BIOINFORMATICS ORIGINAL PAPER

Vol. 26 no. 8 2010, pages 1098–1104
doi:10.1093/bioinformatics/btq075

*Data and text mining*

Advance Access publication February 23, 2010

# Domain adaptation for semantic role labeling in the biomedical domain

Daniel Dahlmeier[1] and Hwee Tou Ng[1,2,*]

[1]NUS Graduate School for Integrative Sciences and Engineering, Singapore 117456 and [2]Department of Computer Science, National University of Singapore, Singapore 117417

Associate Editor: Alfonso valencia

**ABSTRACT**

**Motivation:** Semantic role labeling (SRL) is a natural language processing (NLP) task that extracts a shallow meaning representation from free text sentences. Several efforts to create SRL systems for the biomedical domain have been made during the last few years. However, state-of-the-art SRL relies on manually annotated training instances, which are rare and expensive to prepare. In this article, we address SRL for the biomedical domain as a domain adaptation problem to leverage existing SRL resources from the newswire domain.

**Results:** We evaluate the performance of three recently proposed domain adaptation algorithms for SRL. Our results show that by using domain adaptation, the cost of developing an SRL system for the biomedical domain can be reduced significantly. Using domain adaptation, our system can achieve 97% of the performance with as little as 60 annotated target domain abstracts.

**Availability:** Our BioKIT system that performs SRL in the biomedical domain as described in this article is implemented in Python and C and operates under the Linux operating system. BioKIT can be downloaded at http://nlp.comp.nus.edu.sg/software. The domain adaptation software is available for download at http://www.mysmu.edu/faculty/jingjiang/software/DALR.html. The BioProp corpus is available from the Linguistic Data Consortium http://www.ldc.upenn.edu

**Contact:** nght@comp.nus.edu.sg

Received on September 22, 2009; revised on January 24, 2010; accepted on February 18, 2010

## 1 INTRODUCTION

Advances in biology and life sciences have led to an exponential growth in the amount of biomedical literature. Thus, automatic information retrieval (IR) and information extraction (IE) methods become more and more important to help researchers to keep track of the latest developments in their field. Current IR is still mostly limited to keyword search and unable to infer the relationship between two entities in a text. A system that is able to understand how words in a sentence are related could greatly increase the quality of IE and would allow IR to handle more complex user queries.

Semantic role labeling (SRL) is a shallow semantic processing task that has become increasingly popular in the natural language processing (NLP) community over the last few years. The task is to identify all parts of a sentence that represent arguments for a given predicate and subsequently label each argument with a semantic role. Roughly speaking, SRL can be thought of as the task of finding the words that answer simple questions of the form *Who did what to whom when and where?* The input to the SRL system is a single sentence and a predicate in that sentence. The output is the same sentence, but with labeled semantic roles. Consider the following example:

Input: Transcription factor GATA-3 [stimulates]$_{\text{PRED}}$ HIV-1 expression.
Output: [Transcription factor GATA-3]$_{\text{ARG0}}$ [stimulates]$_{\text{PRED}}$ [HIV-1 expression]$_{\text{ARG1}}$.

In this example, the semantic role ARG0 is the *cause of stimulate* and the semantic role ARG1 is the *thing stimulated* (see Section 2 for a detailed description of semantic roles). This information is most valuable for IE (Surdeanu *et al.*, 2003) and other tasks such as question answering and summarization.

Traditionally, most work in SRL has focused on documents from the newswire domain. While SRL works well on test sentences from the same domain, the models show a sharp performance drop when they are tested on a different domain (Pradhan *et al.*, 2008). Although there have been a number of efforts to apply SRL to the biomedical domain in recent years, the development of state-of-the-art SRL systems for the biomedical domain is hampered by the lack of large biomedical corpora that are labeled with semantic roles. The creation of such corpora is time consuming and expensive.

In this article, we address SRL on biomedical text as a domain adaptation problem. The goal is to adapt an SRL system for the newswire domain (where a large annotated corpus is available) to the biomedical domain (where only a small amount of annotated text is available). This way, we can leverage existing corpora from the newswire domain and significantly reduce the cost of developing an SRL system for the biomedical domain. The main contributions of this article are:

- it is the first work that performs a comparative evaluation of the performance of three recently proposed domain adaptation algorithms on the task of SRL for biomedical text;
- it is the first work that investigates the extent of manual annotation needed to port an SRL system trained on newswire text to biomedical text, by explicitly determining the number of annotated biomedical text examples needed to achieve good performance; and
- we provide a detailed analysis of *why* domain adaptation helps.

*To whom correspondence should be addressed.

