Domain Adaptation of Information Extraction Models

Rahul Gupta
IIT Bombay
grahul@cse.iitb.ac.in

Sunita Sarawagi
IIT Bombay
sunita@iitb.ac.in

ABSTRACT

Domain adaptation refers to the process of adapting an extraction model trained in one domain to another related domain with only unlabeled data. We present a brief survey of existing methods of retraining models to best exploit labeled data from a related domain. These approaches that involve expensive model retraining are not practical when a large number of new domains have to be handled in an operational setting. We describe our approach for adapting record extraction models that exploits the regularity within a domain to jointly label records without retraining any model.

1. INTRODUCTION

The construction of models employed by information extractors is an expensive process requiring tedious effort either in collecting labeled data or hand coding the models. In many cases, it is possible to substantially reduce this effort if a model from a related domain is available. For example, we might find a model for extracting people names from news articles, while we are actually interested in extracting people name mentions in emails. Or, we find a model to identify the polarity of sentiment about home appliances and we are interested in determining the sentiment about audio equipment. In both these cases, although our target domain is related to the original domain, it has a systematic difference so that a blind application of the model is not expected to provide high accuracy. Even within the same domain, an extraction model often needs to be applied on unstructured sources that within themselves display a regularity not foreseeable at the time of model creation. For example, a model trained to extract fields of citation records can be improved significantly by exploiting the regularity of multiple strings from the same web page.

Such forms of domain adaptation will be essential in any large scale information extraction system involving multiple kinds of extraction tasks on evolving and open-ended sources. While domain adaptation is applicable both for manually-coded and machine learning models, in this article we concentrate on machine learning models. We will present an overview of the main techniques that have emerged recently from machine learning and natural language processing communities, and then present an overview of our research in the area.

2. BASICS OF LEARNING-BASED IE MODELS

Many IE tasks, including entity extraction, relationship extraction, and sentiment extraction, are formulated as feature-based prediction models. These models predict a label $\mathbf{y}$ from a space $\mathcal{Y}$ given an input $\mathbf{x}$ based on a feature vector $\mathbf{f}(\mathbf{x}, \mathbf{y}) \in \mathcal{R}^K$ that maps any $(\mathbf{x}, \mathbf{y})$ pair to a vector of K reals. The feature vector is defined by the user and provides a convenient abstraction for capturing many varied kinds of clues known to aid the extraction task. The model associates a weight vector $\mathbf{w}$ corresponding to the feature vector $\mathbf{f}$ and the predicted label is simply the $\mathbf{y}$ with the highest value of $\mathbf{w} \cdot \mathbf{f}(\mathbf{x}, \mathbf{y})$. We briefly illustrate how this works for different kind of extraction tasks. More details can be found in this survey [11].

In sentiment extraction, the space $\mathcal{Y}$ of possible predictions is positive or negative and an entry in the feature vector is typically the counts of occurrences of a particular word or frequent word bigram in the input document $\mathbf{x}$.

In relationship extraction, the input $\mathbf{x}$ is a sentence and two marked entity strings in it. The task is to predict the kind of relationship that exists between the entity pair. The space $\mathcal{Y}$ consists of possible relationship types as defined by the user and a special label "other" indicating none of the above. The feature vector entries capture various syntactic and semantic relationships between the strings, such as the bag of words between the two mentions, the parts-of-speech information of words adjacent to the mention, and so on.

In entity extraction, the task is to label words in a sentence with one of a fixed set of entity types. The prediction $\mathbf{y}$ is therefore a vector of length n for an input sentence of n words and thus the space $\mathcal{Y}$ of possible labels is m^n where m is the number of possible entity types. Instead of explicitly searching over this exponential sized space, we assume that feature vector $\mathbf{f}(\mathbf{x}, \mathbf{y})$ decomposes as a sum of local features that apply over label pairs of adjacent words. This decomposition is exploited for efficient inference over the space of variables $\mathbf{y}$. The local feature vector entries consist of various properties of a word and its neighboring words. Typical properties useful for entity extraction are the case pattern of the word, its orthographic type, match in a dictionary of known types, its part of speech, and so on.

