Zipfian corruptions for robust POS tagging

Anders Søgaard
Center for Language Technology
University of Copenhagen
soegaard@hum.ku.dk

Abstract

Inspired by robust generalization and adversarial learning we describe a novel approach to learning structured perceptrons for part-of-speech (POS) tagging that is less sensitive to domain shifts. The objective of our method is to minimize average loss under random distribution shifts. We restrict the possible target distributions to mixtures of the source distribution and random Zipfian distributions. Our algorithm is used for POS tagging and evaluated on the English Web Treebank and the Danish Dependency Treebank with an average 4.4% error reduction in tagging accuracy.

1 Introduction

Supervised learning approaches have advanced the state of the art on a variety of tasks in natural language processing, often resulting in systems approaching the level of inter-annotator agreement on in-domain data, e.g. in POS tagging, where Shen et al. (2007) report a tagging accuracy of 97.3%. However, performance of state-of-the-art supervised systems is known to drop considerably on out-of-domain data. State-of-the-art POS taggers trained on the Penn Treebank (Marcus et al., 1993) mapped to Google's universal tag set (Petrov et al., 2011) achieve tagging accuracies in the range of 89–91% on Web 2.0 data (Petrov and McDonald, 2012) .

To bridge this gap we may consider using semi-supervised or transfer learning methods to adjust to new target domains (Blitzer et al., 2006; Daume III, 2007), pooling unlabeled data from those domains. However, in many applications this is not possible. If we want to provide an online service or design a piece of software with many potential users covering a wide range of use cases, we do not know the target domain in advance. This is the usual problem of *robust* learning, but in this paper we describe a novel learning algorithm that goes beyond robust learning by making various assumptions about the difference between the source domain and the (unknown) target domain. Under these assumptions we can minimize average loss under (all possible or a representative sample of) domain shifts. We evaluate our approach on two recently introduced cross-domain POS tagging datasets.

Our approach is inspired by work in robust generalization (Ben-Tal and Nemirovski, 1998; Trafalis and Gilbert, 2007) and adversarial learning (Globerson and Roweis, 2006; Dekel and Shamir, 2008; Søgaard and Johannsen, 2012). Our approach also bears similarities to feature bagging (Sutton et al., 2006). Sutton et al. (2006) noted that in learning of linear models useful features are often swamped by correlating, but more indicative features. If the more indicative features are absent in the target domain due to out-of-vocabulary (OOV) effects, we are left with the swamped features which were not updated properly. This is, indirectly, the problem solved in adversarial learning with corrupted data points. Adversarial learning can also be seen as a way of averaging exponentially many models (Hinton et al., 2012).

Adversarial learning techniques have been developed for security-related learning tasks, e.g. where systems need to be robust to failing sensors. We also show how we can do better than straight-forward ap-

Proceedings of NAACL-HLT 2013, pages 668–672,
Atlanta, Georgia, 9–14 June 2013.

plication of adversarial learning techniques by making a second assumption about our data, namely that domains are mixtures of Zipfian distributions over our features. Similar assumptions have been made before in computational linguistics, e.g. by Goldberg and Elhadad (2008).

2 Approach overview

In this paper we consider the structured perceptron (Collins, 2002) – with POS tagging as our practical application. The structured perceptron is prone to feature swamping (Sutton et al., 2006), and we want to prevent that using a technique inspired by adversarial learning (Globerson and Roweis, 2006; Dekel and Shamir, 2008). The modification presented here to the structured perceptron only affects a single line of code in a publicly available implementation (see below), but the consequences are significant.

Online adversarial learning (Søgaard and Johannsen, 2012), briefly, works by sampling random corruptions of our data, or random feature deletions, in the learning phase. A discriminative learner seeing corrupted data points with missing features will not update part of the model and will thus try to find a decision boundary classifying the training data correctly relying on the remaining features. This decision boundary may be very different from the decision boundary found otherwise by the discriminative learner. If we sample enough corruptions, the model learned from the corrupted data will converge on the model minimizing average loss over all corruptions (Dekel and Shamir, 2008).