 

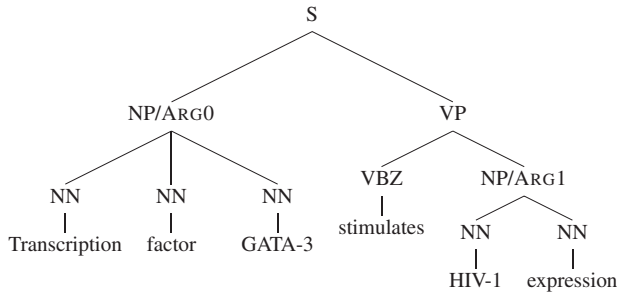


**Fig. 1.** A syntactic parse tree with semantic roles (ARG0 and ARG1) added.

To our knowledge, this is the first detailed study of domain adaptation for SRL in biomedical text, and our work demonstrates that domain adaptation can greatly reduce the cost of developing biomedical SRL systems.

## 2 SRL

The task of SRL is to find all arguments for a given predicate in a sentence and label them with semantic roles. The first step is to parse the sentence into a syntactic parse tree. The parse tree consists of the words in the sentence, their part-of-speech tags (e.g. NN, VBZ, etc.) and nodes with syntactic categories (e.g. S, NP, VP, etc.). Figure 1 shows the syntactic parse tree for the example sentence from Section 1 (the labels ARG0 and ARG1 are not part of the syntactic parse tree).

The next step is the *argument identification* step, where the SRL system has to find the boundaries for all the arguments in the sentence. The annotation standard for semantic roles demands that the boundaries align with nodes in the syntactic parse tree. Thus, argument identification is equivalent to deciding which nodes in the parse tree, including the part-of-speech tags, span arguments. As shown by the example in Figure 1, the system should find that the NP node that dominates *Transcription factor GATA-3* and the NP node that dominates *HIV-1 expression* span arguments and all other nodes do not.

Finally, the system has to determine the semantic role for all identified nodes. This step is called *argument classification*. In our example, the first identified NP node should be labeled ARG0 and the second should be labeled ARG1, as shown in Figure 1. For the predicate *stimulate*, ARG0 and ARG1 represent the *cause of stimulate* and the *thing stimulated*, respectively. In general, ARG0 refers to the *agent* and ARG1 refers to the *theme* of the predicate. Each of the semantic roles ARG2-5 does not have a general meaning that stays consistent across different predicates. The semantic role ARG2, for example, is the *instrument* for the predicate *stimulate*, but for the predicate *increase*, ARG2 is the *amount increased*. The semantic roles ARG0-5 are called *core arguments*, because they represent the essential arguments of a predicate. A predicate and the semantic roles that can appear with it are called a *predicate argument structure (PAS)* or *proposition*. Additional to its core arguments, a predicate can appear with any number of *adjunctive arguments*. Adjunctive arguments express general properties such as time, location, manner, etc. They are labeled with ARGM plus a functional tag, e.g. ARGM-LOC, ARGM-TMP or ARGM-MNR. The combined SRL task involves argument identification followed by argument classification.

**Table 1.** Features used in our SRL system

| | |
|---|---|
| Baseline features (Gildea and Jurafsky, 2002) | |
| pred | Predicate lemma |
| path | Path from constituent to predicate |
| ptype | Syntactic category (NP, PP, etc.) |
| pos | Relative position to the predicate |
| voice | Active or passive voice |
| hw | Syntactic head word of the phrase |
| sub-cat | Rule expanding the predicate's parent |
| Advanced features (Pradhan *et al.*, 2005) | |
| hw POS | POS of the syntactic head word |
| PP hw/POS | Head word and POS of the rightmost NP child if the phrase is a prepositional phrase |
| first/last word | First/last word and POS in the constituent |
| parent ptype | Syntactic category of the parent node |
| parent hw/POS | Head word and POS of the parent |
| sister ptype | Phrase type of left and right sister |
| sister hw/POS | Head word and POS of left and right sister |
| temporal | Temporal key words present |
| partPath | Partial path predicate |
| proPath | Projected path without directions |
| Feature combinations (Xue and Palmer, 2004) | |
| pred&ptype | Predicate and phrase type |
| pred&hw | Predicate and head word |
| pred&path | Predicate and path |
| pred&pos | Predicate and relative position |

## 3 FEATURES AND MACHINE LEARNING METHODS

This section describes how SRL can be solved by supervised machine learning algorithms. First, the input sentence has to be parsed into a syntactic parse tree. In this article, we assume that this step has already been solved and that the correct syntactic parse tree is available to us.

The next step is to learn classifiers for the argument identification and argument classification step. The classifier for argument identification performs a binary classification for every node in the parse tree to decide whether the node spans an argument or not. The classifier for argument classification performs a multiclass classification to predict the semantic role for a node in the parse tree, given that the node spans an argument.

By casting SRL as a machine learning problem, there are two key decisions that have to be made: the choice of features and the choice of the machine learning algorithm. In this article, we adopt the features used in other state-of-the-art SRL systems, which include the seven baseline features from the original work of Gildea and Jurafsky (2002), additional features taken from Pradhan *et al.* (2005) and feature combinations that are inspired by the system in Xue and Palmer (2004). All features can be extracted from the syntactic parse tree. Table 1 lists the features that we use for easy reference.

