A Comparative Study of Methods for Transductive Transfer Learning

Andrew Arnold, Ramesh Nallapati and William W. Cohen
Machine Learning Department, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, USA
{aarnold, nmramesh, wcohen}@cs.cmu.edu

Abstract

The problem of transfer learning, where information gained in one learning task is used to improve performance in another related task, is an important new area of research. In this paper we address the subproblem of domain adaptation, in which a model trained over a source domain is generalized to perform well on a related target domain, where these two domains' data are distributed similarly, but not identically.

Previous work has studied the supervised version of this problem in which labeled data from both source and target domains are available for training. In this work, however, we study the more challenging problem of unsupervised transductive transfer learning, where no labeled data from the target domain are available at training time, but instead, unlabeled target test data are available during training.

We describe some current state-of-the-art inductive and transductive approaches involving three popular learning models, namely the maximum entropy, support vector machines and naive Bayes models. We then adapt these models to the problem of transfer learning for protein name extraction. In the process, we introduce a novel maximum entropy based technique, Iterative Feature Transformation (IFT), and show that it achieves comparable performance with state-of-the-art transductive SVMs.

Finally, we compare the relative strengths and weaknesses of these models across the various learning settings, shedding light both on the algorithms examined and the difficulty of the respective problems. In addition, we show how simple relaxations, such as providing additional information like the proportion of positive examples in the test data, can significantly improve the performance of some of the transductive transfer learners.

1 Introduction

Consider the task of *named entity extraction* (NER). Specifically, you are given a corpus of encyclopedia articles in which all the personal name mentions have been labeled. The standard supervised machine learning problem is to learn a classifier over this training data that will successfully label unseen test data drawn from the same distribution as the training data, where "same distribution" could mean anything from having the train and test articles written by the same author to having them written in the same language. Having successfully trained a named entity classifier on this encyclopedia data, now consider the problem of learning to classify tokens as names in instant messenger data. Clearly the problems of identifying names in encyclopedia articles and instant messages are closely related, and learning to do well on one should help your performance on the other. At the same time, however, there are serious differences between the two problems that need to be addressed. For instance, capitalization, which will certainly be a useful feature in the encyclopedia problem, may prove less informative in the instant messenger data since the rules of capitalization are followed less strictly in that domain. Thus there seems to be some need for altering the classifier learned on the first problem (called the *source domain*) to fit the specifics of the second problem (called the *target domain*). This is the problem of *domain adaptation* and is considered a type of *transfer learning*.

The intuitive solution seems to be to simply train on the target domain data. Since this training data would be drawn from the same distribution as the data you will ultimately test over, this approach avoids the transfer issue entirely. The problem with this idea is that often large amounts of labeled data are not available in the target domain. While it has been shown that even small amounts of labeled target data can greatly improve transfer results [6, 8], there has been relatively little work, however, on the case when there is no labeled target data available, that is, totally unsupervised domain adaptation. In this scenario, one way to adapt a model trained on the source domain is to make the unlabeled *target test data* available to the model during training time. Leveraging (unlabeled) test data during training time is called *transductive learning* and is a well studied problem in the scenario when the training data and test data come from the same domain. However, transduction is not well-studied in a transfer setting, where the training and test data come from different domains. Studying transfer learning in a transductive setting will be the main focus of our work.

The rest of the paper is organized as follows. In section 2, we compare and contrast transductive transfer learning with the more traditional learning paradigms while summarizing relevant work done in the past. Section 3 describes the two discriminative models and one generative model we considered in detail in this work. In this section, we describe the adaptations we used to make these models applicable to the transductive transfer learning setting. One of these adaptations, called IFT, is one of the original contributions of this work. We present our experiments and results in detail in section 4. Lastly, the paper is concluded in sections 5 and 6 where we comment on the results and discuss directions for future work.

2 An overview of learning paradigms and related work

Given an example x and a class label y, the standard statistical classification task is to assign a probability, $p(y|x)$, to x of belonging to class y. In the binary classification case the labels are $Y \in \{0, 1\}$. In the case we examine, each example x_i is represented as a vector of binary features $(f_1(x_i), \cdots, f_F(x_i))$ where F is the number of features. The data consists of two disjoint subsets: the training set $(X_{train}, Y_{train}) = \{(x_1, y_1) \cdots, (x_N, y_N)\}$, available to the model for its training and the test set $X_{test} = (x_1, \cdots, x_M)$, upon which we want to use our trained classifier to make predictions.

We discuss below different paradigms of learning associated with the classification problem. They are also summarized in table 1 for the reader's convenience.

In the paradigm of *inductive learning*, (X_{train}, Y_{train}) are known, while both X_{test} and Y_{test} are completely hidden during training time. In the case of *semi-supervised* inductive learning [27, 22, 11], the learner is also provided with auxiliary unlabeled data $X_{auxiliary}$, that is not part of the test set. It has been noted that such auxiliary data typically helps boost the performance of the classifier significantly.

Another setting that is closely related to semi-supervised learning is *transductive learning* [25, 14, 16], in which X_{test} (but, importantly, not Y_{test}), is known at training time. That is, the learning algorithm knows exactly which examples it will be evaluated on after training. This can be a great asset to the algorithm, allowing it to shape its decision function to match and exploit the properties seen in X_{test}. One can think of transductive learning as a special case of semi-supervised learning in which $X_{auxiliary} = X_{test}$.

