A New Framework for Domain Adaptation without Model Retraining

Gourab Kundu[1] and **Ming-wei Chang**[2] and **Dan Roth**[1] and **Chenxiang Zhai**[1]
[1]University of Illinois at Urbana Champaign
Urbana, IL 61801
{kundu2, danr, czhai@illinois.edu}
[2]Microsoft Research, Redmond, WA
minchang@microsoft.com

Abstract

We propose a principled and effective domain adaptation framework that pursues the goal of Open Domain NLP (train once, test anywhere). Most domain adaptation frameworks *adapt the models* trained on the source domain data by retraining it on target domains (with a mix of labeled and unlabeled data). However, it is time consuming to retrain big models or pipeline systems, and may not even be feasible if you consider a streaming data that may not be coherent (e.g., web data). We propose an adaptation framework that does not require retraining the original model. Instead, our approach *adapts the target domain input* so that it is more similar to the source domain, while preserving the labeling, thus increasing the accuracy of the original model when evaluated on target data. Our experiments on the named entity recognition task in scientific domains show an absolute F1 improvement of 13% over a state-of-the-art named entity recognizer. We also show that without any retraining, the proposed method outperforms the bootstrapping based adaptation method of (Jiang and Zhai, 2007b) that requires multiple rounds of retraining on the target domain data.

1 Introduction

In several NLP tasks, systems trained on annotated text from one domain perform well when tested on text from the same domain but adapt poorly to other domains. For example, in the task of POS tagging, performance drops almost 9% when systems trained on Wall Street Journal domain are tested on the biomedical domain (Blitzer et al., 2006).

Most domain adaptation methods proposed in the literature need to build a new model for every new domain. The works of (Daumé III, 2007; Finkel and Manning, 2009; Jiang and Zhai, 2007a) require some labeled data in the target domain. Using this small amount of labeled data together with a large amount of labeled data from the source domain, they build a new model for the target domain. But labeled data may not exist for every domain. The works of (Blitzer et al., 2006; Glorot et al., 2011; Huang and Yates, 2009) do not require labeled data in the target domain. Instead, they use unlabeled data from the source and the target domain to learn a shared common representation. Using features from this representation, they train a new model for the target domain.

Most existing adaptation systems are not *Open Domain Systems* in the sense that they need to build a new model for every new domain. This may be difficult in NLP, where a lot of the advanced systems are *pipeline systems*. Systems like semantic role labeling (SRL) use a syntactic parser or a shallow parser, a POS tagger and then the SRL model itself, and all components need to be retrained. Many NLP systems rely on *off-the-shelf tools* that may not come with an option for retraining with unlabeled data from the target domain. Decoupling the adaptation module from the trained system is advantageous since it allows the system developer to treat the trained system as a *black box* and provides the flexibility to use one's adaptation module or incorporate target domain specific prior knowledge.

Training a model for every new domain becomes challenging also when there are *many* diverse target domains, e.g., web data. There are scenarios when we have *very little data* from the target domain or data come in a *stream* instead of large batch. For example, in online demo systems, users give one sentence at a time for annotation with an NLP tool. Retraining the model for every new sentence may not be a feasible solution.

There have been recent attempts to study adaptation without retraining (Kundu and Roth, 2011; Kundu et al., 2011). However, while the general direction is promising, the proposed methods are somewhat ad hoc and are limited since they require external resources like WordNet, VerbNet or declarative prior knowledge on the target domain, which may not be available for many domains. Moreover, most target domains come with plenty of unlabeled data which these methods cannot take advantage of.

The key contribution of this paper is a significant extension of the *adaptation without retraining* framework. We propose a new method called *DAT (Domain Adaptation by Translation)*. The key step in this adaptation method is a machine translation step – we *translate* each sentence from the target domain to produce a new sentence that has, in principle, identical labeling but that is more like a sentence taken from the source domain. Consequently, it is expected that the model trained on the source domain labeled data will make better predictions on it than on the original target sentences. Since this framework adapts the text instead of the model, it can use the same model across all domains.

In this paper, we report experiments on the task of named entity recognition in biological text. Here the entities of interest are names of genes, proteins etc. There is a growing body of literature on analyzing scientific text to extract information useful for scientists (Kim et al., 2011; Blitzer et al., 2006). This domain is very important and it is exactly where many NLP tools fail, since it is different enough from the standard domains NLP researchers train models on. Among NLP tasks on scientific data we have chosen the NER task, since it is a fundamental task in NLP, is crucially important in the scientific domain data, and has been studied, so we can compare to existing work (Jiang and Zhai, 2006; Jiang and Zhai, 2007b).

