Learning Field Compatibilities to Extract Database Records from Unstructured Text

Michael Wick, Aron Culotta and **Andrew McCallum**
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
{mwick, culotta, mccallum}@cs.umass.edu

Abstract

Named-entity recognition systems extract entities such as people, organizations, and locations from unstructured text. Rather than extract these mentions in isolation, this paper presents a *record extraction* system that assembles mentions into records (i.e. database tuples). We construct a probabilistic model of the compatibility between field values, then employ graph partitioning algorithms to cluster fields into cohesive records. We also investigate compatibility functions over *sets* of fields, rather than simply pairs of fields, to examine how higher representational power can impact performance. We apply our techniques to the task of extracting contact records from faculty and student homepages, demonstrating a 53% error reduction over baseline approaches.

1 Introduction

Information extraction (IE) algorithms populate a database with facts discovered from unstructured text. This database is often used by higher-level tasks such as question answering or knowledge discovery. The richer the structure of the database, the more useful it is to higher-level tasks.

A common IE task is named-entity recognition (NER), the problem of locating mentions of entities in text, such as people, places, and organizations. NER techniques range from regular expressions to finite-state sequence models (Bikel et al., 1999; Grishman, 1997; Sutton and McCallum, 2006). NER can be viewed as method of populating a database with single-tuple records, e.g. PERSON=*Cecil Conner* or ORGANIZATION= *IBM*.

We can add richer structure to these single-tuple records by extracting the associations among entities. For example, we can populate multi-field records such as a contact record [PERSON=*Steve Jobs*, JOBTITLE = *CEO*, COMPANY = *Apple*, CITY = *Cupertino*, STATE = *CA*]. The relational information in these types of records presents a greater opportunity for text analysis.

The task of associating together entities is often framed as a binary *relation extraction* task: Given a pair of entities, label the relation between them (e.g. *Steve Jobs* LOCATED-IN *Cupertino*). Common approaches to relation extraction include pattern matching (Brin, 1998; Agichtein and Gravano, 2000) and classification (Zelenko et al., 2003; Kambhatla, 2004).

However, binary relation extraction alone is not well-suited for the contact record example above, which requires associating together many fields into one record. We refer to this task of piecing together many fields into a single record as *record extraction*.

Consider the task of extracting contact records from personal homepages. An NER system may label all mentions of cities, people, organizations, phone numbers, job titles, etc. on a page, from both semi-structured an unstructured text. Even with a highly accurate NER system, it is not obvious which fields belong to the same record. For example, a single document could contain five names, three phone numbers and only one email. Additionally, the layout of certain fields may be convoluted or vary across documents.

Intuitively, we would like to learn the *compatibility* among fields, for example the likelihood that the organization *University of North Dakota* is located in the state *North Dakota*, or that phone numbers with area code *212* co-occur with the

city *New York*. Additionally, the system should take into account page layout information, so that nearby fields are more likely to be grouped into the same record.

In this paper, we describe a method to induce a probabilistic compatibility function between sets of fields. Embedding this compatibility function within a graph partitioning method, we describe how to cluster highly compatible fields into records.

We evaluate our approach on personal homepages that have been manually annotated with contact record information, and demonstrate a 53% error reduction over baseline methods.

2 Related Work

McDonald et al. (2005) present clustering techniques to extract *complex relations*, i.e. relations with more than two arguments. Record extraction can be viewed as an instance of complex relation extraction. We build upon this work in three ways: (1) Our system learns the compatibility between *sets* of fields, rather than just pairs of field; (2) our system is not restricted to relations between entities in the same sentence; and (3) our problem domain has a varying number of fields per record, as opposed to the fixed schema in McDonald et al. (2005).

Bansal et al. (2004) present algorithms for the related task of *correlational clustering*: finding an optimal clustering from a matrix of pairwise compatibility scores. The correlational clustering approach does not handle compatibility scores calculated over sets of nodes, which we address in this paper.

