FLORS: Fast and Simple Domain Adaptation for Part-of-Speech Tagging

Tobias Schnabel
Department of Computer Science
Cornell University
tbs49@cornell.edu

Hinrich Schütze
Center for Information & Language Processing
University of Munich
inquiries@cislmu.org

Abstract

We present FLORS, a new part-of-speech tagger for domain adaptation. FLORS uses robust representations that work especially well for unknown words and for known words with unseen tags. FLORS is simpler and faster than previous domain adaptation methods, yet it has significantly better accuracy than several baselines.

1 Introduction

In this paper we describe FLORS, a part-of-speech (POS) tagger that is Fast in training and tagging, uses LOcal context only (as opposed to finding the optimal tag sequence for the entire sentence), performs Robustly on target domains (TDs) in unsupervised domain adaptation (DA) and is Simple in architecture and feature representation.

FLORS constructs a robust representation of the local context of the word v that is to be tagged. This representation consists of distributional features, suffixes and word shapes of v and its local neighbors. We show that it has two advantages.

First, since the main predictors used by FLORS are distributional features (not the word's identity), FLORS *predicts unseen tags of known words better* than prior work on DA for POS. Second, since FLORS uses representations computed from unlabeled text, representations of unknown words are in principle of the same type as representations of known words; this property of FLORS results in *better performance on unknown words* compared to prior work. These two advantages are especially beneficial for TDs that contain high rates of unseen tags of known words and high rates of unknown words. We show that FLORS achieves excellent DA tagging results on the five domains of the SANCL 2012 shared task (Petrov and McDonald, 2012) and outperforms three state-of-the-art taggers on Blitzer et al.'s (2006) biomedical data.

FLORS is also simpler and faster than other POS DA methods. It is simple in that the input representation consists of three simple types of features: distributional count features and two types of binary features, suffix and shape features. Many other word representations that are used for improving generalization (e.g., (Brown et al., 1992; Collobert et al., 2011)) are costly to train or have difficulty handling unknown words. Our representations are fast to build and can be created on-the-fly for unknown words that occur during testing.

The learning architecture is simple and fast as well. We train k binary one-vs-all classifiers that use local context only and no sequence information (where k is the number of tags). Thus, tagging complexity is $O(k)$. Many other learning setups for DA are more complex; e.g., they *learn* representations (as opposed to just counting), they learn *several classifiers* for different subclasses of words (e.g., known vs. unknown) or they combine *left-to-right and right-to-left* taggings.

The next two sections describe experimental data, setup and results. Results are discussed in Section 4. We compare FLORS to alternative word representations in Section 5 and to related work in Section 6. Section 7 presents our conclusions.

2 Experimental data and setup

Data. Our source domain is the Penn Treebank (Marcus et al., 1993) of Wall Street Journal (WSJ)

Transactions of the Association for Computational Linguistics, 2 (2014) 15–26. Action Editor: Sharon Goldwater.
Submitted 9/2013; Revised 11/2013; Published 2/2014.

text. Following Blitzer et al. (2006), we use sections 2-21 for training and 100,000 WSJ sentences from 1988 as unlabeled data in training.

We evaluate on six different TDs. The first five TDs (newsgroups, weblogs, reviews, answers, emails) are from the SANCL shared task (Petrov and McDonald, 2012). Additionally, the SANCL dataset contains sections 22 and 23 of the WSJ for in-domain development and testing, respectively. Each SANCL TD has an unlabeled training set of 100,000 sentences and development and test sets of about 1000 labeled sentences each. The sixth TD is BIO, the Penn BioTreebank data set distributed by Blitzer. It consists of dev and test sets of 500 sentences each and 100,000 unlabeled sentences.

Classification setup. Similar to SVMTool (Giménez and Màrquez, 2004) and Choi and Palmer (2012) (henceforth: C&P), we use *local context only* for tagging instead of performing sequence classification. For a word w occurring as token v_i in a sentence, we build a feature vector for a local window of size $2l + 1$ around v_i. The representation of the object to be classified is this feature vector and the target class is the POS tag of v_i.

