The LODIE team (University of Sheffield) Participation at the TAC2015 Entity Discovery Task of the Cold Start KBP Track

Ziqi Zhang, Jie Gao, Anna Lisa Gentile
Department of Computer Science, University of Sheffield
Regent Court, 211 Portobello
Sheffield, UK, S1 4DP
{ziqi.zhang|j.gao|a.gentile}@sheffield.ac.uk

Abstract

This paper describes the LODIE team (from the OAK lab of the University of Sheffield) participation at TAC-KBP 2015 for the Entity Discovery task in the Cold Start KBP track. We have taken a cross-document coreference resolution approach that starts with Named Entity Recognition to locate and classify mentions of named entities, followed by a clustering procedure that groups mentions referring to the same entity. Our primary interest was studying different features and their effect on the clustering process, as well as scalable methods to cope with very large data. We experimented with several feature combinations and conclude that the best results are obtained using features based on entity surface forms and distributed word embeddings. To cope with large scale data, the clustering process takes a two-step approach to break data to smaller batches. Our method on the 2015 evaluation dataset obtains a best CEAF mention F-measure of 63.2[1].

1 Introduction

The TAC Cold Start KBP track aims to build a Knowledge Base (KB) from scratch using a given document collection and a predefined schema for the entities and relations that will compose the KB. This year the Cold Start KBP consists of two tasks: Entity Discovery (ED) and Slot Filling (SF). The goal of the ED task is to create a KB node for each person (PER), organization (ORG) and geo-political entity (GPE) mentions in the document collection, and to cluster all KB nodes that refer to the same entity. The goal of the SF task is populate specific entities with specific attributes that are to be extracted from the same document collection.

The LODIE team[2] from the OAK research lab of the University of Sheffield participated in the ED task in the Cold Start track. Compared to the previous year, the ED task is essentially the same as the TAC KBP 2014 English Entity Discovery and Linking task, except that (1) all entity mentions are clustered rather than linked to an existing reference KB, and (2) a much larger document collection is to be processed (millions of mentions). Thus a key challenge is to make the system scalable with large dataset.

We have taken a cross-document coreference approach that consists of two stages: (1) Named Entity Recognition (NER) to identify mentions of entities in the document collection and classify them into one of the three types; and (2) clustering named entity mentions that refer to the same entity. Our study focuses on the second stage, while for the first stage we apply a number of state-of-the-art approaches with limited adaptation.

It is known that clustering on very large datasets suffers from two major problems. First, the classic approach based on greedy agglomerative clustering techniques that utilizes pairwise vector comparisons typically requires $\mathcal{O}(k^2)$ computations (Rao et al., 2010; Singh et al., 2011). This can quickly become intractable as the number of mentions (k) increases[3].

[1] Ranked as #3 by the CEAF mention F-measure.

[2] Representing the project team Linked Open Data for Information Extraction (Zhang et al., 2014)

[3] Over 2 million mentions are extracted in this year's dataset.

On the other hand, the problem worsens with high-dimensional sparse feature vectors, which are common when the traditional 'bag-of-words' or 'one-hot' feature model is applied on very large dataset.

Our work in this competition has focused on solving the above challenges. For the first problem, we adopt a heuristic based approach that divides the clustering process into two stages. First, we run a light-weight, efficient string-similarity based approach to group mentions into 'macro-clusters'. Next, agglomerative clustering is performed within the coarse-grained clusters to create smaller clusters, achieving the final result.

For the second problem, we exploit the idea on learning 'word embeddings' (Mikolov et al., 2013) to build a low-dimensional, distributed feature representations of name mentions, which makes computation more tractable.

In the remainder of this paper, we firstly present a brief overview of related work (Section 2), then describe our proposed method in details (Sections 3,4, 5) followed by experiment (Section 6) and finally conclusion (Section 7).

2 Related work

Although the ED task is related to many areas of Natural Language Processing (NLP) and Information Extraction (IE) research, here we focus on the area of coreference resolution. Given a collection of documents and the collection of entity mentions from them, the goal of coreference resolution is to cluster the mentions such that all mentions in the same cluster refer to the same entity. It is an important task in NLP and IE to overcome synonymy and polysemy issues in natural language; i.e. the same entity can have different surface forms or the same surface form can refer to different entities in different context.