McCallum and Wellner (2005) discriminatively train a model to learn binary coreference decisions, then perform joint inference using graph partitioning. This is analogous to our work, with two distinctions. First, instead of binary coreference decisions, our model makes binary *compatibility* decisions, reflecting whether a set of fields belong together in the same record. Second, whereas McCallum and Wellner (2005) factor the coreference decisions into pairs of vertices, our compatibility decisions are made between sets of vertices. As we show in our experiments, factoring decisions into sets of vertices enables more powerful features that can improve performance. These higher-order features have also recently been investigated in other models of coreference, both discriminative (Culotta and McCallum, 2006) and generative (Milch et al., 2005).

Viola and Narasimhan (2005) present a probabilistic grammar to parse contact information blocks. While this model is capable of learning long-distance compatibilities (such as *City* and *State* relations), features to enable this are not explored. Additionally, their work focuses on labeling fields in documents that have been pre-segmented into records. This record segmentation is precisely what we address in this paper.

Borkar et al. (2001) and Kristjannson et al. (2004) also label contact address blocks, but ignore the problem of clustering fields into records. Also, Culotta et al. (2004) automatically extract contact records from web pages, but use heuristics to cluster fields into records.

Embley et al. (1999) provide heuristics to detect record boundaries in highly structured web documents, such as classified ads, and Embley and Xu (2000) improve upon these heuristics for slightly more ambiguous domains using a vector space model. Both of these techniques apply to data for which the records are highly contiguous and have a distinctive separator between records. These heuristic approaches are unlikely to be successful in the unstructured text domain we address in this paper.

Most other work on relation extraction focuses only on binary relations (Zelenko et al., 2003; Miller et al., 2000; Agichtein and Gravano, 2000; Culotta and Sorensen, 2004). A serious difficulty in applying binary relation extractors to the record extraction task is that rather than enumerating over all *pairs* of entities, the system must enumerate over all *subsets* of entities, up to subsets of size k, the maximum number of fields per record. We address this difficulty by employing two sampling methods: one that samples uniformly, and another that samples on a focused subset of the combinatorial space.

3 From Fields to Records

3.1 Problem Definition

Let a field F be a pair $\langle a, v \rangle$, where a is an attribute (column label) and v is a value, e.g. $F_i = \langle$CITY, *San Francisco*$\rangle$. Let record R be a set of fields, $R = \{F_1 \dots F_n\}$. Note that R may contain multiple fields with the same attribute but different values (e.g. a person may have multiple job titles). Assume we are given the output of a named-

entity recognizer, which labels tokens in a document with their attribute type (e.g. NAME or CITY). Thus, a document initially contains a set of fields, $\{F_1 \dots F_m\}$.

The task is to partition the fields in each annotated document into a set of records $\{R_1 \dots R_k\}$ such that each record R_i contains exactly the set of fields pertinent to that record. In this paper, we assume each field belongs to exactly one record.

3.2 Solution Overview

For each document, we construct a fully-connected weighted graph $G = (V, E)$, with vertices V and weighted edges E. Each field in the document is represented by a vertex in V, and the edges are weighted by the compatibility of adjacent fields, i.e. a measure of how likely it is that F_i and F_j belong to the same record.

Partitioning V into k disjoint clusters uniquely maps the set of fields to a set of k records. Below, we provide more detail on the two principal steps in our solution: (1) estimating the compatibility function and (2) partitioning V into disjoint clusters.

3.3 Learning field compatibility

Let $\mathcal{F}$ be a candidate cluster of fields forming a partial record. We construct a compatibility function C that maps two sets of fields to a real value, i.e. $C : \mathcal{F}_i \times \mathcal{F}_j \rightarrow \mathcal{R}$. We abbreviate the value $C(\mathcal{F}_i, \mathcal{F}_j)$ as C_{ij}. The higher the value of C_{ij} the more likely it is that $\mathcal{F}_i$ and $\mathcal{F}_j$ belong to the same record.

For example, in the contact record domain, C_{ij} can reflect whether a city and state should co-occur, or how likely a company is to have a certain job title.

