Hierarchical Bayesian Domain Adaptation

Jenny Rose Finkel and Christopher D. Manning
Computer Science Department
Stanford University
Stanford, CA 94305
{jrfinkel|manning}@cs.stanford.edu

Abstract

Multi-task learning is the problem of maximizing the performance of a system across a number of related tasks. When applied to multiple domains for the same task, it is similar to domain adaptation, but symmetric, rather than limited to improving performance on a target domain. We present a more principled, better performing model for this problem, based on the use of a hierarchical Bayesian prior. Each domain has its own domain-specific parameter for each feature but, rather than a constant prior over these parameters, the model instead links them via a hierarchical Bayesian global prior. This prior encourages the features to have similar weights across domains, unless there is good evidence to the contrary. We show that the method of (Daumé III, 2007), which was presented as a simple "preprocessing step," is actually equivalent, except our representation explicitly separates hyperparameters which were tied in his work. We demonstrate that allowing different values for these hyperparameters significantly improves performance over both a strong baseline and (Daumé III, 2007) within both a conditional random field sequence model for named entity recognition and a discriminatively trained dependency parser.

1 Introduction

The goal of *multi-task learning* is to improve performance on a set of related tasks, when provided with (potentially varying quantities of) annotated data for each of the tasks. It is very closely related to *domain adaptation*, a far more common task in the natural language processing community, but with two primary differences. Firstly, in domain adaptation the different tasks are actually just different domains. Secondly, in multi-task learning the focus is on improving performance across *all* tasks, while in domain adaptation there is a distinction between *source* data and *target* data, and the goal is to improve performance on the target data. In the present work we focus on domain adaptation, but like the multi-task setting, we wish to improve performance across *all* domains and not a single *target* domains. The word *domain* is used here somewhat loosely: it may refer to a topical domain or to distinctions that linguists might term mode (speech versus writing) or register (formal written prose versus SMS communications). For example, one may have a large amount of parsed newswire, and want to use it to augment a much smaller amount of parsed e-mail, to build a higher quality parser for e-mail data. We also consider the extension to the task where the annotation is not the same, but is consistent, across domains (that is, some domains may be annotated with more information than others).

This problem is important because it is omnipresent in real life natural language processing tasks. Annotated data is expensive to produce and limited in quantity. Typically, one may begin with a considerable amount of annotated newswire data, some annotated speech data, and a little annotated e-mail data. It would be most desirable if the aggregated training data could be used to improve the performance of a system on each of these domains.

From the baseline of building separate systems for each domain, the obvious first attempt at domain adaptation is to build a system from the union of the training data, and we will refer to this as a second baseline. In this paper we propose a more principled, formal model of domain adaptation, which not only outperforms previous work, but maintains attractive performance characteristics in terms of training and testing speed. We also show that the domain adaptation work of (Daumé III, 2007), which is presented as an ad-hoc “preprocessing step,” is actually equivalent to our formal model. However, our representation of the model conceptually separates some of the hyperparameters which are not separated in (Daumé III, 2007), and we found that setting these hyperparameters with different values from one another was critical for improving performance.

We apply our model to two tasks, named entity recognition, using a linear chain conditional random field (CRF), and dependency parsing, using a discriminative, chart-based model. In both cases, we find that our model improves performance over both baselines and prior work.

2 Hierarchical Bayesian Domain Adaptation

2.1 Motivation

We call our model *hierarchical Bayesian domain adaptation*, because it makes use of a hierarchical Bayesian prior. As an example, take the case of building a logistic classifier to decide if a word is part of a person’s name. There will be a parameter (weight) for each feature, and usually there is a zero-mean Gaussian prior over the parameter values so that they don’t get too large.[1] In the standard, single-domain, case the log likelihood of the data and prior is calculated, and the optimal parameter values are found. Now, let’s extend this model to the case of two domains, one containing American newswire and the other containing British newswire. The data distributions will be similar for the two domains, but not identical. In our model, we have separate parameters for each feature in each domain. We also have a top level parameter (also to be learned) for each feature. For each domain, the Gaussian prior over the parameter values is now centered around these top level parameters instead of around zero. A zero-mean Gaussian prior is then placed over the top level parameters. In this example, if some feature, say *word=‘Nigel,’* only appears in the British newswire, the corresponding weight for the American newswire will have a similar value. This happens because the evidence in the British domain will push the British parameter to have a high value, and this will in turn influence the top-level parameter to have a high value, which will then influence the American newswire to have a high value, because there will be no evidence in the American data to override the prior. Conversely, if some feature is highly indicative of *isName=true* for the British newswire, and of *isName=false* for the American newswire, then the British parameter will have a high (positive) value while the American parameter will have a low (negative) value, because in both cases the domain-specific evidence will outweigh the effect of the prior.