Coreference resolution is required for both within- and across-document context. While significant progress has been made in within-document coreference (Haghighi and Klein, 2007; Bengtson and Roth, 2008; Haghighi and Klein, 2009; Haghighi and Klein, 2010), much less has been done for the larger problem of cross-document coreference (Singh et al., 2011). Cross-document coreference poses additional challenges. First, the 'one-sense-per-name' assumption adopted in many within-document coreference methods is unlikely to work in the cross-document setting, as it is highly likely to encouter different entities referenced by the same name in a reasonably large corpus (Rao et al., 2010). Further, the most commonly used approach based on agglomerative clustering often fails to scale up to cross-document context. This is because such methods (Bagga and Baldwin, 1998; Mann and Yarowsky, 2003; Gooi and Allan, 2004; Baron and Freedman, 2008) rely on pairwise similarity comparisons between name mentions and easily become intractable as the number of mentions increases. Also, the widely used 'bag-of-words' model for feature representation results in very high-dimensional and sparse feature vectors that increase computation.

To address these issues, Mayfield et al. (2009) applied pre-processing including solving within-document coreference, and using heuristics to filter out unlikely coreferent pair of mentions. Such measures help reduce the number of computations. Rao et al. (2010) proposed an online algorithm that operates in a streaming setting, where input mentions are passed in stream and assigned greedily to entities. Singh et al. (2011) used a distributed inference technique and a hierarchical model of coreference to exploit parallelization.

3 Method overview

The workflow of the proposed method consists of two stages. The document collection firstly passes through an NER component (Section 4) that extracts and classifies mentions of named entities from the documents. This process has the time complexity of $\mathcal{O}(d)$ where d is the number of documents in the collection.

The mentions are then passed to a coreference component (Section 5) that consists of two sub-steps. The first step is a light-weight clustering process that groups mentions based on string similarity (Section 5.1). We call the clusters created in this step 'macro-clusters'. The complexity of this process is $\mathcal{O}(klog(k))$ where k is the number of mentions belonging to the same entity type. In the second step, each macro-cluster that contains mentions from more than one documents is passed to

a further agglomerative clustering process (Section 5.2) whose goal is to further cluster mentions within the macro-cluster to n (automatically determined) optimal clusters. We tried different feature representation models for this process, including bag-of-words, distributed continuous representations based on word embeddings, and the combination of both. Macro-clusters containing mentions only from one document are left as-is, since we hypothesize 'one-sense-per-name' within each document.

Within each macro-cluster, the complexity is $\mathcal{O}(k'^2)$, where k' is the number of mentions within that macro-cluster. Since k' is much smaller than the total number of mentions k of the same type in the entire collection, and that only a fraction of macro-clusters needs to be re-computed, this is much more tractable.

One limitation of this approach is its limited ability to group different name mentions referring to the same entity, which we will discuss later.

4 Named Entity Recognition

The NER component has the goal to identify and classify named entity mentions in the documents. Named Entities (Grishman and Sundheim, 1996) are those expressions that refer to people, organizations, and geographic locations, although fine-grained classification are also used. For the purpose of this work we use the Stanford NER (Finkel et al., 2005), complemented with two additional modules to improve the extraction of GPE and PER.

Stanford NER Standard Standford NER is a CRF-based (Lafferty et al., 2001) statistical NER system, which incorporates constraints to capture non-local structure of entities, by using Gibbs sampling (Finkel et al., 2005). Its implementation[4] comes with recognition models trained using the CoNLL 2003 English training data[5], that annotate entities of PER, ORG and Location (which we consider equivalent to GPE). This standard model is well suited to extract entities from news articles, but fails to recognize some entities in colloquial texts, such as forum posts.

Stanford NER Re-trained To cope with colloquial texts that represent a large proportion of the TAC KBP data, we re-train Stanford NER using only the discussion forum data from the training dataset of the TAC2014 English Entity Discovery and Linking task (EDL).

Gazetteer based GPE annotation To improve the performance of GPE extraction, we implement a simple gazetteer based extractor. We obtained location gazetteers from GATE[6] listing names such as cities, constituencies, countries, country abbreviations, country adjectives, regions, states. The extractor then performs lookup by exact string matching.

Ad-hoc rules Our further analysis of the TAC2014 EDL task shows that a fair amount of usernames are considered as PER entities and are present in structured elements of the discussion forum documents. For example, A poster's username can be embedded as an attribute in an XML element. These can be easily extracted by rules but pose additional challenges to the statistical NER methods. Therefore we implement several rules to capture frequently found structured templates in such documents to extract these usernames to complement the extraction of PER entities.

