Learning on the Test Data: Leveraging "Unseen" Features

Ben Taskar BTASKAR@CS.STANFORD.EDU
Ming Fai Wong MINGFAI.WONG@CS.STANFORD.EDU
Daphne Koller KOLLER@CS.STANFORD.EDU
Computer Science Department, Stanford University, Stanford, CA 94305-9010 USA

Abstract

This paper addresses the problem of classification in situations where the data distribution is not homogeneous: Data instances might come from different locations or times, and therefore are sampled from related but different distributions. In particular, features may appear in some parts of the data that are rarely or never seen in others. In most situations with non-homogeneous data, the training data is not representative of the distribution under which the classifier must operate. We propose a method, based on probabilistic graphical models, for utilizing unseen features during classification. Our method introduces, for each such unseen feature, a continuous hidden variable describing its influence on the class — whether it tends to be associated with some label. We then use probabilistic inference over the test data to infer a distribution over the value of this hidden variable. Intuitively, we "learn" the role of this unseen feature from the test set, generalizing from those instances whose label we are fairly sure about. Our overall probabilistic model is learned from the training data. In particular, we also learn models for characterizing the role of unseen features; these models use "meta-features" of those features, such as words in the neighborhood of an unseen feature, to infer its role. We present results for this framework on the task of classifying news articles and web pages, showing significant improvements over models that do not use unseen features.

1. Introduction

Most statistical learning models make the assumption that data instances are IID samples from some fixed distribution. This assumption is often violated in real-world situations. In many cases, the data are collected from different sources, at different times, locations and under different circumstances. In such cases, the training data is often not representative of the data over which the classifier must operate. For example, in classifying news articles, instances are usually organized chronologically. New events, people and places appear (and disappear) in bursts over time (Kleinberg, 2002). The training data might consist of articles taken over some time period; these are only somewhat representative of future articles. In a task of classifying customers into categories, our training data might be collected from one geographical region, which may not represent the distribution in other regions.

Traditionally, this distribution drift is either ignored or avoided by changing the data to conform better to the IID assumption — all examples are mixed together and training and test are selected randomly. Unfortunately, this homogeneity cannot be ensured in real-world tasks, where only the (non-representative) training data is actually available for training.

One aspect of the non-homogeneity phenomenon is that the test data may contain many features that were never or only rarely observed in the training data. These features may be very useful for classification. For example, in our news article task, these features might include the names of places or people currently in the news. In the customer example, the features might include purchases of products that are specific to a region (e.g., snow suits). However, these features and their effect on the class label cannot be identified from the training data.

In this paper, we propose a method for identifying and utilizing these unseen features. Our approach is to associate with each unseen feature a hidden variable that encodes influence this feature has on the class label. The value of this variable is unknown, and must be "learned" from the test data, without knowledge of the labels in the test data. Our method uses probabilistic inference over a graphical model to infer these values. Probabilistic inference effectively "bootstraps" from instances that are classified with high confidence, identifying the interaction between these unseen features and the class, and thereby helping to classify other new instances.

So what, if anything, can we learn from the training data about these new features? Although the unseen features in the test data do not appear in the training data, the training data might contain other features that play a similar role. For example, in our news article domain, the phrase "XXX said today" might appear in many places in the data, for different values of "XXX". In many cases, "XXX" is the name of a person in the news, and might be (temporarily) a useful feature for determining the topic of an article. In the customer example, we might have features of products that can help us predict their role as unseen features. Our

Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.

algorithm learns a template model which predicts the effect of a unseen feature from from a set of *meta-features*. This template model is learned from seen features in the training data, and applied to unseen features in the test data.

The remainder of this paper is structured as follows. In Section 2 we describe a general framework and probabilistic model for global and unseen features. In Section 3 we describe our algorithm for learning this model from data and in Section 4 we show how it can be used in classification. In Section 5 we provide experimental results on the Reuters news articles dataset and a University web site dataset, showing significant improvement in accuracy. We conclude in Section 6 with discussion and a comparison to related work.

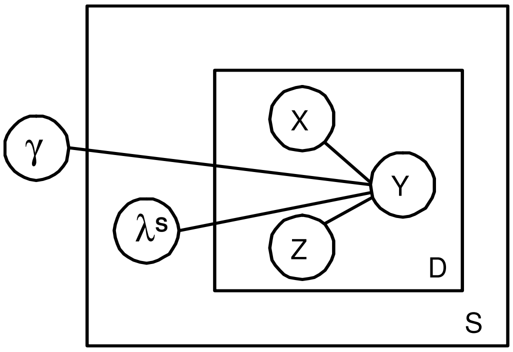

Figure 1. Plate representation of the basic scoped features model. Outer plate represents multiple scopes, inner plate represents multiple data instances.