Example Consider the plot in Figure 1. The solid line with no stars (2d-fit) is the SVM fit in two dimensions, while the dashed line is what that fit amounts to if the feature x is missing in the target. The solid line with stars (1d-fit) is our fit if we could predict the missing feature, training an SVM only with the y feature. The 1d-fit decision boundary only misclassifies a single data point compared to the original fit which misclassifies more than 15 negatives with the x feature missing.

The plot thus shows that the best fit in m dimensions is often not the best in $< m$ dimensions. Consequently, if we think there is a risk that features will be missing in the target, finding the best fit in m dimensions is not necessarily the best we can do. Of course we do not know what features will be missing in advance. The intuition in adversarial learning is that we may obtain more robust decision boundaries by minimizing loss over a set of possible feature deletions. We extend this idea below, modeling not only OOV effects, but a broader class of distributional shifts.

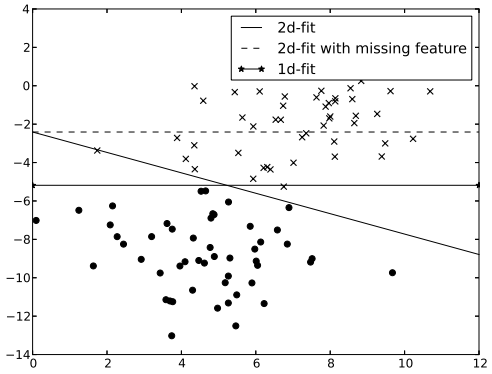

Figure 1: The best fit in m dimensions is often not the best in $< m$ dimensions.

3 Structured perceptron

The structured perceptron (Collins, 2002) models sequences as Markov chains of unobserved variables (POS), each emitting an observed variable (a word form). The structured perceptron is similar to the averaged perceptron (Freund and Schapire, 1999), except data points are sequences of vectors rather than just vectors. Consequently, the structured perceptron does not predict a class label but a sequence of labels (using Viterbi decoding). In learning we update the features at the positions where the predicted labels are different from the true labels. We do this by adding weight to features present in the correct solution and subtracting weight from features only present in the predicted solution. The generic averaged perceptron learning algorithm is presented in Figure 2. A publicly available and easy-to-modify Python reimplementation of the structured perceptron can be found in the LXMLS toolkit.[1] We use the LXMLS toolkit as our baseline with the default feature model, but use the PTB tagset rather than the Google tagset (Petrov et al., 2011) used by default in the LXMLS toolkit.

[1] https://github.com/gracaninja/lxmls-toolkit

```
1: X = {⟨y_i, x_i⟩}_{i=1}^N
2: w^0 = 0, v = 0, i = 0
3: for k ∈ K do
4:    for n ∈ N do
5:       if sign(w · x) ≠ y_n then
6:          w^{i+1} ← update(w^i)
7:          i ← i + 1
8:       end if
9:       v ← v + w^i
10:   end for
11: end for
12: return w = v/(N × K)
```

Figure 2: Generic averaged perceptron

```
1: X = {⟨y_i, x_i⟩}_{i=1}^N
2: w^0 = 0, v = 0, i = 0
3: for k ∈ K do
4:    for n ∈ N do
5:       ξ ← random.zipf(3, M)
6:       if sign(w · x ∘ ξ) ≠ y_n then
7:          w^{i+1} ← update(w^i)
8:          i ← i + 1
9:       end if
10:      v ← v + w^i
11:   end for
12: end for
13: return w = v/(N × K)
```

Figure 3: Z_3SP

4 Minimizing loss under OOV effects

We will think of domain shifts as data point corruptions. Søgaard and Johannsen (2012) model domain shifts using binary vectors of length m where m is the size of of our feature representation. Each vector then represents an expected OOV effect by encoding what features are (predicted to be) missing in the target data, i.e. the ith feature will be missing if the ith element of the binary vector is 0. However, since we are minimizing average loss under OOV effects it makes sense to restrict the class of vectors to encode OOV effects that we are likely to observe. This could, for example, involve fixing an expected rate of missing features or bounding it by some interval, or it could involve distinguishing between features that are likely to be missing in the target and features that are not. Here is what we do in this paper:

Rather than thinking of domain shifts as something that deletes features, we propose to see domain shifts as something making certain features less likely to occur in our data. We will in other words simulate *soft* OOV effects, rather than hard OOV effects. One way to think of this is as an importance weighting of our features. This section provides some intuition for using inverse Zipfian distributions as weight functions.