We then follow some heuristics to combine these different NER modules. The Stanford NER Standard and gazetteers are applied to any documents; the Stanford NER Re-trained and ad-hoc rules are only applied to discussion forum data (which are identifiable by file names in the dataset). The output from all modules are merged. In case of conflicts of types, we retain only the most 'preferrable' module with the order of preference as: ad-hoc rules, Stanford NER Re-trained for forum discussion data and Stanford NER Standard for other data, and gazetteers. In case of conflicting boundaries within the same type, we retain only the longest extraction.

5 Cross-document Coreference

All name mentions extracted and classified before are then passed to cross-document coreference, which starts with a lightweight clustering process based on string similarity matching to create macro-clusters. This is then followed by an agglomera-

[4] http://nlp.stanford.edu/software/CRF-NER.shtml

[5] http://www.cnts.ua.ac.be/conll2003/ner/

[6] https://gate.ac.uk

tive clustering process that effectively splits macro-clusters containing mentions from different documents into final, smaller and more fine-grained clusters.

5.1 String similarity clustering

The goal of string similarity clustering is two-fold. The first is to cluster name mentions using a light process, focusing on the detection and conflation of entity names (e.g., due to lexical and orthographical differences, abbreviations, acronyms). Intuitively, similar names are likely to refer to the same named entity. The second is to 'scale down' the data such that they are manageable by subsequent processes. In our case, we hypothesize that string similarity clustering can 'over-cluster' and the subsequent agglomerative clustering step is applied to further break down the macro-clusters. Alternatively, one may also consider each macro-cluster as a single data point, which is to be further grouped.

Our string similarity clustering is an iterative process shown in Algorithm 1. In each iteration, M denote the pool of remaining name mentions belonging to the same type. We start with taking a random mention $m \in M$ to form a new cluster C_m (line 4 and 5), with the mention being the cluster 'centroid'. Then for every other mention $m' \in M, m' \neq m$, we compute a string similarity score $s(m, m')$. If the score passes a threshold T, we add m' to C_m and remove m' from M. By the end of an iteration, C_m is added to the list of clusters to output ($\mathcal{C}$) and the pool of remaining name mentions would have a size of $|M| - |C_m|$. This is repeated until the pool is empty. The algorithm avoids pair-wise computation of all pairs candidates since the pool of name mentions can be reduced gradually during the iterations, leading to an overall runtime complexity of approximately $\mathcal{O}(klog(k))$ where k is the total number of name mentions of a given type at the beginning. Although the algorithm is essentially non-deterministic, it is not necessarily an issue as we expect string similarity clustering to over-cluster and the subsequent agglomerative clustering process may partially recover this.

To compute string similarity, we use different measures for different named entity types: the *Levenshtein distance (LD)* (normalized by maximum length) for ORG and GPE; and the *Jaro-Winkler*

Algorithm 1 String similarity clustering

```
1: Input: M, T
2: Output: macro-clusters C
3: while M ≠ ∅ do
4:    m ← draw(M)
5:    C_m ← createCluster(m)
6:    for all m' ∈ M, m' ≠ m do
7:       if s(m, m') ⩾ T then
8:          C_m ← C_m ∪ {m'}
9:          M ← M\{m'}
10:      end if
11:      C ← C ∪ C_m
12:   end for
13: end while
```

Name mention 1	Name mention 2	Score
Flair, Rick "The Natureboy"	Flair, Natureboy	0.852
Obama, Barak Hussein	Obama, Barak	0.947
Erving, Dr. J.	Erving, J.	0.949
David Milliband	Ed Milliband	0.792

Table 1: Examples of JW Similarity scores for PER NE names input

Name mention 1	Name mention 2	Score
Dept of the Treasury	Department of the Treasury	0.769
AB Elektronik GmbH	AB ELECTRONIK GMBH	0.944
Ting Tsi River	Tingtze River	0.714
Norwich	Norway	0.571

Table 2: Examples of LD Similarity scores for ORG and GPE names input

(JW) distance (normalized by the longest common prefix) for PER. These are chosen as empirically it has been shown that different measures perform differently for different types of named entities (Cohen et al., 2003; Magnani and Montesi, 2007; Medvedev and Ulanov, 2011).

Tables 1 and Table 2 show examples of similarity scores given by the two different measures. Indeed, even with very high threshold, string similarity clustering can still over-cluster.