2.2 Formal Model

Our domain adaptation model is based on a hierarchical Bayesian prior, through which the domain-specific parameters are tied. The model is very general-purpose, and can be applied to any discriminative learning task for which one would typically put a prior with a mean over the parameters. We will build up to it by first describing a general, single-domain, discriminative learning task, and then we will show how to modify this model to construct our hierarchical Bayesian domain adaptation model. In a typical discriminative probabilistic model, the learning process consists of optimizing the log conditional likelihood of the data with respect to the parameters, $\mathscr{L}_{orig}(\mathscr{D};\theta)$. This likelihood function can take on many forms: logistic regression, a conditional Markov model, a conditional random field, as well as others. It is common practice to put a zero-mean Gaussian prior over the parameters, leading to the following objective, for which we wish to find the optimal parameter values:

$$\arg\max_{\theta}\left(\mathscr{L}_{orig}(\mathscr{D};\theta)-\sum_i \frac{\theta_i^2}{2\sigma^2}\right) \qquad (1)$$

From a graphical models perspective, this looks like Figure 1(a), where μ is the mean for the prior (in our case, zero), σ^2 is the variance for the prior, θ are the parameters, or feature weights, and $\mathscr{D}$ is the data. Now we will extend this single-domain model into a multi-domain model (illustrated in Figure 1(b)). Each feature weight θ_i is replicated once for each domain, as well as for a top-level set of parameters. We will refer to the parameters for domain d as θ_d, with individual components $\theta_{d,i}$, the top-level parameters as θ_*, and all parameters collectively as θ. All of the power of our model stems from the relationship between these sets of param-

[1] This can be regarded as a Bayesian prior or as weight regularization; we adopt the former perspective here.

