Discriminative Training of Clustering Functions:
Theory and Experiments with Entity Identification

Xin Li and Dan Roth
Department of Computer Science
University of Illinois, Urbana, IL 61801
`(xli1,danr)@cs.uiuc.edu`

Abstract

Clustering is an optimization procedure that partitions a set of elements to optimize some criteria, based on a fixed distance metric defined between the elements. Clustering approaches have been widely applied in natural language processing and it has been shown repeatedly that their success depends on defining a good distance metric, one that is appropriate for the task and the clustering algorithm used. This paper develops a framework in which clustering is viewed as a learning task, and proposes a way to train a distance metric that is appropriate for the chosen clustering algorithm in the context of the given task. Experiments in the context of the entity identification problem exhibit significant performance improvements over state-of-the-art clustering approaches developed for this problem.

1 Introduction

Clustering approaches have been widely applied to natural language processing (NLP) problems. Typically, natural language elements (words, phrases, sentences, etc.) are partitioned into non-overlapping classes, based on some distance (or similarity) metric defined between them, in order to provide some level of syntactic or semantic abstraction. A key example is that of class-based language models (Brown et al., 1992; Dagan et al., 1999) where clustering approaches are used in order to partition words, determined to be similar, into sets. This enables estimating more robust statistics since these are computed over collections of "similar" words. A large number of different metrics and algorithms have been experimented with these problems (Dagan et al., 1999; Lee, 1997; Weeds et al., 2004). Similarity between words was also used as a metric in a distributional clustering algorithm in (Pantel and Lin, 2002), and it shows that functionally similar words can be grouped together and even separated to smaller groups based on their senses. At a

higher level, (Mann and Yarowsky, 2003) disambiguated personal names by clustering people's home pages using a TFIDF similarity, and several other researchers have applied clustering at the same level in the context of the entity identification problem (Bilenko et al., 2003; McCallum and Wellner, 2003; Li et al., 2004). Similarly, approaches to coreference resolution (Cardie and Wagstaff, 1999) use clustering to identify groups of references to the same entity.

Clustering is an optimization procedure that takes as input (1) a collection of domain elements along with (2) a distance metric between them and (3) an algorithm selected to partition the data elements, with the goal of optimizing some form of clustering quality with respect to the given distance metric. For example, the K-Means clustering approach (Hartigan and Wong, 1979) seeks to maximize a measure of tightness of the resulting clusters based on the Euclidean distance. Clustering is typically called an unsupervised method, since data elements are used without labels during the clustering process and labels are not used to provide feedback to the optimization process. E.g., labels are not taken into account when measuring the quality of the partition. However, in many cases, supervision is used at the application level when determining an appropriate distance metric (e.g., (Lee, 1997; Weeds et al., 2004; Bilenko et al., 2003) and more).

This scenario, however, has several setbacks. First, the process of clustering, simply a function that partitions a set of elements into different classes, involves no learning and thus lacks flexibility. Second, clustering quality is typically defined with respect to a fixed distance metric, without utilizing any direct supervision, so the practical clustering outcome could be disparate from one's intention. Third, when clustering with a given algorithm and a fixed metric, one in fact makes some implicit assumptions on the data and the task (e.g., (Kamvar et al., 2002); more on that below). For example, the optimal conditions under which for K-means works are that the data is generated from a uniform mixture of Gaussian models; this may not hold in reality.

This paper proposes a new clustering framework that addresses all the problems discussed above. Specifically,

we define clustering as a learning task: in the training stage, a partition function, parameterized by a distance metric, is trained with respect to a specific clustering algorithm, with supervision. Some of the distinct properties of this framework are that: (1) The training stage is formalized as an optimization problem in which a partition function is learned in a way that minimizes a clustering error. (2) The clustering error is well-defined and driven by feedback from labeled data. (3) Training a distance metric with respect to any given clustering algorithm seeks to minimize the clustering error on training data that, under standard learning theory assumptions, can be shown to imply small error also in the application stage. (4) We develop a general learning algorithm that can be used to learn an expressive distance metric over the feature space (e.g., it can make use of kernels).

While our approach makes explicit use of labeled data, we argue that, in fact, many clustering applications in natural language also exploit this information off-line, when exploring which metrics are appropriate for the task. Our framework makes better use of this resource by incorporating it directly into the metric training process; training is driven by true clustering error, computed via the specific algorithm chosen to partition the data.

We study this new framework empirically on the entity identification problem – identifying whether different mentions of real world entities, such as "JFK" and "John Kennedy", within and across text documents, actually represent the same concept (McCallum and Wellner, 2003; Li et al., 2004). Our experimental results exhibit a significant performance improvement over existing approaches ($20\% - 30\%$ F_1 error reduction) on all three types of entities we study, and indicate its promising prospective in other natural language tasks.