2. General Framework

2.1. Scopes

To make our intuitions precise within the framework of a probabilistic model, we need the notion of a *scope*. We assume that data instances are sampled from some set of scopes, each of which is associated with some data distribution. Of course, the problem only makes sense as a coherent learning problem if some aspects of the distributions are shared. We focus on cases where the different distributions share a probabilistic model for some set of *global features*, but can contain a different probabilistic model for a scope-specific set of *local features*. These local features may be rarely or never seen in the scopes comprising the training data.

Let $\mathbf{X}$ denote global features, $\mathbf{Z}$ denote local features, and Y the class variable. We assume, for simplicity, a binary classification problem where $Y \in \{-1, 1\}$; our model can be extended to multi-class classification. In scope-varying data, we assume that for two scopes S and S', $P_S(Y \mid \mathbf{X}, \mathbf{Z})$ and $P_{S'}(Y \mid \mathbf{X}, \mathbf{Z})$ can be quite different, but the effect of the global features is preserved. To make this intuition precise, we fix a particular parametric model for $P_S(Y \mid \mathbf{X}, \mathbf{Z})$ — logistic regression. In this model, for each global feature X_i, there is a parameter γ_i. Additionally, for each scope and each local feature Z_i, there is a parameter λ_i^S. Then, the distribution of Y given all the features and weights is

$$P_S(y \mid \mathbf{x}, \mathbf{z}; \gamma, \lambda^S) \propto \exp\{\gamma \cdot y\mathbf{x} + \lambda^S \cdot y\mathbf{z}\}. \quad (1)$$

Fig. 1 shows a representation of this probabilistic model in terms of plates. The global feature parameters γ are the same across scopes, while the local feature parameters λ^S depend on the scope S. As is often the case, we assume that the global weights can be learned from the training data, so that their values are fixed when we encounter a new scope. The key issue that we try to address in this paper is that the local feature weights are unknown.

2.2. Probabilistic Model

The main idea behind our approach is that these local feature weights can be treated as hidden variables in a graphical model. In such a model, we would like evidence from global features for the labels of some of the instances to modify our beliefs about the role of the local features present in these instances to be consistent with the labels. By learning about the roles of these features, we can then propagate this information to improve accuracy on instances that are harder to classify using global features alone. To implement this idea, we define a joint distribution over λ^S and $y^1, \ldots, y^m$.

For reasons that we discuss below, we use the framework of undirected graphical models, or *Markov networks* (Pearl, 1988). Specifically, we define a *hybrid* Markov network (one containing both continuous and discrete variables), that specifies a single joint distribution over the class variables and features of all of the instances in a scope, and the feature parameters.

We begin by briefly reviewing the framework of Markov networks. We present the log-quadratic parameterization of a hybrid Markov network, as it is more directly suited to our needs. Let $\mathbf{V} = (\mathbf{V}_d, \mathbf{V}_c)$ denote a set of random variables, where $\mathbf{V}_d$ are discrete and $\mathbf{V}_c$ are continuous variables, respectively. A Markov network over $\mathbf{V}$ defines a joint distribution over $\mathbf{V}$, assigning a density over $\mathbf{V}_c$ for each possible assignment $\mathbf{v}_d$ to $\mathbf{V}_d$. A Markov network $\mathcal{M}$ is an undirected graph whose nodes correspond to $\mathbf{V}$. It is parameterization by a set of *potential functions* $\phi_1(\mathbf{C}_1), \ldots, \phi_\ell(\mathbf{C}_\ell)$, such that each $\mathbf{C} \subset \mathbf{V}$ is a fully connected subgraph, or clique, in $\mathcal{M}$, i.e., each $V_i, V_j \in \mathbf{C}$ are connected by an edge in $\mathcal{M}$. (Note that a single node is also considered a clique.) For our purposes, we assume