We represent C_{ij} by a maximum-entropy classifier over the binary variable S_{ij}, which is *true* if and only if field set $\mathcal{F}_i$ belongs to the same record as field set $\mathcal{F}_j$. Thus, we model the conditional distribution

$$P_\Lambda(S_{ij}|\mathcal{F}_i, \mathcal{F}_j) \propto \exp\left(\sum_k \lambda_k f_k(S_{ij}, \mathcal{F}_i, \mathcal{F}_j)\right)$$

where f_k is a binary feature function that computes attributes over the field sets, and $\Lambda = \{\lambda_k\}$ is the set of real-valued weights that are the parameters of the maximum-entropy model. We set $C_{ij} = P_\Lambda(S_{ij} = \textit{true}|\mathcal{F}_i, \mathcal{F}_j)$. This approach can be viewed as a logistic regression model for field compatibility.

Examples of feature functions include formatting evidence ($\mathcal{F}_i$ appears at the top of the document, $\mathcal{F}_j$ at the bottom), conflicting value information ($\mathcal{F}_i$ and $\mathcal{F}_j$ contain conflicting values for the *state* field), or other measures of compatibility (a *city* value in $\mathcal{F}_i$ is known to exist in a state in $\mathcal{F}_j$). A feature may involve more than one field, for example, if a name, title and university occurs consecutively in some order. We give a more detailed description of the feature functions in Section 4.3.

We propose learning the Λ weights for each of these features using supervised machine learning. Given a set of documents $\mathcal{D}$ for which the true mapping from fields to set of records is known, we wish to estimate $P(S_{ij}|\mathcal{F}_i, \mathcal{F}_j)$ for all pairs of field sets $\mathcal{F}_i, \mathcal{F}_j$.

Enumerating all positive and negative pairs of field sets is computationally infeasible for large datasets, so we instead propose two sampling methods to generate training examples. The first simply samples pairs of field sets uniformly from the training data. For example, given a document D containing true records $\{R_1 \dots R_k\}$, we sample positive and negative examples of field sets of varying sizes from $\{R_i \dots R_j\}$. The second sampling method first trains the model using the examples generated by uniform sampling. This model is then used to cluster the training data. Additional training examples are created during the clustering process and are used to retrain the model parameters. This second sampling method is an attempt to more closely align the characteristics of the training and testing examples.

Given a sample of labeled training data, we set the parameters of the maximum-entropy classifier in standard maximum-likelihood fashion, performing gradient ascent on the log-likelihood of the training data. The resulting weights indicate how important each feature is in determining whether two sets of fields belong to the same record.

3.4 Partitioning Fields into Records

One could employ the estimated classifier to convert fields into records as follows: Classify each pair of fields as *positive* or *negative*, and perform transitive closure to enforce transitivity of decisions. That is, if the classifier determines that A and B belong to the same record and that B and C belong to the same record, then by transitivity

A and C must belong to the same record. The drawback of this approach is that the compatibility between A and C is ignored. In cases where the classifier determines that A and C are highly *incompatible*, transitive closure can lead to poor precision. McCallum and Wellner (2005) explore this issue in depth for the related task of noun coreference resolution.

With this in mind, we choose to avoid transitive closure, and instead employ a graph partitioning method to make record merging decisions jointly.

Given a document D with fields $\{F_1 \dots F_n\}$, we construct a fully connected graph $G = (V, E)$, with edge weights determined by the learned compatibility function C. We wish to partition vertices V into clusters with high intra-cluster compatibility.

One approach is to simply use greedy agglomerative clustering: initialize each vertex to its own cluster, then iteratively merge clusters with the highest inter-cluster edge weights. The compatibility between two clusters can be measured using single-link or average-link clustering. The clustering algorithm converges when the inter-cluster edge weight between any pair of clusters is below a specified threshold.

We propose a modification to this approach. Since the compatibility function we have described maps two *sets* of vertices to a real value, we can use this directly to calculate the compatibility between two clusters, rather than performing average or single link clustering.

We now describe the algorithm more concretely.