The rest of this paper discusses existing clustering approaches (Sec. 2) and then introduces our Supervised Discriminative Clustering framework (SDC) (Sec. 3) and a general learner for training in it (Sec. 4). Sec. 5 describes the entity identification problem and Sec. 6 compares different clustering approaches on this task.

2 Clustering in Natural Language Tasks

Clustering is the task of partitioning a set of elements $S \subseteq X$ into a disjoint decomposition [1] $p(S) = \{S_1, S_2, \cdots, S_K\}$ of S. We associate with it a *partition function* $p = p_S : X \to C = \{1, 2, \ldots K\}$ that maps each $x \in S$ to a class index $p_S(x) = k$ iff $x \in S_k$. The subscript S in p_S and $p_S(x)$ is omitted when clear from the context. Notice that, unlike a classifier, the image $x \in S$ under a partition function depends on S.

In practice, a clustering algorithm $\mathcal{A}$ (e.g. K-Means), and a distance metric d (e.g. Euclidean distance), are typ-

ically used to generate a function h to approximate the true partition function p. Denote $h(S) = \mathcal{A}_d(S)$, the partition of S by h. A distance (equivalently, a similarity) function d that measures the proximity between two elements is a pairwise function $X \times X \to R^+$, which can be parameterized to represent a family of functions — metric properties are not discussed in this paper. For example, given any two element $x_1 = <x_1^{(1)}, \cdots, x_1^{(m)}>$ and $x_2 = <x_2^{(1)}, \cdots, x_2^{(m)}>$ in an m-dimensional space, a linearly weighted Euclidean distance with parameters $\theta = \{w_l\}_1^m$ is defined as:

$$d_\theta(x_1, x_2) \equiv \sqrt{\sum_{l=1}^{m} w_l \cdot |x_1^{(l)} - x_2^{(l)}|^2} \qquad (1)$$

When supervision (e.g. class index of elements) is unavailable, the quality of a partition function h operating on $S \subseteq X$, is measured with respect to the distance metric defined over X. Suppose h partitions S into disjoint sets $h(S) = \{S'_k\}_1^K$, one *quality function* used in the K-Means algorithm is defined as:

$$q_S(h) \equiv \sum_{k=1}^{K} \sum_{x \in S'_k} d(x, \mu'_k)^2, \qquad (2)$$

where μ'_k is the mean of elements in set S'_k. However, this measure can be computed irrespective of the algorithm.

2.1 What is a Good Metric?

A good metric is one in which close proximity correlates well with the likelihood of being in the same class. When applying clustering to some task, people typically decide on the clustering quality measure $q_S(h)$ they want to optimize, and then chose a specific clustering algorithm $\mathcal{A}$ and a distance metric d to generate a 'good' partition function h. However, it is clear that without any supervision, the resulting function is not guaranteed to agree with the target function p (or one's original intention).

Given this realization, there has been some work on *selecting* a good distance metric for a family of related problems and on *learning* a metric for specific tasks. For the former, the focus is on developing and selecting good distance (similarity) metrics that reflect well pairwise proximity between domain elements. The "goodness" of a metric is empirically measured when combined with different clustering algorithms on different problems. For example (Lee, 1997; Weeds et al., 2004) compare similarity metrics such as the Cosine, Manhattan and Euclidean distances, Kullback-Leibler divergence, Jensen-Shannon divergence, and Jaccard's Coefficient, that could be applied in general clustering tasks, on the task of measuring distributional similarity. (Cohen et al., 2003) compares a number of string and token-based similarity metrics on the task of matching entity names and found that,

---

[1] Overlapping partitions will not be discussed here.

overall, the best-performing method is a hybrid scheme (SoftTFIDF) combining a TFIDF weighting scheme of tokens with the Jaro-Winkler string-distance scheme that is widely used for record linkage in databases.
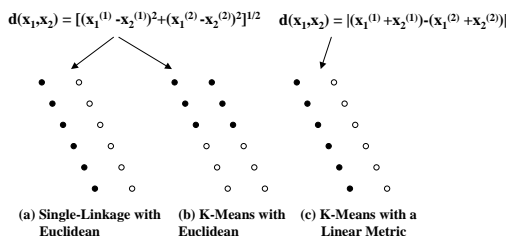

$$d(x_1,x_2) = [(x_1^{(1)} - x_2^{(1)})^2 + (x_1^{(2)} - x_2^{(2)})^2]^{1/2} \qquad d(x_1,x_2) = |(x_1^{(1)} + x_2^{(1)}) - (x_1^{(2)} + x_2^{(2)})|$$

(a) Single-Linkage with Euclidean   (b) K-Means with Euclidean   (c) K-Means with a Linear Metric

Figure 1: **Different combinations of clustering algorithms with distance metrics.** The 12 points, positioned in a two-dimensional space $< X^{(1)}, X^{(2)} >$, are clustered into two groups containing solid and hollow points respectively.