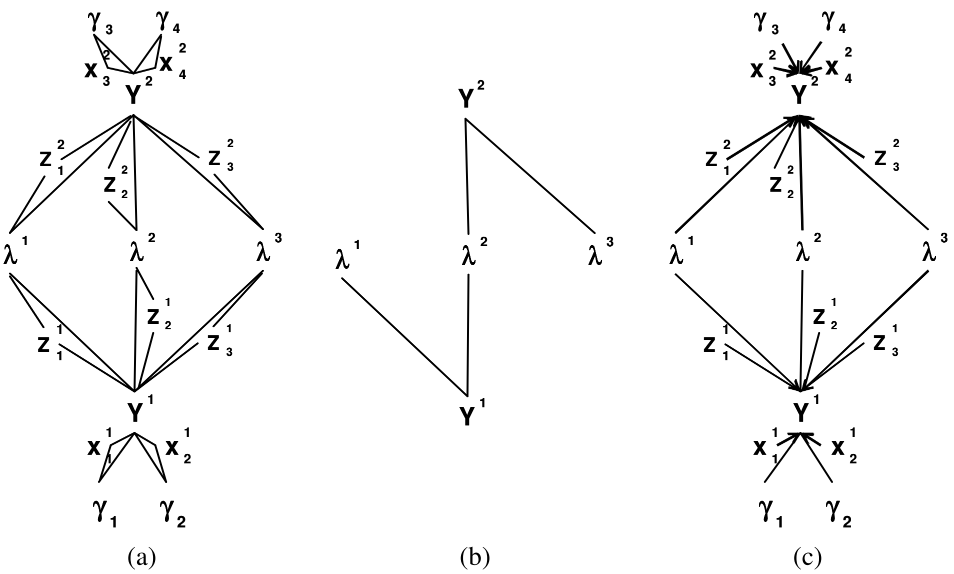

Figure 2. (a) Markov network for two instances, two global features, and three local features. (b) Conditioned Markov network for the same setting in the case of binary local features, assuming that the instance $(\mathbf{x}^1, \mathbf{z}^1, y^1)$ contains the features Z_1 and Z_2, and the instance $(\mathbf{x}^2, \mathbf{z}^2, y^2)$ contains the features Z_2 and Z_3. (c) Hypothetical directed model.

that the function $\phi(\mathbf{C})$ is log-quadratic in variables $\mathbf{u}_c$ for each value $\mathbf{u}_d$ of $\mathbf{C}_d$. The Markov network then represents the distribution: $P(\mathbf{V}) \propto \prod_{i=1}^{\ell} \phi_i(\mathbf{C}_i)$.

In our case, the Markov network for a given scope contains: the global and local features $\mathbf{X}^1, \ldots, \mathbf{X}^m$ and $\mathbf{Z}^1, \ldots, \mathbf{Z}^m$ for all instances; the labels for all instances $Y^1, \ldots, Y^m$; and the global and local feature weights γ and λ^S. Our log-quadratic model contains three types of potentials: One type of potential has the form $\phi(\gamma_i, Y^j, X_i^j) = \exp\{\gamma_i Y^j X_i^j\}$, and relates each global feature X_i^j in instance i to its weight γ_i and the class variables Y_j of the corresponding instance i. A similar potential $\phi(\lambda_i, Y^j, Z_i^j) = \exp\{\lambda_i Y^j Z_i^j\}$ relates the local feature Z_i^j to its weight λ_i and the label Y^j. Finally, as the local feature weights are assumed to be hidden, we introduce a prior over their values, or the form $\phi(\lambda_i^S) = \exp\{-\frac{(\lambda_i^S - \mu_i)^2}{2\sigma^2}\}$.

Overall, our model specifies a joint distribution as follows:

$$\begin{aligned} & P_S(y^1, \ldots, y^m, \lambda^S \mid \mathbf{x}^1, \ldots, \mathbf{x}^m, \mathbf{z}^1, \ldots, \mathbf{z}^m; \gamma) \\ \propto \quad & \pi(\lambda^S) \prod_{j=1}^{m} \exp\{\gamma \cdot y^j \mathbf{x}^j + \lambda^S \cdot y^j \mathbf{z}^j\}, \end{aligned} \qquad (2)$$

where $\pi(\lambda^S) = \prod_i \exp\{-\frac{(\lambda_i^S - \mu_i)^2}{2\sigma^2}\}$.