The machine learning algorithm in our experiments is a maximum entropy (maxent) classifier.[1] Since their introduction by Berger *et al.* (1996), maxent classifiers have successfully been applied to many NLP problems, including SRL (Toutanova *et al.*, 2008; Xue and Palmer, 2004). Maximum entropy classifiers do not require any independence assumptions, which allow great flexibility in encoding linguistic knowledge via features. The model takes

[1]We use the implementation in the DALR package (Jiang and Zhai, 2007)

the form

$$P(y|x) = \frac{1}{Z} \cdot \exp\left\{\sum_{i=1}^{N} \lambda_i f_i(x,y)\right\} \quad (1)$$

where $y$ is a semantic role, $x$ is an input vector, $f_i$ are feature functions, $\lambda_i$ are the weights that are learned during training and $Z$ is a normalization term. A detailed description of maximum entropy classifiers can be found in Ratnaparkhi (1998).

## 4 DOMAIN ADAPTATION ALGORITHMS

The task of domain adaptation is to adapt a classifier that is trained on some source domain to a new target domain. Domain adaptation algorithms can be divided into two categories: *unsupervised* and *supervised* domain adaptation algorithms. Unsupervised algorithms only use unlabeled instances from the target domain, while supervised algorithms assume that there is a small amount of labeled target instances available during training. The algorithms that we evaluate in this article are all supervised. The algorithms are presented below:

- Instance weighting (INSTWEIGHT): The essential problem when applying a classifier to data from another domain is that the joint distribution $P(X,Y)$ of features and class labels in the target domain will be different from the source domain. Instance weighting (Jiang and Zhai, 2007) is a general framework to tune the estimate for $P(X,Y)$. The probability $P(X,Y)$ can be factored in the following way:

$$P(X,Y) = P(Y|X) \times P(X) \quad (2)$$

  The first component is the likelihood of the class given the features and the second is the prior probability of observing the features. The difference in $P(X,Y)$ can arise from either $P(X)$ or $P(Y|X)$. INSTWEIGHT tries to tackle the difference in the conditional probability $P(Y|X)$. By weighting the instances in the training set, the domain adaptation algorithm can try to adjust the probability estimate for the target domain. Intuitively, if the estimated probability density for an instance does not match the probability density in the target domain very well, then the learning algorithm should give less weight to this instance. To do this, the algorithm weights an instance by the ratio $\frac{P_t(Y|X)}{P_s(Y|X)}$ between the probability densities in the target and source domain.
- Augment method (AUGMENT): Daumé III (2007) proposed a domain adaptation strategy that is based on feature space augmentation. The algorithm takes each feature vector and maps it to a feature space of a higher dimension. The mapping depends on whether the instance is from the source or from the target domain. Assume that $\mathbf{x} \in X$ is a feature vector in the original feature space. We define mappings $\Phi^s$ and $\Phi^t$ for the source and target domain, respectively:

$$\Phi^s(\mathbf{x}) = <\mathbf{x}, \mathbf{x}, \mathbf{0}> \quad (3)$$

$$\Phi^t(\mathbf{x}) = <\mathbf{x}, \mathbf{0}, \mathbf{x}> \quad (4)$$

  where $\mathbf{0}$ is a zero vector of length $|\mathbf{x}|$. The transformation can be interpreted in the way that it takes each feature vector and makes three versions out of it: one 'general' version, one 'source-specific' version and one 'target-specific' version. The algorithm is surprisingly easy to implement and is independent of the machine learning algorithm that is used.
- Instance pruning (INSTPRUNE): instance pruning (Jiang and Zhai, 2007) trains a classifier on the target domain instances and uses this classifier to predict class labels for all instances from the source domain. The top $N$ instances that are predicted wrongly, ranked by prediction confidence, are removed from the source domain. The intuition here is that these instances are very different from the target domain and will confuse the classifier during training. The remaining instances from the source domain are then used to train the classifier. Instance pruning is actually another form of instance weighting where the weight for a wrongly predicted source instance is set to zero. INSTPRUNE depends on the parameter $N$. Setting $N$ too low will hurt the performance, because it leaves too many confusing source instances in the training set. Setting $N$ too high will also result in poor performance, because all information from the source domain is pruned away.

In addition to the above domain adaptation algorithms, we implement the following three baseline algorithms:

- Source only (SRCONLY): this baseline simply ignores the target domain data and trains a classifier on only the source domain data.
- Target only (TRGTONLY): at the other extreme, the TRGTONLY baseline trains a classifier on the target domain data only, ignoring any source domain data that are available.
- Source and Target (ALL): the simplest way to combine source and target domain data is to train a classifier on the combined dataset from both domains. This we call the ALL baseline. The potential problem with this algorithm is that when the source domain dataset is much larger than the target domain dataset, the learning algorithm might regard the target domain instances as 'noise' and essentially ignore them.

We evaluate all six algorithms for the SRL task on biomedical text. The details of our experiments are given in the next section.