2.1 Training the weight vector w

During training we are given a labeled set SRC $= \{(\mathbf{x}_\ell, \mathbf{y}_\ell)\}_{\ell=1}^N$ consisting of correct input output pairs and our goal is to find the value of $\mathbf{w}$ that will minimize error on future inputs. This training objective is expressed as:

$$\min_{\mathbf{w}} \sum_{\ell} \text{loss}(\mathbf{x}_\ell, \mathbf{y}_\ell, \mathbf{w}, \mathbf{f}) + C||\mathbf{w}||^\gamma \qquad (1)$$

where $\text{loss}(\mathbf{x}_\ell, \mathbf{y}_\ell, \mathbf{w}, \mathbf{f})$ is a function that measures for input $\mathbf{x}_\ell$ the error in predicting the label $\hat{\mathbf{y}} = \text{argmax}_{\mathbf{y}} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, \mathbf{y})$ given that its correct label is $\mathbf{y}_\ell$. A popular form of the loss function is $\mathbf{w} \cdot \mathbf{f}(\mathbf{x}_\ell, \mathbf{y}_\ell) - \log \sum_{\mathbf{y}} exp(\mathbf{w} \cdot \mathbf{f}(\mathbf{x}_\ell, \mathbf{y}))$. This form of the loss function is both efficient to train and has been found to generalize well to future instances when they follow the same distribution as the training data. The second term $C||\mathbf{w}||^\gamma$ (with γ usually 1 or 2) prevents the model from overfitting $\mathbf{w}$ to the labeled set.

2.2 Domain adaptation

In domain adaptation we are faced with the following situation: there is labeled dataset SRC from one or multiple domains but on the domain on which we need predictions we only have an unlabeled pool DEST. Various existing domain adaptation techniques tend to retrain the model after seeing the examples in DEST [13, 8, 1, 12, 3, 2]. We first give an overview of these methods in Section 3. Next, we give a summary of our method for domain adaptation in record extraction that does not require retraining. Such methods are more useful in settings where we need to handle many target domains and retraining the model for each domain is expensive.

3. CURRENT DOMAIN ADAPTATION TECHNIQUES

The key idea in the various methods of adapting the labeled examples in SRC to provide high accuracy on DEST is to choose a representation of the SRC examples that make them close to the DEST distribution. We discuss three proposed methods of achieving this goal.

3.1 Select relevant examples

One strategy to make the examples in SRC relevant to classifying examples in DEST is to differentially weight examples in SRC such that higher weights are assigned to examples that are similar to our target examples. Let β_i denote the weight assigned to the ith example in SRC. The training objective (Eq. 1) is modified such that the loss of the i-th example is multiplied by β_i. We now discuss how to set β_i.

The ideal weight of each example $\mathbf{x}$ in SRC should be the ratio of its probability in the target and the source domains. As shown in [13], this is the optimum way to align the source distribution to the target distribution. However, this approach is not practical because we do not have a probability distribution over the examples in each domain. Estimating the distribution through the samples is difficult because in general, $\mathbf{x}$ has many dimensions. Therefore, a number of methods have been proposed to estimate the β values directly without first estimating the probability of $\mathbf{x}$ in each of the domains.

A mean matching method proposed in [8] assigns weights β_i to the examples in SRC such that the mean of the weighted examples in SRC matches the means of the examples in DEST. In contrast, [1] uses another classifier to estimate the probability that an example is from the target distribution as against the source distribution. In training this classifier, the examples in DEST are treated as positive examples and the ones in SRC are treated as negative examples.

3.2 Remove irrelevant features

The above method of weighting entire examples is not effective when a few features cause the two domains to differ systematically from each other. For example, if we have a feature called "Is capitalized word" in the source domain but in the target domain every letter is capitalized, then no instance weighting scheme can align the two distributions. In such kinds of mismatch a more effective method of domain adaptation is to differentially weight features instead of examples. A method proposed in [12] is to assign a weight λ_j to each feature j that is equal to the difference in the expected value of the feature in the two domains. The model is then retrained by adding a third term $\sum_j \lambda_j |w_j|$ to the training objective in Equation 1 that penalizes features with large λ_j values so that their role in the final classification is minimized. This method has been shown to provide significant accuracy gains on entity extraction tasks with varying training and test domains [12].