2 Motivating Examples

One of the key reasons for performance degradation of an NLP tool is the widely varying distribution of features between the source and the target domain. Many features such as words/bigrams that are frequent in the target domain, may be infrequent or even non-existent in the source domain. We observed that if a less frequent word in the source domain is replaced by a more frequent word in a way that does not affect the labeling of the sentence, tools trained on the source domain labeled data perform better. For example, consider the following sentence provided as input to the Charniak parser:

> *He finished the worship service as if there had been no brazen attempt to dishonor God and man .*

The word *brazen* only appears once (as a verb) in the Wall Street Journal corpus which causes the Charniak parser to make a wrong tagging decision, resulting subsequently by incorrect attachment decisions. However, once we replace the word *brazen* with *bold* which appears often in training data and is a synonym of *brazen*, the parse gets corrected. Figure 1 shows the original and the corrected parse trees.

For the task of named entity recognition in bio domain, a system trained on annotated data from one specie does not perform well on identifying names from text corresponding to other species. Although the context in which the names appear remains very similar across domains, the patterns of the names vary significantly across domains. The reason is largely that different naming strategies have been adopted for naming of genes across different species.

In Table 1, *ptuf* and *PTP-BL* are the names of two genes in the source and the target domain respectively. In snippet II, the context word *gene* provides a strong clue for *PTP-BL* to be a gene. But the context words in snippet I and III are uninformative for tagging *PTP-BL* as a gene. In all three snipptes, name pattern features like capitalization, hyphenation etc. differ significantly from *PTP-BL* to *ptuf*. So it is difficult for the classifier trained on text containing gene names like *ptuf*, *atonal* etc. to tag *PTP-BL* as a gene specially in snippets I and III. However,

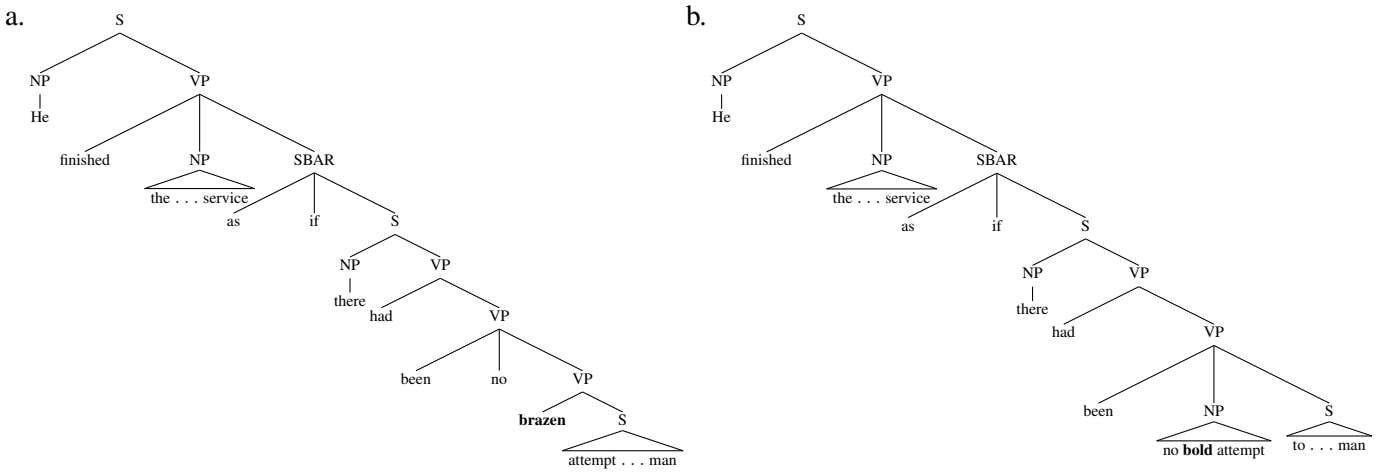

Figure 1: a. Original parse tree b. Corrected parse tree after replacement of infrequent word **brazen** by **bold**

No.	*Target domain Text*	*Translations From Target Domain to Source Domain*
I	**PTP-BL** shares some intriguing ...	**ptuf** shares some intriguing ...
II	... show that the **PTP-BL** gene is expressed ...	... show that the **ptuf** gene is expressed ...
III	... localization for **PTP-BL** in epithelial ...	... localization for **ptuf** in epithelial ...

Table 1: Example Translations. The word **PTP-BL** in original sentences is replaced by **ptuf** in the translated sentences.

in the translated text, the classifier will make the correct predictions in all three snippets since it saw *ptuf* as a gene name in the source domain.

To achieve the translations in Table 1, the translation table needs to contain a word translation pair (*ptuf*, *PTP-BL*). Table 2 shows some contexts in which *ptuf* and *PTP-BL* appear in unlabeled data. The similarity of the contexts can help us to find this word translation pair.