Moreover, it is not clear whether there exists any *universal* metric that is good for many different problems (or even different data sets for similar problems) and is appropriate for any clustering algorithm. For the word-based distributional similarity mentioned above, this point was discussed in (Geffet and Dagan, 2004) when it is shown that proximity metrics that are appropriate for class-based language models may not be appropriate for other tasks. We illustrate this critical point in Fig. 1. (a) and (b) show that even for the same data collection, different clustering algorithms with the same metric could generate different outcomes. (b) and (c) show that with the same clustering algorithm, different metrics could also produce different outcomes. *Therefore, a good distance metric should be both domain-specific and associated with a specific clustering algorithm.*

2.2 Metric Learning via Pairwise Classification

Several works (Cohen et al., 2003; Cohen and Richman, 2002; McCallum and Wellner, 2003; Li et al., 2004) have tried to remedy the aforementioned problems by attempting to learn a distance function in a domain-specific way via pairwise classification. In the training stage, given a set of labeled element pairs, a function $f : X \times X \rightarrow \{0, 1\}$ is trained to classify any two elements as to whether they belong to the same class (1) or not (0), independently of other elements. The distance between the two elements is defined by converting the prediction confidence of the pairwise classifier, and clustering is then performed based on this distance function. Particularly, (Li et al., 2004) applied this approach to measuring name similarity in the entity identification problem, where a pairwise classifier (LMR) is trained using the SNoW learning architecture (Roth, 1998) based on variations of Perceptron and Winnow, and using a collection of relational features between a pair of names. The distance between two names is defined as a softmax

over the classifier's output. As expected, experimental evidence (Cohen et al., 2003; Cohen and Richman, 2002; Li et al., 2004) shows that domain-specific distance functions improve over a fixed metric. This can be explained by the flexibility provided by adapting the metric to the domain as well as the contribution of supervision that guides the adaptation of the metric.

A few works (Xing et al., 2002; Bar-Hillel et al., 2003; Schultz and Joachims, 2004; Mochihashi et al., 2004) outside the NLP domain have also pursued this general direction, and some have tried to learn the metric with limited amount of supervision, no supervision or by incorporating other information sources such as constraints on the class memberships of the data elements. In most of these cases, the algorithm practically used in clustering, (e.g. K-Means), is not considered in the learning procedure, or only implicitly exploited by optimizing the same objective function. (Bach and Jordan, 2003; Bilenko et al., 2004) indeed suggest to learn a metric directly in a clustering task but the learning procedure is specific for one clustering algorithm.

3 Supervised Discriminative Clustering

To solve the limitations of existing approaches, we develop the Supervised Discriminative Clustering Framework (SDC), that can train a distance function with respect to any chosen clustering algorithm in the context of a given task, guided by supervision.
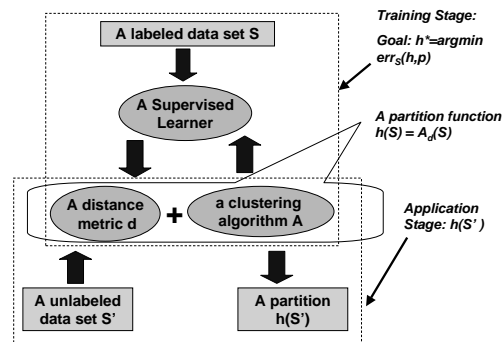

Figure 2: **Supervised Discriminative Clustering**

Fig. 2 presents this framework, in which a clustering task is explicitly split into training and application stages, and the chosen clustering algorithm involves in both stages. In the training stage, supervision is directly integrated into measuring the clustering error $err_S(h, p)$ of a partition function h by exploiting the feedback given by the true partition p. The goal of training is to find a partition function h^* in a hypothesis space H that minimizes the error. Consequently, given a new data set S' in the application stage, under some standard learning theory assumptions, the hope is that the learned partition function

can generalize well and achieve small error as well.

3.1 Supervised and Unsupervised Training

Let p be the target function over X, h be a function in the hypothesis space H, and $h(S) = \{S'_k\}_1^K$. In principle, given data set $S \subseteq X$, if the true partition $p(S) = \{S_k\}_1^K$ of S is available, one can measure the deviation of h from p over S, using an *error function* $err_S(h, p) \to R^+$. We distinguish an error function from a quality function (as in Equ. 2) as follows: an error function measures the disagreement between clustering and the target partition (or one's intention) when supervision is given, while a quality is defined without any supervision.