Say we are interested in making a model $\theta_{\mathcal{D}_1}$ learned from a known distribution $\mathcal{D}_1$ robust against the distributional differences between $\mathcal{D}_1$ and an unknown distribution $\mathcal{D}_2$. These two distributions are somehow related to a distribution $\mathcal{D}_0$ (the underlying language distribution from which the domain distributions are sampled).

It is common to assume that linguistic distributions follow power laws (Zipf, 1935; Goldberg and Elhadad, 2008). We will assume that $\mathcal{D}_1 = \mathcal{D}_0 \times \mathcal{Z}_1$ where $\mathcal{Z}_1$ is some Zipfian distribution. Say $\mathcal{D}_0 \sim \mathcal{Z}_0$ is the master Zipfian distribution of language $\mathcal{L}_0$. If we assume that (otherwise independent) domains $\mathcal{L}_1$ and $\mathcal{L}_2$ follow products of Zipfians $\mathcal{Z}_0 \times \mathcal{Z}_1$ and $\mathcal{Z}_0 \times \mathcal{Z}_2$, we derive the following:

Say $\mathbf{w} = \theta_{\mathcal{Z}_0 \times \mathcal{Z}_1}$ is the model learned from the source data. The ideal model is $\mathbf{w}' = \theta_{\mathcal{Z}_0 \times \mathcal{Z}_2}$, but both Zipfians $\mathcal{Z}_1$ and $\mathcal{Z}_2$ are unknown. Since $\mathcal{Z}_2$ is unknown (and in many applications, we want to model several $\mathcal{Z}_i$), the overall best model we can hope for is $\mathbf{w}' = \theta_{\mathcal{Z}_0}$. $\mathcal{Z}_0$ is also unknown, but we can observe a finite sample $\mathcal{Z}_0 \times \mathcal{Z}_1$. Since the density of $\mathcal{Z}_1$ is directly related to the weights in $\mathbf{w}$, a crude estimate of $\theta_{\mathcal{Z}_0}$ would be $\mathbf{w}' \sim \mathbf{w}\frac{1}{\mathcal{Z}_1}$. Since we cannot observe $\mathcal{Z}_1$, we instead try to minimize average loss under all hypotheses about $\mathcal{Z}_1$.

In practice, we implement the idea of reweighting by random inverse Zipfian distributitons (instead of binary vectors) in the following way: Passing through the data in averaged perceptron learning (Figure 2), we consider one data point at a time. In order to minize loss in all possible domains, we need to consider all possible inverse Zipfian reweightings. This would be possible if we provided a convex formulation of the minimization problem along the lines of Dekel and Shamir (2008), but instead we randomly sample from a Zipfian and factor its inverse into our dataset. The parameter of the Zipfians is set (to 3) on development data (the EWT-email development data). The modified learning algorithm, Z_3SP, is presented in Figure 3.

5 POS tagging

POS tagging is the problem of assigning syntactic categories or POS to tokenized word forms in running text. Most approaches to POS tagging use supervised learning to learn sequence labeling models from annotated ressources. The major ressource for English is the Wall Street Journal (WSJ) sections of the English Treebank (Marcus et al., 1993). POS taggers are usually trained on Sect. 0–18 and evaluated on Sect. 22–24. In this paper we are not interested in in-domain performance on WSJ data, but rather in developing a robust POS tagger that is less sensitive to domain shifts than current state-of-the-art POS taggers and use the splits from a recent parsing shared task rather than the standard POS tagging ones.

6 Experiments

We train our tagger on Sections 2–21 of the WSJ sections of the English Treebank, in the Ontotes 4.0 release. This was also the training data used in the experiments in the Parsing the Web (PTW) shared task at NAACL 2012.[2] In the shared task they used the coarse-grained Google tagset (Petrov et al., 2011). We believe this tagset is too coarse-grained for most purposes (Manning, 2011) and do experiments with the original PTB tagset instead.

Our evaluation data comes from the English Web Treebank (EWT),[3] which was also used in the PTW shared task. The EWT contains development and evaluation data for five domains: answers (from Yahoo!), emails (from the Enron corpus), BBC newsgroups, Amazon reviews, and weblogs. In order not to optimize on in-domain data, we tune on the Email development data and evaluate on the remaining domains (the test sections).