Source domain Snippets	*Target domain Snippets*
.. characterized a novel **gene, patufet (ptuf)** ..	.. show that **the PTP-BL gene** is expressed ..
.. **sequencing of ptuf** shows that ..	.. the **sequencing of PTP-BL** is ..

Table 2: The bolded text indicates the similar contexts for ptuf and PTP-BL

While this is a very simple example and our system will be able to "translate" sentences beyond these simple substitutions, this example serves to illustrate and motivate the approach that we formally introduce next.

3 Adaptation by Translation

Assume that w_s is a model built using source domain labeled data and $\Phi(.)$ is the corresponding feature function. At test time, we find the most probable output by using w_s. The reason we need adaptation algorithm is because a target domain sentence x might not be suitable for the model w_s.

The high-level overview of our framework is listed in Algorithm 1. We list some of the important properties in the following:

- The model w_s need not be retrained in the algorithm.
- The translation model only needs to be built once for a new target domain. (Line 1)
- For a new sentence at test time, we will use the translation model to generate a new sentence (Line 3) and use w_s to generate the final output.

In the following, we explain in details our current implementations for building a translation model (Line 1) in Section 3.1. We describe how to generate the final sentence (Line 3) in Section 3.2.

Algorithm 1 Adaptation by Translation Algorithm

Require: $\mathbf{w}_s$: model built on source domain labeled data,
$\Phi(.)$: feature function used when training $\mathbf{w}_s$
$\mathcal{D}_s$: snippets from unlabeled source domain data,
$\mathcal{D}_t$: snippets from unlabeled target domain data

1: Build a translation model M with $(\mathcal{D}_s, \mathcal{D}_t)$
2: **for** each sentence x_t in the target domain **do**
3: $\hat{x}_t \leftarrow M(x_t)$ {Generate the translation}
4: Output the prediction with

$$\arg\max_y \mathbf{w}_s^T \Phi(\hat{x}_t, y)$$

5: **end for**

3.1 Word-based Snippet Translation Model

We extract snippets of a fixed length from each sentence in the unlabeled data of the source and the target domain to create $\mathcal{D}_s$ and $\mathcal{D}_t$. Since $(\mathcal{D}_s, \mathcal{D}_t)$ is not parallel (i.e. which snippet in $\mathcal{D}_t$ is a translation of which snippet in $\mathcal{D}_s$ is not available), we need to figure out the *snippet alignment* from $\mathcal{D}_s$ to $\mathcal{D}_t$ and the word translation table at the same time. The word translation table is a table of entries (s, t, p) meaning that with probability p, a source domain word s will be translated to a target domain word t.

For a given k-word snippet T from the target domain, we would like to find a corresponding translated snippet S in the source domain:

$$\arg\max_S P(S|T) = \arg\max_S P(S)P(T|S).$$

$P(S)$ can be easily estimated using a language model over the source domain unlabeled data.

$$P(T|S) = \prod_{l=1}^{k} P(t^l|s^l).$$

where t^l and s^l represent the l-th word of T and S, respectively. In analogy of IBM Translation models (Brown et al., 1993), the only allowable *word alignment* across two snippets (S, T) is the alignment where s_l aligns with t_l for $l = 1, \ldots, k$. Since both $\mathcal{D}_s$ and $\mathcal{D}_t$ come from English, we can ignore other word alignments considered in MT since their purpose is to model the word re-orderings that occur across different languages.

Given that $(\mathcal{D}_s, \mathcal{D}_t)$ is not parallel, our objective function becomes

$$\max_\theta \log P(\mathcal{D}_t|\mathcal{D}_s, \theta).$$

where θ represents the parameters for word translation probability $P(t|s)$.

Let n_t and n_s denote the number of snippets in $\mathcal{D}_t$ and $\mathcal{D}_s$ respectively. T_i and S_j denote the ith and jth snippet in $\mathcal{D}_t$ and $\mathcal{D}_s$ respectively.

$$\log P(\mathcal{D}_t|\mathcal{D}_s, \theta) = \sum_{i=1}^{n_t} \log P(T_i|\mathcal{D}_s, \theta)$$
$$= \sum_{i=1}^{n_t} \log \sum_{j=1}^{n_s} P(T_i, A_{ij}|\mathcal{D}_s, \theta)$$
$$= \sum_{i=1}^{n_t} \log \sum_{j=1}^{n_s} P(T_i|\mathcal{D}_s, A_{ij}, \theta) P(A_{ij}|\mathcal{D}_s, \theta).$$

where A_{ij} indicates that T_i and S_j are aligned. We specify $P(A_{ij}|\mathcal{D}_s, \theta)$ as a uniform distribution and solve the EM algorithm.

