SVD and Clustering for Unsupervised POS Tagging

Michael Lamar*
Division of Applied Mathematics
Brown University
Providence, RI, USA
mlamar@dam.brown.edu

Yariv Maron*
Gonda Brain Research Center
Bar-Ilan University
Ramat-Gan, Israel
syarivm@yahoo.com

Mark Johnson
Department of Computing
Faculty of Science
Macquarie University
Sydney, Australia
mjohnson@science.mq.edu.au

Elie Bienenstock
Division of Applied Mathematics
and Department of Neuroscience
Brown University
Providence, RI, USA
elie@brown.edu

Abstract

We revisit the algorithm of Schütze (1995) for unsupervised part-of-speech tagging. The algorithm uses reduced-rank singular value decomposition followed by clustering to extract latent features from context distributions. As implemented here, it achieves state-of-the-art tagging accuracy at considerably less cost than more recent methods. It can also produce a range of finer-grained taggings, with potential applications to various tasks.

1 Introduction

While supervised approaches are able to solve the part-of-speech (POS) tagging problem with over 97% accuracy (Collins 2002; Toutanova et al. 2003), unsupervised algorithms perform considerably less well. These models attempt to tag text without resources such as an annotated corpus, a dictionary, etc. The use of singular value decomposition (SVD) for this problem was introduced in Schütze (1995). Subsequently, a number of methods for POS tagging without a dictionary were examined, e.g., by Clark (2000), Clark (2003), Haghighi and Klein (2006), Johnson (2007), Goldwater and Griffiths (2007), Gao and Johnson (2008), and Graça et al. (2009). The latter two, using Hidden Markov Models (HMMs), exhibit the highest performances to date for fully unsupervised POS tagging.

The revisited SVD-based approach presented here, which we call "two-step SVD" or SVD2, has four important characteristics. First, it achieves state-of-the-art tagging accuracy. Second, it requires drastically less computational effort than the best currently available models. Third, it demonstrates that state-of-the-art accuracy can be realized without disambiguation, *i.e.*, without attempting to assign different tags to different tokens of the same type. Finally, with no significant increase in computational cost, SVD2 can create much finer-grained labelings than typically produced by other algorithms. When combined with some minimal supervision in post-processing, this makes the approach useful for tagging languages that lack the resources required by fully supervised models.

2 Methods

Following the original work of Schütze (1995), we begin by constructing a right context matrix, R, and a left context matrix, L. R_{ij} counts the number of times in the corpus a token of word type i is immediately followed by a token of word type j. Similarly, L_{ij} counts the number of times a token of type i is preceded by a token of type j. We truncate these matrices, including, in the right and left contexts, only the w_1 most frequent word types. The resulting L and R are of dimension $N_{\text{types}} \times w_1$, where N_{types} is the number of word types (spelling forms) in the corpus, and w_1 is set to 1000. (The full $N_{\text{types}} \times N_{\text{types}}$ context matrices satisfy $R = L^{\text{T}}$.)

* These authors contributed equally.

Proceedings of the ACL 2010 Conference Short Papers, pages 215–219,
Uppsala, Sweden, 11-16 July 2010.

Next, both context matrices are factored using singular value decomposition:

$$L = U_L S_L V_L^T$$
$$R = U_R S_R V_R^T.$$

The diagonal matrices S_L and S_R (each of rank 1000) are reduced down to rank $r_1 = 100$ by replacing the 900 smallest singular values in each matrix with zeros, yielding S_L^* and S_R^*. We then form a pair of latent-descriptor matrices defined by:

$$L^* = U_L S_L^*$$
$$R^* = U_R S_R^*.$$

Row i in matrix L^* (resp. R^*) is the left (resp. right) latent descriptor for word type i. We next include a normalization step in which each row in each of L^* and R^* is scaled to unit length, yielding matrices L^{**} and R^{**}. Finally, we form a single descriptor matrix D by concatenating these matrices into $D = [L^{**} R^{**}]$. Row i in matrix D is the complete latent descriptor for word type i; this latent descriptor sits on the Cartesian product of two 100-dimensional unit spheres, hereafter the *2-sphere*.