5.2 Agglomerative clustering

Next, we identify macro-clusters containing name mentions from different documents, and run agglomerative clustering within these macro-clusters to create more fine-grained clusters.

We hypothesize 'one-sense-per-name' within individual document but not across documents (i.e.,

'Mr Blair' are very likely to refer to the same person in the same document but this is much less likely when they are found in different documents). Thus, the macro-clusters should be further divided. In the following we describe the features used (Section 5.2.1) and the clustering algorithm (Section 5.2.2).

5.2.1 Feature extraction

We propose four different feature types and experiment with different combinations of them.

Contextual tokens are the set of n previous and following tokens of an entity mention. Each token is normalized by case folding and lemmatization[7]. Stop words are removed.

Contextual named entities are the set of n previous and following entity mentions that are considered to co-occur with the target entity mention within reasonable proximity.

Both contextual tokens and named entities are expected to capture the local context (i.e. document) of a name mention.

Surface tokens are the set of normalized composing tokens of an entity mention. As an example, 'Mr Blair' has two tokens 'mr' and 'blair'. Note that using surface tokens as features is essentially different from string similarity matching on surface forms. Empirically, the agglomerative clustering step may recover errors made by string similarity clustering when a low threshold is used (e.g., 'Ed Milliband' and 'David Milliband' in Table 1).

All these three features are encoded as traditional 'one-hot' vectors. On large datasets, they (particularly contextual tokens and named entities) generate very high-dimensional sparse feature vectors that are expensive to compute. Therefore, we also use a distributed continuous feature representation. This has proven to be both efficient and effective in many NLP tasks (Collobert and Weston, 2008; Collobert et al., 2011; Kim, 2014) as it encodes latent, abstract features of data objects into a much lower and denser dimension that is lighter for computation. Here we use the approach proposed by Mikolov et al. (Mikolov et al., 2013).

The method uses a very large unlabeled corpus to train word and/or phrase embeddings. Each word or phrase is then represented as an p-dimensional continuous vector, where each dimension corresponds to a latent or abstract feature of the word or phrase. Further, it has been shown (Mikolov et al., 2013) that the representations learned in this way demonstrate additive compositionality, such that simple vector addition often produces meaningful results (e.g., the element-wise addition of the vectors of 'Germany' and 'capital' produces a vector that is close to that of 'Berlin'). This is a desirable property which we exploit to create distributed continuous feature vectors for out-of-vocabulary entity mentions. Specifically, we apply the method to a very large corpus to learn embeddings for the vocabulary extracted from the corpus. This vocabulary can contain both words and phrases. Then if an entity mention is found in the vocabulary, we use the corresponding feature vector as-is. Otherwise, we compute a vector by applying element-wise addition of the vectors of its composing words.

Both surface tokens and word embedding based features capture the global context of entity mentions.

We also experiment with combinations of different features using vector concatenation. We use feature weighting to combine features of different types with different levels of importance. Let w_t be the weight given to a feature type t and V_t^m be the values of the feature type t for the entity mention m, then each value of the feature type t receives a weight as:

$$\frac{w_t}{|V_t^m|} \tag{1}$$

As an example, if both surface tokens and contextual tokens are given the weight of 1.0, given a mention 'Mr Blair' that has 2 previous and following tokens, each surface token receives a weight of 0.5 and each contextual token receives a weight of 0.25. For the word embedding based feature, given a vector of 500 dimensions and a weight of 5.0, each element in the vector receives a weight of 0.01 and hence the corresponding value is reset to be the product of the original value and the weight.

5.2.2 Clustering algorithm

In this step, name mentions from the each macro-cluster are to be further clustered. To compute pairwise distance (or similarity) we use the L_1 norm (i.e. 'Manhattan' distance (Krause, 1987)). Then we run the standard group-average agglomerative