## 5 EXPERIMENTS

### 5.1 Datasets

This section presents the details of the datasets that we used in our experiments. The source domain data comes from the PropBank corpus (Palmer *et al.*, 2005), which is the most commonly used corpus for SRL. The corpus is built from financial news articles from the *Wall Street Journal* and is available through the Linguistic Data Consortium (http://www.ldc.upenn.edu). We use sections 2–21, which form the standard training set used in SRL evaluations, as our source domain dataset. The source domain dataset has a total of 36 090 annotated sentences with their syntactic parse trees and over 90 000 annotated PAS.

The target domain dataset consists of the BioProp corpus (Tsai *et al.*, 2007). The corpus is created from 500 MEDLINE article abstracts. The articles were selected based on the keywords *human*, *blood cells* and *transcription factor*. To our knowledge, BioProp is the only resource for biomedical SRL that uses full syntactic parse trees. The parse trees are taken from the Genia Treebank (GTB; Kim *et al.*, 2003). The GTB is available for download from the Genia project web site ( http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/). During preprocessing of the data, we found that nine abstracts from the BioProp were missing in GTB and that another 45 abstracts did not contain any annotated PAS. The remaining 446 abstracts contain 1635 sentences with a total of 1982 PAS. The statistics of the datasets are given in Table 2. It is obvious that BioProp is much smaller than PropBank, not only in terms of the number of sentences, but also in the number of PAS and verbs that are covered. The reason is that the creators of BioProp concentrated on 30 important or frequent verbs from the biomedical domain, while PropBank annotates PAS for all verbs. We also observe that the semantic roles ARG3 and ARG4 are very rare in BioProp and that ARG5 is not used at all.

### 5.2 Experimental setup

We investigate the performance of the SRL system on argument identification, argument classification and the combined SRL task. All experiments are conducted using 5-fold cross-validation on the target domain dataset. The 446 abstracts in the target domain dataset are split into five equal portions. Thus, there are four portions with 89 abstracts and one portion with 90 abstracts. The split is done randomly to guard against any selection bias. The SRL system is trained on four of the portions plus the complete source domain data and tested on the remaining portion of the target domain data. This is done for each of the five portions in turn and the results are

**Table 2.** Statistics of the source and target domain dataset

| | PropBank | BioProp |
|---|---|---|
| Sentences | 36090 | 1635 |
| Words | 898778 | 46682 |
| Unique verbs | 3101 | 30 |
| PAS | 91122 | 1982 |
| ARG0 | 66329 | 1464 |
| ARG1 | 92958 | 2124 |
| ARG2 | 20547 | 325 |
| ARG3 | 3491 | 8 |
| ARG4 | 2739 | 5 |
| ARG5 | 69 | 0 |
| ARGM | 60962 | 1762 |

ARG0 and ARG1 generally refer to the *agent* and the *theme*, respectively. The semantic roles ARG2-5 do not have a general meaning. ARGM refers to adjunctive arguments.

averaged over all classified instances. We further ensure that all sentences from one abstract end up in the same portion, to avoid a situation where the classifier is trained and tested on sentences from the same abstract. The evaluation metrics are described in the next section. During our experiments, we gradually increase the number of target domain abstracts that are available during training from 8 to 356 abstracts (357 in the case where the fold with 90 abstracts is used for training). This allows us to assess the impact of the target domain data. The order in which abstracts are added is random, but a particular randomly chosen order of abstracts is used in each experiment where abstracts are added incrementally. In all experiments, we use gold standard syntactic parse trees, including part-of-speech tags, which we take from the PropBank and BioProp corpus.

The INSTPRUNE algorithm has a parameter $N$ that needs to be tuned. For each of the five portions of target domain data, we tune $N$ through 4-fold cross-validation on the four portions of target domain data that are used as training data. The classifier is trained on three portions of the target domain data plus the complete source domain data for different values of $N$ and tested on the remaining one portion of target domain that is part of the training data. This is done for each portion of the training data and the results are averaged. The best value of $N$ for each of the five portions is kept. Note that no data from the portion that is used during testing is used to tune the parameter.

### 5.3 Evaluation metrics

Argument identification and the combined SRL task are evaluated in terms of *precision (p)*, *recall (r)* and $F_1$ *measure* ($F_1$). Precision measures how accurate the predictions of a classifier are. It is calculated as the number of correct predictions divided by the total number of predictions: $p = \frac{\#\text{ correct predictions}}{\#\text{ predicted instances}}$. Recall is measured as the number of correct predictions divided by the actual number of relevant instances in the test set: $r = \frac{\#\text{ correct predictions}}{\#\text{ actual instances in the test set}}$. $F_1$ measure combines precision and recall into a single metric by computing the harmonic mean of the two: $F_1 = 2 \times \frac{p \times r}{p+r}$. During argument classification, the boundaries of the relevant instances are already known. In that case, the *accuracy (a)* of the classifier is reported. Accuracy is defined as the number of correct predictions divided by the total number of instances: $a = \frac{\#\text{ correct predictions}}{\#\text{ total instances}}$.