In the three cases discussed above, X_{test} and X_{train} are both assumed to have been drawn from the same distribution, $\mathcal{D}$. In the setting of *transfer learning*, however, we would like to apply our trained classifier to examples drawn from a distribution different from the one upon which it was trained. We therefore assume there are two different distributions, $\mathcal{D}^{source}$ and $\mathcal{D}^{target}$, from which data may be drawn. Given this notation we can then precisely state the transfer learning problem as trying to assign labels Y^{target}_{test} to test data X^{target}_{test} drawn from $\mathcal{D}^{target}$, given training data $(X^{source}_{train}, Y^{source}_{train})$ drawn from $\mathcal{D}^{source}$. In this paper we focus on the subproblem of *domain adaptation*, where we assume Y (the set of possible labels) is the same for both D^{source} and D^{target}, while D^{source} and D^{target} themselves are allowed to vary between domains. This is in contrast to the related subproblem of *multi-task learning* [1, 23] in which the marginal distribution of the data is assumed not to change, while the task (and therefore the labels) is allowed to vary from source to target.

One of the first formulations of the transfer learning problem was presented over 10 years ago by Thrun [24]. More recently there has been a focus on using source data to learn various types of priors for the target data [20]. Other techniques have tried to quantify the generalizability of certain features across domains [9, 13], or tried to exploit the common structure of related problems [2, 4].

Although the case of transfer learning without access to any data drawn from $\mathcal{D}^{target}$ is not completely hopeless [13], in this paper we choose to focus on extensions to the transfer learning setting that allow us to capture some information about D^{target}. One obvious such setting is *inductive transfer learning* where we also provide a few auxiliary labeled data $(X^{target}_{auxiliary}, Y^{target}_{auxiliary})$ from the target domain in addition to the labeled data from the source domain. Due to the presence of labeled target data, this method could also be called *supervised transfer* learning and is the most common setting used by researchers in transfer learning today.

In this work, however, we focus on a new and more challenging paradigm, namely, *transductive transfer learning*, where there is no auxiliary labeled data in the target domain available for training, but where the unlabeled test set on the target domain X^{target}_{test} can be seen during training. Again, due to the lack of labeled target data, this setting could be considered *unsupervised transfer* learning. It is important to point out that *transductive learning* is orthogonal to *transfer learning*. That is, one can have a transductive algorithm that does or does not make the transfer learning assumption, and vice versa. Much of the work in this paper is inspired by the belief that, although distinct, these problems are nevertheless intimately related. More specifically, when trying to solve a transfer problem between two domains, it seems intuitive that looking at the *unlabeled* test data of the target domain during training will improve performance over

Table 1. Learning settings are summarized by the type of auxiliary and test data used. For all settings we assume $(X^{source}_{train}, Y^{source}_{train})$ is available at training time, while Y_{test} is unknown. Settings for which we have run experiments (see table 4) are marked in bold, along with their short name.

Natural name for learning setting	Experiment name	Auxiliary data		Test data	
		Domain	Labels	Domain	X_{test}
Inductive learning	Induct	-	-	$\mathcal{D}^{source}$	unseen
Semi-supervised inductive learning		$\mathcal{D}^{source}$	unseen	$\mathcal{D}^{source}$	unseen
Transductive learning		-	-	$\mathcal{D}^{source}$	seen
Transfer learning		-	-	$\mathcal{D}^{target}$	unseen
Inductive transfer learning	InductTransfer	$\mathcal{D}^{target}$	seen	$\mathcal{D}^{target}$	unseen
Semi-supervised inductive transfer learning		$\mathcal{D}^{source}$	unseen	$\mathcal{D}^{target}$	unseen
Transductive transfer learning	TransductTransfer	-	-	$\mathcal{D}^{target}$	seen
Supervised Transductive transfer learning		$\mathcal{D}^{target}$	seen	$\mathcal{D}^{target}$	seen
Relaxed Transductive transfer learning[1]	RelaxedTransductTransfer	-	-	$\mathcal{D}^{target}$	seen
Semi-supervised transductive transfer learning		$\mathcal{D}^{source}$	unseen	$\mathcal{D}^{target}$	seen

[1] A relaxation of transductive transfer learning in which proportions of labels in the target data is known at training time.

ignoring this source of information.

We note that the setting of *inductive transfer learning*, in which labeled data from both source and target domains are available for training, serves as a rough upper-bound to the performance of a learner based on *transductive transfer learning*, in which no labeled target data is available. Hence, although our primary interest is the transductive transfer setting, we also used the former setting in all our experiments for purposes of illustration.

For similar reasons, we considered an additional artificial setting, which we call *relaxed transductive transfer learning*, in our experiments. This setting is almost equivalent to the transductive transfer setting, but the model is allowed to know the proportion of positive examples in the target domain. Although this technically violates the terms of unsupervision in transductive transfer learning, in practice estimating this single parameter over the target domain does not require nearly as much labeled target data as learning all the parameters of a fully supervised transfer model, and thus serves as a nice compromise between the two extremes of transduction and supervision.

These and a few other interesting settings are summarized in table 1. Note that we only displayed a small subset of the many possible learning settings.

3 Methods considered

In this section, we present the two discriminative models considered in this work, maximum entropy and support vector machines, and one generative model, the naive Bayes classifier. For each model, we first summarize their learning and inference algorithms in the classical inductive learning setting and then describe the adaptations we made for inductive transfer, transductive transfer and relaxed transductive transfer learning settings.