[7] http://dragon.ischool.drexel.edu/

clustering (Murtagh, 1985) with the Silhouette coefficient (Rousseeuw, 1987) to determine a natural number of clusters based on data. This requires repeatedly clustering the data into n groups then compute the Silhouette coefficient on the resulting clusters. The optimal clustering is obtained when the Silhouette is maximized. Given D a dataset with $|D|$ elements, a greedy approach could require up to $|D|$ runs of clustering by varying n from 1 to $|D|$, which can be a very computationally expensive process on large dataset. Here we propose a non-greedy iterative algorithm that searches for a local optimum as an approximation. Starting in the first iteration with D to be clustered, let $n_\vee$ be the minimum possible number of clusters (usually $n_\vee = 1$), and $n_\wedge$ be the maximum possible number of clusters (usually $n_\wedge = m'$). N denotes a sequence of integers indexed by k: $\{n_1, n_2, ..., n_k\}$ such that (1) $n_k = n_\vee$ for $k = 1$ and $n_\wedge$ for $k = |N|$, and (2) each two adjacent elements satisfy $n_k - n_{k-1} = \theta$ for $1 < k < |N|$, i.e., they are equally spaced out by a distance of θ, except for the last element and the one before. We then repeat clustering for $|N|$ times, each time setting the cluster number to be n_k, and compute the Silhouette coefficient on the clustering results (Line 4 and 5 in Algorithm 2). Thus we call N the *cluster trials*. We record the maximum Silhouette coefficient $sh_\wedge$ and the number of clusters $\tilde{n}$ where $sh_\wedge$ is obtained (Line 6 9). In the end of the iteration we reset the minium and maximum possible cluster numbers to $n_\vee = n_{k-1}$ and $n_\wedge = n_{k+1}$ (or n_k if $k = |N|$), then reset the cluster trials to be within the new range and populate its elements to ensure each adjacent elements (except for the last and the one before) are spaced out by a difference of $\theta = \frac{\theta}{2}$ (Line 11 to 14). The computation continues in the next iteration with the new list of cluster trials, which then get updated in the end for the following iteration. This repeats until $\theta = 1$, by which point the value $\tilde{n}$ associated with the maximum Silhouette value $sh_\wedge$ is considered the natural number of clusters for the data, and the data are clustered into $\tilde{n}$ groups.

This step produces the final clusters for each macro-cluster that requires further splitting. Then combined with the other untouched macro-clusters we obtain the final output.

Algorithm 2 Incremental clustering optimization

1: Input: D; $n_\vee$, $n_\wedge$, $N = \{n_1, n_2, ..., n_k\}$ where $n_k = n_\vee$ for $k = 1, n_k = n_\wedge$ for $k = |N|, n_{k-1} + \theta = n_k \; \forall 1 < i < |N|$; $\tilde{n}$= 0, $sh_{max} = 0$
2: Output: the optimal cluster number $\tilde{n}$
3: **for all** $n_k \in N$ **do**
4: $\quad \mathcal{C} \leftarrow cluster(D, n_k)$
5: $\quad sh \leftarrow silhouette(\mathcal{C})$
6: $\quad$ **if** $sh > sh_{max}$ **then**
7: $\quad\quad sh_{max} = sh$
8: $\quad\quad n_i \leftarrow \tilde{n}$
9: $\quad$ **end if**
10: **end for**
11: **if** $\theta > 1$ **then**
12: $\quad reset(N, n_{k-1}, n_{k+1}, \frac{\theta}{2})$
13: $\quad$ **go to** 3
14: **end if**

6 Experiment

6.1 Datasets

We developed our method using both the training and evaluation datasets for the TAC2014 KBP English EDL Track[8]. Training word embeddings requires a very large unlabelled corpus. For this, we used the TAC KBP comprehensive English source corpora distributed from 2013 to 2014, with a total of over 2 million documents. We used the word2vec tool[9] to learn vectors for both words and phrases, using the default pre-processing steps and parameters setting, except from (1) setting the vector dimension to 500, (2) using a negative sampling size of 10, and (3) setting the minimum frequency for a word or phrase to be kept in the vocabulary to 3.

Table 3 shows the statistics of the training and evaluation dataset in the TAC2014 KBP English EDL Track. The TAC2015 KBP Cold Start Track has a much larger corpus, which has over 49,000 documents, from which our NER component extracts over 2 million entity mentions.

6.2 Computing resources

We ran all experiments (for 2014 Train, 2014 Eval, 2015 Cold Start ED datasets) using a maximum of

[8] http://www.nist.gov/tac/2014/KBP/data.html
[9] https://code.google.com/p/word2vec/

	Docs	NE mentions
Train	160	6,349
Eval	138	5,598

Table 3: Training and evaluation datasets of TAC KBP 2014 English EDL Track

	Precision	Recall	F1
Train	63.6	73.1	68.0
Eval	65.5	72.0	68.6

Table 4: NER performance on the training and evaluation datasets.

