichFeats, gives preferential weighting to terms that are immediately around all variants of the person name in question, place names, occupation names, and titles. For marking up place names, occupation names, and titles we used gazetteer[3] lookup without explicit named entity disambiguation. The keywords that appeared in the HTML tag `<META ..>` were also given higher weights. This resulted in an $F_{0.5}$ of 0.664.

2.6 Snippets from the Web

The addition of web snippets as described in Section 2.1 yeilds a significant $F_{0.5}$ improvement to 0.72.

[3]Totalling 19646 terms, gathered from publicly available resources on the web. Further details are available on request.

2.7 Snippets and Rich features

This is a combination of the models mentioned in Sections 2.5 and 2.6. This model combination resulted in a slight degradation of performance over snippets by themselves on the training data but a slight improvement on test data.

Model	K	$F_{0.2}$	$F_{0.5}$
Vanilla BOW	10%	0.702	0.666
BOW + PoS	10%	0.706	0.670
BOW + RichFeats	10%	0.700	0.664
Snippets	10	**0.721**	**0.718**
Snippets + RichFeats	10	0.714	0.712

Table 2: Performance on Training Data

3 Selection of Parameters

The main parameter for K-means clustering is choosing the number of clusters, K. We optimized K over the training data varying K from 10%, 20%,$\cdots$,100% of the number of documents as well as varying absolute K values from 10, 20, $\cdots$ to 100 documents.[4] The evaluation score of F-measure can be highly sensitive to this parameter K, as shown in Table 3. The value of K that gives the best F-measure on training set using vanilla bag of words (BOW) model is K = 10%, however we see in Table 3 that this value of K actually performs much worse on the test data as compared to other K values.

4 Training/Test discrepancy and re-evaluation using cross validation on test data

Table 4 compares cluster statistics between the training and test data. This data was derived from Artiles et. al (2007). The large difference between average number of clusters in training and test sets indicates that the parameter K, optimized on training set cannot be transferred to test set as these two sets belong to a very different distribution. This can be emprically seen in Table 3 where applying the best K on training results in a significant performance drop on test set given this divergence when parameters are optimized for $F_{0.5}$ (although performance does transfer well when parameters are optimized on $F_{0.2}$). This was observed in our primary evaluation system which was optimized for $F_{0.5}$ and resulted in a low official score of $F_{0.5} = .53$ and $F_{0.2} = .65$.

	Train		Test	
K	$F_{0.2}$	$F_{0.5}$	$F_{0.2}$	$F_{0.5}$
10%	.702	**.666**	***.527***	***.600***
20%	.716	.644	.617	.630
30%	.724	.631	.683	.676
40%	.724	.618	.728	.705
50%	.732	.614	.762	.724
60%	.731	.601	.798	.747
70%	.730	.593	.832	.766
80%	**.732**	.586	***.855***	***.773***
90%	.714	.558	.861	.764
100%	.670	.502	.843	.730

Table 3: Selecting the optimal parameter on training data and application to test data

Thus an interesting question is to measure performance when parameters are chosen on data sharing the distributional character of the test data rather than the highly divergent training set. To do this, we used a standard 2-fold cross validation to estimate clustering parameters from a held-out, alternate-half portion of the test data[5], which more fairly represents the character of the other half of the test data than does the very different training data. We divide the test set into two equal halves (taking first fifteen names alphabetically in one set and the rest in another). We optimize K on the first half, test on the other half and vice versa. We report the two K-values and their corresponding F-measures in Table 5 and we also report the average in order to compare it with the results on the test set obtained using K optimized on training. Further, we also report what would be oracle best K, that is, if we optimize K on the entire test data [6]. We can see in Table 5 that how optimizing K on a devlopment set with

[4]We discard the training and test documents that have no text content, thus the absolute value K = 10 and percentage value K = 10% can result in different K's, even if name had originally 100 documents to begin with.

[5]This also prevents overfitting as the two halves for training and testing are disjoint.

