Streaming Cross Document Entity Coreference Resolution

Delip Rao and **Paul McNamee** and **Mark Dredze**
Human Language Technology Center of Excellence
Center for Language and Speech Processing
Johns Hopkins University
delip,mcnamee,mdredze@jhu.edu

Abstract

Previous research in cross-document entity coreference has generally been restricted to the offline scenario where the set of documents is provided in advance. As a consequence, the dominant approach is based on greedy agglomerative clustering techniques that utilize pairwise vector comparisons and thus require $O(n^2)$ space and time. In this paper we explore identifying coreferent entity mentions across documents in high-volume streaming text, including methods for utilizing orthographic and contextual information. We test our methods using several corpora to quantitatively measure both the efficacy and scalability of our streaming approach. We show that our approach scales to at least an order of magnitude larger data than previous reported methods.

1 Introduction

A key capability for successful information extraction, topic detection and tracking, and question answering is the ability to identify equivalence classes of entity mentions. An entity is a real-world person, place, organization, or object, such as the person who serves as the 44th president of the United States. An entity mention is a string which refers to such an entity, such as "Barack Hussein Obama", "Senator Obama" or "President Obama". The goal of coreference resolution is to identify and connect all textual entity mentions that refer to the same entity.

The first step towards this goal is to identify all references within the same document, or *within document coreference resolution*. A document often has a leading canonical reference to the entity ("Barack Obama") followed by additional expressions for the same entity ("President Obama.") An intra-document coreference system must first identify each reference, often relying on named entity recognition, and then decide if these references refer to a single individual or multiple entities, creating a *coreference chain* for each unique entity. Feature representations include surface form similarity, lexical context of mentions, position in the document and distance between references. A variety of statistical learning methods have been applied to this problem, including use of decision trees (Soon et al., 2001; Ng and Cardie, 2002), graph partitioning (Nicolae and Nicolae, 2006), maximum-entropy models (Luo et al., 2004), and conditional random fields (Choi and Cardie, 2007).

Given pre-processed documents, in which entities have been identified and entity mentions have been linked into chains, we seek to identify across an entire document collection all chains that refer to the same entity. This task is called cross document coreference resolution (CDCR). Several of the challenges associated with CDCR differ from the within document task. For example, it is unlikely that the same document will discuss John Phillips the American football player and John Phillips the musician, but it is quite probable that documents discussing each will appear in the same collection. Therefore, while matching entities with the same mention string can work well for within document coreference, more sophisticated approaches are necessary for the cross document scenario where a *one-entity-per-name* assumption is unreasonable.

One of the most common approaches to both within document and cross document coreference resolution has been based on agglomerative clustering, where vectors might be bag-of-word contexts (Bagga and Baldwin, 1998; Mann and

Yarowsky, 2003; Gooi and Allan, 2004; Chen and Martin, 2007). These algorithms creates a $O(n^2)$ dependence in the number of *mentions* – for within document – and *documents* – for cross document. This is a reasonable limitation for within document, since the number of references will certainly be small; we are unlikely to encounter a document with millions of references. In contrast to the small n encountered within a document, we fully expect to run a CDCR system on hundreds of thousands or millions of documents. Most previous approaches cannot handle collections of this size.

In this work, we present a new method for cross document coreference resolution that scales to very large corpora. Our algorithm operates in a *streaming* setting, in which documents are processed one at a time and only a single time. This creates a linear ($O(n)$) dependence on the number of documents in the collection, allowing us to scale to millions of documents and millions of unique entities. Our algorithm uses streaming clustering with common coreference similarity computations to achieve large scale. Furthermore, our method is designed to support both name disambiguation and name variation.

In the next section, we give a survey of related work. In Section 3 we detail our streaming setup, giving a description of the streaming algorithm and presenting efficient techniques for representing clusters over streams and for computing similarity. Section 4 describes the data sets on which we evaluate our methods and presents results. We conclude with a discussion and description of ongoing work.

2 Related Work

Traditional approaches to cross document coreference resolution have first constructed a vector space representation derived from local (or global) contexts of entity mentions in documents and then performed some form of clustering on these vectors. This is a simple extension of Firth's distributional hypothesis applied to entities (Firth, 1957). We describe some of the seminal work in this area.