3.1 Maximum entropy models

3.1.1 Inductive learning: simple MaxEnt

Entropy maximization (MaxEnt) [3, 18] is a way of modeling the conditional distribution of labels given examples. Given a set of training examples $X_{train} \equiv \{x_1, \ldots, x_N\}$, their labels $Y_{train} \equiv \{y_1, \ldots, y_N\}$, and the set of features $\mathcal{F} \equiv \{f_1, \ldots, f_F\}$, MaxEnt learns a model consisting of a set of weights corresponding to each class $\Lambda = \{\lambda_{1,y} \ldots \lambda_{F,y}\}_{y \in \{0,1\}}$ over the features so as to maximize the conditional likelihood of the training data, p($Y_{train}|X_{train}$), given the model p_Λ. In exponential parametric form, this conditional likelihood can be expressed as:

$$p_\Lambda(y_i = y|x_i) = \frac{1}{Z(x_i)} \exp(\sum_{j=1}^{F} f_j(x_i)\lambda_{j,y}) \qquad (1)$$

where Z is the normalization term:

$$Z(x_i) = \sum_{y \in \{0,1\}} \exp(\sum_{j=1}^{F} f_j(x_i)\lambda_{j,y}) \qquad (2)$$

In order to avoid overfitting the training data, these λ's are often further constrained to be near 0 by the use of a regularization term which tries to minimize $\|\Lambda\|_2^2 \equiv \sum_{j,y} (\lambda_{j,y})^2$. Thus the entire expression being optimized is:

$$\underset{\Lambda}{\text{argmax}} \quad \sum_{i=1}^{N} \log \; p_\Lambda(y_i|x_i) - \beta\|\Lambda\|_2^2 \qquad (3)$$

where $\beta > 0$ is a parameter controlling the amount of regularization. Maximizing this likelihood is equivalent to constraining the joint expectations of each feature and label in the learned model, $E_\Lambda[f_j, y]$, to match empirical expectations $E_{train}[f_j, y]$ as shown below:

$$E_{train}\left[f_j, y\right] = \frac{1}{N}\sum_{i}^{N} f_j(x_i)\delta_y(y_i) \qquad (4)$$

$$E_{\Lambda}\left[f_j, y\right] = \frac{1}{N}\sum_{i}^{N} f_j(x_i)P_\Lambda(y|x_i) \qquad (5)$$

where $\delta_y(y_i) = 1$ if $y = y_i$ and 0 otherwise.

In the next few subsections, we will describe how we adapt the model to various scenarios of transfer learning.

3.1.2 Inductive transfer learning

Source trained prior models: One recently proposed method [6] for transfer learning in MaxEnt models involves modifying Λ's regularization term. First a model of the source domain, Λ^{source}, is learned by training on $\{X^{source}_{train}, Y^{source}_{train}\}$. Then a model of the target domain is trained over a limited set of labeled target data $\{X^{target}_{train}, Y^{target}_{train}\}$, but instead of regularizing this Λ^{target} to be near zero by minimizing $\|\Lambda^{target}\|^2_2$, Λ^{target} is instead regularized towards the previously learned source values Λ^{source} by minimizing $\|\Lambda^{target} - \Lambda^{source}\|^2_2$. Thus the modified optimization problem is:

$$\underset{\Lambda^{target}}{\operatorname{argmax}} \sum_{i=1}^{N^{target}_{train}} \log\, p_{\Lambda^{target}}(y_i|x_i) - \beta\|\Lambda^{target} - \Lambda^{source}\|^2_2 \qquad (6)$$

where N^{target}_{train} is the number of labeled training examples in the target domain. It should be noted that this model requires Y^{target}_{train} in order to learn Λ^{target} and is therefore a supervised form of *inductive transfer*.

Feature space expansion: Another approach to the problem of inductive transfer learning is explored by Daumé [8, 9]. Here the idea is that there are certain features that are common between different domains, and others that are particular to one or the other. More specifically, we can redefine our feature set $\mathcal{F}$ as being composed of two distinct subsets $\mathcal{F}^{specific} \bigcup \mathcal{F}^{general}$, where the conditional distribution of the features in $\mathcal{F}^{specific}$ differ between X^{source} and X^{target}, while the features in $\mathcal{F}^{general}$ are identically distributed in the source and target. Given this assumption, there is an EM-like algorithm [9] for estimating the parameters of these distributions. There is also a simpler approach [8] of just making a duplicate copy of each feature in X^{source} and X^{target}, so whereas before you had $x_i = \langle f_1(x_i) ... f_F(x_i)\rangle$, you now have

$$x_i = \langle\ f_1(x_i)^{specific}, f_1(x_i)^{general} \\ ... f_F(x_i)^{specific}, f_F(x_i)^{general}\ \rangle \qquad (7)$$

where $specific$ is $source$ or $target$ respectively, and $f_j(x_i)^{specific}$ is just a copy of $f_j(x_i)^{general}$. The idea is that by expanding the feature space in this way MaxEnt will be able to assign different weights to different versions of the same feature. If a feature is common in both domains its $general$ copy will get most of the weight, while its specific copies (f^{source} and f^{target}) will get less weight, and vice versa.

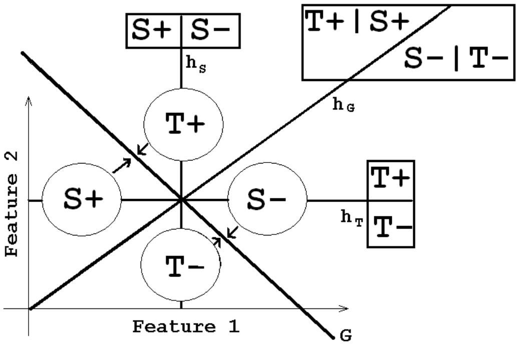

Figure 1. Illustration of feature space transformation in transfer learning problem. h_S and h_T easily separate the source and target data respectively. But a projection onto G is required before h_G can successfully separate both distributions at once.

3.1.3 Transductive transfer learning: Iterative Feature Transformation (IFT)

In this subsection, we present a new approach for the unsupervised setting of transductive transfer learning using MaxEnt. For ease of notation we will use $E^{source}\left[f_j, y\right]$ to mean $E_{x\in\mathcal{D}^{source}}\left[f_j(x), y\right]$, and similarly for $target$.