We next categorize these descriptors into $k_1 = 500$ groups, using a k-means clustering algorithm. Centroid initialization is done by placing the k initial centroids on the descriptors of the k most frequent words in the corpus. As the descriptors sit on the 2-sphere, we measure the proximity of a descriptor to a centroid by the dot product between them; this is equal to the sum of the cosines of the angles—computed on the left and right parts—between them. We update each cluster's centroid as the weighted average of its constituents, the weight being the frequency of the word type; the centroids are then scaled, so they sit on the 2-sphere. Typically, only a few dozen iterations are required for full convergence of the clustering algorithm.

We then apply a second pass of this entire SVD-and-clustering procedure. In this second pass, we use the $k_1 = 500$ clusters from the first iteration to assemble a new pair of context matrices. Now, R_{ij} counts all the cluster-j (j=1… k_1) words to the right of word i, and L_{ij} counts all the cluster-j words to the left of word i. The new matrices L and R have dimension $N_{types} \times k_1$.

As in the first pass, we perform reduced-rank SVD, this time down to rank $r_2 = 300$, and we again normalize the descriptors to unit length, yielding a new pair of latent descriptor matrices L^{**} and R^{**}. Finally, we concatenate L^{**} and R^{**} into a single matrix of descriptors, and cluster these descriptors into k_2 groups, where k_2 is the desired number of induced tags. We use the same weighted k-means algorithm as in the first pass, again placing the k initial centroids on the descriptors of the k most frequent words in the corpus. The final tag of any token in the corpus is the cluster number of its type.

3 Data and Evaluation

We ran the SVD2 algorithm described above on the full Wall Street Journal part of the Penn Treebank (1,173,766 tokens). Capitalization was ignored, resulting in N_{types} = 43,766, with only a minor effect on accuracy. Evaluation was done against the POS-tag annotations of the 45-tag PTB tagset (hereafter PTB45), and against the Smith and Eisner (2005) coarse version of the PTB tagset (hereafter PTB17). We selected the three evaluation criteria of Gao and Johnson (2008): M-to-1, 1-to-1, and VI. M-to-1 and 1-to-1 are the tagging accuracies under the best many-to-one map and the greedy one-to-one map respectively; VI is a map-free information-theoretic criterion—see Gao and Johnson (2008) for details. Although we find M-to-1 to be the most reliable criterion of the three, we include the other two criteria for completeness.

In addition to the best M-to-1 map, we also employ here, for large values of k_2, a *prototype-based* M-to-1 *map*. To construct this map, we first find, for each induced tag t, the word type with which it co-occurs most frequently; we call this word type the *prototype* of t. We then query the annotated data for the most common gold tag for each prototype, and we map induced tag t to this gold tag. This prototype-based M-to-1 map produces accuracy scores no greater—typically lower—than the best M-to-1 map. We discuss the value of this approach as a minimally-supervised post-processing step in Section 5.

4 Results

Low-*k* performance. Here we present the performance of the SVD2 model when k_2, the number of induced tags, is the same or roughly the same as the number of tags in the gold standard—hence small. Table 1 compares the performance of SVD2 to other leading models. Following Gao and Johnson (2008), the number of induced tags is 17 for PTB17 evaluation and 50 for PTB45 evaluation. Thus, with the exception of Graça et al. (2009) who use 45 induced tags for PTB45, the number of induced tags is the same across each column of Table 1.