- **Input:** (1) Graph $G = (V, E)$, where each vertex v_i represents a field F_i. (2) A threshold value τ.
- **Initialization:** Place each vertex v_i in its own cluster $\hat{R}_i$. (The hat notation indicates that this cluster represents a possible record.)
- **Iterate:** Re-calculate the compatibility function C_{ij} between each pair of clusters. Merge the two most compatible clusters, $\hat{R}_i^*, \hat{R}_j^*$.
- **Termination:** If there does not exist a pair of clusters $\hat{R}_i, \hat{R}_j$ such that $C_{ij} > \tau$, the algorithm terminates and returns the current set of clusters.

A natural threshold value is $\tau = 0.5$, since this is the point at which the binary compatibility classifier predicts that the fields belong to different records. In Section 4.4, we examine how performance varies with τ.

3.5 Representational power of cluster compatibility functions

Most previous work on inducing compatibility functions learns the compatibility between *pairs* of vertices, not *clusters* of vertices. In this section, we provide intuition to explain why directly modeling the compatibility of clusters of vertices may be advantageous. We refer to the cluster compatibility function as C_{ij}, and the pairwise (binary) compatibility function as B_{ij}.

First, we note that C_{ij} is a generalization of single-link and average-link clustering methods that use B_{ij}, since the output of these methods can simply be included as features in C_{ij}. For example, given two clusters $\hat{R}_i = \{v_1, v_2, v_3\}$ and $\hat{R}_j = \{v_4, v_5, v_6\}$, average-link clustering calculates the inter-cluster score between $\hat{R}_i$ and $\hat{R}_j$ as

$$S_{AL}(\hat{R}_i, \hat{R}_j) = \frac{1}{|\hat{R}_i||\hat{R}_j|} \sum_{a \in \hat{R}_i, b \in \hat{R}_j} B_{ab}$$

$S_{AL}(\hat{R}_i, \hat{R}_j)$ can be included as a feature for the compatibility function C_{ij}, with an associated weight estimated from training data.

Second, there may exist phenomena of the data that can only be captured by a classifier that considers "higher-order" features. Below we describe two such cases.

In the first example, consider three vertices of mild compatibility, as in Figure 1(a). (For these examples, let $B_{ij}, C_{ij} \in [0, 1]$.) Suppose that these three phone numbers occur nearby in a document. Since it is not uncommon for a person to have two phone numbers with different area codes, the pairwise compatibility function may score any pair of nearby phone numbers as relatively compatible. However, since it is fairly uncommon for a person to have three phone numbers with *three* different area codes, we would not like all three numbers to be merged into the same record.

Assume an average-link clustering algorithm. After merging together the *333* and *444* numbers, B_{ij} will recompute the new inter-cluster compatibility as 0.51, the average of the inter-cluster edges. In contrast, the cluster compatibility function C_{ij} can represent the fact that three numbers with different area codes are to be merged, and can penalize their compatibility accordingly. Thus, in

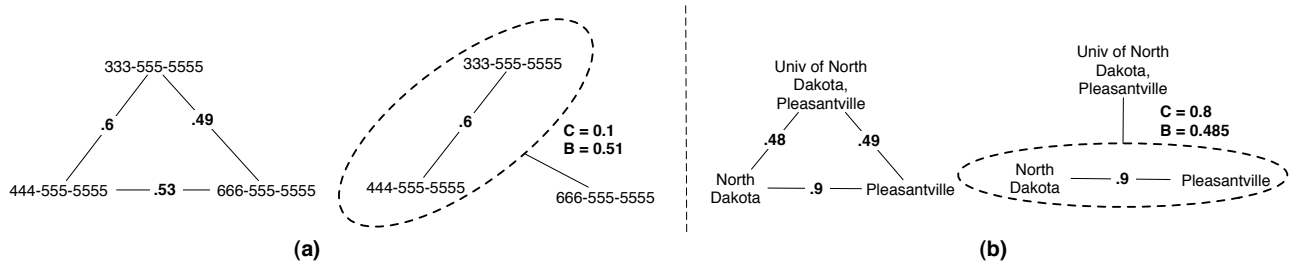

Figure 1: Two motivating examples illustrating why the cluster compatibility measure (**C**) may have higher representational power than the pairwise compatibility measure (**B**). In (a), the pairwise measure *over*-estimates the inter-cluster compatibility when there exist higher-order features such as *A person is unlikely to have phone numbers with three different area codes.* In (b), the pairwise measure *under*-estimates inter-cluster compatibility when weak features like string comparisons can be combined into a more powerful feature by examining multiple field values.

this example, the pairwise compatibility function over-estimates the true compatibility.