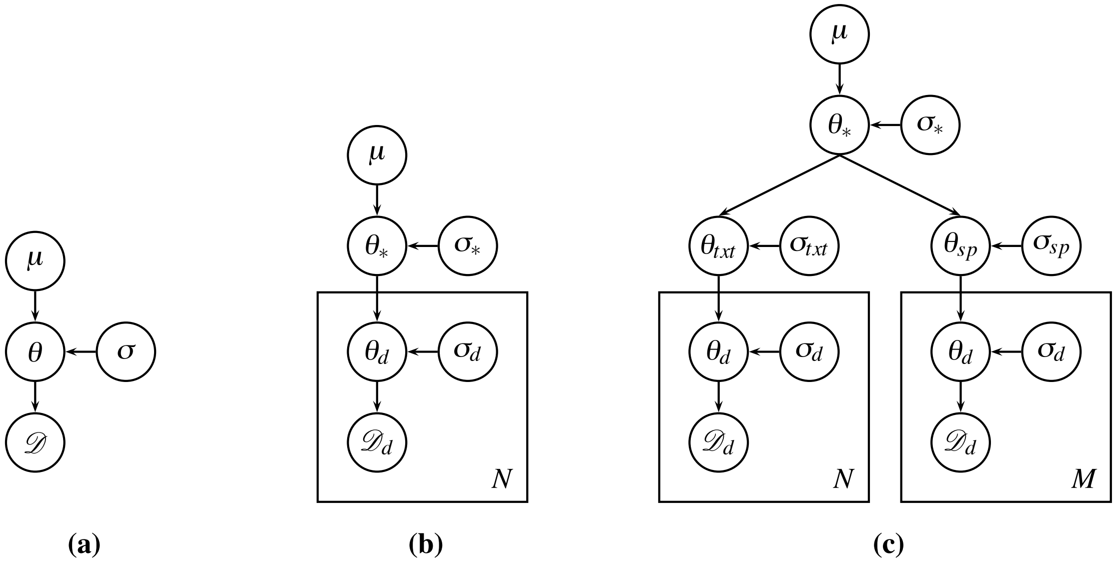

Figure 1: **(a)** No domain adaptation. The model parameters, θ, are normally distributed, with mean μ (typically zero) and variance σ^2. The likelihood of the data, $\mathscr{D}$, is dependent on the model parameters. The form of the data distribution depends on the underlying model (e.g., logistic regression, or a CRF). **(b)** Our hierarchical domain adaptation model. The top-level parameters, θ_*, are normally distributed, with mean μ (typically zero) and variance σ_*^2. There is a plate for each domain. Within each plate, the domain-specific parameters, θ_d are normally distributed, with mean θ_* and variance σ_d^2. **(c)** Our hierarchical domain adaptation model, with an extra level of structure. In this example, the domains are further split into text and speech super-domains, each of which has its own set of parameters (θ_{txt} and σ_{txt} for text and θ_{sp} and σ_{sp} for speech). θ_d is normally distributed with mean θ_{txt} if domain d is in the text super-domain, and θ_{sp} if it is in the speech super-domain.

eters. First, we place a zero-mean Gaussian prior over the top level parameters θ_*. Then, these top level parameters are used as the mean for a Gaussian prior placed over each of the domain-specific parameters θ_d. These domain-specific parameters are then the parameters used in the original conditional log likelihood functions for each domain. The domain-specific parameter values jointly influence an appropriate value for the higher-level parameters. Conversely, the higher-level parameters will largely determine the domain-specific parameters when there is little or no evidence from within a domain, but can be overriden by domain-specific evidence when it clearly goes against the general picture (for instance *Leeds* is normally a *location*, but within the *sports* domain is usually an *organization* (football team)).

The beauty of this model is that the degree of influence each domain exerts over the others, for each parameter, is based on the amount of evidence each domain has about that parameter. If a domain has a lot of evidence for a feature weight, then that evidence will outweigh the effect of the prior. However, when a domain lacks evidence for a parameter the opposite occurs, and the prior (whose value is determined by evidence in the other domains) will have a greater effect on the parameter value.

To achieve this, we modify the objective function. We now sum over the log likelihood for all domains, including a Gaussian prior for each domain, but which is now centered around θ_*, the top-level parameters. Outside of this summation, we have a Gaussian prior over the top-level parameters which is identical to the prior in the original model:

$$\mathscr{L}_{hier}(\mathscr{D};\theta) = \sum_d \left(\mathscr{L}_{orig}(\mathscr{D}_d;\theta_d) - \sum_i \frac{(\theta_{d,i} - \theta_{*,i})^2}{2\sigma_d^2} \right) - \sum_i \frac{(\theta_{*,i})^2}{2\sigma_*^2} \tag{2}$$

where σ_d^2 and σ_*^2 are variances on the priors over the parameters for all the domains, as well as the top-level parameters. The graphical models representation is shown in Figure 1(b).

One potential source of confusion is with respect to the directed or undirected nature of our domain adaptation model, and the underlying model of the data. Our hierarchical Bayesian domain adaptation model is *directed*, as illustrated in Figure 1. However, somewhat counterintuitively, the underlying (original) model of the data can be either *directed* or *undirected*, and for our experiments we use undirected, conditional random field-based models. The directed domain adaptation model can be viewed as a model of the parameters, and those parameter weights are used by the underlying data model. In Figure 1, the entire data model is represented by a single node, $\mathscr{D}$, conditioned on the parameters, θ or θ_d. The form of that model can then be almost anything, including an undirected model.