Viewed graphically, our Markov network contains edges between each Y^j and all of its global features and corresponding parameters, and between Y^j, and each of its local features and corresponding parameters associated with that local feature. Fig. 2(a) shows a simple example. The graph can be simplified considerably, when we account for variables whose values are fixed. In this case, we can simply instantiate the various functions in our log-quadratic models with these values, and omit the variables from the model. As we discussed, we assume that the global feature weights γ are learned from the training data, and hence their value is fixed. Furthermore, in a classification setting, the actual feature values are known. The resulting Markov network is a fully connected bi-partite graph over the set of unobserved variables $\{Y^j\}$ and $\{\lambda_i\}$. However, when the local features Z_i^j are sparse, this network can be simplified significantly. When $Z_i^j = 0$, there is no interaction between Y^j and any of the variables λ_i. In this case, we can simply omit the edge between λ_i and Y^j. For example, if our instances are documents and our local features are words, our model would contain an edge only between the label of a document and the weights of the scope-specific words (local features) that it contains. An example of this conditioned Markov network is shown in Fig. 2(b).

In this model, we can see that the labels of all of the instances are correlated with the local feature weights of features they contain, and thereby with each other. Thus, for example, if we obtain evidence (from global features) about the label Y^1, it would change our posterior beliefs about the local feature weight λ_2, which in turn would change our beliefs about the label Y^2. Thus, by running probabilistic inference over this graphical model, we obtain updated beliefs both about the local feature weights and about the instance labels.

Importantly, this flow of influence depends critically on our choice of single joint undirected graphical model. Suppose, for example, that we had chosen a directed model, where each class label variable Y^j is simply a logistic conditional distribution of the instance features. In this model, illustrated in Fig. 2(c), when the labels (y's) are not observed, they d-separate the weights from each other, and no influence flows from the observed global features, through the labels, to help infer the role of the local features.

3. Learning the Model

We now describe how the model described in the previous section can be learned from data.

3.1. Learning Global Feature Weights

For the case of global features, the problem is an easy one. We simply learn their parameters from the training data, using standard logistic regression. Maximum-likelihood (ML) estimation finds the weights γ that maximize the conditional likelihood of the labels given the global features,

$$\prod_{j=1}^{m} P_\gamma(y^j \mid \mathbf{x}^j) \propto \prod_{j=1}^{m} \exp\{\gamma \cdot y^j \mathbf{x}^j\}.$$

To help avoid overfitting, we assume a "shrinkage" prior over the weights (a zero-mean Gaussian), and use maximum a posteriori (MAP) estimation. More precisely, we assume that parameters are a priori independent and define $P(\gamma) = \prod_i \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\gamma_i^2/2\sigma^2\right\}$. We fit the regression model using conjugate gradient, where the gradient of the log-posterior objective is given by:

$$\sum_{\mathrm{J}=1}^{m} \left(y^j \mathbf{x}^j - \boldsymbol{E}_{y|\mathbf{x}^j}[y\mathbf{x}^j]\right) - \frac{\gamma}{\sigma^2}\,.$$

3.2. Learning Local Feature Distributions

The weights of the local features are unknown in a test scope. Hence, we use a distribution over their values. Most simply, we use a Gaussian distribution with zero mean and some variance σ^2. In this case, we assume that we know nothing about the local features, and hence all of them have an identical prior distribution. More interesting, however, is the case where we can infer something about a local feature, even if we have never seen it before.

As we discussed in the introduction, we often have additional cues that indicate the role of a local feature. For example, consider the problem of trying to distinguish between news articles labeled *grain* and those labeled *trade*. Words such as "corn" or "wheat" are very useful for distinguishing these topics. These words often appear within the phrase "tons of corn (wheat)". Thus, we can learn that if a word "XXX" appears in the context "tons of XXX", it is likely to have positive interaction with the label *grain*. If we now see the phrase "tons of rye" in the test data (where "rye" is rare in the training data), we can infer that "rye" probably has positive interaction with the label *grain*. This conclusion, in turn, will let us better classify documents containing the word "rye", even if that word does not appear in the context "tons of rye".

We can exploit such patterns by learning a model that predicts the prior of the local feature weights using *meta-features* — features of features. More precisely, we learn a model that predicts the prior mean μ_i for λ_i from some set of meta-features $\mathbf{m}_i$. In our news classification example, these meta-features might consist of the words in the vicinity of the word (feature) Z_i from all of the word's occurrences in the scope. Other meta-features might include capitalization, morphological information, presence in a list of proper names, or more. As our predictive model for the mean μ_i we choose to use a linear regression model, setting $\mu_i = \mathbf{w} \cdot \mathbf{m}_i$.

The parameters $\mathbf{w}$ are learned from the training data. We first learn a feature weight for each global feature, and for each feature local to the training scope(s). We then train the regression model weights $\mathbf{w}$ to predict the weight of feature i as $\mathbf{w} \cdot \mathbf{m}_i$. We used a standard linear regression with ridge penalty of 0.1.