Some of the earliest work in CDCR was by Bagga and Baldwin (1998). Key contributions of their research include: promotion of a set-theoretic evaluation measure, *B-CUBED*; introduction of a data set based on 197 New York Times articles which mention a person named *John Smith*; and, use of TF/IDF weighted vectors and cosine similarity in single-link greedy agglomerative clustering.

Mann and Yarowsky (2003) extended Bagga and Baldwin's work and contributed several innovations, including: use of biographical attributes (*e.g.,* year of birth, occupation), and evaluation using *pseudonames*. Pseudonames are sets of artificially conflated names that are used as an efficient method for producing a set of gold-standard disambiguations.[1] Mann and Yarowsky used 4 pairs of conflated names in their evaluation. Their system did not perform as well on named entities with little available biographic information.

Gooi and Allan (2004) expanded on the use of pseudonames by semi-automatically creating a much larger evaluation set, which they called the 'Person-X' corpus. They relied on automated named-entity tagging and domain-focused text retrieval. This data consisted of 34,404 documents where a single person mention in each document was rewritten as 'Person X'. Besides their novel construction of a large-scale resource, they investigated several minor variations in clustering, namely (a) use of Kullback-Leibler divergence as a distance measure, (b) use of 55-word snippets around entity mentions (vs. entire documents or extracted sentences), and (c) scoring clusters using average-link instead of single- or complete-link.

Finally, in more recent work, Chen and Martin (2007) explore the CDCR task in both English and Chinese. Their work focuses on use of both local, and document-level noun-phrases as features in their vector-space representation.

There have been a number of open evaluations of CDCR systems. For example, the Web People Search (WePS) workshops (Artiles et al., 2008) have created a task for disambiguating personal names from HTML pages. A set of ambiguous names is chosen and each is submitted to a popular web search engine. The top 100 pages are then manually clustered.We discuss several other data

[1]See Sanderson (2000) for use of this technique in word sense disambiguation.

sets in Section 4.[2]

All of the papers mentioned above focus on disambiguating personal names. In contrast, our system can also handle organizations and locations. Also, as was mentioned earlier, we are committed to a scenario where documents are presented in sequence and entities must be disambiguated instantly, without the benefit of observing the entire corpus. We believe that such a system is better suited to highly dynamic environments such as daily news feeds, blogs, and tweets. Additionally, a streaming system exposes a set of known entity clusters after each document is processed instead of waiting until the end of the stream.

3 Approach

Our cross document coreference resolution system relies on a streaming clustering algorithm and efficient calculation of similarity scores. We assume that we receive a stream of coreference chains, along with entity types, as they are extracted from documents. We use SERIF (Ramshaw and Weischedel, 2005), a state of the art document analysis system which performs intra-document coreference resolution. BBN developed SERIF to address information extraction tasks in the ACE program and it is further described in Pradhan et al. (2007).

Each unique entity is represented by an entity cluster c, comprised of entity chains from many documents that refer to the same entity. Given an entity coreference chain e, we identify the best known entity cluster c. If a suitable entity cluster is not found, a new entity cluster is formed.

An entity cluster is selected for a given coreference chain using several similarity scores, including document context, predicted entity type, and orthographic similarity between the entity mention and previously discovered references in the entity cluster. An efficient implementation of the similarity score allows the system to identify the top k most likely mentions without considering all m entity clusters. The final output of our system is a collection of entity clusters, each containing a list of coreference chains and their documents. Additionally, due to its streaming nature, the system can be examined at any time to produce this information based on only the documents that have been processed thus far.

In the next sections, we describe both the clustering algorithm and efficient computation of the entity similarity scores.

3.1 Clustering Algorithm

We use a streaming clustering algorithm to create entity clusters as follows. We observe a set of points from a potentially infinite set $\mathcal{X}$, one at a time, and would like to maintain a fixed number of clusters while minimizing the maximum *cluster radius*, defined as the radius of the smallest ball containing all points of the cluster. This setup is well known in the theory and information retrieval community and is referred to as the dynamic clustering problem (Can and Ozkarahan, 1987).