3.3 Add related features

A third strategy proposed in [3, 2] is to add new features to the target domain by aligning them to anchor features in the labeled source domain. Anchor features are those that are frequent in the two domains and are strongly correlated to the class labels. As an example, consider the task of sentiment extraction where the source domain has labeled book reviews and the target domain has unlabeled home appliance reviews. Words like "good", "dissatisfied" and "excellent" that are present in both the domains and strongly correlated with the class labels in the source domain are good candidates for such anchor features. Once a set of anchor features is determined, we find the set of those features from the target and source domains that are strongly correlated with each anchor feature. For example, with the anchor feature "excellent", in the book domain we might find words like "engaging" and in the appliance domain words like "reliable" strongly associated with it. This establishes a correspondence between features "engaging" and "reliable". We refer the reader to [3, 2] for more details on how such a correspondence is quantified and exploited during model retraining.

4. DOMAIN ADAPTATION FOR RECORD EXTRACTION

We now present our approach for domain adaptation which does not do any model retraining. Our approach uses the key observation that record labelings inside a domain tend to have regularity in many aspects. For example, in a citation labeling task, records in the same list tend to use the same ordering of labels. They also tend to use the same font/formatting style for specific labels, such as bold Titles in one domain, or italicized Titles in another. Thus, regularities tend to exist in each domain, although the nature of each such regularity might vary from domain to domain. To illustrate further, Table 1 lists citation records from two domains. All labelings in the first domain start with Author, while those in the second domain start with a Title. All the titles in the second domain are also hyperlinks to the paper. We call each such regularity-rich aspect a *property.*

Properties are not confined to citation labeling tasks. Infact, they exist and can be exploited in any domain whose records enjoy regularity. Consider the task of extracting products, where each catalog corresponds to a domain. In one catalog, each record might end a product price with USD (thus showing regularity), while in another, prices might be listed with EUR. Table 2 lists more properties for these two tasks. Table 1 illustrates the regularity of some of the properties on the citation labeling task.

The key idea behind using properties for domain adaptation is as follows. If we use our error-prone base model on each record in DEST independently, the noisy output labelings will generally not enjoy regularity simply because regularity in the domain is not captured at all, coupled with the errors made by the model. However, if we jointly label all records in DEST together, and provide an extra incentive for the output labelings to be regular with respect to the affected set of properties, many of the errors made earlier will be corrected. As an example, again consider the property *First Label* that returns the first label in a citation record. If under the base model, a majority but not all records in DEST exhibit regularity, then the extra incentive in our scheme will push the remaining records to conform with the other records and take on the correct first label.

Keeping this high-level picture in mind, we describe our approach with the following components:

1. A collection of instance-labeling pairs $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$. Instances and their labelings are usually structured, e.g. $\mathbf{x}_i$ can be a citation record and $\mathbf{y}_i$ its bibliographic segmentation. Each $\mathbf{y}_i$ is probabilistically modeled using a feature-based prediction model as discussed in Section 2. These models are also popularly known as Markov Random Fields (MRFs) in machine learning literature [9]. From Section 2 recall that the scoring function for assigning labeling $\mathbf{y}_i$ to $\mathbf{x}_i$ is $\mathbf{w} \cdot \mathbf{f}(\mathbf{x}_i, \mathbf{y}_i)$, which decomposes over the parts c of the MRF as $\mathbf{w} \cdot \mathbf{f}(\mathbf{x}_i, \mathbf{y}) = \sum_c \mathbf{w} \cdot \mathbf{f}_c(\mathbf{x}_i, \mathbf{y}_c)$. For sentence-like records, a part c is usually a word or a pair of adjacent words, as explained in the entity extraction task in Section 2.
2. A set P of properties where each property $p \in P$ includes in its domain a subset $\mathcal{D}p$ of MRFs and maps each labeling $\mathbf{y}$ of an input $\mathbf{x} \in \mathcal{D}p$ to a discrete value from its range $\mathcal{R}'p$. These properties predominantly take only one value for records in a fixed domain. The dominant value however can vary from domain to domain. For tractability, we assume that each property decomposes over parts of the MRF. We discuss decomposable properties in Section 4.1.
3. A clique potential function $\mathrm{C}p(\{p(\mathbf{x}_i, \mathbf{y}_i)\}_{\mathbf{x}_i \in \mathcal{D}_p})$ for each property p. We call it a clique potential because each property p is thought of as a clique, and each member MRF in $\mathcal{D}p$ a vertex of that clique. This potential is maximized when all the member MRFs get labelings with the same property value. By including these potentials in our objective, we can encourage conformity of properties across labelings of member MRFs. Various symmetric potential functions are discussed in Section 4.2.

