Transfer Learning, Feature Selection and Word Sense Disambguation

Paramveer S. Dhillon and Lyle H. Ungar
Computer and Information Science
University of Pennsylvania, Philadelphia, PA, U.S.A
{pasingh,ungar}@seas.upenn.edu

Abstract

We propose a novel approach for improving Feature Selection for Word Sense Disambiguation by incorporating a feature relevance prior for each word indicating which features are more likely to be selected. We use transfer of knowledge from similar words to learn this prior over the features, which permits us to learn higher accuracy models, particularly for the rarer word senses. Results on the ONTONOTES verb data show significant improvement over the baseline feature selection algorithm and results that are comparable to or better than other state-of-the-art methods.

1 Introduction

The task of WSD has been mostly studied in a supervised learning setting e.g. (Florian and Yarowsky, 2002) and feature selection has always been an important component of high accuracy word sense disambiguation, as one often has thousands of features but only hundreds of observations of the words (Florian and Yarowsky, 2002).

The main problem that arises with supervised WSD techniques, including ones that do feature selection, is the paucity of labeled data. For example, the training set of SENSEVAL-2 English lexical sample task has only 10 labeled examples per sense (Florian and Yarowsky, 2002), which makes it difficult to build high accuracy models using only supervised learning techniques. It is thus an attractive alternative to use transfer learning (Ando and Zhang, 2005), which improves performance by generalizing from solutions to "similar" learning problems. (Ando, 2006) (abbreviated as Ando[CoNLL'06]) have successfully applied the ASO (Alternating Structure Optimization) technique proposed by (Ando and Zhang, 2005), in its transfer learning configuration, to the problem of WSD by doing joint empirical risk minimization of a set of related problems (words in this case). In this paper, we show how a novel form of transfer learning that learns a feature relevance prior from similar word senses, aids in the process of feature selection and hence benefits the task of WSD.

Feature selection algorithms usually put a uniform prior over the features. I.e., they consider each feature to have the same probability of being selected. In this paper we relax this overly simplistic assumption by transferring a prior for feature relevance of a given word sense from "similar" word senses. Learning this prior for feature relevance of a test word sense makes those features that have been selected in the models of other "similar" word senses become more likely to be selected.

We learn the feature relevance prior only from distributionally similar word *senses*, rather than "all" senses of each word, as it is difficult to find words which are similar in "all" the senses. We can, however, often find words which have one or a few similar senses. For example, one sense of "fire" (as in "fire someone") should share features with one sense of "dismiss" (as in "dismiss someone"), but other senses of "fire" (as in "fire the gun") do not. Similarly, other meanings of "dismiss" (as in "dismiss an idea") should not share features with "fire".

As just mentioned, knowledge can only be fruitfully transfered between the shared senses of different words, even though the models being learned are for disambiguating different senses of a single word. To address this problem, we cluster similar word senses of different words, and then use the models learned for all but one of the word senses in the cluster (called the "training word senses") to put a feature relevance prior on which features will be more predictive for the held out test word sense. We hold out each word sense in the cluster once and learn a prior from the remaining word senses in that cluster. For example, we can use the models for discriminating the senses of the words "kill" and the senses of "capture", to put a prior on what features should be included in a model to disambiguate corresponding senses of the distributionally similar word “arrest”.

The remainder of the paper is organized as follows. In Section 2 we describe our “baseline” information theoretic feature selection method, and extend it to our “TRANSFEAT” method. Section 3 contains experimental results comparing TRANSFEAT with the baseline and Ando[CoNLL’06] on ONTONOTES data. We conclude with a brief summary in Section 4.

2 Feature Selection for WSD

We use an information theoretic approach to feature selection based on the Minimum Description Length (MDL) (Rissanen, 1999) principle, which makes it easy to incorporate information about feature relevance priors. These information theoretic models have a ‘dual’ Bayesian interpretation, which provides a clean setting for feature selection.

2.1 Information Theoretic Feature Selection

The state-of-the-art feature selection methods in WSD use either an ℓ_0 or an ℓ_1 penalty on the coefficients. ℓ_1 penalty methods such as Lasso, being convex, can be solved by optimization and give guaranteed optimal solutions. On the other hand, ℓ_0 penalty methods, like stepwise feature selection, give approximate solutions but produce models that are much sparser than the models given by ℓ_1 methods, which is quite crucial in WSD (Florian and Yarowsky, 2002). ℓ_0 models are also more amenable to theoretical analysis for setting thresholds, and hence for incorporating priors.

Penalized likelihood methods which are widely used for feature selection minimize a score:

$$Score = -2log(likelihood) + Fq \qquad (1)$$

where *F* is a function designed to penalize model complexity, and *q* represents the number of features currently included in the model at a given point. The first term in the above equation represents a measure of the in-sample error given the model, while the second term is a model complexity penalty.

