Transfer Learning, Feature Selection and Word Sense Disambiguation

Paramveer S. Dhillon and Lyle Ungar

Computer and Information Science
University of Pennsylvania, Philadelphia, PA, U.S.A

ACL 2009

Outline

The Problem

- Lots of work has been done for WSD in supervised settings, but...
- State-of-the art WSD systems suffer due to paucity of labeled data, For e.g. SENSEVAL-2 task has only 10 labeled examples per sense.
  - Obtaining labeled data is an expensive proposition
  - With limited data it is difficult to build high accuracy supervised learning models (See the winning entries in SENSEVAL-2,3!)
- Can we overcome this data bottleneck ?

The Problem

- Yes... one alternative is to use other sources of data to improve performance, for e.g. Semi -Supervised Learning augments labeled data with unalabeled data.

The Problem

- Yes... one alternative is to use other sources of data to improve performance, for e.g. Semi -Supervised Learning augments labeled data with unalabeled data.
- Or.. use labeled data from other “similar learning tasks" (which have more labeled data available) and build shared models i.e. **Transfer Learning** *[Thrun, ’96; Caruana, ’97; Baxter, ’97; . . .]*
- Ando [CoNLL ’06] have applied Transfer Learning techniques (ASO Alternating Structures Optimization) to WSD .

Our Contribution in a nutshell

- We present a model based on the information theoretic Minimum Description Length (MDL) principle called **TransFeat**.
- Since, Feature Selection is also an important aspect of WSD *[Florian and Yarowsky '02]* so our model integrates Feature Selection with Transfer Learning.
- Besides this it allows us to use an "arbitrary" similar metric when defining similarity among word senses.

Our Contribution in a nutshell

- **Our approach in three steps:**
  1. We break the problem into disambiguation at the level of senses of a word rather than at the level of complete words.
  2. Use data from “similar” words senses to learn a **Feature Relevance Prior** i.e. decide which features are more likely to get selected for a “test" word sense.
  3. Once we have selected the features (using this informed **Feature Relevance Prior**), use a logistic classifier to classify.

Our Contribution in a nutshell

- An example will help to elucidate the concept:

Our Contribution in a nutshell

- An example will help to elucidate the concept:
  1. Words “kill”, “capture”, “arrest” have 4-6 times more labeled data (OntoNotes Dataset) than similar words like “strengthen”.
  2. We use the features for the similar senses of “kill”, “capture”, “arrest” to learn a feature relevance prior for “strengthen”.
  3. The improved Feature Selection (due to a non-uniform prior over the features) coupled with a proper classifier leads to improved accuracy over the baseline and the state-of-the-art.

Outline of the Algorithm

- Learn a separate “baseline” (one-vs-all) model (explained later) for each sense of a word.
- Cluster word senses based on positively correlated features.

Outline of the Algorithm

- Learn a separate “baseline” (one-vs-all) model (explained later) for each sense of a word.
- Cluster word senses based on positively correlated features.
- For each “target” word sense to be predicted, use the selected features of other $(n-1)$ word senses in its cluster to estimate the probability of the features being relevant for disambiguating the “target" word sense.

Outline of the Algorithm

- Learn a separate “baseline” (one-vs-all) model (explained later) for each sense of a word.
- Cluster word senses based on positively correlated features.
- For each “target” word sense to be predicted, use the selected features of other $(n-1)$ word senses in its cluster to estimate the probability of the features being relevant for disambiguating the “target" word sense.
- Combine all these “improved models” learnt at the sense level; and disambiguate the word as a whole by picking the predicted sense as the one whose model gave the best score.

Feature Selection and MDL Background

- Our approach is based on a version of ℓ_0 penalty based methods, called penalized likelihood methods.

Feature Selection and MDL Background

- Our approach is based on a version of ℓ_0 penalty based methods, called penalized likelihood methods.
- In our case the MDL message (S) is composed of two parts S_E and S_M.
- $S_E \longmapsto$ # bits for encoding the residual errors given the model
  $S_M \longmapsto$ # bits for encoding the model.

Feature Selection and MDL Background

- Our approach is based on a version of ℓ_0 penalty based methods, called penalized likelihood methods.
- In our case the MDL message (S) is composed of two parts S_E and S_M.
- $S_E \longmapsto$ # bits for encoding the residual errors given the model
  $S_M \longmapsto$ # bits for encoding the model.
- The goal at each step is to maximize the following:
  $\Delta S_j = \Delta S_{jE} - \Delta S_{jM}$
  i.e. Reduction in description length by adding a feature 'j' to the model

Coding Schemes used

- S_{jE} = We use a Gaussian likelihood model.