Our aim is now to jointly label the N records so as to maximize the sum of the individual MRF specific scores from our trained model, and the clique potentials coupling these MRFs via their property functions. This is given by:

$$\max_{(\mathbf{y}_1, \ldots, \mathbf{y}_N)} \sum_{i=1}^{N} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}_i, \mathbf{y}_i) + \sum_{p \in P} \mathrm{C}p(\{p(\mathbf{x}_i, \mathbf{y}_i)\}_{i \in \mathcal{D}_p}) \tag{2}$$

Equation 2 is NP-hard to optimize even for simple cases. Hence, we look at an approximate optimization method instead. Our method uses the well-known paradigm of message passing in a cluster graph [5]. Message passing breaks the computation into two kinds of efficient subroutines: one that exploits the decomposability of the properties, and the other uses algorithms tailored to the specific symmetric potentials being used.

In the subsequent sections, we discuss decomposable properties, symmetric clique potentials, and the joint labeling computation algorithm.

4.1 Decomposable Properties

A property maps a $(\mathbf{x}, \mathbf{y})$ pair to a discrete value in its range. Table 2 gives examples of some properties for various record extraction tasks.

For tractability, we consider only *decomposable properties*, viz. properties that can be broken over the parts c of the MRF of labeling $\mathbf{y}$, just like our base model $\mathbf{w} \cdot \mathbf{f}$. We formally describe decomposable properties as:

DEFINITION 4.1. *A decomposable property $p(\mathbf{x}, \mathbf{y})$ is composed out of component level properties $p(\mathbf{x}, \mathbf{y}_c, c)$*

Domain	Record	p_1	p_2	p_3	p_4
1	Bhardwaj, P. (2001). Delegating Pricing Decisions. *Marketing Science* 20(2). 143-169.	Author	'.'	Venue	Volume
1	Balasubramaniam, S. and P. Bhardwaj (2004). When not all conflict is bad: Manufacturing marketing conflict and strategic incentive design. *Management Science* 50(4). 489-502.	Author	'.'	Venue	Volume
1	Bhardwaj, P. and S. Balasubramaniam (2005). Managing Channel Profits: The Role of Managerial Incentives. Forthcoming *Quantitative Marketing and Economics.*	Author	'.'	Venue	End
2	A Simulator for estimating Railway Line Capacity. *In APORS - 2003.*	Title	Start	Venue	Year
2	Scheduling Loosely Connected Task Graphs. *Journal of Computer and System Sciences , August 2003 .*	Title	Start	Venue	Month
2	Devanagari Pen-written Character Recognition. *In ADCOM - 2001 .*	Title	Start	Venue	Year

Table 1: Illustration of properties $p_1, \ldots, p_4$ applied to records from two bibliographic domains.

Id	Property $p(\mathbf{x},\mathbf{y})$	Range	Decomposable?
p_1	First non-Other label in $\mathbf{y}$	$Y \setminus \{\text{Other}\}$	Yes
p_2	Token before Title in $\mathbf{y}$	All seen tokens $\cup$ {Start,NoTitle}	Yes
p_3	First non-Other label after Title in $\mathbf{y}$	$Y \setminus \{\text{Other}\} \cup \{\text{End,NoTitle}\}$	Yes
p_4	First non-Other label after Venue in $\mathbf{y}$	$Y \setminus \{\text{Other}\} \cup \{\text{End,NoVenue}\}$	Yes
p_5	Does Title appear after Author in $\mathbf{y}$?	Boolean $\cup$ {NoAuthor,NoTitle}	No
p_6	Number of Titles in $\mathbf{y}$	$\mathbb{N} \cup \{0\}$	No
p_7	Label of fixed token t in $\mathbf{y}$	Y	Yes
p_8	HTML tag containing ProductName in $\mathbf{y}$	All HTML tags	Yes
p_9	Token after Price in $\mathbf{y}$?	USD,GBP,EUR,CAD etc.	Yes
p_{10}	Lowest common ancestor of ProductName and Price	Seen DOM XPaths	No

Table 2: Examples of properties for Bibliographic record extraction and Product extraction. Y is the set of all labels.