From an implementation perspective, the objective function is not much more difficult to implement than the original single-domain model. For all of our experiments, we optimized the log likelihood using L-BFGS, which requires the function value and partial derivatives of each parameter. The new partial derivatives for the domain-specific parameters (but not the top-level parameters) utilize the same partial derivatives as in the original model. The only change in the calculations is with respect to the priors. The partial derivatives for the domain-specific parameters are:

$$\frac{\partial \mathscr{L}_{hier}(\mathscr{D};\theta)}{\partial \theta_{d,i}} = \frac{\partial \mathscr{L}_{d}(\mathscr{D}_d,\theta_d)}{\partial \theta_{d,i}} - \frac{\theta_{d,i} - \theta_{*,i}}{\sigma_d^2} \qquad (3)$$

and the derivatives for the top level parameters θ_* are:

$$\frac{\partial \mathscr{L}_{hier}(\mathscr{D};\theta)}{\partial \theta_{*,i}} = \left(\sum_d \frac{\theta_{*,i} - \theta_{d,i}}{\sigma_d^2}\right) - \frac{\theta_{*,i}}{\sigma_*^2} \qquad (4)$$

This function is convex. Once the optimal parameters have been learned, the top level parameters can be discarded, since the runtime model for each domain is the same as the original (single-domain) model, parameterized by the parameters learned for that domain in the hierarchical model. However, it may be useful to retain the top-level parameters for use in adaptation to further domains in the future.

In our model there are d extra hyper-parameters which can be tuned. These are the variances σ_d^2 for each domain. When this value is large then the prior has little influence, and when set high enough will be equivalent to training each model separately. When this value is close to zero the prior has a strong influence, and when it is sufficiently close to zero then it will be equivalent to completely tying the parameters, such that $\theta_{d_1,i} = \theta_{d_2,i}$ for all domains. Despite having many more parameters, for both of the tasks on which we performed experiments, we found that our model did not take much more time to train that a baseline model trained on all of the data concatenated together.

2.3 Model Generalization

The model as presented thus far can be viewed as a two level tree, with the top-level parameters at the root, and the domain-specific ones at the leaves. However, it is straightforward to generalize the model to any tree structure. In the generalized version, the domain-specific parameters would still be at the leaves, the top-level parameters at the root, but new mid-level parameters can be added based on beliefs about how similar the various domains are. For instance, if one had four datasets, two of which contained speech data and two of which contained newswire, then it might be sensible to have two sets of mid-level parameters, one for the speech data and one for the newswire data, as illustrated in Figure 1(c). This would allow the speech domains to influence one another more than the newswire domains, and vice versa.

2.4 Formalization of (Daumé III, 2007)

As mentioned earlier, our model is equivalent to that presented in (Daumé III, 2007), and can be viewed as a formal version of his model.[2] In his presentation, the adapation is done through feature augmentation. Specifically, for each feature in the original version, a new version is created for each domain, as well as a general, domain-independent version of the feature. For each datum, two versions of each original feature are present: the version for that datum's domain, and the domain independent one.

The equivalence between the two models can be shown with simple arithmetic. Recall that the log likelihood of our model is:

$$\sum_d \left(\mathscr{L}_{orig}(\mathscr{D}_d;\theta_d) - \sum_i \frac{(\theta_{d,i} - \theta_{*,i})^2}{2\sigma_d^2}\right) - \sum_i \frac{(\theta_{*,i})^2}{2\sigma_*^2}$$

We now introduce a new variable $\psi_d = \theta_d - \theta_*$, and plug it into the equation for log likelihood:

$$\sum_d \left(\mathscr{L}_{orig}(\mathscr{D}_d;\psi_d + \theta_*) - \sum_i \frac{(\psi_{d,i})^2}{2\sigma_d^2}\right) - \sum_i \frac{(\theta_{*,i})^2}{2\sigma_*^2}$$

The result is the model of (Daumé III, 2007), where the ψ_d are the domain-specific feature weights, and θ_d are the domain-independent feature weights. In his formulation, the variances $\sigma_d^2 = \sigma_*^2$ for all domains d.