One problem with transfer in MaxEnt is that the joint distribution of the features with labels differs between the source and target domains. In other words, $E^{source}\left[f_j, y\right]$ does not necessarily equal $E^{target}\left[f_j, y\right]$. If the expectations in the train and test datasets are similar, then the Λ learned on the training data will generalize well to the test data. The more these distributions differ, however, the less well the trained model will perform. Figure 1 illustrates this phenomenon. In this example, there are two features comprising the feature space. The distribution of the positive (+) and negative (-) classes of the source (S) and target (T) distributions are plotted with respect to these features. The supervised, non-transfer problems are simple in this setting since the source and target data are each easily separable in this feature space, by h_S and h_T respectively. For transfer learning, however, if we train on the source, we might learn the classifier h_S, which depends only on *feature 1*. If we then attempt to classify the target data we will fail, since *feature 1* is a poor discriminator of the target data. What we would like to do is transform the feature space so that the distribution of the positive and negative classes in that transformed feature space is the same for both domains. This transformation is represented by G in the figure, a line upon which the data have been projected. Given this new transformation, h_G can easily be learned over the source data and subsequently performs equally well when transfered to the target data. Phrased in terms of maximum entropy, we are trying to learn a transformation $G()$ of the feature space $\mathcal{F}$ such that the joint distributions of the source and target features with their labels are aligned:

$$E^{target}\left[G(f_j), y\right] = E^{source}\left[G(f_j), y\right], \forall f_j \in \mathcal{F} \quad (8)$$

One could relax this condition even further by arguing that it is enough to transform only one of the domains, say the source data, so that data from both domains could be separated by a single hyperplane. In the figure, if we project only the source data onto G, but leave the target data untouched, the hyperplane h_G would still be able to classify the target data accurately. In maximum entropy phraseology, the relaxed transformation can be expressed as:

$$E^{target}\left[f_j, y\right] = E^{source}\left[G(f_j), y\right], \forall f_j \in \mathcal{F} \quad (9)$$

The problem with this, of course, is that in the unsupervised transductive transfer case, we do not have Y^{target} and therefore cannot estimate $E^{target}\left[f_j, y\right]$. Hence we approximate $E^{target}\left[f_j, y\right]$ using the joint estimates on the target unlabeled data from a model learned from the source data as shown below.

$$\begin{aligned} E^{target}\left[f_j, y\right] &\approx E^{target}_{\Lambda_{source}}\left[f_j, y\right] \\ &= \frac{1}{N^{target}_{test}} \sum_{i=1}^{N^{target}_{test}} f_j(x_i) P_{\Lambda_{source}}(y, x_i) \end{aligned}$$

where N^{target}_{test} is the number of target domain (unlabeled) test examples. These estimates may not reflect the true target expectations, but it is the best we could do in the unsupervised transductive setting. Now we use these expectations to define the source domain transformation G as follows:

$$\forall_{i=1}^{N^{source}_{train}} \quad G(f_j(x_i)) = f_j \frac{E^{target}_{\Lambda_{source}}\left[f_j, y_i\right]}{E^{source}[f_j, y_i]} \quad (10)$$

where $E^{source}[f_j, y_i]$ is given by the formula in (4) and N^{source}_{train} is the number of labeled training data in the source domain. It is easy to show that the empirical feature-label joint expectations of the transformed source data given by $E^{source}[G(f_j, y)]$ defined this way is equal to $E^{target}_{\Lambda_{source}}[f_j, y]$, the model expectations of the original features in the target domain, satisfying the condition in (9). The effect is to rescale $f_j(x)$, putting more weight on features that occur frequently in the target but rarely in the source (in a conditional sense), and downweighting features that are common in the source but seldom seen in the target. This algorithm can be implemented in an iterative fashion by first training the source model, computing the target expectations using the source model, transforming the source features and then retraining the source model.

In practice, since the target expectation $E^{target}_{\Lambda_{source}}[f_j, y]$ is only approximate, we smooth the transformed features with the original ones in each iteration as follows:

$$G'(f_j(x_i)) = \theta f_j(x_i) + (1 - \theta) G(f_j(x_i)) \quad (11)$$

where the free parameter θ controls the degree to which we use the target conditional estimates to alter the source conditionals.

3.1.4 Relaxed transductive transfer learning: biased thresholding

A natural way to exploit the known value of the proportion of positive class labels in the target domain is to adjust the decision threshold of the MaxEnt classifier so that the percentage of unlabeled target examples predicted as positive by the source-trained classifier is equal to the known value. We call this intuitive algorithm *biased thresholding*, to reflect the fact that the decision threshold is biased towards the known information on class ratio.

3.2 Support vector machines

3.2.1 Inductive learning: inductive SVMs

Support vector machines (SVM's) [15] take a different approach to the binary classification problem. Instead of explicitly modeling the conditional distribution of the data and using these estimates to predict labels, SVMs try to model the data geometrically. Each example is represented as an F-dimensional real-valued vector of features and is then projected as a point in F-dimensional space.

The *inductive SVM* exploits the label information of the training data and fits a discriminative hyperplane between the positively and negatively labeled training examples in this space, so as to best separate the two classes. This separation is called the margin, and thus SVMs belong to the margin based approach to classification. This formulation has proven very successful as inductive SVMs currently have some of the best general performance of any popular machine learning algorithm.

3.2.2 Inductive transfer learning: inductive SVMs with concatenated data

Recall that in the supervised inductive transfer case, we are given the training sets $(X_{train}^{source}, Y_{train}^{source})$ and $(X_{train}^{target}, Y_{train}^{target})$. Since the SVM does not explicitly model the data distribution, we simply concatenate the source and target labeled data together and provide the entire data for training. The hope is that it will improve on an SVM trained purely on labeled source data, by re-adjusting its hyperplane based on the labeled target data. It is possible to do better than such a naive approach [1], but we used this as a reasonable baseline.