The Web 2.0 data used for evaluation contains a lot of non-canonical language use. An example is the sentence *you r retarded.* from the Email section. The POS tagger finds no support for *r* as a verb in the training data, but needs to infer this from the context.

We also include experiments on the Danish Dependency Treebank (DDT) (Buch-Kromann, 2003), which comes with meta-data enabling us to single out four domains: newspaper, law, literature and magazines. We train our tagger on the newspaper data and evaluate on the remaining three sections.

[2]https://sites.google.com/site/sancl2012/home/shared-task

[3]LDC Catalog No.: LDC2012T13.

	SP	BSP	Z_3SP
EWT-answers	85.22	85.45	**85.59**
EWT-newsgroups	86.82	86.94	**87.42**
EWT-reviews	84.92	85.14	85.67
EWT-weblogs	87.00	87.06	**87.39**
DDT-law	92.38	92.80	**93.35**
DDT-lit	93.61	93.80	**93.85**
DDT-mag	**94.71**	94.44	94.68

Table 1: Results. BSP samples binary vectors with probabilities $\{0 : 0.1, 1 : 0.9\}$

6.1 Results

The results are presented in Table 1. We first note that improvements over the structured perceptron are statistically significant with $p < 0.01$ across all domains, except DDT-mag. We also note that using inverse Zipfian reweightings is better than using binary vectors in almost all cases. We believe that these are strong results given that we are assuming *no* knowledge of the target domain, and our modification of the learning algorithm does not affect computational efficiency at training or test time. The average error reduction of Z_3SP over the structured perceptron (SP) is 8%. Since using inverse Zipfian reweightings seems more motivated for node potentials than for edge potentials, we also tried using BSP for edge potentials and Z_3SP for node potentials. This mixed model acchieved 93.70, 93.91 and 94.35 on the DDT data, which on average is slightly better than Z_3SP.

7 Conclusions

Inspired by robust generalization and adversarial learning we introduced a novel approach to learning structured perceptrons for sequential labeling, which is less sensitive to OOV effects. We evaluated our approach on POS tagging data from the EWT and the DDT with an average 4.4% error reduction over the structured perceptron.

Acknowledgements

Anders Søgaard is funded by the ERC Starting Grant LOWLANDS No. 313695.

References

Aharon Ben-Tal and Arkadi Nemirovski. 1998. Robust convex optimization. *Mathematics of Operations Research*, 23(4).

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

Matthias Buch-Kromann. 2003. The Danish Dependency Treebank and the DTAG Treebank Tool. In *TLT*.

Michael Collins. 2002. Discriminative training methods for Hidden Markov Models. In *EMNLP*.

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

Ofer Dekel and Ohad Shamir. 2008. Learning to classify with missing and corrupted features. In *ICML*.

Yoav Freund and Robert Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37:277–296.

Amir Globerson and Sam Roweis. 2006. Nightmare at test time: robust learning by feature deletion. In *ICML*.

Yoav Goldberg and Michael Elhadad. 2008. splitSVM: fast, space-efficient, non-heuristic, polynomial kernel computation for NLP applications. In *ACL*.

Geoffrey Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. http://arxiv.org/abs/1207.0580.

Chris Manning. 2011. Part-of-speech tagging from 97%˜to 100%: Is it time for some linguistics? In *CICLing*.

Mitchell Marcus, Mary Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 Shared Task on Parsing the Web. In *Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL)*.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2011. A universal part-of-speech tagset. CoRR abs/1104.2086.

Libin Shen, Giorgio Satta, and Aravind Joshi. 2007. Guided learning for bidirectional sequence classification. In *ACL*.

Anders Søgaard and Anders Johannsen. 2012. Robust learning in random subspaces: equipping NLP against OOV effects. In *COLING*.

Charles Sutton, Michael Sindelar, and Andrew McCallum. 2006. Reducing weight undertraining in structured discriminative learning. In *NAACL*.

T Trafalis and R Gilbert. 2007. Robust support vector machines for classification and computational issues. *Optimization Methods and Software*, 22:187–198.

George Zipf. 1935. *The psycho-biology of language*. Houghton Mifflin.