16 CPUs and 64G memory. Only the agglomerative clustering component is parallelized as all the other components can run at a reasonable speed in a single thread. Implementation of clustering is based on R and its high-performance and parallel computing libraries[10]. For string similarity measures, we used the SimMetrics toolkit (Chapman, 2009) which produces normalized similarity coefficients in the range or $[0, 1]$.

6.3 Named Entity Recognition

We firstly assessed the performance of the NER component, which the coreference component depends on. Using the training and evaluation datasets we compiled a list of mentions for each document in the datasets, and evaluated the NER component by the standard Precision, Recall, and F1 measures. Table 4 shows the results based on the TAC2014 datasets.

6.4 Cross-document coreference

We evaluated cross-document coreference using the official CEAF mention measures, i.e., CEAFmP, CEAFmR, and CEAFmF. We created different settings to study the contribution of each individual features described in Section 5.2.1. For the string similarity clustering component (Section 5.1), we tried two thresholds T=0.7 and 0.9 (same thresholds for both similarity measures), and we denote these settings as $ss_{0.7}$, $ss_{0.9}$ respectively. Next, the output from each setting is passed to the agglomerative clustering component, where we tried each of the four features individually, configured as below:

- Previous and following 5 tokens (tok_5);

[10]https://cran.r-project.org/

	P	R	F
$ss_{0.7}$	44.1	50.7	47.2
$ss_{0.7}$+tok_5	44.1	50.7	47.2
$ss_{0.7}$+ne_3	44.1	50.7	47.2
$ss_{0.7}$+sf	53.0	60.9	56.6
$ss_{0.7}$+$dvec$	52.0	59.8	55.6
$ss_{0.9}$	54.1	62.2	57.9
$ss_{0.9}$+tok_5	54.1	62.2	57.9
$ss_{0.9}$+ne_3	54.1	62.2	57.9
$ss_{0.9}$+sf	54.6	62.7	58.4
$ss_{0.9}$+$dvec$	54.6	62.7	58.4

Table 5: Results obtained on the training dataset

	P	R	F
$ss_{0.7}$	47.8	52.4	50.0
$ss_{0.7}$+tok_5	47.8	52.4	50.0
$ss_{0.7}$+ne_3	47.8	52.4	50.0
$ss_{0.7}$+sf	55.5	61.0	58.1
$ss_{0.7}$+$dvec$	54.8	60.1	57.3
$ss_{0.9}$	56.2	61.8	58.9
$ss_{0.9}$+tok_5	56.2	61.8	58.9
$ss_{0.9}$+ne_3	56.2	61.8	58.9
$ss_{0.9}$+sf	56.8	62.4	59.5
$ss_{0.9}$+$dvec$	57.0	62.6	59.6

Table 6: Results obtained on the evaluation dataset

- Previous and following 3 entity mentions (ne_3);
- Surface tokens of the entity mention (sf);
- Distributed continuous feature vector of the entity mention, trained using the tool and corpus described above ($dvec$).

Hence combined with the two different settings for the string similarity clustering component, we obtained ten settings, each tested on both the training and evaluation (Tables 5 and 6) datasets.

Firstly, comparing $ss_{0.7}$ and $ss_{0.9}$, it shows that stricter string match boosts accuracy by a significant margin. This suggests that the datasets indeed have a fair proportion of entity mentions that satisfy 'one-sense-per-name' even across documents. Secondly, contextual tokens (tok_5) and entity mentions (ne_3) have unnoticeable benefit to string similarity clustering when used as features for agglomerative clustering[11]. This suggests that our modelling of lo-

[11]Different window sizes have also been tried but made little difference.

	P	R	F
$ss_{0.7}$	62.9	62.9	62.9
$ss_{0.7}$+tok_5	62.9	62.9	62.9
$ss_{0.7}$+ne_3	62.9	62.9	62.9
$ss_{0.7}$+sf	74.3	74.3	74.3
$ss_{0.7}$+$dvec$	72.5	72.5	72.5
$ss_{0.9}$	75.9	75.9	75.9
$ss_{0.9}$+tok_5	75.9	75.9	75.9
$ss_{0.9}$+ne_3	75.9	75.9	75.9
$ss_{0.9}$+sf	76.3	76.3	76.3
$ss_{0.9}$+$dvec$	76.3	76.3	76.3

Table 7: Ceiling performance obtained with ground truth NER output on the training dataset