For clustering, there is generally no direct way to compare the true class index $p(x)$ of each element with that given by a hypothesis $h(x)$, so an alternative is to measure the disagreement between p and h over pairs of elements. Given a labeled data set S and $p(S)$, one error function, namely *weighted clustering error*, is defined as a sum of the pairwise errors over any two elements in S, weighted by the distance between them:

$$err_S(h, p) \equiv \frac{1}{|S|^2} \sum_{x_i, x_j \in S} [d(x_i, x_j) \cdot A_{ij} + (D - d(x_i, x_j)) \cdot B_{ij}]$$

(3)

where $D = \max_{x_i, x_j \in S} d(x_i, x_j)$ is the maximum distance between any two elements in S and I is an indicator function. $A_{ij} \equiv I[(p(x_i) = p(x_j) \ \& \ h(x_i) \neq h(x_j)]$ and $B_{ij} \equiv I[(p(x_i) \neq p(x_j) \ \& \ h(x_i) = h(x_j)]$ represent two types of pairwise errors respectively.

Just like the quality defined in Equ. 2, this error is a function of the metric d. Intuitively, the contribution of a pair of elements that should belong to the same class but are split by h, grows with their distance, and vice versa. However, this measure is significantly different from the quality, in that it does not just measure the tightness of the partition given by h, but rather the difference between the tightness of the partitions given by h and by p.

Given a set of observed data, the goal of training is to learn a good partition function, parameterized by specific clustering algorithms and distance functions. Depending on whether training data is labeled or unlabeled, we can further define supervised and unsupervised training.

Definition 3.1 *Supervised Training: Given a labeled data set S and $p(S)$, a family of partition functions H, and the error function $err_S(h, p)(h \in H)$, the problem is to find an optimal function h^* s.t.*

$$h^* = argmin_{h \in H} \ err_S(h, p).$$

Definition 3.2 *Unsupervised Training: Given an unlabeled data set S ($p(S)$ is unknown), a family of partition functions H, and a quality function $q_S(h)(h \in H)$, the problem is to find an optimal partition function h^* s.t.*

$$h^* = argmax_{h \in H} \ q_S(h).$$

With this formalization, SDC along with supervised training, can be distinguished clearly from (1) unsupervised clustering approaches, (2) clustering over pairwise classification; and (3) related works that exploit partial supervision in metric learning as constraints.

3.2 Clustering via Metric Learning

By fixing the clustering algorithm in the training stage, we can further define supervised metric learning, a special case of supervised training.

Definition 3.3 *Supervised Metric Learning: Given a labeled data set S and $p(S)$, and a family of partition functions $H = \{h\}$ that are parameterized by a chosen clustering algorithm $\mathcal{A}$ and a family of distance metrics d_θ ($\theta \in \Omega$), the problem is to seek an optimal metric d_{θ^*} with respect to $\mathcal{A}$, s.t. for $h(S) = \mathcal{A}_{d_\theta}(S)$*

$$\theta^* = argmin_\theta \ err_S(h, p).$$

(4)

Learning the metric parameters θ requires parameterizing h as a function of θ, when the algorithm $\mathcal{A}$ is chosen and fixed in h. In the later experiments of Sec. 5, we try to learn weighted Manhattan distances for the single-link algorithm and other algorithms, in the task of entity identification. In this case, when pairwise features are extracted for any elements $x_1, x_2 \in X$, $(x_1, x_2) = < \phi_1, \phi_2, \cdots, \phi_m >$, the *linearly weighted Manhattan distance*, parameterized by ($\theta = \{w_l\}_1^m$) is defined as:

$$d(x_1, x_2) \equiv \sum_{l=1}^m w_l \cdot \phi_l(x_1, x_2)$$

(5)

where w_l is the weight over feature $\phi_l(x_1, x_2)$. Since measurement of the error is dependent on the metric, as shown in Equ. 3, one needs to enforce some constraints on the parameters. One constraint is $\sum_{l=1}^m |w_l| = 1$, which prevents the error from being scale-dependent (e.g., metrics giving smaller distance are always better).

4 A General Learner for SDC

In addition to the theoretical SDC framework, we also develop a practical learning algorithm based on gradient descent (in Fig. 3), that can train a distance function for any chosen clustering algorithm (such as Single-Linkage and K-Means), as in the setting of supervised metric learning. The training procedure incorporates the clustering algorithm (step 2.a) so that the metric is trained with respect to the specific algorithm that will be applied in evaluation. The convergence of this general training procedure depends on the convexity of the error as a function of θ. For example, since the error function we use is *linear* in θ, the algorithm is guaranteed to converge to a global minimum. In this case, for rate of convergence, one can appeal to general results that typically imply, when there exists a parameter vector with zero error, that convergence rate

depends on the 'separation'' of the training data, which roughly means the minimal error archived with this parameter vector. Results such as (Freund and Schapire, 1998) can be used to extend the rate of convergence result a bit beyond the separable case, when a small number of the pairs are not separable.