We use the linear L2-regularized L2-loss SVM implementation provided by LIBLINEAR (Fan et al., 2008) to train k one-vs-all classifiers on the training set where k is the number of POS tags in the training set (in our case $k = 45$). We train with untuned default parameters; in particular, $C = 1$. In the special case of linear SVMs, the value of C does not need to be tuned exhaustively as the solution remains constant after C has reached a certain threshold value C^* (Keerthi and Lin, 2003). Training can easily be parallelized by giving each binary SVM its own thread.

Windows. The local context for tagging token v_i is a window of size $2l + 1$ centered around v_i: $(v_{i-l}, \ldots, v_i, \ldots, v_{i+l})$. We pad sentences on either side with $\langle$BOUNDARY$\rangle$ to ensure sufficient context for all words. Given a mapping f from words to feature vectors (see below), the representation F of a token v_i is the concatenation of the $2l + 1$ word vectors in its window

$$F(v_i) = f(v_{i-l}) \oplus \ldots \oplus f(v_{i+l})$$

where $\oplus$ is vector concatenation.

Word features. We represent each word w by four components: (i) counts of left neighbors, (ii) counts of right neighbors, (iii) binary suffix features and (iv) binary shape features. These four components are concatenated:

$$f(w) = f_{\text{left}}(w) \oplus f_{\text{right}}(w) \oplus f_{\text{suffix}}(w) \oplus f_{\text{shape}}(w)$$

We consider these sources of information equally important and normalize each of the four component vectors to unit length. Normalization also has a beneficial effect on SVM training time because it alleviates numerical problems (Fan et al., 2008).

Distributional features. We follow a long tradition of older (Finch and Chater, 1992; Schütze, 1993; Schütze, 1995) and newer (Huang and Yates, 2009) work on creating distributional features for POS tagging based on local left and right neighbors.

Specifically, the i^{th} entry x_i of $f_{\text{left}}(w)$ is the weighted number of times that the *indicator word* c_i occurs immediately to the left of w:

$$x_i = \text{tf}\left(\text{freq}\left(\text{bigram}(c_i, w)\right)\right)$$

where c_i is the word with frequency rank i in the corpus, $\text{freq}\left(\text{bigram}(c_i, w)\right)$ is the number of times the bigram "c_i w" occurs in the corpus and we weight the non-zero frequencies logarithmically: $\text{tf}(x) = 1 + \log(x)$. tf-weighting has been used by other researchers (Huang and Yates, 2009) and showed good performance in our own previous work.

$f_{\text{right}}(w)$ is defined analogously. We restrict the set of indicator words to the $n = 500$ most frequent words in the corpus. To avoid zero vectors, we add an entry x_{n+1} to each vector that counts omitted contexts:

$$x_{n+1} = \text{tf}\left(\sum_{j:j>n} \text{freq}\left(\text{bigram}(c_j, w)\right)\right)$$

We compute distributional vectors on the joint corpus $\mathcal{D}_{\text{ALL}}$ of all labeled and unlabeled text of source domain and TD. The text is preprocessed by lowercasing everything – which is often done when computing word representations, e.g., by Turian et al. (2010) – and by padding sentences with $\langle$BOUNDARY$\rangle$ tokens.