Others have attempted to use an incremental clustering approach, such as Gooi and Allan (2004) (who eventually prefer a hierarchical clustering approach), and Luo et al. (2004), who use a Bell tree approach for incrementally clustering within document entity mentions. Our work closely follows the *Doubling Algorithm* of Charikar et al. (1997), which has better performance guarantees for streaming data. Streaming clustering means potentially linear performance in the number of observations since each document need only be examined a single time, as opposed to the quadratic cost of agglomerative clustering.[3]

The Doubling Algorithm consists of two stages: update and merge. Update adds points to existing clusters or creates new clusters while merge combines clusters to prevent the clusters from exceeding a fixed limit. New clusters are created according to a threshold set using development data. We selected a threshold of 0.5 since it worked well in preliminary experiments. Since the number of entities grows with time, we have skipped the merge step in our initial experiments so as not to limit cluster growth.

We use a dynamic caching scheme which backs the actual clusters in a disk based index, but re-

[2]We preferred other data sets to the WePS data in our evaluation because it is not easily placed in temporal order.

[3]It is possible to implement hierarchical agglomerative clustering in $\mathrm{O}(n \log m)$ time where n is the number of points and m in the number of clusters. However this is still superlinear and expensive in situations where m continually increases like in streaming coreference resolution.

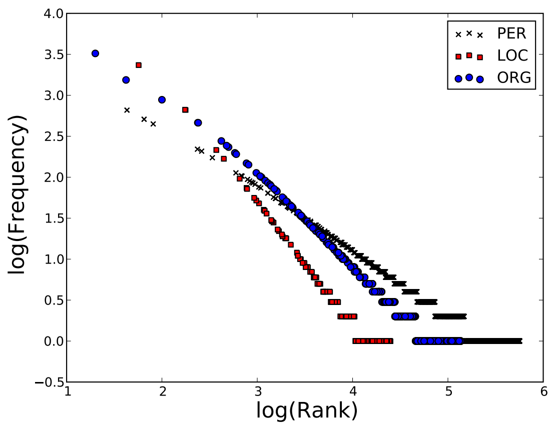

Figure 1: Frequency vs. rank for 567k people, 136k organizations, and 25k locations in the New York Times Annotated Corpus (Sandhaus, 2008).

tains basic cluster information in memory (see below). Doing so improves paging performance as observed in Omiecinski and Scheuermann (1984). Motivated by the Zipfian distribution of named entities in news sources (Figure 1), we organize our cluster store using an LRU policy, which facilitates easy access to named entities that were observed in the recent past. We obtain additional performance gains by hashing the clusters based on the constituent mention string (details below). This allows us to quickly retrieve a small but related number of clusters, k. It is always the case that $k << m$, the current number of clusters.

3.2 Candidate Cluster Selection

As part of any clustering algorithm, each new item must be compared against current clusters. As we see more documents, the number of unique clusters (entities) grows. Therefore, we need efficient methods to select candidate clusters.

To select the top candidate clusters, we obtain those that have high orthographic similarity with the head name mention in the coreference chain e. We compute this similarity using the dice score on either word unigrams or character skip bigrams. For each entity mention string associated with a cluster c, we generate all possible n-grams using one of the above two policies. We then index the cluster by each of its n-grams in a hash maintained in memory. In addition, we keep the number of n-grams generated for each cluster.

When given a new head mention e for a coreference chain, we generate all of the n-grams and look up clusters that contain these n-grams using the hash. We then compute the dice score:

$$\text{dice}(e, c) = \frac{|\{\text{ngram}(e)\} \cap \{\text{ngram}(c)\}|}{|\{\text{ngram}(e)\} \cup \{\text{ngram}(c)\}|},$$

where $\{\text{ngram}(e)\}$ are the set of n-grams in entity mention e and $\{\text{ngram}(c)\}$ are the set of n-grams for all entity mentions in cluster c. Note that we can calculate the numerator (the intersection) by looking up the n-grams of e in the hash and counting matches with c. The denominator is equivalent to the number of n-grams unique to e and to c plus the number that are shared. The number that are shared is the intersection. The number unique to e is the total number of n-grams in e minus the intersection. The final term, the number unique to c, is computed by taking the total number of n-grams in c (a single integer stored in memory) minus the intersection.

Through this strategy, we can select only those clusters that have the highest orthographic similarity to e without requiring the cluster contents, which may not be stored in memory. In our experiments, we evaluate settings where we select all candidates with non-zero score and a pruned set of the top k dice score candidates. We also include in the n-gram list known aliases to facilitate orthographically dissimilar, but reasonable matches (*e.g.,* IBM or 'Big Blue' for 'International Business Machines, Inc.').[4]

For further efficiency, we keep separate caches for each named entity type.[5] We then select the appropriate cache based on the automatically determined type of the named entity provided by the named entity tagger, which also prevents spurious matches of non-matching entity types.