Coding Schemes used

- S_{jE} = We use a Gaussian likelihood model.
- $S_{jM} = l_f + l_\theta$
- Coding cost for:
  - The index of the feature $\longmapsto l_f$
  - The coefficient of the feature $\longmapsto l_\theta$

Simple ℓ_0 regression - “Baseline”

- Assume a simple linear regression model: $Y = WX + \epsilon$

Simple ℓ_0 regression - "Baseline"

- Assume a simple linear regression model: $Y = WX + \epsilon$
- In this case: S_E = Gaussian likelihood model.
- $S_M = log(m) + 2$ i.e. Bits to code the feature (RIC penalty) and its coefficient (AIC Penalty) *[Risannen '83, '99]*.

Simple ℓ_0 regression - "Baseline"

- Assume a simple linear regression model: $Y = WX + \epsilon$
- In this case: S_E = Gaussian likelihood model.
- $S_M = log(m) + 2$ i.e. Bits to code the feature (RIC penalty) and its coefficient (AIC Penalty) *[Risannen '83, '99]*.
- Equivalently we can write: $-log(P(f_i = 1)) = -\log\left(\frac{1}{m}\right)$; i.e. a uniform prior over all the features.

TransFeat Formulation

- Likelihood $\longmapsto$ Bernoulli (either a feature is selected or not)
- Prior $\longmapsto$ Beta (conjugate of Bernoulli)
- Posterior = Likelihood $\times$ Prior

TransFeat Formulation

- Likelihood $\longmapsto$ Bernoulli (either a feature is selected or not)
- Prior $\longmapsto$ Beta (conjugate of Bernoulli)
- Posterior = Likelihood $\times$ Prior
- The predictive distribution can be obtained as:
  $p(f_i = 1|\mathcal{D}_i) = \frac{k+a}{k+l+a+b}$
  $k \longmapsto$ # times that the i^{th} feature is selected (in the models of "similar" tasks)
  $l \longmapsto$ # times that the i^{th} feature is not selected
  $a, b \longmapsto$ Hyperparameters of the Beta Prior.
  $\mathcal{D}_{f_i} \longmapsto$ Data for the i^{th} feature.
- Refer the Paper for detailed analysis

TransFeat Formulation

- The new cost of coding the model for the test-task is:

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

TransFeat Formulation

- The new cost of coding the model for the test-task is:

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

- If we choose the hyperparameters $a = 1$ and $b = m - 1$ where m is the total number of features, then in the case of no transfer i.e. $k = l = 0$, the above equation reduces to the baseline.

TransFeat Formulation

- The new cost of coding the model for the test-task is:

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

- If we choose the hyperparameters $a = 1$ and $b = m - 1$ where m is the total number of features, then in the case of no transfer i.e. $k = l = 0$, the above equation reduces to the baseline.
- Intuitively we have relaxed the "harsh" assumption of $p(f_i = 1) = \frac{1}{m}$; and replaced it with a more liberal assumption by transferring from similar tasks.

Experimental Setup

- Set of 172 verbs *[Schuler and Palmer '06]*, with varying number of senses and data (OntoNOTES data–coarse grained senses).

Experimental Setup

- Set of 172 verbs *[Schuler and Palmer '06]*, with varying number of senses and data (OntoNOTES data–coarse grained senses).
- Since, most of the word senses are singleton and may not be similar to other word senses; we use Foreground- Background clustering *[Kandylas et. al. '07]* which puts all the singleton words into background.

Experimental Setup

- The features that we used were quite similar to the features used by *[Chen, Dligach & Palmer '07]*.
    - Local and Global Context, POS tags, wordnet syns, hypernyms etc.
    - Besides this there were some novel features like "topic features".
- A typical feature vector looked like:

```
word_added pos_vbd morph_normal subj_use
subjsyn_16840 subjsyn_17218 subjsyn_11365663
dobjsyn_16993 prt_up prep_to to_money tosyn_16993
word-2_foreign pos-2_jj word-1_investments
pos-1_nns word+1_up pos+1_rp tp_account tp_actual
tp_advance tp_border tp_city
```

Experimental Results (OntoNOTES Dataset)

Table: 10-fold CV (test) accuracies of various methods *Note*: In each setting we only disambiguated the verbs, if atleast one of their senses fell into foreground

Method	Setting 1	Setting 2	Setting 3
TransFeat	85.8	85.1	85.4
Ando *[CoNLL '06]*	85.9	85.0	85.5
SVM Poly. Kernel	83.8	83.4	83.6
Baseline Feat. Selection	83.5	83.1	83.3
Most. Freq. Sense	76.6	77.1	77.2

- In Setting 1, 2 and 3 we clustered word senses based on “semantic+syntactic",“only syntactic" and “only syntactic" similarity respectively.
- All results are significant at (5% level, Paired t-test)

Conclusion

- TransFeat provides an elegant way to introduce prior information from related tasks.
- Gave results which are comparable or better than the state of the art.
- The performance is affected little by introducing different linguistic factors like (syntax and semantics) into clustering
- Information Theoretic MDL methods capture the spirit of Bayesian priors without the strong commitment to specific models.

Thanks

Thanks