The E-step is:

$$\alpha_{ij} = P(A_{ij}|T_i, S_j, \theta) \propto P(T_i|S_j, \theta).$$

The M-step is:

$$P(t|s) = \frac{\sum_{i,j} \alpha_{ij} (\sum_{d=1}^{k} [T_i^d == t][S_j^d == s])}{\sum_{i,j} \alpha_{ij} (\sum_{d=1}^{k} [S_j^d == s])}.$$

E-step calculates the probability of alignment for each snippet in $\mathcal{D}_s$ to each snippet in $\mathcal{D}_t$ assuming a fixed θ. M-step calculates θ using the fixed probabilities of alignment from D_s to $\mathcal{D}_t$.

To make the translation model label preserving such that the translated sentences will have identical labeling as the input sentence, we use pivot words and snippet pruning.

Pivot Words We want to use the wisdom that those words that occur in similar contexts are similar themselves. For the setting of domain adaptation, if one source domain-specific word and one target domain-specific word occur in similar contexts, they can form a potential word translation pair. So we need a translation model that will translate each of the words occurring in both source and target domain to itself. The hope is that by keeping

these translations fixed, we can find proper translation pairings for the domain specific words. In this work, we take all the common words in both domains as pivot words.

Snippet Pruning If a snippet $S \in \mathcal{D}_s \cup \mathcal{D}_t$ does not have at least one domain-specific word, it is pruned from both $\mathcal{D}_s$ and $\mathcal{D}_t$.

If a snippet $S \in \mathcal{D}_s$ contains a source domain specific word s with label l and all pivot words in S are weakly correlated with l, S can lead to errors in estimating the translation table. A word is weakly correlated with a label l if it is equally likely to occur in the contexts of words with label l and contexts of words with labels other than l. As for example, Let S = ***ptuf is a***, where $S \in \mathcal{D}_s$, *ptuf* is a source domain specific word with the label *Gene*. The pivot words ***is,a*** in S are weakly correlated with the label *Gene* since they can occur in the context of both gene and non-gene entities. Let a snippet T = ***Mouse is a*** where $T \in \mathcal{D}_t$ and *Mouse* is a target domain specific word with the label *O* indicating non-gene. Now aligning S with T will create the word translation pair (ptuf, Mouse) which is not label preserving. To eliminate snippets like S, we use a scoring function, Score: Snippet⇒R, and remove all snippets from $\mathcal{D}_s$ except the top-k highest scoring snippets.

$Score_l(S) = \sum_{w \in S \text{ and } w \in C} MI_l(w)$

Here S is a snippet, C is the set of pivot words, l is a label and $MI_l(w)$ is the mutual information of word w with respect to l calculated as the fraction of times w appears in a 2 word window of a word with label l.

3.2 Decoding

At test time, given a target domain sentence T, we find the sentence S that maximizes $P(T|S)P(S)$. If some target domain specific words do not appear in any word translation pair in the translation table, these words remain unchanged.

4 Experimental Setup

In this section, we discuss our experimental setups. The data we used is from BioCreAtIvE Task 1B (Hirschman et al., 2005). This data set contains three subsets of MEDLINE abstracts with gene and protein names from three species (fly, mouse, and yeast). We used the data set from (Jiang and Zhai, 2006). The original BioCreAtIvE 1B data did not have direct annotations. Instead for each abstract, they provided a list of entities that were mentioned in the abstract. A synonym list was also given for each species. The authors in (Jiang and Zhai, 2006) used a string matching method with relaxation to tag the mentions in the abstracts using the list of names. They took 7500 sentences from each specie for their experiments, where half of the sentences contain named entities. They further split the 7500 sentences of each specie into two sections, section A of 5000 sentences and section B of 2500 sentences.

5 Results

Single Source Domain In these experiments, we want to evaluate our method in typical adaptation scenarios, where we train on one domain and test on a different domain. Each time we train on the section A of one specie and test it on the section A data of the remaining two species. This gives us six adaptation experiments.

For training, we used JLIS (Chang et al., 2010b) to train a structured SVM with sequential tagging. We set C = 1.0 while learning the structured SVM classifier. The features used were adopted from a system (Finkel et al., 2005) that achieved the state-of-the-art in gene recognition in BioCreative Evaluation. The same system was adopted in (Jiang and Zhai, 2006; Jiang and Zhai, 2007b). The features used mostly were shape features capturing patterns like capitalization, number etc., lexical features in a window, POS tags, abbreviations, suffix or prefix features etc. For details, please refer to (Finkel et al., 2005).