3.3 Similarity Metric

After filtering by orthographic information to quickly obtain a small set of candidate clusters, a full similarity score is computed for the current

[4]We generated alias lists for entities from Freebase.

[5]Persons (PER), organizations (ORG), and locations (LOC).

entity coreference chain and each retrieved candidate cluster. These computations require information about each cluster, so the cluster's sufficient statistics are loaded using the LRU cache described above.

We define several similarity metrics between coreference chains and clusters to deal with both name variation and disambiguation. For name variation, we define an orthographic similarity metric to match similar entity mention strings. As before, we use word unigrams and character skip bigrams. For each of these methods, we compute a similarity score as dice(e, c) and select the highest scoring cluster.

To address name disambiguation, we use two types of context from the document. First, we use *lexical* features represented as TF/IDF weighted vectors. Second, we consider *topic* features, in which each word in a document is replaced with the topic inferred from a topic model. This yields a distribution over topics for a given document. We use an LDA (Blei et al., 2003) model trained on the New York Times Annotated Corpus (Sandhaus, 2008). We note that LDA can be computed over streams (Yao et al., 2009).

To compare context vectors we use cosine similarity, where the cluster vector is the average of all document vectors assigned to the cluster. Note that the filtering step in Section 3.2 returns only those candidates with some orthographic similarity with the coreference chain, so a similarity metric that uses context only is still restricted to orthographically similar entities.

Finally, we consider a combination of orthographic and context similarity as a linear combination of the two metrics as:

$$\text{score}(e, c) = \alpha\, \text{dice}(e, c) + (1 - \alpha)\text{cosine}(e, c)\ .$$

We set $\alpha = 0.8$ based on initial experiments.

4 Evaluation

We used several corpora to evaluate our methods, including two data sets commonly used in the coreference community. We also created a new test set using artificially conflated names. And finally to test scalability, we ran our algorithm over a large text collection that, while it did not have ground truth entity clusters, was useful for computing other performance statistics. Properties for each data set are given in Table 1.

Attribute	*smith*	*nytac*	*ace08*	*kbp09*
Total Documents	197	1.85M	10k	1.2M
Annotated Docs	197	19,360	415	**
Annotated Entities	35	200	3,943	**

Table 1: Data sets used in our experiments. For the *kbp09* data we did not have annotations.

4.1 John Smith corpus

Bagga and Baldwin (1998) evaluated their disambiguation system on a set of 197 articles from the New York Times that mention a person named 'John Smith'. This data exhibits no name variants and is strictly a disambiguation task. We include this data (*smith*) to allow comparison to previous work.

4.2 NYTAC Pseudo-name corpus

To study the effects of word sense ambiguity and disambiguation several researchers have artificially conflated dissimilar words together and then attempted to disambiguate them (Sanderson, 2000). The obvious advantage is cheaply obtained ground truth for disambiguation.

The same trick has also been employed in person name disambiguation (Mann and Yarowsky, 2003; Gooi and Allan, 2004). We adopt the same method on a somewhat larger scale using annotations from the New York Times Annotated Corpus (*NYTAC*) (Sandhaus, 2008), which annotates documents based on whether or not they mention an entity. The NYTAC data contains documents from 20 years of the New York Times and contains rich metadata and document-level annotations that indicate when an entity is mentioned in the document using a standard lexicon of entities. (Note that mention strings are not tagged.) Using these annotations we created a set of 100 pairs of conflated person names.