Algorithm: SDC-Learner
Input: S and $p(S)$: the labeled data set. $\mathcal{A}$: the clustering algorithm. $err_S(h, p)$: the clustering error function. $\alpha > 0$: the learning rate. T (typically T is large) : the number of iterations allowed.
Output: θ^* : the parameters in the distance function d.

1. In the initial (I-) step, we randomly choose θ^0 for d. After this step we have the initial d^0 and h^0.

2. Then we iterate over t ($t = 1, 2, \cdots$),

    (a) Partition S using $h^{t-1}(S) \equiv \mathcal{A}_{d^{t-1}}(S)$;
    (b) Compute $err_S(h^{t-1}, p)$ and update θ using the formula: $\theta^t = \theta^{t-1} - \alpha \cdot \frac{\partial err_S(h^{t-1}, p)}{\partial \theta^{t-1}}$.
    (c) Normalization: $\theta^t = \frac{1}{Z} \cdot \theta^t$, where $Z = ||\theta^t||$.

3. Stopping Criterion: If $t > T$, the algorithm exits and outputs the metric in the iteration with the least error.

Figure 3: **A general training algorithm for SDC**

For the weighted clustering error in Equ. 3, and linearly weighted Manhattan distances as in Equ. 5, the update rule in Step $2(b)$ becomes

$$w_l^t = w_l^{t-1} - \alpha \cdot [\psi_l^{t-1}(p, S) - \psi_l^{t-1}(h, S)]. \quad (6)$$

where $\psi_l(p, S) \equiv \frac{1}{|S|^2} \sum_{x_i, x_j \in S} \phi_l(x_i, x_j) \cdot I[p(x_i) = p(x_j)]$ and $\psi_l(h, S) \equiv \frac{1}{|S|^2} \sum_{x_i, x_j \in S} \phi_l(x_i, x_j) \cdot I[h(x_i) = h(x_j)]$, and $\alpha > 0$ is the learning rate.

5 Entity Identification in Text

We conduct experimental study on the task of entity identification in text (Bilenko et al., 2003; McCallum and Wellner, 2003; Li et al., 2004). A given entity – representing a person, a location or an organization – may be mentioned in text in multiple, ambiguous ways. Consider, for example, an open domain question answering system (Voorhees, 2002) that attempts, given a question like: "When was President Kennedy born?" to search a large collection of articles in order to pinpoint the concise answer: "on May 29, 1917." The sentence, and even the document that contains the answer, may not contain the name "President Kennedy"; it may refer to this entity as "Kennedy", "JFK" or "John Fitzgerald Kennedy". Other documents may state that "John F. Kennedy, Jr. was born on November 25, 1960", but this fact refers to our target entity's son. Other mentions, such as "Senator Kennedy" or "Mrs. Kennedy" are even "closer" to the writing of the target entity, but clearly refer to different entities. Understanding natural language requires identifying whether different mentions of a name, within and across documents, represent the same entity.

We study this problem for three entity types – People, Location and Organization. Although deciding the coreference of names within the same document might be relatively easy, since within a single document identical mentions typically refer to the same entity, identifying coreference across-document is much harder. With no standard corpora for studying the problem in a general setting – both within and across documents, we created our own corpus. This is done by collecting about $8,600$ names from 300 randomly sampled 1998-2000 New York Times articles in the TREC corpus (Voorhees, 2002). These names are first annotated by a named entity tagger, then manually verified and given as input to an entity identifier.

Since the number of classes (entities) for names is very large, standard multi-class classification is not feasible. Instead, we compare SDC with several pairwise classification and clustering approaches. Some of them (for example, those based on SoftTFIDF similarity) do not make use of any domain knowledge, while others do exploit supervision, such as LMR and SDC. Other works (Bilenko et al., 2003) also exploited supervision in this problem by discriminative training of a pairwise classifier but were shown to be inferior.

1. *SoftTFIDF Classifier* – a pairwise classifier deciding whether any two names refer to the same entity, implemented by thresholding a state-of-art SoftTFIDF similarity metric for string comparison (Cohen et al., 2003). Different thresholds have been experimented but only the best results are reported.

2. *LMR Classifier (P|W)* – a SNoW-based pairwise classifier (Li et al., 2004) (described in Sec. 2.2) that learns a linear function for each class over a collection of relational features between two names: including string and token-level features and structural features (listed in Table 1).