	M-to-1		1-to-1		VI	
Model	**PTB17**	**PTB45**	**PTB17**	**PTB45**	**PTB17**	**PTB45**
SVD2	**0.730**	**0.660**	0.513	0.467	**3.02**	**3.84**
HMM-EM	0.647	0.621	0.431	0.405	3.86	4.48
HMM-VB	0.637	0.605	0.514	0.461	3.44	4.28
HMM-GS	0.674	**0.660**	0.466	**0.499**	3.46	4.04
HMM-Sparse(32)	0.702(2.2)	0.654(1.0)	0.495	0.445		
VEM (10^{-1},10^{-1})	0.682(0.8)	0.546(1.7)	**0.528**	0.460		

Table 1. Tagging accuracy under the best M-to-1 map, the greedy 1-to-1 map, and VI, for the full PTB45 tagset and the reduced PTB17 tagset. HMM-EM, HMM-VB and HMM-GS show the best results from Gao and Johnson (2008); HMM-Sparse(32) and VEM (10^{-1},10^{-1}) show the best results from Graça et al. (2009).

The performance of SVD2 compares favorably to the HMM models. Note that SVD2 is a deterministic algorithm. The table shows, in parentheses, the standard deviations reported in Graça et al. (2009). For the sake of comparison with Graça et al. (2009), we also note that, with $k_2 = 45$, SVD2 scores 0.659 on PTB45. The NVI scores (Reichart and Rappoport 2009) corresponding to the VI scores for SVD2 are 0.938 for PTB17 and 0.885 for PTB45. To examine the sensitivity of the algorithm to its four parameters, w_1, r_1, k_1, and r_2, we changed each of these parameters separately by a multiplicative factor of either 0.5 or 2; in neither case did M-to-1 accuracy drop by more than 0.014.

This performance was achieved despite the fact that the SVD2 tagger is mathematically much simpler than the other models. Our MATLAB implementation of SVD2 takes only a few minutes to run on a desktop computer, in contrast to HMM training times of several hours or days (Gao and Johnson 2008; Johnson 2007).

High-*k* performance. Not suffering from the same computational limitations as other models, SVD2 can easily accommodate high numbers of induced tags, resulting in fine-grained labelings. The value of this flexibility is discussed in the next section. Figure 1 shows, as a function of k_2, the tagging accuracy of SVD2 under both the best and the prototype-based M-to-1 maps (see Section 3), for both the PTB45 and the PTB17 tagsets. The horizontal one-tag-per-word-type line in each panel is the theoretical upper limit for tagging accuracy in non-disambiguating models (such as SVD2). This limit is the fraction of all tokens in the corpus whose gold tag is the most frequent for their type.

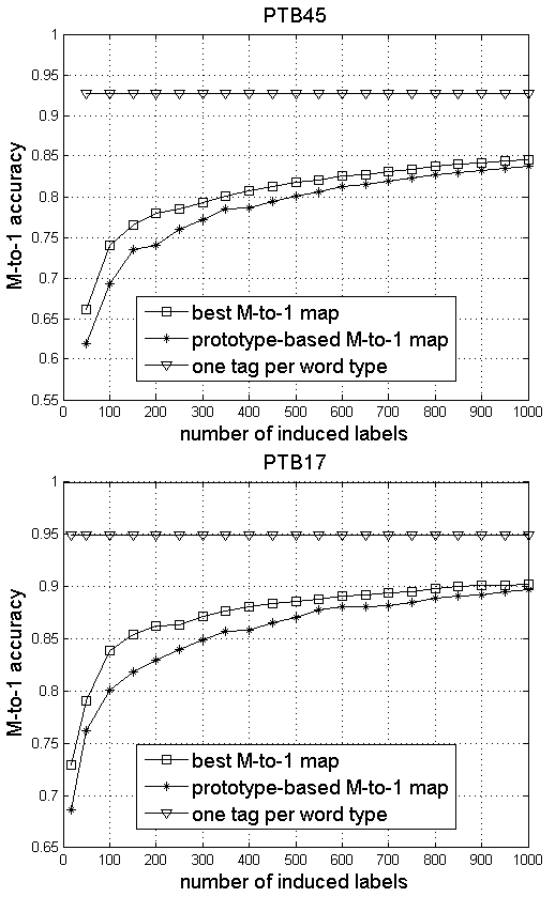

Figure 1. Performance of the SVD2 algorithm as a function of the number of induced tags. Top: PTB45; bottom: PTB17. Each plot shows the tagging accuracy under the best and the prototype-based M-to-1 maps, as well as the upper limit for non-disambiguating taggers.