This separation of the domain-specific and independent variances was critical to our improved performance. When using a Gaussian prior there are

[2] Many thanks to David Vickrey for pointing this out to us.

two parameters set by the user: the mean, μ (usually zero), and the variance, σ^2. Technically, each of these parameters is actually a vector, with an entry for each feature, but almost always the vectors are uniform and the same parameter is used for each feature (there are exceptions, e.g. (Lee et al., 2007)). Because Daumé III (2007) views the adaptation as merely augmenting the feature space, each of his features has the same prior mean and variance, regardless of whether it is domain specific or independent. He could have set these parameters differently, but he did not.[3] In our presentation of the model, we explicitly represent different variances for each domain, as well as the top level parameters. We found that specifying different values for the domain specific versus domain independent variances significantly improved performance, though we found no gains from using different values for the different domain specific variances. The values were set based on development data.

3 Named Entity Recognition

For our first set of experiments, we used a linear-chain, conditional random field (CRF) model, trained for named entity recognition (NER). The use of CRFs for sequence modeling has become standard so we will omit the model details; good explanations can be found in a number of places (Lafferty et al., 2001; Sutton and McCallum, 2007). Our features were based on those in (Finkel et al., 2005).

3.1 Data

We used three named entity datasets, from the CoNLL 2003, MUC-6 and MUC-7 shared tasks. CoNLL is British newswire, while MUC-6 and MUC-7 are both American newswire. Arguably MUC-6 and MUC-7 should not count as separate domains, but because they were annotated separately, for different shared tasks, we chose to treat them as such, and feel that our experimental results justify the distinction. We used the standard train and test sets for each domain, which for CoNLL corresponds to the (more difficult) testb set. For details about the number of training and test words in each dataset, please see Table 1.

	# Train Words	# Test Words
MUC-6	165,082	15,032
MUC-7	89,644	64,490
CoNLL	203,261	46,435

Table 1: Number of words in the training and test sets for each of the named entity recognition datasets.

One interesting challenge in dealing with both CoNLL and MUC data is that the label sets differ. CoNLL has four classes: *person*, *organization*, *location*, and *misc*. MUC data has seven classes: *person*, *organization*, *location*, *percent*, *date*, *time*, and *money*. They overlap in the three core classes (*person*, *organization*, and *location*), but CoNLL has one additional class and MUC has four additional classes.

The differences in the label sets led us to perform two sets of experiments for the baseline and hierarchical Bayesian models. In the first set of experiments, at training time, the model allows any label from the union of the label sets, regardless of whether that label was legal for the domain. At test time, we would ignore guesses made by the model which were inconsistent with the allowed labels for that domain.[4] In the second set of experiments, we restricted the model at training time to only allow legal labels for each domain. At test time, the domain was specified, and the model was once again restricted so that words would never be tagged with a label outside of that domain's label set.

3.2 Experimental Results and Discussion

In our experiments, we compared our model to several strong baselines, and the full set of results is in Table 2. The models we used were:

TARGET ONLY. Trained and tested on only the data for that domain.

ALL DATA. Trained and tested on data from all domains, concatenated into one large dataset.

ALL DATA*. Same as ALL DATA, but restricted possible labels for each word based on domain.

DAUME07. Trained and tested using the same technique as (Daumé III, 2007). We note that they present results using per-token label accuracy, while we used the more standard entity precision, recall, and F score (as in the CoNLL 2003 shared task).

[3] Although he alludes to the potential for something similar in the last section of his paper, when discussing the kernelization interpretation of his approach.

[4] We treated them identically to the background symbol. So, for instance, labelling a word a *date* in the CoNLL data had no effect on the score.

Named Entity Recognition

Model	Precision	Recall	F1
	MUC-6		
Target Only	86.74	80.10	83.29
All Data*	85.04	83.49	84.26
All Data	86.00	82.71	84.32
Daume07*	87.83	83.41	85.56
Daume07	87.81	82.23	85.46
Hier Bayes*	88.59	84.97	86.74
Hier Bayes	**88.77**	**85.14**	**86.92**
	MUC-7		
Target Only	81.17	70.23	75.30
All Data*	81.66	76.17	78.82
All Data	82.20	70.91	76.14
Daume07*	**83.33**	75.42	79.18
Daume07	83.51	75.63	79.37
Hier Bayes*	82.90	76.95	79.82
Hier Bayes	83.17	**77.02**	**79.98**
	CoNLL		
Target Only	85.55	84.72	85.13
All Data*	86.34	84.45	85.38
All Data	86.58	83.90	85.22
Daume07*	86.09	85.06	85.57
Daume07	86.35	**85.26**	**85.80**
Hier Bayes*	86.33	85.06	85.69
Hier Bayes	**86.51**	85.13	**85.81**

Table 2: Named entity recognition results for each of the models. With the exception of the Target Only model, all three datasets were combined when training each of the models.