The names were selected to be medium frequency (*i.e.,* occurring in between 50 and 200 articles) and each pair matches in gender. The first 50 pairs are for names that are topically similar, for example, Tim Robbins and Tom Hanks (both actors); Barbara Boxer and Olympia Snowe (both

	smith			*nytac*			*ace08*		
Approach	P	R	F	P	R	F	P	R	F
Baseline	1.000	0.178	0.302	1.000	0.010	0.020	1.000	0.569	0.725
ExactMatch	0.233	1.000	0.377	0.563	0.897	0.692	0.977	0.697	0.814
Ortho	0.603	0.629	0.616	0.611	0.784	0.687	0.975	0.694	0.811
BoW	0.956	0.367	0.530	0.930	0.249	0.349	0.989	0.589	0.738
Topic	0.847	0.592	0.697	0.815	0.244	0.363	0.983	0.605	0.750
Ortho+BoW	0.603	0.634	0.618	0.801	0.601	0.686	0.976	0.691	0.809
Ortho+Topic	0.603	0.634	0.618	0.800	0.591	0.680	0.975	0.704	0.819

Table 2: Best B^3 performance on the *smith*, *nytac*, and *ace08* test sets.

US politicians). We imagined that this would be a more challenging subset because of presumed lexical overlap. The second set of 50 name pairs were arbitrarily conflated. We sub-selected the data to ensure that no two entities in our collection co-occur in the same document and this left us with 19,360 documents for which ground-truth was known. In each document we rewrote the conflated name mentions using a single gender-neutral name; any middle initials or names were discarded.

4.3 ACE 2008 corpus

The NIST ACE 2008 (*ace08*) evaluation studied several related technologies for information extraction, including named-entity recognition, relation extraction, and cross-document coreference for person names in both English and Arabic. Approximately 10,000 documents from several genres (predominantly newswire) were given to participants, who were expected to cluster person and organization entities across the entire collection. However, only a selected set of about 400 documents were annotated and used to evaluate system performance. Baron and Freedman (2008) describe their work in this evaluation, which included a separate task for within-document coreference.

4.4 TAC-KBP 2009 corpus

The NIST TAC 2009 Knowledge Base Population track (*kbp09*) (McNamee and Dang, 2009) conducted an evaluation of a system's ability to link entity mentions to corresponding Wikipedia-derived knowledge base nodes. The TAC-KBP task focused on ambiguous person, organization, and geo-political entities mentioned in newswire, and required systems to cope with name variation (*e.g.,* "Osama Bin Laden" / "Usama Bin Laden" or "Mark Twain" / "Samuel Clemens") as well as name disambiguation. Furthermore, the task required detection of when no appropriate KB entry exists, which is a departure from the conventional disambiguation problem. The collection contains over 1.2 million documents, primarily newswire. Wikipedia was used as a surrogate knowledge base, and it has been used in several previous studies (*e.g.,* Cucerzan (2007)). This task is closely related to CDCR, as mentions that are aligned to the same knowledge base entry create a coreference cluster. However, there are no actual CDCR annotations for this corpus, though we used it nonetheles as a benchmark corpus to evaluate speed and to demonstrate scalability.

5 Discussion

5.1 Accuracy

In Table 2 we report cross document coreference resolution performance for a variety of experimental conditions using the B^3 method, which includes precision, recall, and calculated $F_{\beta=1}$ values. For each of the three evaluation corpora (*smith*, *nytac*, and *ace08*) we report values for two baseline methods and for similarity metrics using different types of features. The first baseline, called *Baseline*, places each coreference chain in its own cluster while the second baseline, called *ExactMatch*, merges all mentions that match exactly orthographically into the same cluster.

Use of name similarity scores as features (in addition to their use for candidate cluster selection) is indicated by rows labeled *Ortho*. Use of lexical features is indicated by *BoW* and use of topic model features by *Topic*.

Using topic models as features was more helpful than lexical contexts on the *smith* corpus.

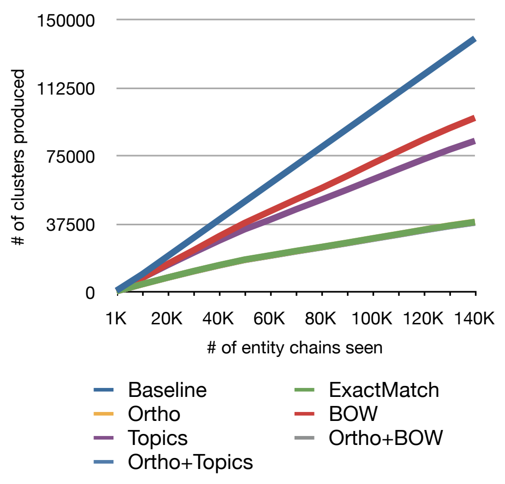

Figure 2: Number of clusters produced vs. number of entity chains observed in the stream. Number of entity chains is proportional to the number of documents.