Note that we train the meta-feature model using both global and local features. This decision reflects an assumption that the meta-features help predict the weights of any feature, local or global. In other words, features that have similar context tend to interact similarly with the class label. This assumption allows us to utilize the information from global features to learn our probabilistic model for local feature weights.

The other parameter characterizing the distribution over local feature weights is the variance. Intuitively, if the variance of a weight is small, its posterior distribution must stay fairly close to its original mean. The larger it is, the more flexibility there is for the test instances to change the posterior mean, updating our beliefs about the role that this feature plays in the test scope. We currently do not learn the variance parameter, but set it via cross-validation.

4. Using the Model

Now that we have discussed both the basic model and how it is learned from data, we describe the overall use of the model in the context of an actual learning task.

Given a training set, we first learn the model, as described in the previous section. We note that, in the training set, there local and global features are treated identically. When applying the model to the test set, however, our first decision is to determine the set of local and global features. In some cases, such as the work of (Blei et al., 2002), prior knowledge helps us distinguish global and local features. When we have binary-valued features (such as words), we can use a simple heuristic, and select as local features those features that have zero or low frequency in the training set and high frequency in the test set. The intuition behind this rule is twofold: First, it is clear that the probabilistic model for these features is fairly different in the training and test set, so they are likely to be scope-specific. Second, the model learned for these features in the training data is likely to be quite poor, so a local model is often better. In general, the problem of distinguishing local features from global ones is far from trivial, and is an important topic for future work.

Our next step is to generate the Markov network for the test set, as described in Section 2.2. This Markov network defines a joint probability distribution over the two types of hidden variables: the weights of the local features, and the instance labels. As we discussed, probabilistic inference over this model will precisely implement the bootstrapping process described above for inferring the effect of local features. It will also label the instances in a way that takes advantage of this information.

The major difficulty is that this joint distribution is over a very high-dimensional space. However, the presentation as a graphical model reveals certain structure that can be exploited for inference. In very simple cases, the graph may be sufficiently structured that exact (up to numerical error) inference can be used (Lerner et al., 2001). In our experiments, however, the graphs contain thousands of nodes and are quite densely connected. Exact inference is completely intractable in these cases. We therefore resort to approximate inference.

Several approximate inference methods can be used to solve this problem, including variational (Jordan et al., 1999; Blei et al., 2002), or Monte Carlo sampling. We chose to use *expectation propagation* for its simplicity and relative efficiency and accuracy. Expectation Propagation (EP) is a local message passing algorithm (Minka, 2001; Heskes & Zoeter, 2003) akin to Belief Propagation (BP) (Pearl, 1988; Yedidia et al., 2000). Like BP, EP maintains approximate beliefs (marginals) over nodes of the Markov network and iteratively adjusts them to achieve local consistency. In our networks, we maintain independent Gaussian beliefs over λ variables, and discrete beliefs over the Y variables. The messages $m(\cdot)$ and beliefs $b(\cdot)$ are initialized to uniform. Nodes then iteratively pass messages to each other, and update their local beliefs.

In our network, each edge connects between a node representing a local feature weight λ_i and a node corresponding to a label Y^j. We therefore have two types of messages: $m_i(Y^j)$ that goes from the weight to the label, and $m_j(\lambda_i)$ that goes the other way. These messages are computed as follows:

$$\begin{aligned} m_i(Y^j) &\leftarrow \alpha \int \frac{b(\lambda_i)\ e^{\lambda_i Y^j z_i^j}}{m_j(\lambda_i)} d\lambda_i \\ m_j(\lambda_i) &\leftarrow \alpha \sum_{Y^j} \frac{b(Y^j)\ e^{\lambda_i Y^j z_i^j}}{m_i(Y^j)}, \end{aligned}$$

where α is a normalizing constant. Computing the message $m_i(Y^j)$ involves one-dimensional numerical integration which can be done efficiently using Gaussian quadrature (Press et al., 1988).

In BP, the beliefs at each node are computed as the product of the prior and the incoming messages. In hybrid networks such as the one in our application, the resulting function is a very complex multi-modal function, and is computationally prohibitive to maintain. In EP, we approximate these beliefs as a Gaussian, and update them using "weak marginalization", where the product of messages from Y's and the prior is approximated by matching its mean and variance:

$$b(\lambda_i) \approx b'(\lambda_i) = \alpha \pi(\lambda_i) \prod_j m_j(\lambda_i).$$

Again, we use Gaussian quadrature to compute the moments of $b'(\lambda_i)$. The beliefs of the label nodes $b(Y^j) = \alpha\ e^{\gamma \cdot Y^j \mathbf{x}^j} \prod_i m_i(Y^j)$ can be computed exactly given the messages.