As is obvious from Eq. 1, the description length of the MDL (Minimum Description Length) message is composed of two parts: S_E, the number of bits for encoding the residual errors given the models and S_M, the number of bits for encoding the model. Hence the description length can be written as: $S = S_E + S_M$. Now, when we evaluate a feature for possible addition to our model, we want to **maximize** the reduction of “description length” incurred by adding this feature to the model. This change in description length is: $\Delta S = \Delta S_E - \Delta S_M$; where $\Delta S_E \geq 0$ is the number of bits saved in describing residual error due to increase in the likelihood of the data given the new feature and $\Delta S_M > 0$ is the extra bits used for coding this new feature.

In our baseline feature selection model, we use the following coding schemes:

Coding Scheme for S_E :

The term S_E represents the cost of coding the residual errors given the models and can be written as:

$$S_E = -\log(P(y|w,x))$$

ΔS_E represents the increase in likelihood (in bits) of the data by adding this new feature to the model. We assume a Gaussian model, giving:

$$P(y|w,x) \sim \exp\left(-\left(\frac{\sum_{i=1}^{n}(y_i - w \cdot x_i)^2}{2\sigma^2}\right)\right)$$

where *y* is the response (word senses in our case), *x’s* are the features, *w’s* are the regression weights and σ^2 is the variance of the Gaussian noise.

Coding Scheme for ΔS_M : For describing S_M, the number of bits for encoding the model, we need the bits to code the index of the feature (i.e., which feature from amongst the total m candidate features) and the bits to code the coefficient of this feature.

The total cost can be represented as:

$$S_M = l_f + l_\theta$$

where l_f is the cost to code the index of the feature and l_θ is the number of bits required to code the coefficient of the selected feature.

In our baseline feature selection algorithm, we code l_f by using $log(m)$ bits (where m is the total number of candidate features), which is equivalent to the standard RIC (or the Bonferroni penalty) (Foster and George, 1994) commonly used in information theory. The above coding scheme[1] corresponds to putting a uniform prior over all the features; I.e., each feature is equally likely to get selected.

For coding the coefficients of the selected feature we use 2 bits, which is quite similar to the AIC

[1]There is a duality between Information Theory and Bayesian terminology: If there is $\frac{1}{k}$ probability of a fact being true, then we need $-log(\frac{1}{k}) = log(k)$ bits to code it.

(Akaike Information Criterion) (Rissanen, 1999). Our final equation for S_M is therefore:

$$S_M = log(m) + 2 \quad (2)$$

2.2 Extension to TRANSFEAT

We now extend the baseline feature selection algorithm to include the feature relevance prior. We define a binary random variable $f_i \in \{0,1\}$ that denotes the event of the i^{th} feature being in or not being in the model for the test word sense. We can parameterize the distribution as $p(f_i = 1|\theta_i) = \theta_i$. I.e., we have a Bernoulli Distribution over the features.

Given the data for the i^{th} feature for all the training word senses, we can write: $\mathcal{D}_i = \{f_{i1}, ..., f_{iv}, ..., f_{it}\}$. We then construct the likelihood functions from the data (under the i.i.d assumption) as:

$$p(\mathcal{D}_{f_i}|\theta_i) = \prod_{v=1}^{t} p(f_{iv}|\theta_i) = \prod_{v=1}^{t} \theta^{f_{iv}}(1-\theta_i)^{1-f_{iv}}$$

The posteriors can be calculated by putting a prior over the parameters θ_i and using Bayes rule as follows:

$$p(\theta_i|\mathcal{D}_{f_i}) = p(\mathcal{D}_{f_i}|\theta_i) \times p(\theta_i|a, b)$$

where a and b are the hyperparameters of the Beta Prior (conjugate of Bernoulli). The predictive distribution of θ_i is:

$$\begin{aligned} p(f_i = 1|\mathcal{D}_{f_i}) &= \int_0^1 \theta_i p(\theta_i|\mathcal{D}_{f_i}) d\theta_i = \mathbb{E}[\theta_i|\mathcal{D}_{f_i}] \\ &= \frac{k+a}{k+l+a+b} \quad (3) \end{aligned}$$

where k is the number of times that the i^{th} feature is selected and l is the complement of k, i.e. the number of times the i^{th} feature is not selected in the training data.

In light of above, the coding scheme, which incorporates the prior information about the predictive quality of the various features obtained from similar word senses, can be formulated as follows:

$$S_M = -\log\left(p(f_i = 1|\mathcal{D}_{f_i})\right) + 2$$

In the above equation, the first term represents the cost of coding the features, and the second term codes the coefficients. The negative signs appear due to the duality between Bayesian and Information-Theoretic representation, as explained earlier.

3 Experimental Results

In this section we present the experimental results of TRANSFEAT on ONTONOTES data.

3.1 Similarity Determination

To determine which verbs to transfer from, we cluster verb senses into groups based on the TF/IDF similarity of the vector of features selected for that verb sense in the baseline (non-transfer learning) model. We use only those features that are positively correlated with the given sense; they are the features most closely associated with the given sense. We cluster senses using a "foreground-background" clustering algorithm (Kandylas et al., 2007) rather than the more common k-means clustering because many word senses are not sufficiently similar to any other word sense to warrant putting into a cluster. Foreground-background clustering gives highly cohesive clusters of word senses (the "foreground") and puts all the remaining word senses in the "background". The parameters that it takes as input are the $\%$ of data points to put in "background" (i.e., what would be the singleton clusters) and a similarity threshold which impacts the number of "foreground" clusters. We experimented with putting 20% and 33% data points in background and adjusted the similarity threshold to give us $50 - 100$ "foreground" clusters. The results reported below have 20% background and $50 - 100$ "foreground" clusters.