In the second example (Figure 1(b)), we consider the opposite case. Consider three edges, two of which have weak compatibility, and one of which has high compatibility. For example, perhaps the system has access to a list of city-state pairs, and can reliably conclude that *Pleasantville* is a city in the state *North Dakota*.

Deciding that *Univ of North Dakota, Pleasantville* belongs in the same record as *North Dakota* and *Pleasantville* is a bit more difficult. Suppose a feature function measures the string similarity between the city field *Pleasantville* and the company field *Univ of North Dakota, Pleasantville*. Alone, this string similarity might not be very strong, and so the pairwise compatibility is low. However, after *Pleasantville* and *North Dakota* are merged together, the cluster compatibility function can compute the string similarity of the *concatenation* of the city and state fields, resulting in a higher compatibility. In this example, the pairwise compatibility function underestimates the true compatibility.

These two examples show that the cluster compatibility score can have more representational power than the average of pairwise compatibility scores.

FirstName	MiddleName
LastName	NickName
Suffix	Title
JobTitle	CompanyName
Department	AddressLine
City1	City2
State	Country
PostalCode	HomePhone
Fax	CompanyPhone
DirectCompanyPhone	Mobile
Pager	VoiceMail
URL	Email
InstantMessage	

Table 1: The 25 fields annotated in the contact record dataset.

4 Experiments

4.1 Data

We hand-labeled a subset of faculty and student homepages from the WebKB dataset[1]. Each page was labeled with the 25 fields listed in Table 1. In addition, we labeled the records to which each field belonged. For example, in Figure 2, we labeled the contact information for Professor Smith into a separate record from that of her administrative assistant. There are 252 labeled pages in total, containing 8996 fields and 16679 word tokens. We perform ten random samples of 70-30 splits of the data for all experiments.

4.2 Systems

We evaluate five different record extraction systems. With the exception of **Transitive Closure**, all methods employ the agglomerative clustering

[1] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/

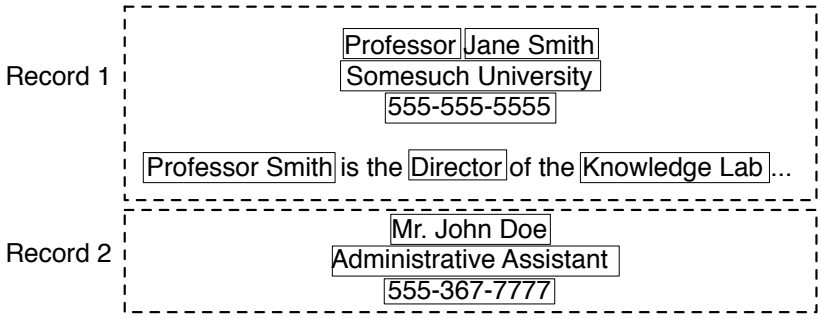

Figure 2: A synthetic example representative of the labeled data. Note that Record 1 contains information both from an address block and from free text, and that Record 2 must be separated from Record 1 even though fields from each may be nearby in the text.

algorithm described previously. The difference is in how the inter-cluster compatibility is calculated.

- **Transitive Closure:** The method described in the beginning of Section 3.4, where hard classification decisions are made, and transitivity is enforced.
- **Pairwise Compatibility:** In this approach, the compatibility function only estimates the compatibility between *pairs* of fields, not *sets* of fields. To compute inter-cluster compatibility, the mean of the edges between the clusters is calculated.
- **McDonald:** This method uses the pairwise compatibility function, but instead of calculating the mean of inter-cluster edges, it calculates the geometric mean of all pairs of edges in the potential new cluster. That is, to calculate the compatibility of records R_i and R_j, we construct a new record R_{ij} that contains all fields of R_i and R_j, then calculate the geometric mean of all pairs of fields in R_{ij}. This is analogous to the method used in McDonald et al. (2005) for relation extraction.
- **Cluster Compatibility (uniform):** Inter-cluster compatibility is calculated directly by the cluster compatibility function. This is the method we advocate in Section 3. Training examples are sampled uniformly as described in Section 3.3.
- **Cluster Compatibility (iterative):** Same as above, but training examples are sampled using the iterative method described in Section 3.3.