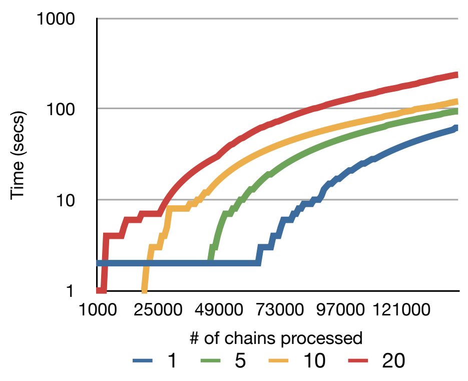

Figure 3: Elapsed processing time as a function of bounding the number of candidate clusters considered for an entity. When fewer candidates are considered, clustering decisions can be made much faster.

When used alone *topic* beats *BoW*, but in combination with the *ortho* features performance is equivalent. For both *nytac* and *ace08* heavy reliance on orthographic similarity proved hard to beat. On the *ace08* corpus Baron and Freedman (2008) report B^3 F-scores of 83.2 for persons and 67.8 for organizations, and our streaming results appear to be comparable to their offline method.

The cluster growth induced by the various measures can be seen in Figure 2. The two baseline methods, *Baseline* and *ExactMatch*, provide bounds on the cluster growth with all other methods falling in between.

5.2 Hashing Strategies for Candidate Selection

Table 3 contains B^3 F-scores when different hashing strategies are employed for candidate selection. The trend appears to be that stricter matching outperforms fuzzier matching; full mentions tended to beat words, which beat use of the character bigrams. This agrees with the results described in the previous section, which show heavy reliance on orthographic similarity.

5.3 Timing Results

Figure 3 shows how processing time increases with the number of entities observed in the *ace08* document stream. We experimented with using an upper bound on the number of candidate clusters to consider for an entity.

Figure 4 compares the efficiency of using three different methods for candidate cluster identification. The most restrictive hashing strategy, using exact mention strings, is the most efficient, followed by the use of words, then the use of character skip bigrams. This makes intuitive sense – the strictest matching reduces the number of candidate clusters that have to be considered when processing an entity.[6]

The *ace08* corpus contained over 10,000 documents and is one of the largest CDCR test sets. In Figure 5 we show how processing time grows when processing the *kbp09* corpus. Doubling the number of entities processed increases the runtime by about a factor of 5. The curve is not linear due to the increasing number of entity cluster's that must be considered. Future work will examine how to keep the number of clusters considered constant over time, such as ignoring older entities.

6 Conclusion

We have presented a new streaming cross document coreference resolution system. Our approach is substantially faster than previous sys-

[6]In the limit, if names were unique, hashing on strings would completely solve the CDCR problem and processing an entity would be O(1)

	smith			*nytac*			*ace08*		
Approach	bigrams	words	mention	bigrams	words	mention	bigrams	words	mention
Ortho	0.382	0.553	0.616	0.120	0.695	0.687	0.540	0.797	0.811
BoW	0.480	0.530	0.467	0.344	0.339	0.349	0.551	0.700	0.738
Topic	0.697	0.661	0.579	0.071	0.620	0.363	0.544	0.685	0.750
Ortho+BoW	0.389	0.554	0.618	0.340	0.691	0.686	0.519	0.783	0.809
Ortho+Topic	0.398	0.555	0.618	0.120	0.477	0.680	0.520	0.776	0.819

Table 3: B^3 F-scores using different hashing strategies for candidate selection. Name/cluster similarity could be based on character skip bigrams, words appear in names, or exact matching of mention.

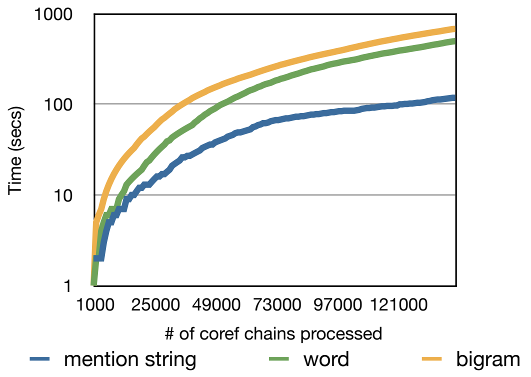

Figure 4: Comparison of three hashing strategies for identifying candidate clusters for a given entity. The more restrictive strategies lead to faster processing as fewer candidates are considered.