## 6 RESULTS

This section presents the results of our experiments. Before we started experiments on the target domain data, we performed a test on the PropBank corpus to ensure that our SRL system represents a strong baseline. We trained the classifier on sections 2–21 and tested on section 23, which is the standard evaluation setting.

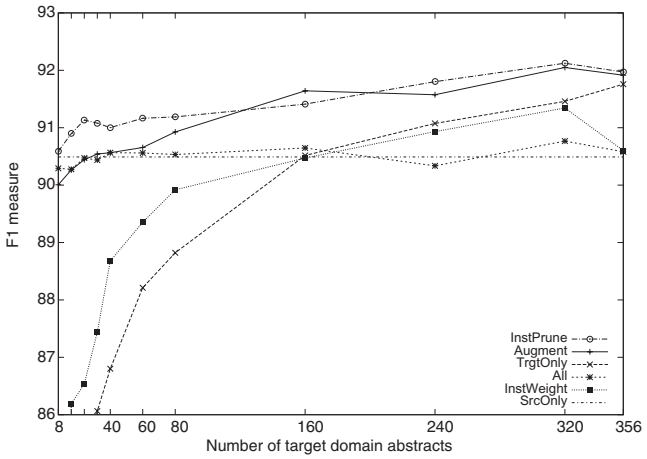


**Fig. 2.** Argument identification results for INSTWEIGHT, AUGMENT and INSTPRUNE. The $x$-axis denotes the number of target domain abstracts that are available during training. The $y$-axis denotes the averaged $F_1$ measure, using 5-fold cross-validation.

The results are 95.11% $F_1$ measure for argument identification, 90.58% accuracy for argument classification, and 86.79% $F_1$ measure for the combined SRL task. This confirms that our model performs comparably with other state-of-the-art SRL systems (Xue and Palmer, 2004).

### 6.1 Argument identification

The first experiment examines the system's performance for the argument identification step. The learning curves for the domain adaptation algorithms and the baselines are shown in Figure 2. The first observation that can be made is that the SRCONLY baseline achieves a high $F_1$ measure of 90.49%, only 5% lower than the $F_1$ measure on PropBank. The TRGTONLY baseline performs poorly in the beginning but improves as more target domain abstracts are added. The ALL baseline shows no significant improvement over SRCONLY. For the domain adaptation algorithms, INSTPRUNE and AUGMENT perform better than INSTWEIGHT and also better than the three baselines.

### 6.2 Argument classification

The second experiment examines the system's performance for the argument classification step. The learning curves for the domain adaptation algorithms and the baselines are shown in Figure 3. The SRCONLY baseline achieves 81.50% accuracy, a drop of over 9% from 90.58% accuracy on PropBank. The TRGTONLY baseline improves quickly with more target domain data. The three domain adaptation algorithms perform similar to or slightly above the TRGTONLY baseline. None of the domain adaptation algorithms can clearly outperform the others.

### 6.3 Combined SRL task

The third experiment examines the system's performance for the combined SRL task. The results are shown in Figure 4. The SRCONLY baseline achieves 76.46% $F_1$ measure, a drop of over 10% from the PropBank results of 86.79%. The TRGTONLY baseline (INSTWEIGHT algorithm) only achieves comparable results when 60 (32) or more

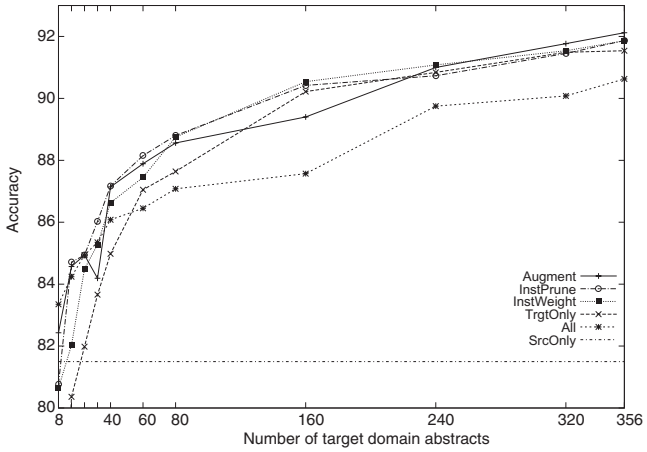


**Fig. 3.** Argument classification results for INSTWEIGHT, AUGMENT and INSTPRUNE. The *x*-axis denotes the number of target domain abstracts that are available during training. The *y*-axis denotes the averaged accuracy, using 5-fold cross-validation.