Daume07*. Same as Daume07, but restricted possible labels for each word based on domain.

Hier Bayes. Our hierarchical Bayesian domain adaptation model.

Hier Bayes*. Same as Hier Bayes, but restricted possible labels for each word based on the domain.

For all of the baseline models, and for the top level-parameters in the hierarchical Bayesian model, we used $\sigma = 1$. For the domain-specific parameters, we used $\sigma_d = 0.1$ for all domains.

The Hier Bayes model outperformed all baselines for both of the MUC datasets, and tied with the Daume07 for CoNLL. The largest improvement was on MUC-6, where Hier Bayes outperformed Daume07*, the second best model, by 1.36%. This improvement is greater than the improvement made by that model over the All Data* baseline. To assess significance we used a document-level paired t-test (over all of the data combined), and found that Hier Bayes significantly outperformed all of the baselines (not including Hier Bayes*) with greater than 95% confidence.

For both the Hier Bayes and Daume07 models, we found that performance was better for the variant which did not restrict possible labels based on the domain, while the All Data model did benefit from the label restriction. For Hier Bayes and Daume07, this result may be due to the structure of the models. Because both models have domain-specific features, the models likely learned that these labels were never actually allowed. However, when a feature does not occur in the data for a particular domain, then the domain-specific parameter for that feature will have positive weight due to evidence present in the other domains, which at test time can lead to assigning an illegal label to a word. This information that a word may be of some other (unknown to that domain) entity type may help prevent the model from mislabeling the word. For example, in CoNLL, nationalities, such as *Iraqi* and *American*, are labeled as *misc*. If a previously unseen nationality is encountered in the MUC testing data, the MUC model may be tempted to label is as a *location*, but this evidence from the CoNLL data may prevent that, by causing it to instead be labeled *misc*, a label which will subsequently be ignored.

In typical domain adaptation work, showing gains is made easier by the fact that the amount of training data in the *target* domain is comparatively small. Within the multi-task learning setting, it is more challenging to show gains over the All Data baseline. Nevertheless, our results show that, so long as the amount of data in each domain is not widely disparate, it is possible to achieve gains on all of the domains simultaneously.

4 Dependency Parsing

4.1 Parsing Model

We also tested our model on an untyped dependency parsing task, to see how it performs on a more structurally complex task than sequence modeling. To our knowledge, the discriminatively trained dependency model we used has not been previously published, but it is very similar to recent work on discriminative constituency parsing (Finkel et al., 2008). Due to space restrictions, we cannot give a complete treatment of the model, but will give an overview.

We built a CRF-based model, optimizing the likelihood of the parse, conditioned on the words and parts of speech of the sentence. At the heart of our model is the Eisner dependency grammar chart-parsing algorithm (Eisner, 1996), which allows for efficient computation of inside and outside scores. The Eisner algorithm, originally designed for generative parsing, decomposes the probability of a dependency parse into the probabilities of each attachment of a dependent to its parent, and the probabilities of each parent stopping taking dependents. These probabilities can be conditioned on the child, parent, and direction of the dependency. We used a slight modification of the algorithm which allows each probability to also be conditioned on whether there is a previous dependent. While the unmodified version of the algorithm includes stopping probabilities, conditioned on the parent and direction, they have no impact on which parse for a particular sentence is most likely, because all words must eventually stop taking dependents. However, in the modified version, the stopping probability is also conditioned on whether or not there is a previous dependent, so this probability does make a difference.