5 Discussion

At the heart of the algorithm presented here is the reduced-rank SVD method of Schütze (1995), which transforms bigram counts into latent descriptors. In view of the present work, which achieves state-of-the-art performance when evaluation is done with the criteria now in common use, Schütze's original work should rightly be praised as ahead of its time. The SVD2 model presented here differs from Schütze's work in many details of implementation—not all of which are explicitly specified in Schütze (1995). In what follows, we discuss the features of SVD2 that are most critical to its performance. Failure to incorporate any one of them signifi-

cantly reduces the performance of the algorithm (M-to-1 reduced by 0.04 to 0.08).

First, the reduced-rank left-singular vectors (for the right and left context matrices) are scaled, *i.e.*, multiplied, by the singular values. While the resulting descriptors, the rows of L^* and R^*, live in a much lower-dimensional space than the original context vectors, they are mapped by an angle-preserving map (defined by the matrices of right-singular vectors V_L and V_R) into vectors in the original space. These mapped vectors best approximate (in the least-squares sense) the original context vectors; they have the same geometric relationships as their equivalent high-dimensional images, making them good candidates for the role of word-type descriptors.

A second important feature of the SVD2 algorithm is the unit-length normalization of the latent descriptors, along with the computation of cluster centroids as the weighted averages of their constituent vectors. Thanks to this combined device, rare words are treated equally to frequent words regarding the length of their descriptor vectors, yet contribute less to the placement of centroids.

Finally, while the usual drawback of *k*-means-clustering algorithms is the dependency of the outcome on the initial—usually random—placement of centroids, our initialization of the k centroids as the descriptors of the k most frequent word types in the corpus makes the algorithm fully deterministic, and improves its performance substantially: M-to-1 PTB45 by 0.043, M-to-1 PTB17 by 0.063.

As noted in the Results section, SVD2 is fairly robust to changes in all four parameters w_1, r_1, k_1, and r_2. The values used here were obtained by a coarse, greedy strategy, where each parameter was optimized independently. It is worth noting that dispensing with the second pass altogether, *i.e.*, clustering directly the latent descriptor vectors obtained in the first pass into the desired number of induced tags, results in a drop of Many-to-1 score of only 0.021 for the PTB45 tagset and 0.009 for the PTB17 tagset.

Disambiguation. An obvious limitation of SVD2 is that it is a non-disambiguating tagger, assigning the same label to all tokens of a type. However, this limitation *per se* is unlikely to be the main obstacle to the improvement of low-k performance, since, as is well known, the theoretical upper limit for the tagging accuracy of non-disambiguating models (shown in Fig. 1) is much higher than the current state-of-the-art for unsupervised taggers, whether disambiguating or not.

To further gain insight into how successful current models are at disambiguating when they have the power to do so, we examined a collection of HMM-VB runs (Gao and Johnson 2008) and asked how the accuracy scores would change if, after training was completed, the model were forced to assign the same label to all tokens of the same type. To answer this question, we determined, for each word type, the *modal* HMM state, *i.e.*, the state most frequently assigned by the HMM to tokens of that type. We then relabeled all words with their modal label. The effect of thus eliminating the disambiguation capacity of the model was to slightly *increase* the tagging accuracy under the best M-to-1 map for every HMM-VB run (the average increase was 0.026 for PTB17, and 0.015 for PTB45). We view this as a further indication that, in the current state of the art and with regards to tagging accuracy, limiting oneself to non-disambiguating models may not adversely affect performance.