4.3 Features

For the pairwise compatibility classifier, we exploit various formatting as well as knowledge-based features. Formatting features include the number of hard returns between fields, whether the fields occur on the same line, and whether the fields occur consecutively. Knowledge-based features include a mapping we compiled of cities and states in the United States and Canada. Additionally, we used compatibility features, such as which fields are of the same type but have different values.

In building the cluster compatibility classifier, we use many of the same features as in the binary classifier, but cast them as first-order existential features that are generated if the feature exists between any pair of fields in the two clusters. Additionally, we are able to exploit more powerful compatibility and knowledge-base features. For example, we examine if a title, a first name and a last name occur consecutively (i.e., no other fields occur in-between them). Also, we examine multiple telephone numbers to ensure that they have the same area codes. Additionally, we employ count features that indicate if a certain field occurs more than a given threshold.

4.4 Results

For these experiments, we compare performance on the *true* record for each page. That is, we calculate how often each system returns a complete and accurate extraction of the contact record pertaining to the owner of the webpage. We refer to

this record as the *canonical record* and measure performance in terms of precision, recall and F1 for each field in the canonical record.

Table 2 compares precision, recall and F1 across the various systems. The cluster compatibility method with iterative sampling has the highest F1, demonstrating a 14% error reduction over the next best method and a 53% error reduction over the transitive closure baseline.

Transitive closure has the highest recall, but it comes at the expense of precision, and hence obtains lower F1 scores than more conservative compatibility methods. The McDonald method also has high recall, but drastically improves precision over the transitivity method by taking into consideration all edge weights.

The pairwise measure yields a slightly higher F1 score than McDonald mostly due to precision improvements. Because the McDonald method calculates the mean of *all* edge weights rather than just the inter-cluster edge weights, inter-cluster weights are often outweighed by intra-cluster weights. This can cause two densely-connected clusters to be merged despite low inter-cluster edge weights.

To further investigate performance differences, we perform three additional experiments. The first measures how sensitive the algorithms are to the threshold value τ. Figure 3 plots the precision-recall curve obtained by varying τ from 1.0 to 0.1. As expected, high values of τ result in low recall but high precision, since the algorithms halt with a large number of small clusters. The highlighted points correspond to $\tau = 0.5$. These results indicate that setting τ to 0.5 is near optimal, and that the cluster compatibility method outperforms the pairwise across a wide range of values for τ.

In the second experiment, we plot F1 versus the size of the canonical record. Figure 4 indicates that most of the performance gain occurs in smaller canonical records (containing between 6 and 12 fields). Small canonical records are most susceptible to precision errors simply because there are more extraneous fields that may be incorrectly assigned to them. These precision errors are often addressed by the cluster compatibility method, as shown in Table 2.

In the final experiment, we plot F1 versus the total number of fields on the page. Figure 5 indicates that the cluster compatibility method is best at handling documents with large number of fields.

	F1	Precision	Recall
Cluster (I)	**91.81** (.013)	92.87 (.005)	90.78 (.007)
Cluster (U)	90.02 (.012)	**93.56 (.007)**	86.74 (.011)
Pairwise	90.51 (.013)	91.07 (.004)	89.95 (.006)
McDonald	88.36 (.012)	83.55 (.004)	93.75 (.005)
Trans Clos	82.37 (.002)	70.75 (.009)	**98.56** (.020)

Table 2: Precision, recall, and F1 performance for the record extraction task. The standard error is calculated over 10 cross-validation trials.