While the Eisner algorithm computes locally normalized probabilities for each attachment decision, our model computes unnormalized scores. From a graphical models perspective, our parsing model is undirected, while the original model is directed.[5] The score for a particular tree decomposes the same way in our model as in the original Eisner model, but it is globally normalized instead of locally normalized. Using the inside and outside scores we can compute partial derivatives for the feature weights, as well as the value of the normalizing constant needed to determine the probability of a particular parse. This is done in a manner completely analogous to (Finkel et al., 2008). Partial derivatives and the function value are all that is needed to find the optimal feature weights using L-BFGS.[6]

Features are computed over each attachment and stopping decision, and can be conditioned on the parent, dependent (or none, if it is a stopping decision), direction of attachment, whether there is a previous dependent in that direction, and the words and parts of speech of the sentence. We used the same features as (McDonald et al., 2005), augmented with information about whether or not a dependent is the first dependent (information they did not have).

4.2 Data

For our dependency parsing experiments, we used LDC2008T04 OntoNotes Release 2.0 data (Hovy et al., 2006). This dataset is still in development, and includes data from seven different domains, labeled for a number of tasks, including PCFG trees. The domains span both newswire and speech from multiple sources. We converted the PCFG trees into dependency trees using the Collins head rules (Collins, 2003). We also omitted the WSJ portion of the data, because it follows a different annotation scheme from the other domains.[7] For each of the remaining six domains, we aimed for an 75/25 data split, but because we divided the data using the provided sections, this split was fairly rough. The number of training and test sentences for each domain are specified in the Table 3, along with our results.

4.3 Experimental Results and Discussion

We compared the same four domain adaptation models for dependency parsing as we did for the named entity experiments, once again setting $\sigma = 1.0$ and $\sigma_d = 0.1$. Unlike the named entity experiments however, there were no label set discrepencies between the domains, so only one version of each domain adaptation model was necessary, instead of the two versions in that section.

Our full dependency parsing results can be found in Table 3. Firstly, we found that DAUME07, which had outperformed the ALL DATA baseline for the sequence modeling task, performed worse than the

[5] The dependencies themselves are still *directed* in both cases, it is just the underlying graphical model used to compute the likelihood of a parse which changes from a directed model to an undirected model.

[6] In (Finkel et al., 2008) we used stochastic gradient descent to optimize our weights because our function evaluation was too slow to use L-BFGS. We did not encounter this problem in this setting.

[7] Specifically, all the other domains use the "new" Penn Treebank annotation style, whereas the WSJ data is still in the "traditional" annotation style, familiar from the past decade's work in Penn Treebank parsing. The major changes are in hyphenation and NP structure. In the new annotation style, many hyphenated words are separated into multiple tokens, with a new part-of-speech tag given to the hyphens, and leftward-branching structure inside noun phrases is indicated by use of a new NML phrasal category. The treatment of hyphenated words, in particular, makes the two annotation styles inconsistent, and so we could not work with all the data together.

Dependency Parsing

	Training		Testing		TARGET	ALL		HIER
	Range	# Sent	Range	# Sent	ONLY	DATA	DAUME07	BAYES
ABC	0–55	1195	56–69	199	83.32%	**88.97%**	87.30%	88.68%
CNN	0–375	5092	376–437	1521	85.53%	87.09%	86.41%	**87.26%**
MNB	0–17	509	18–25	245	77.06%	86.41%	84.70%	**86.71%**
NBC	0–29	552	30–39	149	76.21%	**85.82%**	85.01%	85.32%
PRI	0–89	1707	90–112	394	87.65%	90.28%	89.52%	**90.59%**
VOA	0–198	1512	199–264	383	89.17%	**92.11%**	90.67%	**92.09%**

Table 3: Dependency parsing results for each of the domain adaptation models. Performance is measured as unlabeled attachment accuracy.

baseline here, indicating that the transfer of information between domains in the more structurally complicated task is inherently more difficult. Our model's gains over the ALL DATA baseline are quite small, but we tested their significance using a sentence-level paired t-test (over all of the data combined) and found them to be significant at $p < 10^{-5}$. We are unsure why some domains improved while others did not. It is not simply a consequence of training set size, but may be due to qualities of the domains themselves.