defined over parts c of $\mathbf{y}$. $p : (\mathbf{x}, \mathbf{y}_c, c) \mapsto \mathcal{R}_p \cup \{\emptyset\}$ where the special symbol $\emptyset$ means that the property is not applicable to $(\mathbf{x}, \mathbf{y}_c, c)$. $p(\mathbf{x}, \mathbf{y})$ is composed as:

$$p(\mathbf{x},\mathbf{y}) \triangleq \begin{cases} \emptyset & \textit{if } \forall c : p(\mathbf{x},\mathbf{y}_c,c) = \emptyset \\ v & \textit{if } \forall c : p(\mathbf{x},\mathbf{y}_c,c) \in \{v,\emptyset\} \\ \perp & \textit{otherwise.} \end{cases} \quad (3)$$

The first case occurs when the property does not fire over any of the parts of $\mathbf{y}$. The last case occurs when $\mathbf{y}$ has more than one parts where the property has a valid value but the values are different. The new range $\mathcal{R}'p$ now consists of $\mathcal{R}_p$ and the two special symbols $\perp$ and $\emptyset$.

We show that even with decomposable properties we can express many useful types of regularities in labeling multiple MRFs arising in domain adaptation.

Example 1. Consider a property that expresses regularity in the order of labels in a collection of bibliography records. Let $\mathbf{x}$ be a bibliographic record and $\mathbf{y}$ its labeling. Define property p_3, which returns the first non-Other label in $\mathbf{y}$ after a Title. A label 'End' marks the end of $\mathbf{y}$. So $\mathcal{R}_p$ contains 'End' and all labels except Other. So when p_3 is applied to a part c labeled Title, it returns the first non-Other label in $\mathbf{y}$ after c. Thus,

$$p_3(\mathbf{x}, y_c, c) \triangleq \begin{cases} \beta & (y_c = \text{Title}) \text{ and } (y_{c+i} = \beta) \text{ and} \\ & (y_{c+j} = \text{Other},\ j = 1, \ldots, i-1) \\ \text{End} & (y_c = \text{Title}) \text{ and } (\mathbf{y} \text{ ends at } c) \\ \emptyset & y_c \neq \text{Title} \end{cases}$$

Therefore,

$$p_3(\mathbf{x},\mathbf{y}) \triangleq \begin{cases} \emptyset & \mathbf{y} \text{ has no Title} \\ \beta & \beta \text{ is the first non-Other label after } \textbf{each} \\ & \text{Title in } \mathbf{y} \\ \perp & \text{otherwise} \end{cases}$$

Example 2. Consider a property, denoted by p_2, whose range is the space of tokens. This property returns the identity of the token before a Title in $\mathbf{y}$. So,

$$p_2(\mathbf{x}, y_c, c) \triangleq \begin{cases} x_{c-1} & y_c = \text{Title and } (c > 0) \\ \text{'Start'} & y_c = \text{Title and } (c = 0) \\ \emptyset & y_c \neq \text{Title} \end{cases}$$

Therefore,

$$p_2(\mathbf{x},\mathbf{y}) \triangleq \begin{cases} \emptyset & \text{No Title in } \mathbf{y} \\ \text{'Start'} & \text{The only Title in } \mathbf{y} \text{ is at the start} \\ t & \text{All Titles in } \mathbf{y} \text{ preceded by token } t \\ \perp & \mathbf{y} \text{ has multiple Titles with different} \\ & \text{preceding tokens} \end{cases}$$

4.2 Symmetric Clique Potentials

A symmetric clique potential depends only on the number of clique vertices taking a property value v, denoted by n_v, and not on the identity of those vertices. In other words, such a potential is invariant under any permutation of its arguments and the potential's value

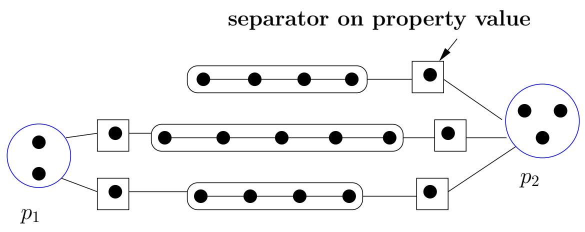

Figure 1: Cluster graph for a toy example with three chain-shaped MRF instances and two properties.

is derived from the histogram of counts $\{n_v | \forall v \in V\}$, where V is the set of possible property values. We denote this histogram by the vector $\mathbf{n}$. An associative potential is maximized when $n_v = n$ for some v, i.e. one value is given to all the clique vertices.