[6]By *oracle best* K we mean the K obtained by optimizing over the entire test data. Note that, the oracle best K is just for comparison because it would be unfair to claim results by optimizing K on the entire test set, all our claimed results for different models are based on 2-fold cross validation.

same distribution as test set can give us F-measure in the range of 77%, a significant increase as compared to the F-measure obtained by optimizing K on given training data. Further, Table 5, also indicates results by a custom clustering method, that takes the best K-means clustering using vanilla bag of words model, retains the largest cluster and splits all the other clusters into singleton clusters. This method gives an improved 2-fold F-measure score over the simple bag of words model, implying that most of the namesakes in test data have one (or few) dominant cluster and a lot of singleton clusters. Table 6 shows a full enumeration of model variance under this cross validated test evaluation. POS and RichFeats yield small gains, and a best $F_{0.5}$ performance of .776.

Data set	cluster size		# of clusters	
	Mean	Variance	Mean	Variance
Train	5.4	144.0	10.8	146.3
Test	3.1	26.5	45.9	574.1

Table 4: Cluster statistics from the test and training data

Data set	K	$F_{0.2}$	$F_{0.5}$
$F_{0.5}$ Best K on train	10%	.702	.666
$F_{0.2}$ Best K on train	10	.707	.663
Best K on train	10%	.527	.560
applied to test	10	.540	.571
2Fold on Test	80	.847	.748
	80%	.862	.793
		*.854**	*.771**
2Fold on Single	80	.847	.749
Largest Cluster	80	.866	.795
		.856*	**.772***
Oracle on Test	80	.858	.774

Table 5: Comparision of training and test results using Vanilla Bag-of-words model. The values indicated with * represent the average value.

5 Conclusion

We presented a K-means clustering approach for the task of person name disambiguation using several augmented feature sets including HTML meta features, part-of-speech-filtered features, and inclusion of additional web snippets extracted from Google to facilitate term bridging. The latter showed significant empirical gains on the training data. Best performance on test data, when parameters are optimized for $F_{0.2}$ on training (Table 3), yielded a top performing $F_{0.2}$ of .855 on test data (and $F_{0.5}$=.773 on test data). We also explored the striking discrepancy between training and test data characteristics and showed how optimizing the clustering parameters on given training data does not transfer well to the divergent test data. To control for similar training and test distributional characteristics, we re-evaluated our test results estimating clustering parameters from alternate held-out portions of the test set. Our models achieved cross validated $F_{0.5}$ of .77-.78 on test data for all feature combinations, further showing the broad strong performance of these techniques.

Model	K	$F_{0.2}$	$F_{0.5}$
Vanilla BOW	80/ 80%	.847/.862 Avg = .854	.749/.793 Avg = .771
BOW + PoS	80%/ 80%	.844/.865 Avg = .854	.749/.795 Avg = .772
BOW RichFeats	80%/ 80%	.847/.868 Avg = .858	.754/.798 Avg = **.776**
Snippets	50%/ 50%	.842/.875 Avg = **.859**	.746/.800 Avg = .773
Snippets + RichFeats	40%/ 50%	.836/.874 Avg = .855	.750/.798 Avg = .774

Table 6: Performance on 2Fold Test Data

References

Reema Al-Kamha and David W. Embley. 2004. Grouping search-engine returned citations for person-name queries. In *Proceedings of the 6th annual ACM international workshop on Web information and data management*, pages 96–103.

Javier Artiles, Julio Gonzalo, and Felisa Verdejo. 2007. Evaluation: Establishing a benchmark for the web people search task. In *Proceedings of Semeval 2007, Association for Computational Linguistics*.

Ron Bekkerman and Andrew McCallum. 2005. Disambiguating web appearances of people in a social network. In *Proceedings of the 14th international conference on World Wide Web*, pages 463–470.

Gideon S. Mann and David Yarowsky. 2004. Unsupervised personal name disambiguation. In *Proceedings of the seventh conference on Natural language learning (CONLL)*, pages 33–40.

Xiaojun Wan, Jianfeng Gao, Mu Li, and Binggong Ding. 2005. Person resolution in person search results: Webhawk. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 163–170.