5 Related Work

We already discussed the relation of our work to (Daumé III, 2007) in Section 2.4. Another piece of similar work is (Chelba and Acero, 2004), who also modify their prior. Their work is limited to two domains, a source and a target, and their algorithm has a two stage process: First, train a classifier on the source data, and then use the learned weights from that classifier as the mean for a Gaussian prior when training a new model on just the target data.

Daumé III and Marcu (2006) also took a Bayesian approach to domain adaptation, but structured their model in a very different way. In their model, it is assumed that each datum within a domain is either a domain-specific datum, or a general datum, and then domain-specific and general weights were learned. Whether each datum is domain-specific or general is not known, so they developed an EM based algorithm for determining this information while simultaneously learning the feature weights. Their model had good performance, but came with a 10 to 15 times slowdown at training time. Our slowest dependency parser took four days to train, making this model close to infeasible for learning on that data.

Outside of the NLP community there has been much similar work making use of hierarchical Bayesian priors to tie parameters across multiple, similar tasks. Evgeniou et al. (2005) present a similar model, but based on support vector machines, to predict the exam scores of students. Elidan et al. (2008) make us of an *undirected* Bayesian transfer hierarchy to jointly model the shapes of different mammals. The complete literature on related multi-task learning is too large to fully discuss here, but we direct the reader to (Baxter, 1997; Caruana, 1997; Yu et al., 2005; Xue et al., 2007). For a more general discussion of hierarchical priors, we recommend Chapter 5 of (Gelman et al., 2003) and Chapter 12 of (Gelman and Hill, 2006).

6 Conclusion and Future Work

In this paper we presented a new model for domain adaptation, based on a hierarchical Bayesian prior, which allows information to be shared between domains when information is sparse, while still allowing the data from a particular domain to override the information from other domains when there is sufficient evidence. We outperformed previous work on a sequence modeling task, and showed improvements on dependency parsing, a structurally more complex problem, where previous work failed. Our model is practically useful and does not require significantly more time to train than a baseline model using the same data (though it does require more memory, proportional to the number of domains). In the future we would like to see if the model could be adapted to improve performance on data from a new domain, potentially by using the top-level weights which should be less domain-dependent.

Acknowledgements

The first author is supported by a Stanford Graduate Fellowship. We also thank David Vickrey for his helpful comments and observations.

References

J. Baxter. 1997. A bayesian/information theoretic model of learning to learn via multiple task sampling. In *Machine Learning*, volume 28.

R. Caruana. 1997. Multitask learning. In *Machine Learning*, volume 28.

Ciprian Chelba and Alex Acero. 2004. Adaptation of a maximum entropy capitalizer: Little data can help a lot. In *EMNLP 2004*.

M. Collins. 2003. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637.

Hal Daumé III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Conference of the Association for Computational Linguistics (ACL)*, Prague, Czech Republic.

Jason Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, Copenhagen.

Gal Elidan, Benjamin Packer, Geremy Heitz, and Daphne Koller. 2008. Convex point estimation using undirected bayesian transfer hierarchies. In *UAI 2008*.

T. Evgeniou, C. Micchelli, and M. Pontil. 2005. Learning multiple tasks with kernel methods. In *Journal of Machine Learning Research*.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL 2005*.

Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. 2008. Efficient, feature-based conditional random field parsing. In *ACL/HLT-2008*.

Andrew Gelman and Jennifer Hill. 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

A. Gelman, J. B. Carlin, H. S. Stern, and Donald D. B. Rubin. 2003. *Bayesian Data Analysis*. Chapman & Hall.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: The 90% solution. In *HLT-NAACL 2006*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML 2001*.

Su-In Lee, Vassil Chatalbashev, David Vickrey, and Daphne Koller. 2007. Learning a meta-level prior for feature relevance from multiple related tasks. In *ICML '07: Proceedings of the 24th international conference on Machine learning*.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *ACL 2005*.

Charles Sutton and Andrew McCallum. 2007. An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press.

Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. 2007. Multi-task learning for classification with dirichlet process priors. *J. Mach. Learn. Res.*, 8.

Kai Yu, Volker Tresp, and Anton Schwaighofer. 2005. Learning gaussian processes from multiple tasks. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*.