3.2.3 Transductive transfer learning: transductive SVMs

Transduction with SVMs, in contrast to probabilistic models, is quite intuitive. Whereas, in the supervised case, we tried to fit a hyperplane to best separate the labeled training data, in the transductive case, we add in unlabeled testing data which we must also separate. Since we do not know the labels of the testing data, however, we cannot perform a straight forward margin maximization, as in the supervised case. Instead, one can use an iterative algorithm [14] similar in flavor to the MaxEnt iterative feature transformation (IFT) algorithm of section 3.1.3. Specifically, a hyperplane is trained on the labeled source data and then used to classify the unlabeled testing data. As in IFT, one can adjust how confident the hyperplane must be in its prediction in order to use a pseudo-label during the next phase of training (since there are no probabilities, large margin values are used as a measure of confidence). The pseudo-labeled testing data is then, in turn, incorporated in the next round of training. The idea is to iteratively adjust the hyperplane (by switching presumed pseudo-labels) until it is very confident on most of the testing points, while still performing well on the labeled training points.

Transductive SVMs were originally designed for the case where the training and test sets were drawn from the same domain. Again, since SVMs do not model the data distribution, it is not immediately obvious how one would model different distributions in the SVM algorithm. Hence in this work, we directly test the applicability of transductive SVMs to the transductive transfer setting.

3.2.4 Relaxed transductive transfer learning: biased thresholding

As with the maximum entropy approaches described in section 3.1.4, transductive SVMs used for transfer can also be adjusted to match the prior proportion of positive examples in the target domain.

Specifically, whereas the SVM usually just considers which side of the hyperplane a test example is on in determining its label (i.e., a threshold of 0), this threshold can be moved so that some points that lie nearest on the negative side of the hyperplane and would normally be given a negative label, would instead receive a positive one, or vice verse. This is very similar to the biased thresholding technique used in MaxEnt, hence we retain the same name.

3.3 Naive Bayes classifier

3.3.1 Inductive learning: maximum likelihood estimation

Naive Bayes [17] is one of the most popular and effective generative classifiers for many text-classification tasks. Like any generative model, its decision rule is given by the posterior probability of the class y given the example x, given by $P(y|x)$, which is computed using Bayes' rule as follows:

$$P(y|x) = \frac{P(x|\theta(y))\pi(y)}{\sum_{y'} P(x|\theta(y'))\pi(y')} \qquad (12)$$

where $\theta(y)$ are the class-conditional parameters and $\pi(y)$ are the prior probabilities. The naive Bayes model makes the somewhat unrealistic yet practical assumption of conditional-independence between the features of each example, given its class. That is:

$$P(x|\theta(y)) = \prod_{j=1}^{F} P(f_j(x)|\theta_j(y)) \qquad (13)$$

In our case, since the features are all binary, we use the Bernoulli distribution to model each feature as follows:

$$P(x|\theta(y)) = \prod_{j=1}^{F} (\theta_j(y))^{f_j(x)}(1-\theta_j(y))^{1-f_j(x)} \qquad (14)$$

where $\theta_j(y)$ can be interpreted as the probability that the feature f_j assumes a value 1 given the class y. The Bernoulli parameters $\theta_j(y)$ and $\pi(y)$ are estimated using Maximum Likelihood training with the labeled training data $(X_{train}, Y_{train}) = \{(x_1, y_1), \cdots, (x_N, y_N)\}$ as below:

$$\begin{aligned} \theta_j(y) &= \frac{\sum_{i=1}^{N} f_j(x_i)\delta_y(y_i) + \lambda}{\sum_{i=1}^{N} \delta_y(y_i) + 2\lambda} \\ \pi(y) &= \frac{\sum_{i=1}^{N} \delta_y(y_i)}{N} \end{aligned} \qquad (15)$$

where $\delta_y(y_i) = 1$ if $y = y_i$ and 0 otherwise; and λ is the Laplace smoothing parameter, which we set to 0.05 in our experiments.

[1] For example, one could impose a higher penalty for classification errors on the target data than on the source data.

3.3.2 Inductive transfer learning: maximum likelihood estimation with concatenated data

In the *inductive transfer* case, similar to the SVMs, we concatenate the entire labeled data $(X^{source}_{train}, Y^{source}_{train})$ and $(X^{target}_{train}, Y^{target}_{train})$ to generate a single training set. Then, we learn the parameters $\theta_j(y)$ and $\pi(y)$ using the maximum likelihood estimators shown in the classic supervised case (see eqn. 15). Although more sophisticated approaches are possible, we tried this algorithm as a simple baseline.

3.3.3 Transductive transfer learning: source-initialized EM

In the transductive transfer case, $(X^{target}_{train}, Y^{target}_{train})$ are not available for training, but X^{target}_{test} is available at training time. Learning from unlabeled examples in the generative framework is done typically using the standard Expectation Maximization algorithm [19]. The algorithm is iterative, and consists of two steps: in the E-step corresponding to the t^{th} iteration, we compute the posterior probability of each label for all the unlabeled examples w.r.t. the old parameter values $\theta^{(t)}_j(y), \pi^{(t)}(y)$ as follows:

$$\forall_y P(y|x, \theta^{(t)}, \pi^{(t)}) = \frac{P(x|\theta^{(t)}(y))\pi^{(t)}(y)}{\sum_{y'} P(x|\theta^{(t)}(y'))\pi^{(t)}(y')} \quad (16)$$