    For pairwise classifiers like LMR and SoftTFIDF, prediction is made over pairs of names so transitivity of predictions is not guaranteed as in clustering.

3. *Clustering over SoftTFIDF* – a clustering approach based on the SoftTFIDF similarity metric.

4. *Clustering over LMR (P|W)* – a clustering approach (Li et al., 2004) by converting the LMR classifier into a similarity metric (see Sec. 2.2).

5. *SDC* – our new supervised clustering approach. The distance metric is represented as a linear function over a set of pairwise features as defined in Equ. 5.

The above approaches (2), (4) and (5) learn a classifier or a distance metric using the same feature set as in Table 1. Different clustering algorithms [2], such as Single-Linkage, Complete-Linkage, Graph clustering (George,

---

[2]The clustering package *Cluster* by Michael Eisen at Stanford University is adopted for K-medoids and *CLUTO* by (George, 2003) is used for other algorithms. Details of these algorithms can be found there.

Honorific Equal	active if both tokens are honorifics and identical.
Honorific Equivalence	active if both tokens are honorifics, not identical, but equivalent.
Honorific Mismatch	active for different honorifics.
Equality	active if both tokens are identical.
Case-Insensitive Equal	active if the tokens are case-insensitive equal.
Nickname	active if tokens have a "nickname" relation.
Prefix Equality	active if the prefixes of both tokens are equal.
Substring	active if one of the tokens is a substring of the other.
Abbreviation	active if one of the tokens is an abbreviation of the other.
Prefix Edit Distance	active if the prefixes of both tokens have an edit-distance of 1.
Edit Distance	active if the tokens have an edit-distance of 1.
Initial	active if one of the tokens is an initial of another.
Symbol Map	active if one token is a symbolic representative of the other.
Structural	recording the location of the tokens that generate other features in two names.

Table 1: **Features employed by LMR and SDC.**

2003) – seeking a minimum cut of a nearest neighbor graph, Repeated Bisections and K-medoids (Chu et al., 2001) (a variation of K-means) are experimented in (5). The number of entities in a data set is always given.

6 Experimental Study

Our experimental study focuses on (1) evaluating the supervised discriminative clustering approach on entity identification; (2) comparing it with existing pairwise classification and clustering approaches widely used in similar tasks; and (3) further analyzing the characteristics of this new framework.

We use the TREC corpus to evaluate different approaches in identifying three types of entities: People, Locations and Organization. For each type, we generate three pairs of training and test sets, each containing about 300 names. We note that the three entity types yield very different data sets, exhibited by some statistical properties[3]. Results on each entity type will be averaged over the three sets and ten runs of two-fold cross-validation for each of them. For SDC, given a training set with annotated name pairs, a distance function is first trained using the algorithm in Fig. 3 (in 20 iterations) with respect to a clustering algorithm and then be used to partition the corresponding test set with the same algorithm.

For a comparative evaluation, the outcomes of each approach on a test set of names are converted to a classification over all possible pairs of names (including non-matching pairs). Only examples in the set M_p, those that are predicated to belong to the same entity (positive predictions) are used in the evaluation, and are compared with the set M_a of examples annotated as positive. The performance of an approach is then evaluated by F_1 value, defined as: $F_1 = \frac{2|M_p \bigcap M_a|}{|M_p|+|M_a|}$.

6.1 Comparison of Different Approaches

Fig. 4 presents the performance of different approaches (described in Sec. 5) on identifying the three entity types. We experimented with different clustering algorithms but only the results by Single-Linkage are reported for *Cluster over LMR (P|W)* and SDC, since they are the best.

SDC works well for all three entity types in spite of their different characteristics. The best F_1 values of SDC are 92.7%, 92.4% and 95.7% for people, locations and organizations respectively, about $20\% - 30\%$ error reduction compared with the best performance of the other approaches. This is an indication that this new approach which integrates metric learning and supervision in a unified framework, has significant advantages [4].

6.2 Further Analysis of SDC

In the next experiments, we will further analyze the characteristics of SDC by evaluating it in different settings.

Different Training Sizes Fig. 5 reports the relationship between the performance of SDC and different training sizes. The learning curves for other learning-based approaches are also shown. We find that SDC exhibits good learning ability with limited supervision. When training examples are very limited, for example, only 10% of all 300 names, pairwise classifiers based on Perceptron and Winnow exhibit advantages over SDC. However, when supervision become reasonable ($30\%+$ examples), SDC starts to outperform all other approaches.