Taking the section A data of one specie as source domain and section A data of another specie as the target domain, a translation model is learned. Note that our estimation of the translation model is unsupervised and so it does not use the gold labels from section A of the target domain. We set snippet length = 3. We also keep 1 word left and right context for each snippet. A snippet in the target domain can only align to those snippets in the source domain that have identical left and right contexts as the current snippet. For all experiments, we use a trigram language model from the section A data of

the source domain. We run EM for 15 iterations. We used a beam search based decoder of beam size 25. We tune top-k by doing 5 fold cross validation over the source domain training data. In this process of cross validation, test set itself is a portion of the original training data. Since the test set comes from the same domain as the training set, the classifier tends to be very accurate and there is less need for translation, so the value of top-k that gives optimal results in this cross validation is very small. We scale top-k by a factor of 2 for out-of-domain experiments and this works well in all six experiments in Table 3.

Source	Target	bl-P	DAT-P	bl-R	DAT-R	bl-F1	DAT-F1
F	M	44.1	**45.9**	23.4	**26.3**	30.6	**33.4**
F	Y	47.9	**50.0**	32.2	**35.2**	38.5	**41.3**
M	F	66.5	**73.1**	10.8	**28.8**	18.6	**41.4**
M	Y	**72.4**	65.9	39.2	**48.2**	50.8	**55.7**
Y	F	50.2	**63.7**	5.8	**24.9**	10.4	**35.8**
Y	M	56.3	**60.5**	20.2	**36.9**	29.7	**45.9**
Avg.		56.2	**59.9**	21.9	**33.4**	29.8	**42.3**

Table 3: Results on adaptation experiments from a single source domain to a single target domain. bl-P = Precision of baseline, bl-R = Recall of baseline, bl-F1 = F1 score of baseline, DAT-P = Precision of DAT, DAT-R = Recall of DAT, DAT-F1 = F1 of DAT, F = fly data set, M = mouse data set, Y = yeast data set. DAT improves precision over baseline on 5 out of 6 experiments. DAT improves both recall and F1 on all 6 experiments. On average, DAT improves F1 13% over baseline.

The baseline in Table 3 is direct application of the state-of-the-art system (Finkel et al., 2005) (trained on the source domain) tested on the target domain without any adaptation. Note that DAT translates the target domain text and then applies the same model as the baseline. We see that DAT outperforms the baseline model on average by 13%. Out of 6 cases, precision improved in 5 cases. In all 6 cases, recall improved.

The results for in-domain experiments are reported in Table 4. The classifier was trained from the section A of one specie and tested on the section B from that same specie. These numbers can be regarded as the upper bounds for adaptation methods. The results in Table 4 show that there is still much room to further improve the adaptation accuracy.

Multiple Source Domains In this section, we conduct experiments similar to (Jiang and Zhai, 2007b) and compare to their results. In (Jiang and Zhai, 2007b), the focus was to identify the generalizable features from the labeled data of multiple domains. To this goal, they trained on the combined section A data (10000 sentences) from two domains and tested it on section B of a third domain (2500 sentences). We conduct experiments in similar settings and report our results in Table 5. In each of these experiments, we estimate a translation model from the 10000 sentences of the two source domains and 7500 sentences of the target domain. For tuning top-k, we take advantage of the training data from two different domains, we train on each of the two domains and test on the other and select the value of top-k that gives the best average performance.

Source	Target	bl-P	bl-R	bl-F1
fly	fly	79.6	59.6	68.2
mouse	mouse	69.2	42.2	52.5
yeast	yeast	85.2	86.0	85.6

Table 4: Results of applying the baseline model on a test set that is from the same domain as the training set. bl-P = Precision of baseline, bl-R = Recall of baseline, bl-F1 = F1 score of baseline

The authors in (Jiang and Zhai, 2007b) presented a two stage approach for domain adaptation. The goal of the first stage was to learn a better classifier from the training data of several domains. In the first stage, they had three methods: BL, DA-1 and DA-2. BL was the baseline model learned from the combined training data of the two source domains, neglecting the domain difference similar to ours. DA-1 and DA-2 involved specialized methods to generalize from the training data of the two source domains in order to learn a generalized classifier that will perform well on the third (target) domain. The number of top features h was one of the parameters of BL, DA-1 and DA-2. In (Jiang and Zhai, 2007b), the authors evaluated BL, DA-1 and DA-2 for different values of h over the target domain test data. Since they did not outline any process for selecting the optimal h without evaluating on the test data, we compare with the performance of BL, DA-1 and DA-2 for the case where h is set to the total number of features. Since DA-1 has better performance than BL and DA-2 on all three experiments, we report only the results for DA-1. From Table 5, we see that in

2 of 3 experiments (mouse+yeast to fly & fly+yeast to mouse), DAT outperforms DA-1 significantly. On average, the improvement of DAT over DA-1 is 7%. DA-1 does not look into the target domain data. So we believe the performance improvement of DA-1 and DAT should be orthogonal and combining them should give further improvements. It is also worth noting that our baseline system gives similar performance to the baseline system of (Jiang and Zhai, 2007b).