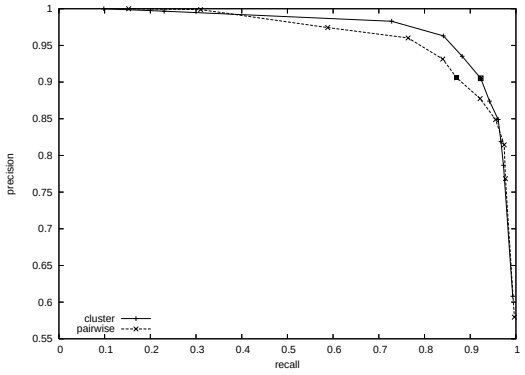

Figure 3: Precision-recall curve comparing cluster and pairwise compatibility methods. The graph is obtained by varying the stopping threshold τ from 1.0 to 0.1. The highlighted points correspond to $\tau = 0.5$.

When there are over 80 fields in the document, the performance of the pairwise method drops dramatically, while cluster compatibility only declines slightly. We believe the improved precision of the cluster compatibility method explains this trend as well.

We also examine documents where cluster compatibility outperforms the pairwise methods. Typically, these documents contain interleaving contact records. Often, it is the case that a single pair of fields is sufficient to determine whether a cluster should *not* be merged. For example, the cluster classifier can directly model the fact that a contact record should not have multiple first or last names. It can also associate a weight with the fact that several fields overlap (e.g., the chances that a cluster has two first names, two last names and two cities). In contrast, the binary classifier only examines pairs of fields in isolation and averages these probabilities with other edges. This averaging can dilute the evidence from a single pair of fields. Embarrassing errors may result, such as

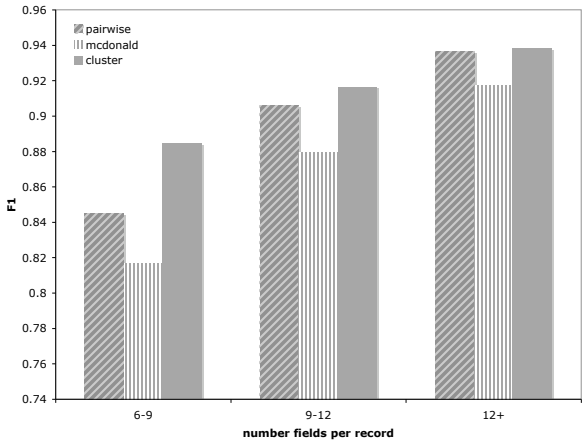

Figure 4: Field F1 as the size of the canonical record increases. This figure suggests that cluster compatibility is most helpful for small records.

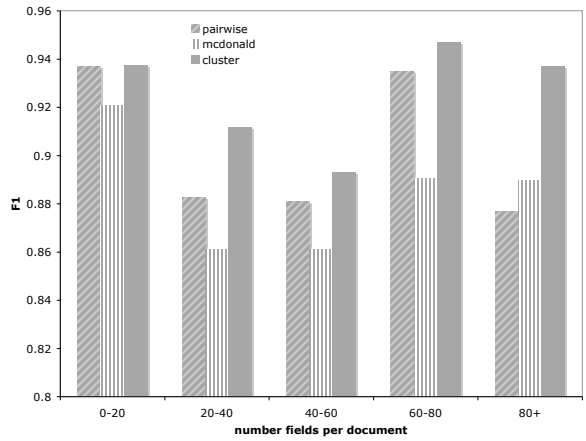

Figure 5: Field F1 as the number of fields in the document increases. This figure suggests that cluster compatibility is most helpful when the document has more than 80 fields.

a contact record with two first names or two last names. These errors are particularly prevalent in interleaving contact records since adjacent fields often belong to the same record.

5 Conclusions and Future Work

We have investigated graph partitioning methods for discovering database records from fields annotated in text. We have proposed a cluster compatibility function that measures how likely it is that two *sets* of fields belong to the same cluster. We argue that this enhancement to existing techniques provides more representational power.

We have evaluated these methods on a set of hand-annotated data and concluded that (1) graph partitioning techniques are more accurate than performing transitive closure, and (2) cluster compatibility methods can avoid common mistakes made by pairwise compatibility methods.