	P	R	F
$ss_{0.7}$	64.9	64.8	64.8
$ss_{0.7}$+tok_5	64.9	64.8	64.8
$ss_{0.7}$+ne_3	64.9	64.8	64.8
$ss_{0.7}$+sf	74.9	74.9	74.9
$ss_{0.7}$+$dvec$	73.7	73.7	73.7
$ss_{0.9}$	76.3	76.2	76.2
$ss_{0.9}$+tok_5	76.3	76.2	76.2
$ss_{0.9}$+ne_3	76.3	76.2	76.2
$ss_{0.9}$+sf	76.9	76.8	76.8
$ss_{0.9}$+$dvec$	76.8	76.8	76.8

Table 8: Ceiling performance obtained with ground truth NER output on the evaluation dataset

cal context is not useful. This could be attributed to the extremely sparse feature vectors when using such features. On the contrary, features capturing global context (sf and $dvec$) can further improve over string similarity clustering. This may be another indication that a large proportion of entities in the datasets have used (nearly) identical entity mentions both within and across-documents. For the surface token features (sf), this may also suggest that the clustering algorithm is able to interpret surface strings in a different way that complements string similarity measures.

We also experimented with combinations of multiple feature types with different weights, such as combining sf with tok_5 or $dvec$ and using the weight factor to balance the contribution of each feature type. However, this led to only negligible improvement.

To isolate the performance of the NER component on the overall task, we ran our coreference component on the ground truth NER output by removing NER and simply outputting the entity mentions from the gold standard for each document. These results (Tables 7 and 8) can be considered as the ceiling performance obtainable with our cross-document coreference method.

The large difference between the ceiling and actual performance on both datasets suggests that our cross-document coreference method can be hampered by the mediocre NER component. In terms of the performance of the coreference component itself, the results have shown consistent patterns.

In the end, we chose the following five settings to run on the 2015 TAC KBP Cold Start evaluation dataset:

	P	R	F
run1	62.4	64.0	63.2
run2	62.1	63.7	62.9
run3	62.4	64.0	63.2
run4	62.4	64.0	63.2
run5	62.4	64.0	63.2

Table 9: Final performance (CEAFm) obtained on 2015 TAC KBP Cold Start ED task

- **run1**[12]: $ss_{0.9}$+sf+$dvec$, where sf is given a weight of 1.0 and $dvec$ 250 (equivalent to 0.5 for each of the 500 elements in the vector);
- **run2**: $ss_{0.9}$ only;
- **run3**: $ss_{0.9}$+$dvec$;
- **run4**[13]: $ss_{0.7}$+sf+$dvec$, same as run1 but using a string similarity threshold of 0.7;
- **run5**: $ss_{0.7}$+$dvec$;

We did not use local context based features as they are very high-dimensional (expensive to compute) and not effective. On the 2015 TAC KBP Cold Start evaluation dataset, we obtained results shown in Table 9. Our best result of 63.2 ranks as #3 by CAEF mention F-measure.

It is interesting to note that on the final evaluation dataset, the distributed feature representation ($dvec$) brought the most improvement, regardless of

[12]On the 2014 training and evaluation datasets this setting obtained identical results to $ss_{0.9}$+sf and $ss_{0.9}$+$dvec$ respectively.

[13]Same observation as run1

the threshold used by the string similarity clustering component.

7 Conclusion

We described a method for cross-document coreference and applied it to the bench-marking datasets published under TAC KBP Tracks. The method employs NER to firstly identify and classify entity mentions in a document collection, then uses clustering techniques to resolve coreference across document context. To cope with scalability, string similarity based clustering process is firstly performed to create macro-clusters by name matching, which are subsequently broken down into smaller and fine-grained clusters by running agglomerative clustering within these macro-clusters. On the 2015 evaluation dataset, we obtain a best CEAFmF score of 63.2.

An important lesson learnt on dealing with very large scale cross-document coreference is that, some light-weight pre-processing (e.g., name matching, name pair filtering) may be necessary to re-structure the data to more manageable scale and reduce unnecessary computations. However, ultimately a method that runs in a streaming fashion and a distributed architecture utilizing the MapReduce framework on a cluster may be necessary to cope with even larger scale.