3.2 Description of Data and Results

We performed our experiments on ONTONOTES data of 172 verbs (Hovy et al., 2006). The data consists of a rich set of linguistic features which have proven to be beneficial for WSD.

A sample feature vector for the word "add", given below, shows typical features.

```
word_added pos_vbd morph_normal
subj_use subjsyn_16993 dobj_money
dobjsyn_16993 pos+1+2+3_rp+to+cd
tp_account tp_accumulate tp_actual
```

The 172 verbs each had between 1,000 and 10,000 nonzero features. The number of senses varied from 2 (For example, "add") to 15 (For example, "turn").

We tested our transfer learning algorithm in three slightly varied settings to tease apart the contributions of different features to the overall performance. In our main setting, we cluster the word senses based on the “semantic + syntactic” features. In Setting 2, we do clustering based only on “semantic” features (topic features) and in Setting 3 we cluster based on only “syntactic” (pos, dobj etc.) features.

Table 1: 10-fold CV (microaveraged) accuracies of various methods for various Transfer Learning settings. **Note:** These are true cross-validation accuracies; No parameters have been tuned on them.

Method	Setting 1	Setting 2	Setting 3
TRANSFEAT	85.75	85.11	85.37
Baseline Feat. Sel.	83.50	83.09	83.34
SVM (Poly. Kernel)	83.77	83.44	83.57
Ando[CoNLL’06]	85.94	85.00	85.51
Most Freq. Sense	76.59	77.14	77.24

We compare TRANSFEAT against Baseline Feature Selection, Ando[CoNLL’06], SVM (libSVM package) with a cross-validated polynomial kernel and a simple most frequent sense baseline. We tuned the “d” parameter of the polynomial kernel using a separate cross validation.

The results for the different settings are shown in Table 1 and are significantly better at the 5% significance level (Paired t-test) than the baseline feature selection algorithm and the SVM. It is comparable in accuracy to Ando[CoNLL’06]. Settings 2 and 3, in which we cluster based on only “semantic” or “syntactic” features, respectively, also gave significant (5% level in a Paired t-Test) improvement in accuracy over the baseline and SVM model. But these settings performed slightly worse than Setting 1, which suggests that it is a good idea to have clusters in which the word senses have “semantic” as well as “syntactic” distributional similarity.

Some examples will help to emphasize the point that we made earlier that transfer helps the most in cases in which the target word sense has much less data than the word senses from which knowledge is being transferred. “kill” had roughly 6 times more data than all other word senses in its cluster (i.e., “arrest”, “capture”, “strengthen”, etc.) In this case, TRANSFEAT gave $3.19 - 8.67\%$ higher accuracies than competing methods[2] on these three words. Also, for the case of word “do,” which had roughly 10 times more data than the other word senses in its cluster (E.g., “die” and “save”), TRANSFEAT gave $4.09 - 6.21\%$ higher accuracies than other methods. Transfer makes the biggest difference when the target words have much less data than the word senses they are generalizing from, but even in cases where the words have similar amounts of data we still get a $1.5 - 2.5\%$ increase in accuracy.

[2]TRANSFEAT does better than Ando[CoNLL’06] on these words even though on average over all 172 verbs, the difference is slender.

4 Summary

This paper presented a Transfer Learning formulation which learns a prior suggesting which features are most useful for disambiguating ambiguous words. Successful transfer requires finding similar word senses. We used “foreground/background” clustering to find cohesive clusters for various word senses in the ONTONOTES data, considering both “semantic” and “syntactic” similarity between the word senses. Learning priors on features was found to give significant accuracy boosts, with both syntactic and semantic features contributing to successful transfer. Both feature sets gave substantial benefits over the baseline methods that did not use any transfer and gave comparable accuracy to recent Transfer Learning methods like Ando[CoNLL’06]. The performance improvement of our Transfer Learning becomes even more pronounced when the word senses that we are generalizing from have more observations than the ones that are being learned.

References

R. Ando and T. Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 6:1817–1853.

R. Ando. 2006. Applying alternating structure optimization to word sense disambiguation. In *(CoNLL)*.

R. Florian and D. Yarowsky. 2002. Modeling consensus: classifier combination for word sense disambiguation. In *EMNLP ’02*, pages 25–32.

D. P. Foster and E. I. George. 1994. The risk inflation criterion for multiple regression. *The Annals of Statistics*, 22(4):1947–1975.

E. H. Hovy, M. P. Marcus, M. Palmer, L. A. Ramshaw, and R. M. Weischedel. 2006. Ontonotes: The 90% solution. In *HLT-NAACL*.

V. Kandylas, S. P. Upham, and L. H. Ungar. 2007. Finding cohesive clusters for analyzing knowledge communities. In *ICDM*, pages 203–212.

J. Rissanen. 1999. Hypothesis selection and testing by the mdl principle. *The Computer Journal*, 42:260–269.