Three popular associative clique potential families are listed in Table 3. We specifically discuss a very prominent symmetric clique potential — Potts potential, which we also use in our experiments.

Potts Potential

The Potts potential corresponds to the negative Gini index of the property values at the clique vertices:

$$\mathrm{C}^{\text{Potts}} = \mathrm{C}(n_1, \ldots, n_{|V|}) = \frac{\lambda}{n} \sum_{v \in V} n_v^2 \qquad (4)$$

where n is the number of vertices. Potts potential counts (upto a constant) the number of clique edges, both of whose end vertices have the same property value. Potts potential have been extensively used in statistical physics in the garb of Ising models, in image pixel labeling tasks e.g. [4], in associative Markov networks [15] to model associative labeling tasks, and in skip-chain MRFs [14] to model non-local associative dependencies between repeated occurrences of a word.

4.3 Joint Record Labeling

The individual MRFs coupled with symmetric clique potentials form a natural cluster graphical model, such as the toy model shown in Figure 1. To compute the labelings of all MRFs jointly, we perform message passing on this cluster graph.

Message passing on the cluster graph can be seen as an optimization-by-parts heuristic to solve Equation 2. In a single iteration of this heuristic, each cluster computes its own belief of what its own labeling should be and passes this message to the neighboring clusters, which in turn compute their beliefs about the labels of their member nodes. These computations are interdependent because the clusters have overlapping members.

In our setup, this leads to computing messages from MRFs to property cliques and vice versa. Complete message computation details are provided in [7]. We only provide the outline here. First, to compute a message from an MRF to a property clique, we look at the message computation algorithm *inside* the MRF and fold-in the property. Folding is possible because the properties are decomposable in the same way as the MRF model. Second, to compute the reverse message, we invoke highly efficient potential-specific combinatorial algorithms at the clique. Some such algorithms are presented in [6] and [7] for a variety of potentials.

After sufficient rounds of interaction via message passing, each MRF reports its best labeling independently.

5. EXPERIMENTAL RESULTS

We demonstrate the domain adaptability of our approach and show that using a good set of properties can bring down the test error significantly.

We focus on the bibliographic information extraction task, where the aim is to adapt a base model across widely varying publications pages of authors. Our dataset consists of 433 bibliographic entries from the webpages of 31 authors, hand-labeled with 14 labels such as Title, Author, Venue, Location and Year. Bibliographic entries across different authors differ in various aspects like label-ordering, missing labels, punctuation, HTML formatting and bibliographic style.

Varying fractions of 31 domains were used to train a base model. We used standard extraction features in a window around each token, along with label transition features [10].

For our collective labeling framework, we use properties $p_1, \ldots, p_4$ from Table 2. We use Potts potential for each property, with $\lambda = 1$. Some of these properties, e.g. p_3, operate on non-adjacent labels, and thus are not Markovian. This can be easily rectified by making 'Other' an extension of its predecessor label, e.g. an 'Other' segment after 'Title' can be relabeled as 'After-Title'.

The performance results of the collective model with the above properties versus the baseline model are presented in Table 4. For the test domains, we report token F1 accuracy of the important labels — Title, Author and Venue. F1 accuracy of a label l is the harmonic mean of the precision and recall of l, defined as:

$$\text{Prec}(l) = \frac{\text{\# tokens correctly marked } l \text{ by the model}}{\text{\# tokens marked } l \text{ by the model}}$$

$$\text{Recall}(l) = \frac{\text{\# tokens correctly marked } l \text{ by the model}}{\text{\# tokens with true label } l}$$

F1 accuracies reported in Table 4 are averaged over five trials. The collective model leads to up to 25% reduction in the test error for Venue and Title, labels for which we had defined related properties. Figure 2 shows the error reduction on individual test domains for one particular split when five domains were used for training and 26 for testing. The errors are computed from the combined token F1 scores of Title, Venue and Author. For some domains the errors are reduced by more than 50%. Collective inference increases errors in only two domains. In those domains, a majority of the labelings output by the base model take on wrong property values. Thus by encouraging conformity, the

Name	Form	Remarks
MAX	$\max_v f_v(n_v)$	f_v is a non-decreasing function
SUM	$\sum_v f_v(n_v)$	f_v non-decreasing. Includes the Potts potential $= \frac{\lambda}{n} \sum_v n_v^2$
MAJORITY	$f_a(\mathbf{n})$, where $a = \text{argmax}_v n_v$	f_a is typically linear

Table 3: Various families of symmetric clique potentials. $\mathbf{n} = (n_1, \ldots, n_{|V|})$ is the histogram of property values in the clique.