Suffix features. Suffixes are promising for DA because basic morphology rules are the same in different domains. In contrast to other work on tagging

	model	classifier	features
1	TnT	HMM	$p_{-\{0,1,2\}}$, v_0, suffixes (for OOVs)
2	Stanford	bidir. MEMM	$p_{\pm\{0,1,2\}}$, $v_{\pm\{0,1\}}$, affixes, orthography
3	SVMTool	SVM	$p_{\pm\{0,1,2,3\}}$, $v_{\pm\{0,1,2,3\}}$, affixes, orthography, word length
4	C&P	SVM	$p_{\pm\{0,1,2,3\}}$, $v_{\pm\{0,1,2,3\}}$, affixes, orthography
5	FLORS	SVM	distributions of $v_{\pm\{0,1,2\}}$, suffixes, orthography

Table 1: Overview of baseline taggers and FLORS. v_i: token, p_i: POS tag. Positions included in the sets of token indices are relative to the position i of the word v_0 to be tagged; e.g., $p_{\pm\{0,1,2\}}$ is short for $\{p_{-0}, p_{-1}, p_{-2}, p_0, p_1, p_2\}$. To represent tokens v_i, models 1–4 use vocabulary indices and FLORS uses distributional representations. Models 2–4 use combinations of features (e.g., tag-word) as well.

(e.g., Ratnaparkhi (1996), Toutanova et al. (2003), Miller et al. (2007)) we simply use *all (lowercase) suffixes* to avoid the need for selecting a subset of suffixes; and we treat *all words* equally as opposed to using suffix features for only a subset of words. For suffix s, we set the dimension corresponding to s in $f_{\text{suffix}}(w)$ to 1 if lowercased w ends in s and to 0 otherwise. Note that w is a suffix of itself.[1]

Shape features. We use the Berkeley parser word signatures (Petrov and Klein, 2007). Each word is mapped to a bit string encompassing 16 binary indicators that correspond to different orthographic (e.g., does the word contain a digit, hyphen, uppercase character) and morphological (e.g., does the word end in -ed or -ing) features. There are 50 unique signatures in WSJ. We set the dimension of $f_{\text{shape}}(w)$ that corresponds to the signature of w to 1 and all other dimensions to 0. We note that the shape features we use were designed for English and probably would have to be adjusted for other languages.

Baselines. We address the problem of unsupervised domain adaptation for POS tagging. For this problem, we consider three types of baselines: (i) high-performing publicly available systems, (ii) the taggers used at SANCL and (iii) POS DA results published for BIO.

Most of our experiments use taggers from category (i) because we can ensure that experimental conditions are directly comparable. The four baselines in category (i) are shown in Table 1. Three have near state-of-the-art performance on WSJ: SVMTool (Giménez and Màrquez, 2004), Stanford (Toutanova et al., 2003) (a birectional MEMM) and C&P. TnT (Brants, 2000) is included as a representative of fast and simple HMM taggers. In addition, C&P is a tagger that has been extensively tested in DA scenarios with excellent results. Unless otherwise stated, we train all models using their default configuration files. We use the optimized parameter configuration published by C&P for the C&P model.

Test set results will be compared with the SANCL taggers (category (ii)) at the end of Section 3.

As far as category (iii) is concerned, most work on POS DA has been evaluated on BIO. We discuss our concerns about the BIO evaluation sets in Section 4, but also show that FLORS beats previously published results on BIO as well (see Table 6).

3 Experimental results

We train k binary SVM classifiers on the training set. A token in the test set is classified by building its feature vector, running the classifiers on it and then assigning it to the POS class whose one-vs-all LIBLINEAR classifier returns the largest score.

Results for ALL accuracy (accuracy for all tokens) and OOV accuracy (accuracy for tokens not occurring in the labeled WSJ data) are reported in Table 2. Results with an asterisk are significantly worse than a column's best result using McNemar's test ($p < .001$). We use the same test and p-value throughout this paper.

The basic FLORS model (Table 2, line 5) uses window size 5 ($l = 2$). Each word in the window has 1002 distributional features (501 left and right), 91,161 suffix features and 50 shape features. The final feature vector for a token has a dimensionality of about 500,000, but is very sparse.