As information extraction systems become more reliable, it will become increasingly important to develop accurate ways of associating disparate fields into cohesive records. This will enable more complex reasoning over text.

One shortcoming of this approach is that fields are not allowed to belong to multiple records, because the partitioning algorithm returns non-overlapping clusters. Exploring overlapping clustering techniques is an area of future work.

Another avenue of future research is to consider syntactic information in the compatibility function. While performance on contact record extraction is highly influenced by formatting features, many fields occur within sentences, and syntactic information (such as dependency trees or phrase-structure trees) may improve performance.

Overall performance can also be improved by increasing the sophistication of the partitioning method. For example, we can examine "block moves" to swap multiple fields between clusters in unison, possibly avoiding local minima of the greedy method (Kanani et al., 2006). This can be especially helpful because many mistakes may be made at the start of clustering, before clusters are large enough to reflect true records.

Additionally, many personal web pages contain a time-line of information that describe a person's educational and professional history. Learning to associate time information with each contact record enables *career path modeling*, which presents interesting opportunities for knowledge discovery techniques, a subject of ongoing work.

Acknowledgments

We thank the anonymous reviewers for helpful suggestions. This work was supported in part by the Center for Intelligent Information Retrieval, in part by U.S. Government contract #NBCH040171 through a subcontract with BBNT Solutions LLC, in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0326249, and in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, un-

der contract number NBCHD030010. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*.

Nikhil Bansal, Avrim Blum, and Shuchi Chawla. 2004. Correlation clustering. *Machine Learining*, 56:89–113.

Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning*, 34:211–231.

Vinayak R. Borkar, Kaustubh Deshmukh, and Sunita Sarawagi. 2001. Automatic segmentation of text into structured records. In *SIGMOD Conference*.

Sergey Brin. 1998. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology*.

Aron Culotta and Andrew McCallum. 2006. Practical Markov logic containing first-order quantifiers with application to identity uncertainty. In *HLT Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*, June.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *ACL*.

Aron Culotta, Ron Bekkerman, and Andrew McCallum. 2004. Extracting social networks and contact information from email and the web. In *First Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA.

David W. Embley and Lin Xu. 2000. Record location and reconfiguration in unstructured multiple-record web documents. In *WebDB*, pages 123–128.

David W. Embley, Xiaoyi Jiang, and Yiu-Kai Ng. 1999. Record-boundary discovery in web documents. In *SIGMOD Conference*, pages 467–478.

Ralph Grishman. 1997. Information extraction: Techniques and challenges. In *SCIE*, pages 10–27.

Nanda Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *ACL*.

Pallika Kanani, Andrew McCallum, and Chris Pal. 2006. Improving author coreference by resource-bounded information gathering from the web. Technical note.

Trausti Kristjannson, Aron Culotta, Paul Viola, and Andrew McCallum. 2004. Interactive information extraction with conditional random fields. *Nineteenth National Conference on Artificial Intelligence (AAAI 2004)*.

Andrew McCallum and Ben Wellner. 2005. Conditional models of identity uncertainty with application to noun coreference. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA.

Ryan McDonald, Fernando Pereira, Seth Kulick, Scott Winters, Yang Jin, and Pete White. 2005. Simple algorithms for complex relation extraction with applications to biomedical ie. In *43rd Annual Meeting of the Association for Computational Linguistics*.

Brian Milch, Bhaskara Marthi, Stuart Russell, David Sontag, Daniel L. Ong, and Andrey Kolobov. 2005. BLOG: Probabilistic models with unknown objects. In *IJCAI*.

Scott Miller, Heidi Fox, Lance A. Ramshaw, and Ralph Weischedel. 2000. A novel use of statistical parsing to extract information from text. In *ANLP*.

Charles Sutton and Andrew McCallum. 2006. An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press. To appear.

Paul Viola and Mukund Narasimhan. 2005. Learning to extract information from semi-structured text using a discriminative context free grammar. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 330–337, New York, NY, USA. ACM Press.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.