Different Clustering Algorithms Fig. 6 shows the performance of applying different clustering algorithms (see Sec. 5) in the SDC approach. Single-Linkage and Complete-Linkage outperform all other algorithms. One possible reason is that this task has a great number of

---

[3]The average SoftTFIDF similarity between names of the same entity is 0.81, 0.89 and 0.95 for people, locations and organizations respectively.

[4]We note that in this experiment, the relative comparison between the pairwise classifiers and the clustering approaches over them is not consistent for all entity types. This can be partially explained by the theoretical analysis in (Li et al., 2004) and the difference between entity types.
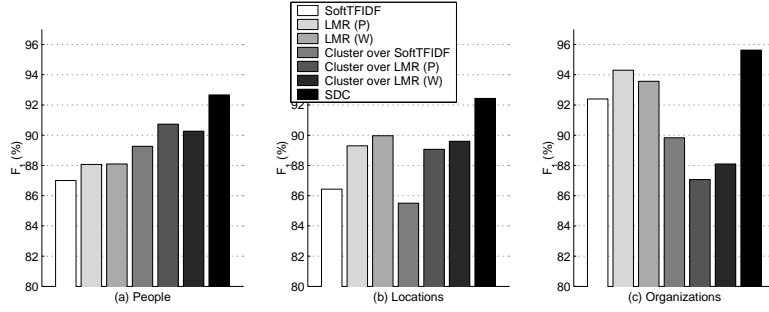
Figure 4: **Performance of different approaches.** The results are reported for SDC with a learning rate $\alpha = 100.0$. The Single-Linkage algorithm is applied whenever clustering is performed. Results are reported in F_1 and averaged over the three data sets for each entity type and 10 runs of two-fold cross-validation. Each training set typically contains 300 annotated names.
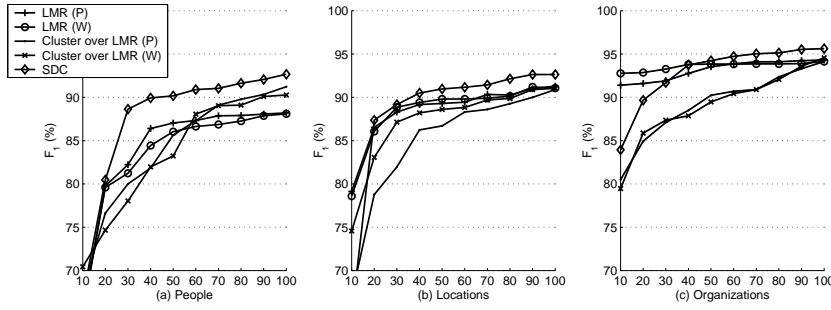

Figure 5: **Performance for different training sizes.** Five learning-based approaches are compared. Single-Linkage is applied whenever clustering is performed. X-axis denotes different percentages of 300 names used in training. Results are reported in F_1 and averaged over the three data sets for each entity type.
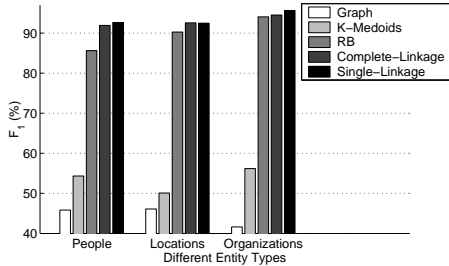

Figure 6: **Different clustering algorithms.** Five clustering algorithms are compared in SDC ($\alpha = 100.0$). Results are averaged over the three data sets for each entity type and 10 runs of two-fold cross-validations.
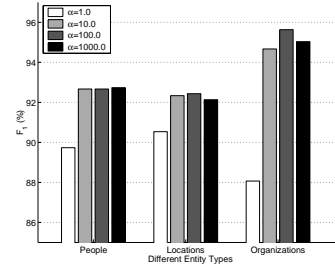

Figure 7: **Performance for different learning rates.** SDC with different learning rates ($\alpha = 1.0, 10.0, 100.0, 1000.0$) compared in this setting. Single-Linkage clustering algorithm is applied.

6.3 Discussion

The reason that SDC can outperform existing clustering approaches can be explained by the advantages of SDC – training the distance function with respect to the chosen clustering algorithm, guided by supervision, but they do not explain why it can also outperform the pairwise classifiers. One intuitive explanation is that supervision in the entity identification task or similar tasks is typically given on whether two names correspond to the same entity – entity-level annotation. Therefore it does not necessarily mean whether they are similar in appearance. For exam-

classes ($100 - 200$ entities) for 300 names in each single data set. The results indicate that the metric learning process relies on properties of the data set, as well as the clustering algorithm. Even if a good distance metric could be learned in SDC, choosing an appropriate algorithm for the specific task is still important.