FLORS outperforms all baselines on the five TDs

[1] One could also compute these suffixes for _w (w prefixed by underscore) instead of for w to include words as distinguishable special suffixes. We test this alternative in Table 2, line 15.

				newsgroups		reviews		weblogs		answers		emails		wsj	
				ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV
1	TnT			88.66*	54.73*	90.40*	56.75*	93.33*	74.17*	88.55*	48.32*	88.14*	58.09*	95.75*	88.30
2	Stanford			89.11*	56.02*	91.43*	58.66*	94.15*	77.13*	88.92*	49.30*	88.68*	58.42*	**96.83**	90.25
3	SVMTool			89.14*	53.82*	91.30*	54.20*	94.21*	76.44*	88.96*	47.25*	88.64*	56.37*	96.63	87.96
4	C&P			89.51*	57.23*	91.58*	59.67*	94.41*	78.46*	89.08*	48.46*	88.74*	58.62*	96.78	88.65
5	FLORS		basic	90.86	66.42	92.95	75.29	94.71	83.64	90.30	62.15	89.44*	62.61	96.59	90.37
6			$n = 250$	**90.93**	**67.03**	92.93	75.45	94.69	83.69	90.29	62.20	**89.63**	63.43	96.56*	89.45
7		$v_{\{0\}}$	$n = 0$	89.14*	55.59*	91.80*	66.31*	93.40*	72.55*	89.47*	55.82*	88.21*	57.83*	96.29*	85.55*
8		$v_{\{0\}}$	no suffixes	90.60	65.17	92.74	71.94*	94.77	**84.92**	89.77*	58.71*	89.30*	62.09	96.28*	88.88
9		$v_{\{0\}}$	no shapes	89.70*	63.10*	92.24*	68.70*	92.60*	74.72*	89.55*	59.08*	**89.63**	**64.17**	95.52*	83.94*
10		$v_{\{0,1\}}$	$n = 0$	90.61*	65.95	92.76*	75.56	94.62	84.62	90.23	61.87	89.40*	63.82	96.51*	90.02
11		$v_{\{0,1\}}$	no suffixes	90.66	64.78*	92.88	75.08	**94.83**	84.52	90.36	61.92	89.42	62.74	96.64	89.45
12		$v_{\{0,1\}}$	no shapes	90.74	**67.03**	**93.02**	**75.88**	94.57	83.83	90.23	61.73	89.41*	63.49	96.57	90.25
13			$l = 1$	90.44*	63.62*	92.69*	75.72	94.48*	84.03	90.02*	62.66	89.17*	62.71	96.44*	88.65
14			L-to-R	90.56*	66.08	92.97	75.40	94.57	83.79	**90.43**	**62.80**	89.43	63.13	96.53*	**90.94**
15			voc. indices	**90.93**	66.64	92.91	75.03	94.71	84.08	90.27	61.92	89.37*	62.26	96.63	90.60

Table 2: Tagging accuracy of four baselines and FLORS on the dev sets. The table is structured as follows: baselines (lines 1–4), basic FLORS setup (lines 5–6), effect of omitting one of the three feature types if the word to be tagged is changed compared to the basic FLORS setup (lines 7–9) and if the word to be tagged is not changed compared to basic FLORS (lines 10–12), effect of three important configuration choices on tagging accuracy: window size (line 13), inclusion of prior tagging decision (line 14) and vocabulary index (line 15). n: number of indicator words. $2l+1$: size of the local context window. Lines 10–12: Only the neighbors of v_0 are modified compared to basic (line 5). Lines 7–9: All five token representations (including v_0) are modified. A column's best result is bold.

(line 5 vs. lines 1–4). Only in-domain on WSJ, three baselines are slightly superior. The baselines are slightly better on ALL accuracy because they were designed for tagging in-domain data and use feature sets that have been found to work well on the source domain. Generally, C&P performs best for DA among the baselines. On answers and WSJ, however, Stanford has better overall accuracies. These results are in line with C&P.

On lines 6–15, we investigate how different modifications of the basic FLORS model affect performance. First, we examine the effect of leaving out components of the representation: distributional features ($f_{\text{left}}(w)$, $f_{\text{right}}(w)$), suffixes ($f_{\text{suffix}}(w)$) and shape features ($f_{\text{shape}}(w)$).