To the contrary, this limitation may actually benefit an approach such as SVD2. Indeed, on difficult learning tasks, simpler models often behave better than more powerful ones (Geman et al. 1992). HMMs are powerful since they can, in theory, induce both a system of tags and a system of contextual patterns that allow them to disambiguate word types in terms of these tags. However, carrying out both of these unsupervised learning tasks *at once* is problematic in view of the very large number of parameters to be estimated compared to the size of the training data set.

The POS-tagging subtask of disambiguation may then be construed as a challenge in its own right: demonstrate *effective* disambiguation in an unsupervised model. Specifically, show that tagging accuracy *decreases* when the model's disambiguation capacity is removed, by re-labeling all tokens with their modal label, defined above.

We believe that the SVD2 algorithm presented here could provide a launching pad for an approach that would successfully address the disambiguation challenge. It would do so by allowing a gradual and carefully controlled amount of ambiguity into an initially non-disambiguating model. This is left for future work.

Fine-grained labeling. An important feature of the SVD2 algorithm is its ability to produce a fine-grained labeling of the data, using a number of clusters much larger than the number of tags

in a syntax-motivated POS-tag system. Such fine-grained labelings can capture additional linguistic features. To achieve a fine-grained labeling, only the final clustering step in the SVD2 algorithm needs to be changed; the computational cost this entails is negligible. A high-quality fine-grained labeling, such as achieved by the SVD2 approach, may be of practical interest as an input to various types of unsupervised grammar-induction algorithms (Headden et al. 2008). This application is left for future work.

Prototype-based tagging. One potentially important practical application of a high-quality fine-grained labeling is its use for languages which lack any kind of annotated data. By first applying the SVD2 algorithm, word types are grouped together into a few hundred clusters. Then, a prototype word is automatically extracted from each cluster. This produces, in a completely unsupervised way, a list of only a few hundred words that need to be hand-tagged by an expert. The results shown in Fig. 1 indicate that these prototype tags can then be used to tag the entire corpus with only a minor decrease in accuracy compared to the best M-to-1 map—the construction of which requires a fully annotated corpus. Fig. 1 also indicates that, with only a few hundred prototypes, the gap left between the accuracy thus achieved and the upper bound for non-disambiguating models is fairly small.

References

Alexander Clark. 2000. Inducing syntactic categories by context distribution clustering. In *The Fourth Conference on Natural Language Learning*.

Alexander Clark. 2003. Combining distributional and morphological information for part of speech induction. In *10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 59–66.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing – Volume 10*.

Jianfeng Gao and Mark Johnson. 2008. A comparison of bayesian estimators for unsupervised Hidden Markov Model POS taggers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 344–352.

Stuart Geman, Elie Bienenstock and René Doursat. 1992. Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, 4 (1), pages 1–58.

Sharon Goldwater and Tom Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 744–751.

João V. Graça, Kuzman Ganchev, Ben Taskar, and Fernando Pereira. 2009. Posterior vs. Parameter Sparsity in Latent Variable Models. In Neural Information Processing Systems Conference (NIPS).

Aria Haghighi and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 320–327, New York City, USA, June. Association for Computational Linguistics.

William P. Headden, David McClosky, and Eugene Charniak. 2008. Evaluating unsupervised part-of-speech tagging for grammar induction. In *Proceedings of the International Conference on Computational Linguistics* (*COLING '08)*.

Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers? In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pages 296–305.

Marina Meilă. 2003. Comparing clusterings by the variation of information. In Bernhard Schölkopf and Manfred K. Warmuth, editors, *COLT 2003: The Sixteenth Annual Conference on Learning Theory*, volume 2777 of *Lecture Notes in Computer Science*, pages 173–187. Springer.

Roi Reichart and Ari Rappoport. 2009. The NVI Clustering Evaluation Measure. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL)*, pages 165–173.

Hinrich Schütze. 1995. Distributional part-of-speech tagging. In Proceedings of the seventh conference on European chapter of the Association for Computational Linguistics, pages 141–148.

Noah A. Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 354–362.

Kristina Toutanova, Dan Klein, Christopher D. Manning and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL* 2003, pages 252-259.