5. Experimental Results

We performed experiments on two data sets – Reuters, a news articles collection, and Webkb2, a collection of web pages from universities.

5.1. Reuters

We used the ModApte set of Reuters-21578 corpus.[1] We selected four categories in the data set that contain a substantial number of documents (≈ 500); these categories are *grain*, *crude*, *trade*, and *money-fx*. We eliminated documents that are labeled with more than one of these four labels. We split the data into articles, tokenized it, and represented each document as a bag of words.

Using this data set, we created six experimental setups, by using all possible pairings of categories from the four categories that we have chosen. For each pairing, we combined all documents with those two labels, and sorted them

[1]The Reuters-21578 corpus was downloaded from http://www.research.att.com/lewis/reuters21578.html

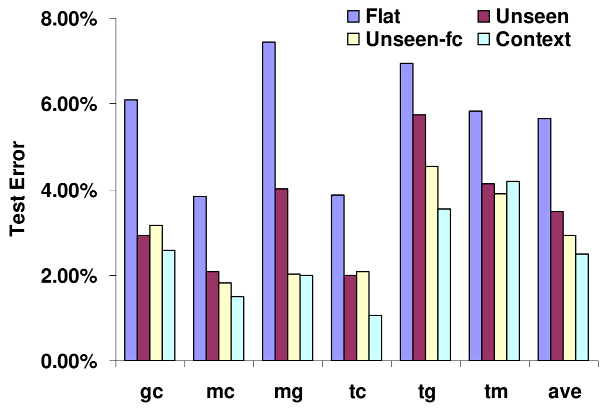

Figure 3. Comparison of scope models on the Reuters dataset. Each column is an average of 5 folds. Six sets of four columns show error on different pairing of categories, while the seventh set is the overall average. All Unseen models significantly outperform the baseline Flat model.

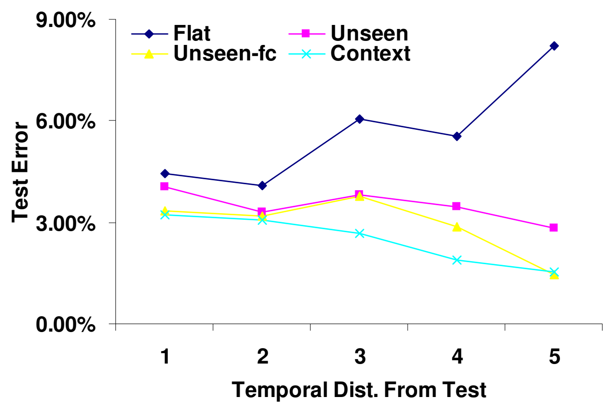

Figure 4. Comparison of test error as the temporal distance between training and test grows. Flat model performs worse with increasing distance, while the scope models do not suffer from this problem and performed roughly the same regardless of the temporal distance.

temporally. We divided the resulting sequence into nine time segments with roughly the same number of documents (≈ 100) in each segment. Thus, documents from the same segment represent articles that are published at around the same period of time, and hence can be assumed to correspond to a scope. In each setup, we set aside segments 7–9 (≈ 300 documents) as our test set, and segment 6 as a validation set. We then trained five models, using each of the segments 1–5 separately as a training set. We used the data in segment 6 to select the parameter σ for the Gaussian prior used for regularization for all models, and evaluated the accuracy on the test set.

Our baseline model Flat is a logistic regression model that uses words as features. We tried different variations for the baseline model, including one that learned using only "global words" — words that appear in the training set with enough frequency (> 8 documents). However, our best model is a Flat model using all words, with each document normalized according to its length.

Unseen is the basic model that leverages on unseen features. For seen features, it uses parameters learned from Flat. As for unseen features, it models the uncertainty over their weights with a zero mean Gaussian and variance set to α, which we set using cross-validation.

Unseen-fc is almost identical to Unseen, except that we set variance to be $\frac{\beta}{|X^j|}$, where $|X^j|$ is the number of times feature j appeared in the test set. Again, we set β using cross-validation.

Context builds on top of Unseen-fc by allowing the prior means of unseen features to be changed by their meta-features, which in this case are words appearing in their contexts. We used the individual words in a window of ± 3 around the feature word as meta-features, as well as consecutive pairs of words within the same window. Finally, we eliminated meta-features that appear too frequently (> 250 times) or too infrequently (< 10 times) as context words.