Distributional features boost performance in all domains: ALL and OOV accuracies are consistently worse for $n = 0$ (line 7) than for $n \in \{250, 500\}$ (lines 6&5). FLORS with $n = 250$ has better OOV accuracies in 5 of 6 domains. However, ALL accuracy for FLORS with $n = 500$ is better in the majority of domains. The main result of this comparison is that FLORS does not seem to be very sensitive to the value of n if n is large enough.

Shape features also improve results in all domains, with one exception: emails (lines 9 vs 5). For emails, shape features decrease ALL accuracy by .19 and OOV accuracy by 1.56. This may be due to the fact that many OOVs are NNP/NN and that tagging conventions for NNP/NN vary between domains. See Section 4 for discussion.

Performance benefits from suffixes in all domains but weblogs (lines 8 vs 5). Weblogs contain many foreign names such as *Abdul* and *Yasim*. For these words, shapes apparently provide better information for classification than suffixes. ALL accuracies suffer little when leaving out suffixes, but the feature space is much smaller: about 3000 dimensions. Thus, for domains where we expect few OOVs, omitting suffix features could be considered.

Lines 7–9 omit one of the components of $f(v_i)$ for all five words in the local context: $i \in \{-2, -1, 0, 1, 2\}$. Lines 10–12 omit the same components for the neighbor words only – i.e., $i \in \{-2, -1, 1, 2\}$ – and leave $f(v_0)$ unchanged. 14 of the 6×3 ALL accuracies on lines 10–12 are worse than FLORS basic, 4 are better. The largest differences are .25 for newsgroups and .19 for reviews (lines 5 vs 10), but differences for the other domains are negligible. This shows that the most important

feature representation is that of v_0 (not surprisingly) and that the distributional features of the other words can be omitted at the cost of some loss in accuracy if a small average number of active features is desired.

Another FLORS parameter is the size of the local context. Surprisingly, OOV accuracies benefit a bit in four domains if we reduce l from 2 to 1 (lines 13 vs 5). However, ALL accuracy consistently drops in all six domains. This argues for using $l = 2$, i.e., a window size of 5.

Results for left-to-right (L-to-R) tagging are given on line 14. Similar to SVMTool and C&P, each sentence is tagged from left to right and previous tagging decisions are used for the current classification. In this setting, we use the previous tag p_{i-1} as one additional feature in the feature vector of v_i.

The effect of left-to-right is similar to the effect of omitting suffixes: OOV accuracies go up in some domains, but ALL accuracies decrease (except for an increase of .02 for reviews). This is in line with the experiments in (Schnabel and Schütze, 2013) where sequential information in a CRF was not robust across domains. OOV tagging may benefit from correct previous tags because the larger left context that is indirectly made available by left-to-right tagging compensates partially for the lack of information about the OOV word.

In contrast to standard approaches to POS tagging, the FLORS basic representation does not contain vocabulary indices. Line 15 shows what happens if we add them; the dimensionality of the feature vector is increased by $5|V|$ – where V is the training set vocabulary – and in training one binary feature is set to one for each of the five local context words. Performance is almost indistinguishable from FLORS basic, suggesting that only using suffixes – which can be viewed as "ambiguous" vocabulary indices, e.g., "at" is on for "at", "mat", "hat", "laundromat" etc – is sufficient.

In summary, we find that distributional features, word signatures and suffixes all contribute to successful POS DA. Factors with only minor impact on performance are the number of indicator words used for the distributional representations, the window size l and the tagging scheme (L-to-R vs. non-L-to-R). Unknown words and known words behave differently with respect to certain feature choices.