In (Jiang and Zhai, 2007b), the goal of the second stage was to make better use of the target domain unlabeled data. The authors used semi-supervised learning (SSL) over target domain unlabeled data. They reported results for three methods: BL-SSL, BL-SSL-2 and DA-2-SSL. For BL-SSL and DA-2-SSL, the base model used to start the bootstrapping process was respectively the baseline model and DA-2 model from the first stage for *optimal value of* h obtained by running across test data as discussed before. So our results are not comparable to the results of BL-SSL and DA-2-SSL since we do not make use of the gold labels in the test data in any way. For BL-SSL-2, h was set to the total number of features and so we compare our results with it. From Table 5, we see that without any retraining, DAT outperforms BL-SSL-2 that involves multiple rounds of retraining.

System	F+M⇒Y	M+Y⇒F	F+Y⇒M	Avg.	Retrain
bl	53.0	14.0	38.3	35.1	×
blj	52.2	11.4	39.4	34.3	×
DA-1	58.3	12.3	39.2	36.6	×
DAT	54.7	**30.0**	45.6	**43.4**	×
BL-SSL-2	**62.7**	19	**46**	42.6	✓

Table 5: F1 scores of adaptation experiments from two source domains to a single target domain. F+M⇒Y = experiment with fly and mouse data sets as source domain and yeast data set as target domain, M+Y⇒F = experiment with mouse and yeast data sets as source domain and fly data set as target domain, F+Y⇒M = experiment with fly and yeast data sets as source domain and mouse data set as target domain. DAT outperforms both DA-1 and BL-SSL-2 from (Jiang and Zhai, 2007b). Note that without any retraining, DAT outperforms BL-SSL-2 that involves multiple rounds of retraining. bl and blj refer to the baselines of ours and (Jiang and Zhai, 2007b) respectively.

Online Adaptation for Multiple Source Domains There are adaptation scenarios where significant unlabeled data does not exist in the target domain. There are also cases where data come in a stream instead of in batch. For example, in online demo systems for NLP tools, users typically give inputs as one sentence at a time. It is difficult to apply retraining based adaptation algorithms in these scenarios. If the model needs to be retrained for every new sentence, it will be very time consuming and the response time of the system for the user will be the model training time for every sentence input.

Since we can adapt without retraining our model, our framework applies nicely for these cases. DAT-O is the **O**nline version of DAT. For DAT-O, we do not perform any EM iteration to make it fast. We select the common words of the two source domains as pivots. We assume that each time only a sentence from the target domain is given as input. We build a translation model from this sentence and the source domain data and translate the sentence and apply the source domain model. From Table 6, we see that DAT-O improves 5% over baseline and even outperforms DA-1. Nevertheless, DAT-O is behind DAT since DAT looks at the entire unlabeled data and so it can see multiple contexts of a word and build a better translation model.

Experiment	bl	DA-1	DAT-O	DAT
F+M⇒Y	52.0	**58.3**	48.7	**54.7**
M+Y⇒F	12.4	12.3	**29.1**	**30.0**
F+Y⇒M	37.9	39.2	**40.2**	**45.6**
average	34.4	36.6	**39.3**	**43.4**

Table 6: F1 scores for experiments where data come in one sentence at a time. Online version of DAT, DAT-O outperforms DA-1 from (Jiang and Zhai, 2007b). DAT performs better than DAT-O since DAT can look at entire unlabeled data from target domain and DAT-O looks at only one sentence at a time from the target domain. (bold number is the best, bold underline is the second best)

Estimating a translation model is 12 times faster than training the supervised NER model. Retraining based adaptation approaches should be slower than this since they also need to build a shared feature space across the source and target domain and then retrain the model. Run times are computed on a 6-core machine with $48G$ memory.

Some entries in the word translation table for the

adaptation experiment from yeast to fly are listed as examples in Table 7. For each source domain word, we list 3 target domain words that are most probable translations of the source domain word, sorted by decreasing translation probability. In almost all cases, both the words in the word translation pair are named entities. So in the translated text, it is likely that names in the target domain will be replaced by the names in the source domain and the classifier will perform better.

(Source Domain Word, Target Domain Word)
(TNF-alpha, tor), (TNF-alpha, p127), (TNF-alpha, Sx1)
(BTBD1, BJ1), (BTBD1, rutabaga), (BTBD1, dunce)
(FAS, Ddc) , (FAS, Dpp), (FAS, stg)

Table 7: Examples of word translation pairs. The gene names are underlined. Our translation model almost always learns a gene name in the source domain and a gene name in the target domain as a word translation pair.