A number of questions remains to be answered in the future. First, empirical experiments seem to suggest the prevalance of one-sense-per-name both within and across document context. However, no systematic analysis have been done to quantify this in the datasets. Better understanding this may help making better choices of features for the task. Second, we expect local context to capture context-specific semantics of entity name mentions. However, our local context based features have been ineffective in our experiment. Further analysis is needed to reveal the underlying reasons, which may lead to better design of local context features. As examples, we may extend the idea of distributed vector representations to also incorporate contextual tokens and entity mentions; or we may exploit large-scale topic modelling techniques to assign best n latent topics of a document to all its containing entity mentions. Third, our current method that follows a 'divide-and-conquer' principle may need to be fundamentally changed, as it is limited in capturing different lexicalizations of named entities. Last but not least, our current NER component simply uses existing state-of-the-art tools with limited adaptation. More work can be done to better adapt these tools and methods to the domain.

References

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 17th International Conference on Computational Linguistics*, COLING '98, pages 79–85, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alex Baron and Marjorie Freedman. 2008. Who is who and what is what: Experiments in cross-document coreference. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 274–283, Boston, Massachusetts, USA. Association for Computational Linguistics.

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 294–303, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sam Chapman. 2009. Simmetrics. *URL http://sourceforge. net/projects/simmetrics/. SimMetrics is a Similarity Metric Library, eg from edit distance¡¯s (Levenshtein, Gotoh, Jaro etc) to other metrics,(eg Soundex, Chapman). Work provided by UK Sheffield University funded by (AKT) an IRC sponsored by EPSRC, grant number GR N*, 15764.

William Cohen, Pradeep Ravikumar, and Stephen Fienberg. 2003. A comparison of string metrics for matching names and records. In *Kdd workshop on data cleaning and object consolidation*, volume 3, pages 73–78.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA. ACM.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, November.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling.

In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics.

Chung Heong Gooi and James Allan. 2004. Cross-document coreference on a large scale corpus. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 9–16, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Ralph Grishman and Beth Sundheim. 1996. Message understanding conference-6: A brief history. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 1*, COLING '96, pages 466–471, Stroudsburg, PA, USA. Association for Computational Linguistics.

Aria Haghighi and Dan Klein. 2007. Unsupervised coreference resolution in a nonparametric bayesian model. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 848–855, Prague, Czech Republic, June. Association for Computational Linguistics.

Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP '09, pages 1152–1161, Stroudsburg, PA, USA. Association for Computational Linguistics.

Aria Haghighi and Dan Klein. 2010. Coreference resolution in a modular, entity-centered model. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 385–393, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751, Boston, Massachusetts, USA. Association for Computational Linguistics.

Eugene F. Krause. 1987. *Taxicab Geometry: An Adventure in Non-Euclidean Geometry*. Dover Publications.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Matteo Magnani and Danilo Montesi. 2007. A study on company name matching for database integration. Technical report, Technical Report UBLCS-07-15.

Gideon S. Mann and David Yarowsky. 2003. Unsupervised personal name disambiguation. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, CONLL '03, pages 33–40, Stroudsburg, PA, USA. Association for Computational Linguistics.

James Mayfield, David Alexander, Bonnie Dorr, Jason Eisner, Tamer Elsayed, Tim Finin, Clay Fink, Marjorie Freedman, Nikesh Garera, Paul McNamee, Saif Mohammad, Douglas Oard, Christine Piatko, Asad Sayeed, Zareen Syed, Ralph Weischedel, Tan Xu, and David Yarowsky. 2009. Cross-document coreference resolution: A key technology for learning by reading. In *AAAI Spring Symposium on Learning by Reading and Learning to Read*, pages 65–70, California, USA. Association for the Advancement of Artificial Intelligence.

Timofey Medvedev and Alexander Ulanov. 2011. Company names matching in the large patents dataset. *Hewlett-Packard Development Company, LP*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C.j.c. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

Fionn Murtagh. 1985. Multidimensional clustering algorithm. *COMPSTAT Lectures 4. Wuerzburg: Physica-Velag*.

Delip Rao, Paul McNamee, and Mark Dredze. 2010. Streaming cross document entity coreference resolution. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1050–1058, Stroudsburg, PA, USA. Association for Computational Linguistics.

Peter Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(1):53–65, November.

Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. 2011. Large-scale cross-document coreference using distributed inference and hierarchical models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 793–803, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ziqi Zhang, Anna Lisa Gentile, and Isabelle Augenstein. 2014. Linked data as background knowledge for information extraction on the web. *SIGWEB Newsletter.*, pages 5:1–5:9, Jul.

Acknowledgments

Part of this research has been sponsored by the EPSRC funded project LODIE: Linked Open Data for Information Extraction, EP/J019488/1; and the

SPEEAK-PC collaboration agreement 101947 of the
Innovative UK.