The different behavior of unknown and known words suggests that training and optimizing two separate models – an approach used by SVMTool – would further increase tagging accuracy. Note that there has been at least one publication (Schnabel and Schütze, 2013) on optimizing a separate model for unknown words that has in some cases better performance on OOV accuracy than what we publish here.[2] However, this would complicate the architecture of FLORS. We opted for a maximally simple model in this paper, potentially at the cost of some performance.

Test set results. Table 3 reports results on the test sets. FLORS again performs significantly better on all five TDs, both on ALL and OOV. Only in-domain on WSJ, ALL performance is worse.

Finally, we compare our results to the POS taggers for which performance was reported at SANCL 2012 (Petrov and McDonald, 2012, Table 4). Constituency-based parsers – which also tag words as a by-product of deriving complete parse trees – are excluded from the comparison because they are trained on a richer representation, the syntactic structure of sentences.[3] FLORS' results are better than the best non-parsing-based results at SANCL 2012, which were accuracies of 92.32 on newsgroups (HIT), 90.65 on reviews (HIT) and 91.07 on answers (IMS-1).

4 Discussion

Advantages of FLORS representation. As we can see in Table 1, the main representational difference between FLORS and the other taggers is that the FLORS representation does not include vocabulary indices of the word to be tagged or its neighbors – the FLORS vector only consists of distributional, suffix and shape features.

This is an obvious advantage for OOVs. In other representational schemes, OOVs have representations that are fundamentally different from known

[2] Schnabel and Schütze (2013) report OOV accuracies of 56.62 (newsgroups), 64.61 (reviews), 71.86 (weblogs), 54.28 (answers), 61.05 (emails) and 64.64 (BIO) for their basic model and even higher OOV accuracies if parameters are optimized on a per-domain basis.

[3] DCU-Paris13 is listed in the dependency parser tables, but DCU-Paris13 results are derived from a constituency parser. DCU also developed sophisticated preprocessing rules for the different domains, which can be viewed as a kind of manual domain adaptation.

		newsgroups		reviews		weblogs		answers		emails		wsj	
		ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV
1	TnT	90.85*	56.60*	89.67*	50.98*	91.37*	62.65*	89.36*	51.82*	87.38*	55.12*	96.57*	86.27
2	Stanford	91.25*	57.96*	90.30*	51.87*	92.32*	67.85*	89.74*	53.41*	87.77*	57.10*	97.43	**88.71**
3	SVMTool	91.21*	54.40*	90.01*	45.05*	92.05*	63.59*	89.90*	51.07*	87.74*	53.23*	97.26	86.47
4	C&P	91.68*	60.58*	90.42*	51.12*	92.22*	66.91*	89.90*	53.31*	87.91*	54.47*	**97.44**	88.20
5	FLORS basic	**92.41**	**66.91**	**92.25**	**70.87**	**93.14**	**75.32**	**91.17**	**67.93**	**88.67**	**61.09**	97.11*	87.79

Table 3: Tagging accuracy of four baselines and FLORS on the test sets.

		newsgroups	reviews	weblogs	answers	emails	wsj	bio
pct tokens	unknown tag	0.31	0.06	0.00	0.25	0.80	0.00	0.98
	OOV	10.34	6.84	8.45	8.53	10.56	2.72	19.86
	unseen word+tag	2.44	2.22	1.46	2.91	3.47	0.61	2.50
accuracy on unseen word+tag	TnT	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Stanford	3.66	5.74	9.40	5.46	2.77	15.23	4.64
	SVMTool	0.00	0.16	0.00	0.00	0.10	0.00	0.00
	C&P	14.47	14.75	20.51	13.37	10.29	38.07	8.98
	FLORS basic	**21.06**	**21.97**	**21.65**	**17.19**	**15.13**	**41.12**	**12.69**

Table 4: Top: Percentage of unknown tags, OOVs and unseen word+tag combinations (i.e., known words tagged with unseen tags) in the dev sets. Bottom: Tagging accuracy on unseen word+tag.

words – since their vocabulary index does not occur in the training set and cannot be used for prediction. In contrast, given enough unlabeled TD data, FLORS represents known and unknown words in essentially the same way and prediction of the correct tag is easier. This explanation is supported by the experiments in Table 2: FLORS beats all other systems on OOVs – even in-domain on WSJ.