6 Related Work

Over the years, many adaptation techniques have been proposed in the literature. In (Daumé III, 2007), each training example had a fully duplicated set of general features and domain-specific features. The method proposed in (Finkel and Manning, 2009) was an extension of (Daumé III, 2007) where the domain specific features had a hierarchical bayesian prior in order to encourage the features to share similar weights across domains. In (Jiang and Zhai, 2007a), an instance weighting scheme was presented where the target instances were weighted heavier than the source instances. All these methods require some amount of labeled data in the target domain. Labeling unfortunately is very expensive and time-consuming. Other adaptation methods try to build a common shared representation from the unlabeled data of the source and the target domain. Using this common representation as shared features, they augment the feature space and train a new model. For example, (SCL) (Blitzer et al., 2006) finds a set of pivot features to align the feature space from the source and the target domain and find a projection operator to a low dimensional real valued space. In (Huang and Yates, 2009), the common representation learned from both domains was a hidden markov model that aligned similar words from both domains by assigning similar hidden states. Self-training (McClosky et al., 2006) has been used successfully for adaptation of syntactic parsing. In (Glorot et al., 2011), deep learning was used to discover the shared representation in a hierarchical manner from the unlabeled data of different domains. In (Chang et al., 2010a), clustering of words was used as the shared representation. However, all these methods still need to build new intermediate representations/clusters if confronted with a significantly different domain and then build a new model. Compared to them, we can avoid retraining since we adapt text instead of the model.

Recently there have been several attempts for adaptation without retraining. In (Umansky-Pesin et al., 2010), a POS tag was predicted for every unknown word in the new domain by considering contexts of that word collected by web search queries. For different contexts, applying the classifier yielded different predictions for the same word and they combined those predictions into a single prediction by averaging to find the POS tag for the word. Unfortunately their approach was specifically for the POS tagging task. It is not clear how to use their method for more complicated tasks whereas our approach is general and can be used for any task. However, our method can also be augmented to use additional unlabeled data collected through web search queries. In (Kundu and Roth, 2011), the authors used external resources like WordNet, VerbNet to create new sentences from each sentence in the target domain such that the new sentences will resemble more like the training domain. Compared to them, our work does not rely on using external resources. External resources like gazetter/list can be hard to find for many domains. WordNet will not contain the names of the genes used in our experiments. But using the unlabeled data, we can learn correspondence information and translate the text without needing any external resource to provide the correspondence. In (DaumeIII and Jagarlamudi, 2011), a dictionary was mined from the comparable corpora of different languages and it was integrated in the phrase translation table to achieve domain adaptation for MT systems. Compared to them, our work is on adaptation on domains belonging to the same language and so we can use pivots

for alignment. Instead of focusing on MT, our approach is general and can be applied for adaptation of any system.

Our approach can be related to *paraphrase extraction* approaches (Androutsopoulos and Malakasiotis, 2010). We want to generate paraphrases that have identical labeling as the input sentence but are more similar to text from the source domain. For example, consider the sentences t_1 : *PTP-BL gene causes cancer* and t_2 : *ptuf gene causes cancer*. Here PTP-BL and ptuf are names of two genes in the target and the source domain respectively. t_1 and t_2 are not paraphrases of each other but from our perspective, we want to translate t_1 to t_2 since the NER of t_2 is the same as NER of t_1 and t_2 is more similar to source domain than t_1. Again, for the task of syntactic parsing, consider two sentences t_1 : *X causes Y* and t_2 : *X is a cause of Y*. t_1 and t_2 are paraphrases of each other but we do not want to translate t_1 into t_2 or vice versa since the two sentences have different parse trees. However, exploring how to use existing paraphrase generation techniques to our setting is a very interesting direction to explore.

Named entity recognition has been extensively studied in the community (Sang and F., 2002; Sang et al., 2003). Many named entity recognition methods have been proposed in the literature that include models such as HMMs (Zhou and Su, 2002), MEMMs (Bender et al., 2003), CRFs (McCallum and Li, 2003), combination of classifiers (Florian et al., 2003). But unfortunately, when these systems are trained on a domain and tested on a different domain, performance degrades drastically. For example, a system trained on CoNLL 2003 corpus that achieves 90.8 F1 score in a test set drawn from the same domain, achieves only 64.3 F1 score in a test set drawn from the Wall Street Journal corpus (Ciaramita and Altun, 2005). There has also been a large body of work on unsupervised or semi-supervised named entity recognition methods (Collins and Singer, 1999; Etzioni et al., 2005; Evans, 2003). But these systems are usually outperformed by systems trained on labeled data from a domain.