In the M-step, we estimate the new parameters $\theta^{(t+1)}_j(y), \pi^{(t+1)}(y)$ using the posterior probabilities as follows.

$$\theta^{(t+1)}_j(y) = \frac{\sum_{i=1}^N f_j(x_i)P(y|x_i, \theta^{(t)}_j(y))}{\sum_{i=1}^N P(y|x_i, \theta^{(t)}_j(y))} \quad (17)$$

$$\pi^{(t+1)}(y) = \frac{\sum_{i=1}^N P(y|x_i, \theta^{(t)}_j(y))}{N} \quad (18)$$

where N is the number of unlabeled examples available during training. In our case, this is the size of the set X^{target}_{test}. The iterations are continued until the likelihood of the unlabeled data converges to a maximum value. In the completely unsupervised case of the EM algorithm, the model parameters are initialized to random values before starting the iterations. In our case, since we have $(X^{source}_{train}, Y^{source}_{train})$ at our disposal, we first do a classic supervised training of our model using the labeled source data, and initialize the parameters to the ones learned from the source data, before we start the EM iterations. This encodes the information available from the source data into the model, while allowing the EM algorithm to discover its optimal parameters on the target domain.

Table 2. Summary of data used in experiments

Corpus name (Abbr.)	Abstracts	Tokens	% Positive
UTexas (UT)	748	216,795	6.6%
Yapex (Y)	200	60,530	15.0%
Yapex-train (YTR)	160	48,417	15.1%
Yapex-test (YTT)	40	12,113	14.5%

3.3.4 Relaxed transductive transfer learning: redefining the prior

In the case when the values of the prior probability of each class in the target data is available, we simply fix $\pi(y)$ to these values and only estimate $\theta(y)$ using eqn. 17 in the M-step of the EM algorithm.

4 Investigation

4.1 Domain

We now turn to *protein name extraction*, an interesting problem domain [21, 26, 12] in which to test these methods. In this setting you are given text related to biological research (usually abstracts, captions, and full body text from biological journal articles) which is known to contain mentions of protein names. The goal is to identify which words are part of a protein name mention, and which are not. One major difficulty is that there is a large variance in how these proteins are mentioned and annotated between different authors, journals, and sub-disciplines of biology. Because of this variance it is often difficult to collect a large corpus of truly identically distributed training examples. Instead, researchers are often faced with heterogeneous sources of data, both for training and testing, thus violating one of the key assumptions of most standard machine learning algorithms. Hence the setting of transfer learning is very relevant and appropriate to this problem.

4.2 Data and evaluation

Our corpora are abstracts from biological journals coming from two sources: University of Texas, Austin (UT) [5] and Yapex [10]. Each abstract was tokenized and each token was hand-labeled as either being part of a protein name or not. We used a standard natural language toolkit [7] to compute tens of thousands of binary features on each of these tokens, encoding such information as capitalization patterns and contextual information of surrounding words.

Some summary statistics for these data are shown in table 2. We purposely chose corpora that differed in two important dimensions: the total amount of data collected and the relative proportion of positively labeled examples in each dataset. Specifically, UT has over three times as many tokens as Yapex but has only half the proportion of

Table 3. Training and testing data used in the settings of Inductive learning (I), Inductive Transfer (IT), Transductive Transfer (TT) and Relaxed Transductive Transfer (RTT). Abbreviations of data sets are described in table 2.

Setting	Source-train	Target-train	Target-test
I	-	YTR	YTT
IT	UT	YTR	YTT
TT	UT	-	Y
RTT	UT	-	Y

positively labeled protein names. This disparity is not uncommon in the domain and could be attributed to differing ways the data sources were collected and annotated. Specifically, if the protein mention annotations in Yapex tend to be longer (that is, extend for more tokens) then the proportion of positively labeled tokens will be higher in Yapex. For all our experiments, we used the larger UT dataset as our source domain and the smaller Yapex dataset as our target. We also split the Yapex data into two parts: *Yapex-train* (YTR) consisting of 80% of the data, and *Yapex-test* (YTT), consisting of the remaining 20%.

In table 3, we display the subsets of data used for various learning settings in our experiments. Note that the transductive methods use different testing data from the inductive methods. This choice is made deliberately to provide a chance for the classifiers in each setting to achieve their peak performance, i.e., transductive algorithms work best when there is abundance of unlabeled test data and inductive algorithms work best when there is plenty of labeled data. However, since the data is slightly different between inductive and transductive settings, one must use caution in comparing the transductive results to the inductive ones.

Because of the relatively small proportion of positive examples in both the UT and Yapex datasets, we are more interested in achieving both high precision and recall of protein name mentions instead of simply maximizing classification accuracy. Since we were dealing with binary, and not sequential classification, the definition of these measures is straightforward as summarized below:

$$\begin{aligned}
\text{accuracy} &= \frac{\text{\# of tokens labeled correctly by the model}}{\text{total \# of tokens}} \\
\text{precision} &= \frac{\text{\# of POS-tokens labeled POS by the model}}{\text{\# of tokens labeled POS by the model}} \\
\text{recall} &= \frac{\text{\# of POS-tokens labeled POS by the model}}{\text{\# of POS-tokens}} \\
\text{F1} &= \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}} \qquad (19)
\end{aligned}$$

We use the $F1$ measure, which combines precision and recall into one metric, as our main evaluation measure.

4.3 Experiments and results

We assessed the relative performance of the various methods on four different settings described in section 3. In addition to running the corresponding adaptations of each model for each of the settings, we did a few additional runs across the settings for purposes of illustration. For example, we ran the transductive SVM not only on the transductive settings, but also on the two inductive settings. Note that TSVM, when run on the inductive case corresponds to transductive learning (see table 1) and when run on the inductive transfer case, corresponds to the supervised transductive transfer learning in table 1. There are other extra runs we did for the purposes of comparison, which will become apparent from the following discussion.