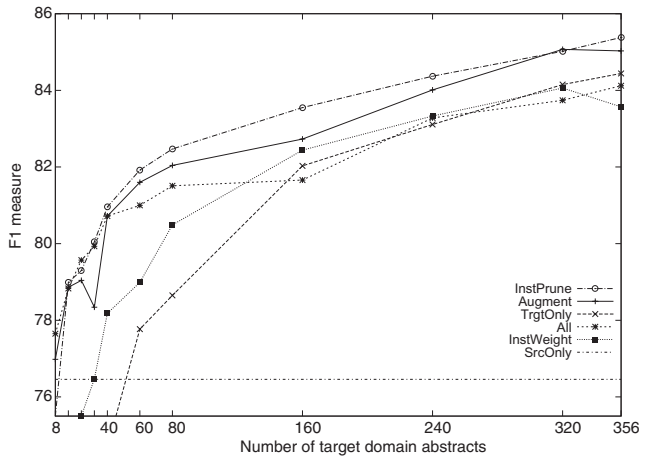


**Fig. 4.** Results for INSTWEIGHT, AUGMENT and INSTPRUNE on the combined SRL task. The *x*-axis denotes the number of target domain abstracts that are available during training. The *y*-axis denotes the averaged $F_1$ measure, using 5-fold cross-validation.

abstracts from the target domain are added. The ALL baseline performs decently. Our initial concern that the larger source domain data would dominate the effect of the target domain data appears to be unjustified.

The best performing algorithm is INSTPRUNE, followed by AUGMENT. INSTPRUNE shows a consistent improvement over all three baselines for 32 or more target domain abstracts. We recall that INSTPRUNE is actually a version of INSTWEIGHT where misclassified source domain instances are weighted with zero weights. Our experiments show that the more aggressive strategy of INSTPRUNE shows better results than INSTWEIGHT.

We performed the Wald test for statistical significance to determine whether the improvement for INSTPRUNE and AUGMENT could have occurred by chance. The test was always performed against the best performing baseline algorithm. The domain adaptation algorithm performed worse than or equal to the baseline for two data points for INSTPRUNE and four data points for AUGMENT. For INSTPRUNE, the improvement is statistically significant ($P < 0.05$) for all remaining data points. For AUGMENT, the improvement is statistically significant ($P < 0.05$) for all remaining data points, except for 16 abstracts. The detailed results of all six algorithms are given in Table 3. Overall, we observe that by using PropBank data and domain adaptation algorithms, the SRL system can achieve accurate results with only a fraction of the target domain abstracts. For example, with 60 abstracts ($\sim$17% of the data) and INSTPRUNE, we can get 97% ($= \frac{81.92\%}{84.44\%}$) of the performance that we get when using 356 abstracts without domain adaptation.

## 7 ANALYSIS

In this section, we analyze the results to better understand why SRL on BioProp is difficult and why domain adaptation helps. One reason why SRL on Bioprop is difficult is that the vocabularies in the newswire domain and in the biomedical domain are very different, so there are many words in the target domain that the model has not seen during training. Another reason is that a word can have a different dominant meaning in the source and target domain. Consider the following two examples for the predicate *increase*:

> Source domain: $[\text{Sales}]_{\text{ARG1}}$ increased $[\text{a more modest } 4.8\%]_{\text{ARG2}}$ $[\text{in the South}]_{\text{ARGM-LOC}}$.
> Target domain: $[\text{LTB4}]_{\text{ARG0}}$ increased [the expression of the c-fos gene]$_{\text{ARG1}}$ [in a time- and concentration-dependent manner]$_{\text{ARGM-MNR}}$.

In the first example, *increase* has an intransitive usage where the subject is ARG1 (*thing increasing*). This usage can typically be found in the source domain. That is why the SRCONLY baseline wrongly predicts ARG1 for *LTB4*. In the target domain, we often observe *increase* with a transitive usage where the subject is ARG0 (*causer of increase*) and the object is ARG1. With domain adaptation, the system correctly classifies *LTB4* as ARG0, for all three domain adaptation algorithms. This example suggests that predicates with different usage in the source and target domain are more difficult to predict than predicates with similar usage in both domains. To quantify this difference, we split the target domain data into two sets: one set contained only instances of predicates that have a similar usage and the other set only contained instances for predicates which have a different usage. To decide which predicates have similar or different usage, we referred to the data provided in Tsai *et al.* (2007). We tested the INSTPRUNE algorithm, which performed best in our previous experiments, on the combined SRL task for the two datasets, using the same experimental setup as before. The results are shown in Figures 5 and 6. We observe a significant gap between the SRCONLY baseline results in Figures 5 and 6. This empirically confirms our conjecture that predicates with different usage are more difficult to predict without domain adaptation. When domain adaptation is used, predicates with different usage can be predicted as accurately as predicates with the same usage.