Fig. 3 shows the results. Six sets of three columns represent test error of the three models for each pair of categories (*grain*,*crude*, *trade*, and *money-fx*) abbreviated by the first letter. Last set of columns is the average test error. Each column is an average of 5 folds. As shown in the figure, both Unseen and Unseen-fc perform better than Flat. The relative reduction in error is 38.51% and 48.47%, respectively. Moreover, Context performs better than the stronger unseen feature model Unseen-fc, with a 15.19% relative reduction in error. The p-values from paired t-test for each of the models Unseen, Unseen-fc and Context with flat are 0.0000566, 0.0000192 and 0.00000749 respectively. Note that the meta-features of the Context model are also a part of the global features and are already used by the Flat model. Hence the improvement that comes from the Context model is due to the indirect impact of these meta-features, which provide the model with a reasonable guess for the weights of the unseen features.

We also compared our models to transductive SVMs using the SVM^{light} package (Joachims, 1999). For the best value of the SVM parameter C, transductive SVMs with a linear kernel achieve average error of 8.59% (averaged over all folds and pairings), which is significantly worse than our models.

Fig. 4 shows how the performance of models varies with the temporal distance between training and test sets. As evident from the graph, the Flat model gradually performs worse with increasing temporal distance. This is because the distribution of words between training and test sets becomes more and more different as the temporal distance between them increases. While the Flat model learned words that are useful for prediction in one time period but not the other, the models utilizing unseen features are able to adapt to changing distributions, and thus do not suffer from the same problem.

It is interesting to examine what meta-features had high impact on determining the function of the unseen features. Some examples with high regression weight include: *tonnes of (wheat/rye/rice)*, and *export* for the category *grain*, and *kuwait*, *saudi* and *demand for (petroleum/oil)* for the category *crude*. Moreover, the model successfully inferred that many features that were not seen on the training were very useful in classification on the test set. Some examples with high posterior mean are *texaco* and *chevron* for the category *grain* and *ventures* and *prosperity* for the category *trade*.

5.2. Webkb2

This data set consists of hand-labeled web pages from Computer Science department web sites of four schools: Berkeley, CMU, MIT and Stanford. We chose four categories based on the requirement that they each have a substantial number of documents in each school ($\approx$ 100 per school). The categories are *faculty*, *student*, *course* and *organization*. As in Reuters, we tokenized each web page and represented it as a bag of words.

We created six experimental setups by using all possible pairings of categories from the four categories that we have chosen. In this dataset, each school naturally corresponds to a scope. Thus, for each pairing, we train a model using each school, using one of the other three schools for validation and the remaining two for test.

Fig. 5 shows the results of our runs on Webkb2 using two of the models described above, Flat and Context. Six sets of two columns represent test error for each pair of categories (*faculty*,*student*, *course*, and *organization*) abbreviated by the first letter. Last set of columns is the average test error. Each column is an average of 8 folds.

As shown in the figure, Context performs better than Flat on all but one pairing, and is superior on average. The reduction of error is 5.5%, and the p-value from paired t-test is 0.0368.

Although our model outperforms the baseline, the improvements on this data set are not as large as in Reuters. There are several possible reasons. First, the words in web pages might not be as well suited as local features when compared

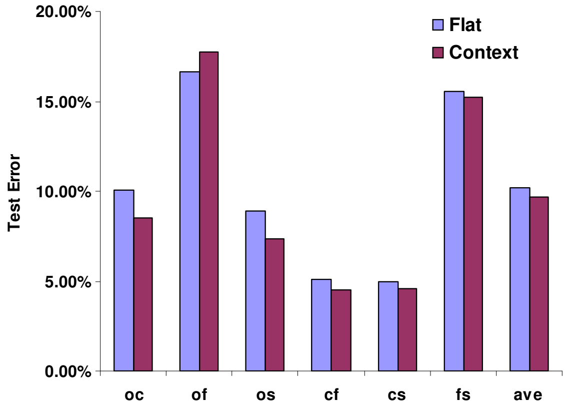

Figure 5. Comparison of Context model against Flat model on the Webkb2 data set. Each column is an average of eight folds. Six sets of four columns show error on different pairing of categories, while the seventh set is the overall average. Context performs better than Flat on all but one pairing.

to the case of news articles. Unlike words in news articles, which are closely associated with the topic of the article, web pages contain more heterogeneous text. Features of html formatting of web pages might serve as better local features; although html formatting changes from school to school, within a particular school, it may be a good indicator of the type of web page. A second reason for the reduction in improvement is that the average performance of the Flat model in the case of Reuters is better. This improved base gives our models a better starting point for inferring the values of unseen features.