Different Learning Rates We also experimented with different learning rates in the SDC approach as shown in Fig. 7. It seems that SDC is not very sensitive to different learning rates as long as it is in a reasonable range.

70

ple, "Brian" and "Wilson" could both refer to a person "Brian Wilson" in different contexts, and thus this name pair is a positive example in training a pairwise classifier. However, with features that only capture the appearance similarity between names, such apparently different names become training noise. This is what exactly happened when we train the LMR classifier with such name pairs. SDC, however, can employ this entity-level annotation and avoid the problem through transitivity in clustering. In the above example, if there is "Brian Wilson" in the data set, then "Brian" and "Wilson" can be both clustered into the same group with "Brian Wilson". Such cases do not frequently occur for locations and organization but still exist .

7 Conclusion

In this paper, we explicitly formalize clustering as a learning task, and propose a unified framework for training a metric for any chosen clustering algorithm, guided by domain-specific supervision. Our experiments exhibit the advantage of this approach over existing approaches on Entity Identification. Further research in this direction will focus on (1) applying it to more NLP tasks, e.g. coreference resolution; (2) analyzing the related theoretical issues, e.g. the convergence of the algorithm; and (3) comparing it experimentally with related approaches, such as (Xing et al., 2002) and (McCallum and Wellner, 2003).

References

F. R. Bach and M. I. Jordan. 2003. Learning spectral clustering. In *NIPS-03*.

A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. 2003. Learning distance functions using equivalence relations. In *ICML-03*, pages 11–18.

M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg. 2003. Adaptive name matching in information integration. *IEEE Intelligent Systems*, pages 16–23.

M Bilenko, S. Basu, and R. J. Mooney. 2004. Integrating constraints and metric learning in semi-supervised clustering. In *ICML-04*, pages 81–88.

P. Brown, P. deSouza R. Mercer, V. Pietra, and J. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

C. Cardie and K. Wagstaff. 1999. Noun phrase coreference as clustering. In *EMNLP-99*, pages 82–89.

S. C. Chu, J. F. Roddick, and J. S. Pan. 2001. A comparative study and extensions to k-medoids algorithms. In *ICOTA-01*.

W. Cohen and J. Richman. 2002. Learning to match and cluster large high-dimensional data sets for data integration. In *KDD-02*, pages 475–480.

W. Cohen, P. Ravikumar, and S. Fienberg. 2003. A comparison of string metrics for name-matching tasks. In *IIWeb Workshop 2003*, pages 73–78.

I. Dagan, L. Lee, and F. Pereira. 1999. Similarity-based models of word cooccurrence probabilities. *Machine Learning*, 34(1-3):43–69.

Y. Freund and R. Schapire. 1998. Large margin classification using the Perceptron algorithm. In *COLT-98*.

M. Geffet and I. Dagan. 2004. Automatic feature vector quality and distributional similarity. In *COLING-04*.

K. George. 2003. Cluto: A clustering toolkit. Technical report, Dept of Computer Science, University of Minnesota.

J. Hartigan and M. Wong. 1979. A k-means clustering algorithm. *Applied Statistics*, 28(1):100–108.

S. Kamvar, D. Klein, and C. Manning. 2002. Interpreting and extending classical agglomerative clustering algorithms using a model-based approach. In *ICML-02*, pages 283–290.

L. Lee. 1997. *Similarity-Based Approaches to Natural Language Processing*. Ph.D. thesis, Harvard University, Cambridge, MA.

X. Li, P. Morie, and D. Roth. 2004. Identification and tracing of ambiguous names: Discriminative and generative approaches. In *AAAI-04*, pages 419–424.

G. Mann and D. Yarowsky. 2003. Unsupervised personal name disambiguation. In *CoNLL-03*, pages 33–40.

A. McCallum and B. Wellner. 2003. Toward conditional models of identity uncertainty with application to proper noun coreference. In *IJCAI Workshop on Information Integration on the Web*.

D. Mochihashi, G. Kikui, and K. Kita. 2004. Learning non-structural distance metric by minimum cluster distortions. In *COLING-04*.

P. Pantel and D. Lin. 2002. Discovering word senses from text. In *KDD-02*, pages 613–619.

D. Roth. 1998. Learning to resolve natural language ambiguities: A unified approach. In *AAAI-98*, pages 806–813.

M. Schultz and T. Joachims. 2004. Learning a distance metric from relative comparisons. In *NIPS-04*.

E. Voorhees. 2002. Overview of the TREC-2002 question answering track. In *TREC-02*, pages 115–123.

J. Weeds, D. Weir, and D. McCarthy. 2004. Characterising measures of lexical distributional similarity. In *COLING-04*.

E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. 2002. Distance metric learning, with application to clustering with side-information. In *NIPS-02*.