In our analysis we found that apart from better handling of OOVs there is a second beneficial effect of distributional representations: they facilitate the correct tagging of *known words occurring with tags unseen in the training set*, which we call *unseen word+tags*. Table 4 gives statistics on this case and shows that unseen word+tags occur at least two times as often out-of-domain (e.g., 1.46% for weblogs) than in-domain (.61% for WSJ). The bottom part of the table shows performance of the five taggers on unseen word+tags. FLORS is the top performer on all seven domains, with large differences of more than 5% in some domains.

The explanation is similar to the OOV case: FLORS does not restrict the set of possible POS's of a word. The other taggers in Table 2 use the vocabulary index of the word to be tagged and will therefore give a strong preference to seen tags. Since FLORS uses distributional features, it can more easily assign an unseen tag as long as it is compatible with the overall pattern of distribution, suffixes and shapes typical of the tag. C&P also perform relatively well on unseen word+tag due to the ambiguity classes in their model, but FLORS representations are better for every domain. We take these results to mean that constraints on a word's possible POS tags may well be helpful for in-domain data, but for out-of-domain data an overly strong bias for a word's observed tags is harmful.

It is important to stress that representations similar to FLORS representations have been used for a long time; we would expect many of them to have similar advantages for unseen word+tags. E.g., Brown clusters (Brown et al., 1992) and word embeddings (Collobert et al., 2011) are similar to FLORS in this respect. However, FLORS representations are extracted by simple counting whereas the computation of Brown clusters or word embeddings is much more expensive. The speed with which FLORS representations can be computed is particularly beneficial when taggers need to be adapted to new domains. FLORS can easily adapt its representations on the fly – as each new occurrence of a word is encountered, the counts that are the basis

for the x_i can simply be incremented. We present a direct comparison of FLORS representations with other representations in Section 5.

"Local context" vs. sequence classification. The most common approach to POS tagging is to tag a sentence with its most likely sequence; in contrast, independent tagging of local context is not guaranteed to find the best sequence. Recent work on English suggests that window-based tagging can perform as well as sequence-based methods (Liang et al., 2008; Collobert et al., 2011). Toutanova et al. (2003) report similar results. In our experiments, we also did not find consistent improvements when we incorporated sequence constraints (Table 2, line 14). However, there may be languages and applications involving long-distance relationships where local-context classification is suboptimal.

Local-context classification has two advantages compared to sequence classification. (i) It simplifies the classification and tagging setup: we can use any existing statistical classifier. Sequence classification limits the range of methods that can be applied; e.g., it is difficult to find a good CRF implementation that can handle real-valued features – which are of critical importance for our representation.

(ii) The time complexity of FLORS in tagging is $O(skf)$ where s is the length of the sentence, k is the number of tags and f is the number of non-zero features per local-context representation. In contrast, sequence decoding complexity is $O(sk^2f)$. This difference is not of practical importance for standard English POS sets, but it could be an argument against sequence classification for tagging problems with much larger tag sets.

In summary, replacing sequence classification with local-context classification is attractive for large-scale, practical tagging.

What DA can and cannot do. Despite the superior DA tagging results we report for FLORS in this paper, there is still a gap of 2%–7% (depending on the domain) between in-domain WSJ accuracy and DA accuracy on SANCL. In our analysis of this gap, we found some evidence that DA performance can be further improved – especially as more unlabeled TD data becomes available. But we also found two reasons for low performance that unsupervised DA cannot do anything about: differences in tag sets – or unknown tags – and differences in annotation guidelines.