Table 4 summarizes the results under all four settings. The inductive experiment is dominated by Naive Bayes, achieving an F1 of 86% compared to MaxEnt's 82% and TSVM's 73%. This should not be surprising since generative models are known be robust when large amount of labeled training data is available.

Moving to the transductive transfer setting causes all three methods' performances to fall, but MaxEnt falls most sharply, causing it to lose its entire lead over TSVM. Note that in this setting, basic MaxEnt and ISVM have equivalent performance of about 54% F1. The inductive Naive Bayes (using maximum likelihood estimator) proves to the top performer in this setting. TSVM, on the other-hand, is able to adjust its hyperplane in light of the transfer test data and stabilize its performance at 60%, even though it is unlabeled, because it knows where these points lie relative to the labeled training points in feature space. Similarly, we see the effect of our iterative feature transformation algorithm (*IFT*, section 3.1.3) on MaxEnt's transductive transfer performance. We use a conservative value of θ (.95) to ease the transition from source- to target-based conditional feature expectations. Indeed, as expected from our previous analysis, iteratively combining the approximate joint feature-label expectations in the target data with the true joints of the source data improves the overall performance on the target data. It seems this method is bounded, however, by the quality of the initial target labels generated by the source-trained classifier. Given a relatively poor initial classification, how can we bootstrap our way up to higher and higher performance? This is certainly a question worthy of future study. The transductive version of the naive Bayes (using EM), however, fares worse than its inductive counterpart. Since EM's optimization function is the marginal log-likelihood of the test data, it is not guaranteed to improve the classification performance in some cases.

In the relaxed transductive transfer setting, finally, where the target dataset is still unlabeled but all algorithms are told the expected proportion of positive examples, TSVM ex-

Table 4. Summary of % accuracy (**Acc**), precision (**Prec**), recall (**Rec**), and F1 for regular maximum entropy (**Basic**), Iterative Feature Transformation MaxEnt (**IFT**), prior-based regularized MaxEnt (**Regularize**), and feature expansion MaxEnt (**Expand**), inductive SVM (**ISVM**), transductive SVM (**TSVM**), Maximum Likelihood Naive Bayes (**NB-ML**), and EM based Naive Bayes (**NB-EM**) models under the conditions of classic inductive learning, (**Induction**), unsupervised transductive transfer learning, (**TransductTransfer**), relaxed transductive transfer, (**RelaxTransductTransfer**), and supervised inductive transfer (**InductTransfer**).

Method	Induction				TransductTransfer				RelaxTransductTransfer				InductTransfer			
	Acc	Prec	Rec	**F1**	Acc	Prec	Rec	**F1**	Acc	Prec	Rec	**F1**	Acc	Prec	Rec	**F1**
MAXIMUM ENTROPY																
Basic	95	85	78	**82**	89	75	42	**54**	90	65	68	**67**	91	81	54	**65**
IFT, 1 iter	-	-	-	-	79	41	90	**56**	-	-	-	-	-	-	-	-
IFT, 2 iters	-	-	-	-	82	45	86	**59**	-	-	-	-	-	-	-	-
Regularize	-	-	-	-	-	-	-	-	-	-	-	-	96	87	84	**85**
Expand	-	-	-	-	-	-	-	-	-	-	-	-	93	84	62	**72**
SUPPORT VECTOR MACHINES																
ISVM	92	78	58	**67**	90	86	40	**54**	90	86	40	**55**	92	86	52	**65**
TSVM	92	68	79	**73**	91	86	46	**60**	92	72	75	**73**	93	86	58	**70**
NAIVE BAYES																
NB-ML	95	80	93	**86**	85	50	81	**62**	84	48	85	**61**	86	55	84	**67**
NB-EM	-	-	-	-	79	40	84	**54**	80	41	82	**55**	-	-	-	-

cels. Again, while MaxEnt is able to make significant use of this information (note the jump to 67% from 54%), it seems TSVM does a better job leveraging the prior knowledge into better performance. Maximum Likelihood based Naive Bayes, on the other hand loses out. It seems that the class conditional probability is more critical in naive Bayes than the prior, so tuning the latter's value does not have any positive impact on its performance. Also, notice that the EM based naive Bayes is even worse, repeating the pattern in the transductive transfer case.

Finally, the last column of table 4 compares the performance of the three methods for inductive transfer learning: the prior-based regularized maximum entropy method (*Regularize*, described in section 3.1.2), and the feature expanding version (*Expand*, described in section 3.1.2). We can see that both methods handily outperform the transductive transfer methods described in the second column of table 4, and for the most part outperform even the relaxed transductive transfer versions in column three. This should not be surprising given the fact that the inductive transfer methods can actually see some labeled examples from the target domain and thus, in the case of MaxEnt, better estimate the conditional expectation of the features in the target data. Likewise, since they have access to labeled target data, they can also assess the proportion of positive examples and adjust their decision functions accordingly. What is more surprising, however, is the fact that these methods do not significantly outperform the inductive learning methods described in the first column of table 4. This suggests that these inductive transfer methods are relying almost entirely on their labeled target data in order to train their classifiers, and are not making full use of the large amount of labeled source data. One might assume that having access to almost four times as much related data, in the form of the labeled source data, would significantly boost their ability to classify the target data (this is, after all, one of the stated goals of transfer learning). Dishearteningly, in this instance, this seems not to be the case. The regularized maximum entropy model does outperform[2] the basic MaxEnt in the inductive setting, but not by as much as might have been hoped for.