Finally, we wanted to find out if domain adaptation only improves the performance for predicates that appear frequently in the target domain, or if infrequent predicates see an improvement as well. We again split the target domain data into two sets, depending on whether the predicate belongs to the most frequent verbs in the target domain or not. The information on which predicates are frequent can be found in Tsai *et al.* (2007). Again, we tested the

**Table 3.** $F_1$ measure for the combined SRL task for different number of target domain abstracts that were available to the systems during training

| Algorithm | 0 | 8 | 16 | 24 | 32 | 40 | 60 | 80 | 160 | 240 | 320 | 356 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SrcOnly | 76.46 | – | – | – | – | – | – | – | – | – | – | – |
| TrgtOnly | – | 57.04 | 68.03 | 71.84 | 73.24 | 74.69 | 77.77 | 78.65 | **82.03** | 83.11 | **84.15** | **84.44** |
| All | – | **77.65** | **78.83** | **79.57** | **79.93** | **80.72** | **81.00** | **81.51** | 81.66 | **83.28** | 83.74 | 84.12 |
| InstWeight | – | 67.62 | 74.04 | 75.50 | 76.47 | 78.18 | 78.99 | 80.49 | 82.44 | 83.33 | 84.06 | 83.57 |
| Augment | – | **76.98** | 78.86 | 79.04 | 78.34 | 80.72 | 81.60* | 82.04* | 82.73* | 84.01* | **85.07*** | 85.03 |
| InstPrune | – | 75.49 | **78.99*** | **79.30** | **80.05*** | **80.96*** | **81.92*** | **82.47*** | **83.55*** | **84.37*** | 85.02* | **85.38*** |

The best baseline and domain adaptation algorithm for each column are printed in bold face. Statistically significant improvements for Augment and InstPrune over the best baseline algorithm are marked with an 'asterisk'. All results are obtained using 5-fold cross-validation.

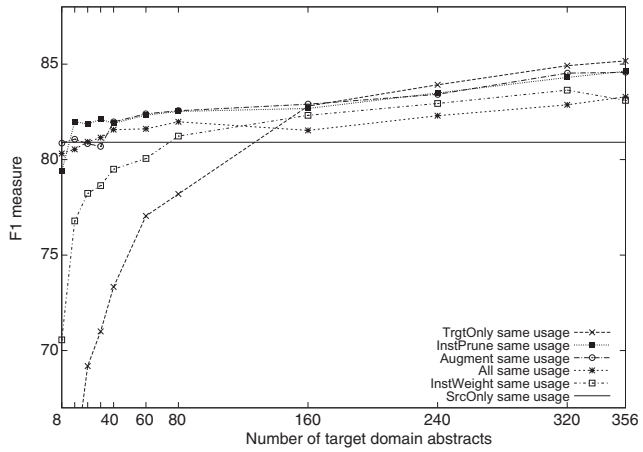


**Fig. 5.** Results for InstPrune on the combined SRL task for predicates which have similar usage in source and target domain. The $x$-axis denotes the number of target domain abstracts that are available during training. The $y$-axis denotes the averaged $F_1$ measure, using 5-fold cross-validation.

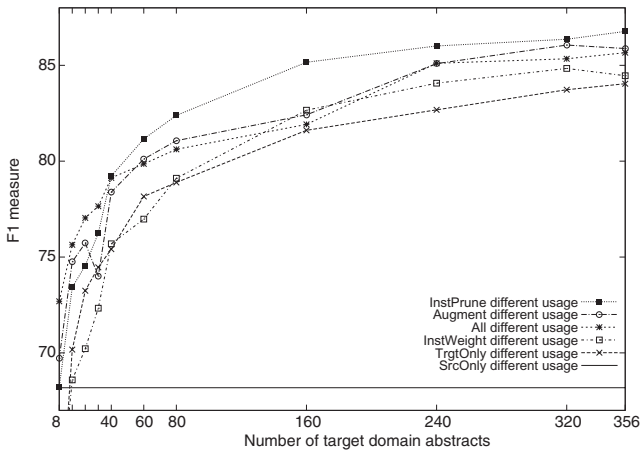


**Fig. 6.** Results for InstPrune on the combined SRL task for predicates which have different usage in source and target domain. The $x$-axis denotes the number of target domain abstracts that are available during training. The $y$-axis denotes the averaged $F_1$ measure, using 5-fold cross-validation.

InstPrune algorithm on the combined SRL task on the two datasets, using the same experimental setup as before. Our results show that SRL performance for infrequent predicates is about 1–2% lower in $F_1$ measure than the performance for frequent predicates. Thus, even for infrequent predicates, domain adaptation improves SRL performance, although performance is slightly lower.

## 8 RELATED WORK

In the last few years, there have been a number of efforts to bring SRL to the biomedical domain. Wattarujeekrit *et al.* (2004) developed PASBio that contains and analyzes PAS for over 30 verbs and has become a standard for annotating PAS for molecular events. Shah and Bork (2006) applied SRL in the LSAT system to identify sentences that talk about gene transcripts. Kogan *et al.* (2005) analyzed PAS in medical case reports, but they did not present a functioning SRL system. Paek *et al.* (2006) applied SRL to abstracts from randomized controlled trial reports, but they limited the scope to five verbs only. Bethard *et al.* (2008) presented an SRL system that extracted information about protein movement. They created a corpus for 34 verbs and 4 semantic roles. Their work was more problem-specific than general SRL, as they only focused on a very problem specific set of semantic roles. Most recently, Barnickel *et al.* (2009) presented a neural network-based SRL system for relation extraction. Their emphasis was not on accurate SRL but on fast processing speed.