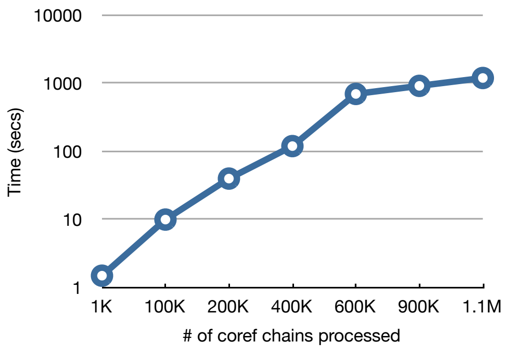

Figure 5: The number of coreference chains processed over time in the *kbp09* corpus. The processing of over 1 million coreference chains is at least an order of magnitude larger than previous systems reported.

tems, and our experiments have demonstrated scalability to an order of magnitude larger data than previously published evaluations. Despite its speed and simplicity, we still obtain competitive results on a variety of data sets as compared with batch systems. In future work, we plan to investigate additional similarity metrics that can be computed efficiently, as well as experiments on web scale corpora.

References

Artiles, Javier, Satoshi Sekine, and Julio Gonzalo. 2008. Web people search: results of the first evaluation and the plan for the second. In *World Wide Web (WWW)*.

Bagga, Amit and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Conference on Computational Linguistics (COLING)*.

Baron, Alex and Marjorie Freedman. 2008. Who is Who and What is What: Experiments in cross-document co-reference. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Blei, D.M., A.Y. Ng, and M.I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research (JMLR)*, 3:993–1022.

Can, F. and E. Ozkarahan. 1987. A dynamic cluster maintenance system for information retrieval. In *Conference on Research and Development in Information Retrieval (SIGIR)*.

Charikar, Moses, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. 1997. Incremental clustering and dynamic information retrieval. In *ACM Symposium on Theory of Computing (STOC)*.

Chen, Ying and James Martin. 2007. Towards robust unsupervised personal name disambiguation. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Choi, Y. and C. Cardie. 2007. Structured local training and biased potential functions for conditional random fields with application to coreference resolution. In *North American Chapter of the Association* *for Computational Linguistics (NAACL)*, pages 65–72.

Cucerzan, Silviu. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 708–716.

Firth, J.R. 1957. A synopsis of linguistic theory 1930-1955. In *Studies in Linguistic Analysis*, pages 1–32. Oxford: Philological Society.

Gooi, Chung Heong and James Allan. 2004. Cross-document coreference on a large scale corpus. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Luo, X., A. Ittycheriah, H. Jing, N. Kambhatla, and S. Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the bell tree. In *Association for Computational Linguistics (ACL)*.

Mann, Gideon S. and David Yarowsky. 2003. Unsupervised personal name disambiguation. In *Conference on Natural Language Learning (CONLL)*.

McNamee, Paul and Hoa Dang. 2009. Overview of the TAC 2009 knowledge base population track. In *Text Analysis Conference (TAC)*.

Ng, V. and C. Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Association for Computational Linguistics (ACL)*, pages 104–111.

Nicolae, C. and G. Nicolae. 2006. Bestcut: A graph algorithm for coreference resolution. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 275–283. Association for Computational Linguistics.

Omiecinski, Edward and Peter Scheuermann. 1984. A global approach to record clustering and file reorganization. In *Conference on Research and Development in Information Retrieval (SIGIR)*.

Pradhan, S.S., L. Ramshaw, R. Weischedel, J. MacBride, and L. Micciulla. 2007. Unrestricted coreference: Identifying entities and events in ontonotes. In *International Conference on Semantic Computing (ICSC)*.

Ramshaw, L. and R. Weischedel. 2005. Information extraction. In *IEEE ICASSP*.

Sanderson, Mark. 2000. Retrieving with good sense. *Information Retrieval*, 2(1):45–65.

Sandhaus, Evan. 2008. The new york times annotated corpus. Linguistic Data Consortium, Philadelphia.

Soon, Wee Meng, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*.

Yao, L., D. Mimno, and A. McCallum. 2009. Efficient methods for topic model inference on streaming document collections. In *Knowledge discovery and data mining (KDD)*.