In order to measure how much these inductive transfer methods' explicit modeling of the transfer problem was responsible for their performance, we compared them to the baselines of ISVM, TSVM, MaxEnt and Naive Bayes trained on a simple concatenation of the labeled source and target training data. These transfer-agnostic methods clearly benefited from the addition of labeled target data (as compared to column *TransductiveTransfer*), yet still yielded consistently lower F1 than the transfer-aware *Regularize* and *Expand* methods, suggesting that the mere presence of labeled sets of both types (source and target) of data is not enough to account for the transfer methods' superior results.

[2] *Regularize* has F1 of 85 vs. *MaxEnt*'s 82. Significance was determined by comparing the 99% binomial confidence intervals for each method's recall and precision.

Instead, it seems it is the modeling of the different domains in the transfer problem, even in simple ways, that provides the extra boost to performance.

5 Conclusions

These experiments and analysis have shed light on a number of important issues and considerations related to the problems of transduction and transfer learning.

We have seen that in the case of discriminative models, even a small amount of prior knowledge about the target domain can greatly improve performance in a transductive transfer problem. Generative model is not able to exploit this information. For all these models, we notice that even large amounts of source data cannot overcome the advantage of having access to labeled data drawn from the target distribution.

We have also seen the degree to which pseudo-labeling based schemes (in both TSVM's margin-based model and our MaxEnt's IFT-based model) can improve performance by incorporating the unlabeled structure of the target domain. However, this improvement is not seen in the generative Naive Bayes model. We believe this is because discriminative models directly optimize classification accuracy, while the EM based Naive Bayes model optimizes an unrelated function, namely, the marginal log-likelihood.

Finally, we have seen that the generative Naive Bayes model is robust in the inductive setting with large amount of labeled data, while the discriminative models are at least as good or better in the transductive setting. Of the two discriminative models considered, the margin based SVM seems to adapt better to the unlabeled data.

6 Future work

Given the promising results of our MaxEnt based feature transformation methods, we would like to further investigate the theoretical properties of the IFT-type algorithms. In particular, we would like to be able to guarantee convergence.

In terms of the named entity extraction application, we are also looking towards applying these techniques to the sequential, rather than just binary labeling problem. Most transfer learning results have emphasized the use of structure in relating the source and target domain, and it seems sequential classifiers like conditional random fields [23] would be better equipped to exploit this structure.

References

[1] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. In *JMLR 6*, pages 1817 – 1853, 2005.

[2] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. In *NIPS 20*, Cambridge, MA, 2007. MIT Press.

[3] A. L. Berger, S. D. Pietra, and V. J. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

[4] J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *EMNLP*, Sydney, Australia, 2006.

[5] R. Bunescu, R. Ge, R. Kate, E. Marcotte, R. Mooney, A. Ramani, and Y. Wong. Comparative experiments on learning information extractors for proteins and their interactions. In *Journal of AI in Medicine*, 2004. Data from *ftp://ftp.cs.utexas.edu/pub/mooney/bio-data/proteins.tar.gz*.

[6] C. Chelba and A. Acero. Adaptation of maximum entropy capitalizer: Little data can help a lot. In D. Lin and D. Wu, editors, *EMNLP 2004*, pages 285–292. ACL, 2004.

[7] W. W. Cohen. Minorthird: Methods for identifying names and ontological relations in text using heuristics for inducing regularities from data. http://minorthird.sourceforge.net, 2004.

[8] H. Daumé III. Frustratingly easy domain adaptation. In *ACL*, 2007.

[9] H. Daumé III and D. Marcu. Domain adaptation for statistical classifiers. In *Journal of Artificial Intelligence Research 26*, pages 101–126, 2006.

[10] K. Franzén, G. Eriksson, F. Olsson, L. Asker, P. Lidn, and J. Cöster. Protein names and how to find them. In *International Journal of Medical Informatics*, 2002.

[11] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *CAP*, Nice, France, 2005.

[12] K. Ji, M. Ohta, and Y. Tsujii. Tuning support vector machines for biomedical named entity recognition. In *ACL Workshop on Natural Language Processing in the Biomedical Domain.*, 2002.

[13] J. Jiang and C. Zhai. Exploiting domain structure for named entity recognition. In *Human Language Technology Conference*, pages 74 – 81, 2006.

[14] T. Joachims. Transductive inference for text classification using support vector machines. In *ICML 16*, 1999.

[15] T. Joachims. *Learning to Classify Text Using Support Vector Machines*. Kluwer, 2002.

[16] T. Joachims. Transductive learning via spectral graph partitioning. In *ICML*, 2003.

[17] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *In AAAI Workshop on Learning for Text Categorization*, 1998.

[18] K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification, 1999.

[19] K. Nigam, A. K. McCallum, S. Thrun, and T. M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.

[20] R. Raina, A. Y. Ng, and D. Koller. Transfer learning by constructing informative priors. In *ICML 22*, 2006.

[21] L. Shi and F. Campagne. Building a protein name dictionary from full text: a machine learning term extraction approach. In *BMC Bioinformatics 6:88*, 2005.

[22] V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In *ICML*, pages 824–831. ACM, 2005.

[23] C. Sutton and A. McCallum. Composition of conditional random fields for transfer learning. In *HLT/EMLNLP*, 2005.

[24] S. Thrun. Is learning the n-th thing any easier than learning the first? In *NIPS*, volume 8, pages 640–646. MIT, 1996.

[25] V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[26] R. C. Wang, A. Tomasic, R. E. Frederking, and W. W. Cohen. Learning to extract gene-protein names from weakly-labeled text in preparation. In *preparation*, 2006.

[27] X. Zhu. Semi-supervised learning literature survey. In *Technical Report 1530*. University of Wisconsin, 2005.