6. Discussion

Our work is not the first to address the task of learning with unseen features or scope. Recently, Blei et al. (2002) addressed the problem of local features in the context of webpage classification and information extraction. There are some important conceptual differences between our work and theirs. In their work, local features were disjoint from global features, with local features consisting of html formatting and global features of words on the page. This clean partition simplifies their setting. A second key difference is that they learn very little concerning unseen features from the training data, whereas our approach learns the context characterizing features. At a more technical level, their work assumes that the global probabilistic model is naive Bayes, a model whose unrealistic independence assumptions often lead to suboptimal results for text.

The problem of discovering regularities and adapting to the test set is related to several threads of work. Slattery and Mitchell (2000) designed an iterative procedure that exploits web-specific unseen features (directory pages) in order to improve classification of web pages. However, their procedure is limited to using these very specific features and lacks a global underlying model (probabilistic or otherwise).

Our approach is also related to the transductive learning setting (Vapnik, 1995), in which the learner is presented the (unlabeled) test set together with the training data, and can attempt to optimize its performance on just the examples in the test set. Joachims (1999) defines transductive support vector machines and shows improvements over purely inductive methods for text classification. However, transductive learners assume the presence of both training and test set at training time and cannot adapt to multiple test sets. A probabilistic approach proposed by Nigam et al. (2000) uses the EM algorithm to combine labeled and unlabeled data to improve classification. However, the underlying assumption in that work is that the distribution in training and test are the same.

In general, however, the problem of heterogeneous data distributions is a complex one, and none of the work done so far provides a comprehensive solution. We propose a probabilistic framework that explicitly accommodates for the notion of different scopes with varying data distributions. We also describe a model that accounts for one source of variability between scopes — the appearance of new features. Clearly, however, this model addresses only part of the problem: Even a feature that is common in one scope can still take on a different meaning in another. Thus, we may want to relax the strict partition into local and global features, allowing the meaning of a feature to vary, to some degree, between scopes. In this case, we would need to construct both a mechanism that learns the role of a feature in one scope, but allows it to vary (to some extent) in others.

Finally, our work assumes that we are given a partition of data into scopes. In many settings, this assumption is a reasonable one. In others, however, this partition may not be known, or (as in the temporal setting) may not even be a sharp partition, but rather a gradual shift. It would be very interesting to try and detect changes in the distribution automatically, and update the model accordingly.

Acknowledgments

This work was supported by ONR Contract F3060-01-2-0564-P00002 under DARPA's EELD program.

References

Blei, D. M., Bagnell, J. A., & McCallum, A. K. (2002). Learning with scope, with application to information extraction and classification. *Proceedings for Uncertainty in Artificial Intelligence (UAI)*.

Heskes, T., & Zoeter, O. (2003). Generalized belief propagation for approximate inference in hybrid bayesian networks. *In Proc. AISTATS*.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. *Proc. ICML99* (pp. 200–209). Morgan Kaufmann Publishers, San Francisco, US.

Jordan, M. I., Ghahramani, Z., Jaakkola, T., & Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, *37*, 183–233.

Kleinberg, J. (2002). Bursty and hierarchical structure in streams. *In Proc. 8th ACM SIGKDD*.

Lerner, U., Segal, E., & Koller, D. (2001). Exact inference in networks with discrete children of continuous parents. *Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI)* (pp. 319–238).

Minka, T. (2001). *A family of algorithms for approximate bayesian inference*. Doctoral dissertation, MIT Media Lab.

Nigam, K., McCallum, A. K., Thrun, S., & Mitchell, T. M. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning*, *39*, 103–134.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems*. San Francisco: Morgan Kaufmann.

Press, W., Teukolsky, S., Vetterling, W., & Flannery, B. (1988). *Numerical recipes in C*. Cambridge University Press.

Slattery, S., & Mitchell, T. (2000). Discovering test set regularities in relational domains. *Proc. ICML00* (pp. 895–902).

Vapnik, V. (1995). *The nature of statistical learning theory*. New York, New York: Springer-Verlag.

Yedidia, J., Freeman, W., & Weiss, Y. (2000). Generalized belief propagation. *NIPS* (pp. 689–695).