7 Conclusion

This paper proposes a translation based approach for domain adaptation without retraining. The key goal of the approach is to support porting of learned models to new domains without the need to modify the model. We believe that supporting multiple domains with a single model is crucially important in NLP, where many systems make use of multiple learning based components and third party tools, and where the text can come from diverse domains such as the web. Adaptation without retraining allows the decoupling of the adaptation module from the NLP tool itself and may allow the users to use their own adaptation module where they can use any target domain specific prior knowledge. We believe that text adaptation instead of model adaptation can make domain adaptation more flexible and efficient. We showed that even a simpler, on-line, version of our model is effective, potentially promising more progress towards adapting on the fly to data coming from diverse domains.

Our framework can be extended in several ways. First, by using a more sophisticated translation model, for example, a phrase based translation model. Second, we want to apply our framework to additional natural language tasks such as POS tagging and parsing tasks. Finally, we want to extend our framework by combining our text adaptation techniques with the traditional model adaptation techniques.

References

Ion Androutsopoulos and Prodromos Malakasiotis. 2010. A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research*, 38.

Oliver Bender, Franz Josef Och, and Hermann Ney. 2003. Maximum entropy models for named entity recognition. In *Proceedings of CoNLL*.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *EMNLP*.

Peter Brown, Stephen Della Pietra, Vincent Della Pietra, and Robert Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*.

Ming Chang, Michael Connor, and Dan Roth. 2010a. The necessity of combining adaptation methods. In

Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Massachusetts, USA.

Ming-Wei Chang, Vivek Srikumar, Dan Goldwasser, and Dan Roth. 2010b. Structured output learning with indirect supervision. In *Proceedings of ICML*.

Massimiliano Ciaramita and Yasemin Altun. 2005. Named-entity recognition in novel domains with external lexical knowledge. In *Workshop on Advances in Structured Learning for Text and Speech Processing(NIPS)*.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *EMNLP*.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *ACL*.

Hal DaumeIII and Jagadeesh Jagarlamudi. 2011. Domain adaptation for machine translation by mining unseen words. In *Proceedings of ACL*.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana maria Popescu, Tal Shaked, Stephen Soderl, Daniel S. Weld, and Er Yates. 2005. Unsupervised named entity extraction from the web: An experimental study. *Artificial Intelligence*, 165:91–134.

Richard Evans. 2003. A framework for named entity recognition in the open domain. In *Proceedings of Recent Advances in Natural Language Processing(RANLP)*.

Jenny Finkel and Christopher Manning. 2009. Hierarchical bayesian domain adaptation. In *NAACL*.

Jenny Finkel, Shipra Dingare, Christopher D. Manning, Malvina Nissim, Beatrice Alex, and Claire Grover. 2005. Exploring the boundaries: Gene and protein identification in biomedical text. *BMC Bioinformatics*, 6.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of CoNLL*.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of ICML*.

Lynette Hirschman, Marc Colosimo, Alexander Morgan, and Alexander Yeh. 2005. Overview of biocreative task 1b: normailized gene lists. *BMC Bioinformatics*, 6.

Fei Huang and Alexander Yates. 2009. Distributional representations for handling sparsity in supervised sequence-labeling. In *ACL*.

Jing Jiang and ChengXiang Zhai. 2006. Exploiting domain structure for named entity recognition. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*.

Jing Jiang and ChengXiang Zhai. 2007a. Instance weighting for domain adaptation in nlp. In *ACL*.

Jing Jiang and Chenxiang Zhai. 2007b. A two-stage approach to domain adaptation for statistical classifiers. In *Proceedings of CIKM*.

Jin-Dong Kim, Sampo Pyysalo, Tomoko Ohta, Robert Bossy, and Junichi Tsujii. 2011. Overview of bionlp shared task 2011. In *BioNLP 2011 Workshop*.

Gourab Kundu and Dan Roth. 2011. Adapting text instead of the model: An open domain approach. In *Proceedings of CoNLL*.

Gourab Kundu, Ming wei Chang, and Dan Roth. 2011. Prior knowledge driven domain adaptation. In *International Conference on Machine Learning Workshop on Combining Learning Strategies to Reduce Label Cost*.

Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of CoNLL*.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of HLT-NAACL*.

Tjong Kim Sang and Erik F. 2002. Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL*.

Tjong Kim Sang, F. Erik, and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL*.

Shulamit Umansky-Pesin, Roi Reichart, and Ari Rappoport. 2010. A multi-domain web-based algorithm for pos tagging of unknown words . In *Proceedings of Coling*.

Guodong Zhou and Jian Su. 2002. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of ACL*.