	Title		Venue		Author	
Train %	Base	CI	Base	CI	Base	CI
5	70.7	74.8	58.8	62.5	74.1	74.3
10	78.0	82.1	69.2	72.2	75.6	75.9
20	85.8	88.6	76.7	78.9	80.7	80.7
30	91.7	93.0	81.5	82.6	87.7	88.0
50	92.3	94.2	83.5	84.5	89.4	90.0

Table 4: Token-F1 of the Collective and Base Models on the Bibliographic Extraction Task

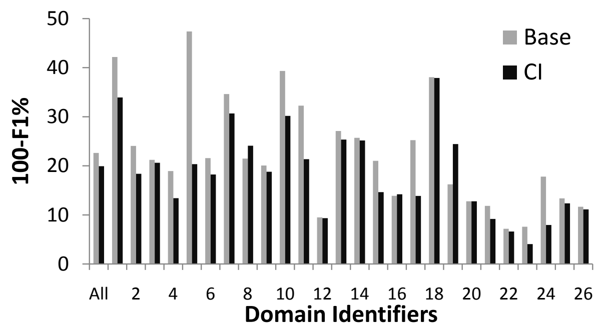

Figure 2: Per-domain error for the base and collective inference (CI) model

remaining labelings also take on the wrong value causing a slight dip in accuracy.

6. CONCLUSION

We presented a mini-survey of domain adaptation approaches for information extraction. We summarized our proposed approach that jointly labels records in a target domain without retraining any model. Our framework encourages records to jointly take labelings that are conformant with respect to a set of domain-independent properties. We also presented the joint-labeling algorithm for the new graphical model that results from our setup. Finally, we demonstrated our framework on a bibliographic task, where we showed significant gains by exploiting intra-domain regularity.

7. REFERENCES

[1] S. Bickel, M. Brückner, and T. Scheffer. Discriminative learning for differing training and test distributions. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, 2007.

[2] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, 2007.

[3] J. Blitzer, R. McDonald, and F. Pereira. Domain Adaptation with Structural Correspondence Learning. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP)*, 2006.

[4] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, 2001.

[5] J. Duchi, D. Tarlow, G. Elidan, and D. Koller. Using combinatorial optimization within max-product belief propagation. In *Advances in Neural Information Processing Systems (NIPS 2006)*, 2007.

[6] R. Gupta, A. A. Diwan, and S. Sarawagi. Efficient inference with cardinality-based clique potentials. In *Proceedings of the 24th International Conference on Machine Learning (ICML), USA*, 2007.

[7] R. Gupta and S. Sarawagi. A generalized framework for collective inference with applications in domain adaptation, Under Preparation.

[8] J. Huang, A. Smola, A. Gretton, K. Borgwardt, and B. Schölkopf. Correcting Sample Selection Bias by Unlabeled Data. In *Advances in Neural Information Processing Systems 20*, Cambridge, MA, 2007. MIT Press.

[9] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning (ICML-2001)*, Williams, MA, 2001.

[10] F. Peng and A. McCallum. Accurate information extraction from research papers using conditional random fields. In *HLT-NAACL*, pages 329–336, 2004.

[11] S. Sarawagi. Information extraction. *FnT Databases*, 1(3), 2008.

[12] S. Satpal and S. Sarawagi. Domain adaptation of conditional probability models via feature subsetting. In *ECML/PKDD*, 2007.

[13] H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, pages 227–244, 2000.

[14] C. Sutton and A. McCallum. Collective segmentation and labeling of distant entities in information extraction. Technical Report TR # 04-49, University of Massachusetts, July 2004. Presented at ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields.

[15] B. Taskar, V. Chatalbashev, and D. Koller. Learning associative markov networks. In *Twenty First International Conference on Machine Learning (ICML04), Banff, Canada.*, 2004.