All of the above SRL systems use a word-by-word or chunk-by-chunk approach instead of a full constituent-by-constituent, syntax-based approach, although the latter approach is the state-of-the-art in SRL. To our knowledge, the only system for biomedical SRL on full syntactic parse trees is the BIOSMILE system by Tsai *et al.* (2007). They observed that their SRL system did not perform well on BioProp if it was only trained on PropBank. However, their results are not directly comparable with ours, because they only used a smaller portion of PropBank to train their model and they did not use any domain adaptation algorithms. Thus, their work did not investigate how well a state-of-the-art SRL system performs on biomedical text if it is trained on the whole PropBank corpus and uses domain adaptation algorithms.

## 9 CONCLUSION

In this article, we study the effect of domain adaptation for SRL in the biomedical domain. We evaluate three different domain adaptation algorithms on the BioProp corpus using a competitive,

state-of-the-art SRL classifier. We conduct a systematic, detailed comparison of different domain adaptation algorithms for different number of target domain training examples. Our results show that by using just the existing SRL resources and domain adaptation, significant improvements can be achieved with only a small number of annotated target domain data. We believe that our findings will be helpful for applying SRL to new domains in the biomedical field.

## ACKNOWLEDGEMENTS

We thank Jing Jiang for allowing us to use her DALR software in our experiments and Wen-Lian Hsu for sharing a prerelease version of the BioProp dataset with us.

*Funding*: This work was funded by National University of Singapore Academic Research Fund (research grant R-252-000-225-112).

*Conflict of Interest*: none declared.

## REFERENCES

Barnickel,T. *et al*. (2009) Large scale application of neural network based semantic role labeling for automated relation extraction from biomedical texts. *PLoS ONE*, **4**, Article no. e6393.

Berger,A.L. *et al*. (1996) A maximum entropy approach to natural language processing. *Comput. Linguist.*, **22**, 39–71.

Bethard,S. *et al*. (2008) Semantic role labeling for protein transport predicates. *BMC Bioinformatics*, **9**, Article no. 277.

Daumé III,H. (2007) Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, ACL, Prague, Czech Republic, pp. 256–263.

Gildea,D. and Jurafsky,D. (2002) Automatic labeling of semantic roles. *Comput. Linguist.*, **28**, 245–288.

Jiang,J. and Zhai,C. (2007) Instance weighting for domain adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, ACL, Prague, Czech Republic, pp. 264–271.

Kim,J.-D. *et al*. (2003) Genia corpus – a semantically annotated corpus for bio-textmining. *Bioinformatics*, **19** (Suppl. 1), i180–i182.

Kogan,Y. *et al*. (2005) Towards semantic role labeling & IE in the medical literature. In *Proceedings of the American Medical Informatics Association (AMIA) Annual Symposium*, AMIA, Washington DC, USA, pp. 410–414.

Paek,H. *et al*. (2006) Shallow semantic parsing of randomized controlled trial reports. In *Proceedings of the American Medical Informatics Association (AMIA) Annual Symposium*, AMIA, Washington DC, USA, pp. 604–608.

Palmer,M. *et al*. (2005) The proposition bank: an annotated corpus of semantic roles. *Comput. Linguist.*, **31**, 71–106.

Pradhan,S.S. *et al*. (2005) Support vector learning for semantic argument classification. *Mach. Learn.*, **60**, 11–39.

Pradhan,S.S. *et al*. (2008) Towards robust semantic role labeling. *Comput. Linguist.*, **34**, 289–310.

Ratnaparkhi,A. (1998) Maximum entropy models for natural language ambiguity resolution. PhD. Thesis, University of Pennsylvania, Philadelphia, PA, USA.

Shah,P.K. and Bork,P. (2006) LSAT: learning about alternative transcripts in MEDLINE. *Bioinformatics*, **22**, 857–865.

Surdeanu,M. *et al*. (2003) Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003)*, ACL, Sapporo, Japan, pp. 8–15.

Toutanova,K. *et al*. (2008) A global joint model for semantic role labeling. *Comput. Linguist.*, **34**, 161–191.

Tsai,R.T.-H. *et al*. (2007) BIOSMILE: A semantic role labeling system for biomedical verbs using a maximum-entropy model with automatically generated template features. *BMC Bioinformatics*, **8**, Article no. 325.

Wattarujeekrit,T. *et al*. (2004) PASBio: predicate-argument structures for event extraction in molecular biology. *BMC Bioinformatics*, **5**, Article no. 155.

Xue,N. and Palmer,M. (2004) Calibrating features for semantic role labeling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, ACL, Barcelona, Spain, pp. 88–94.