Table 4 shows that *unknown tags* occur in five of the seven TDs at rates between 0% (weblogs) and 1% (BIO). Each token that is tagged with an unknown tag is necessarily an error in unsupervised DA. Furthermore, the unknown tag can also impact tagging accuracy in the local context[4] – so the unknown tag rates in Table 4 are probably lower bounds for the error that is due to unknown tags. Based on these considerations, it is not surprising that tagging accuracy (e.g., of FLORS basic) and unknown tag rate are correlated as we can see in Tables 2, 4 and 6; e.g., we get the highest accuracies in the two domains that do not have unknown tags (weblogs and WSJ) and the lowest accuracy in the domain with the highest rate (BIO).

Since unknown tags cannot be predicted correctly, one could simply report accuracy on known tags. However, given the negative effect of unknown tags on tagging accuracy of the local context in which they occur, excluding unknown tags does not fully address the problem. For this reason, it is probably best to keep the common practice of simply reporting accuracy on all tokens, including unknown tags. But the percentages of unknown tags should also be reported for each dataset as a basis for a more accurate interpretation of results.

Another type of error that cannot be avoided in unsupervised DA is due to differences in *annotation guidelines*. There are a few such problems in SANCL; e.g., file names like "Services.doc" are annotated as NN in the email domain. But their distributional and grammatical behavior is more similar to NNPs; as a consequence, most file names are incorrectly tagged. In general, it is difficult to discriminate NNs from NNPs. The Penn Treebank annotation guidelines (Santorini, 1990) are compatible with either tag in many cases and it may simply be impossible to write annotation guidelines that avoid these problems (cf. Manning (2011)). NN-NNP inconsistencies are especially problematic for OOV tagging since most OOVs are NNs or NNPs.

[4]For example, there is a special tag ADD in the web domain for web addresses. The last two words of the sentence "I would like to host my upcoming website to/IN Liquidweb.com/ADD" are mistagged by Stanford tagger as "... to/TO Liquidweb.com/VB". So the missing tag in this case also affects the tagging of surrounding words.

	bio dev		wsj train
	OOV	ALL	ALL
NN	62.4	25.4	14.4
JJ	15.9	8.9	6.2
NNS	10.2	7.5	6.3
NNP	0.5	0.2	9.5
NNPS	0.0	0.0	0.3

Table 5: Frequency of some tags (percent of tokens) for bio dev and wsj train.

While the amount of inconsistent annotation is limited for SANCL, it is a serious problem for BIO. Table 5 shows that the proportion of NNPs in BIO is less than a tenth of that in WSJ (.2 in BIO vs. 9.5 in WSJ). This is due to the fact that many bio-specific names, in particular genes, are annotated as NN. In contrast, the distributionally and orthographically most similar names in WSJ are tagged as NNP. For example, we find "One cell was teased out, and its DNA/NNP extracted" in WSJ vs. "DNA/NN was isolated" in BIO.

		standard setup		NNP→NN	
		ALL	OOV	ALL	OOV
	TnT	87.49*	59.08*	91.75*	78.33*
	Stanford	88.46*	62.55*	92.36*	79.19*
	SVMTool	88.33*	61.30*	92.47	79.46*
	C&P	87.82*	60.60*	92.06*	79.30*
FLORS	basic	88.90	64.74	92.91	82.58
FLORS	$n = 250$	88.90	64.51	92.93	82.47
FLORS	$n = 0$	87.27*	57.75*	90.91*	73.57*
FLORS	no suffixes	88.09*	62.20*	91.98*	79.27*
FLORS	no shapes	87.78*	59.82*	91.81*	77.31*
FLORS	$l = 1$	**89.12**	**65.52**	**92.99**	**82.90**

Table 6: Tagging accuracy on bio dev. NNP→NN results were obtained by replacing NNPs with NNs.

Given this large discrepancy in the frequency of the tag NNP – which arguably is due to different annotation guidelines, not due to underlying differences between the two genres – BIO should probably not be used for evaluating DA. This is why we did not include it in our comparison in Table 2.

For sake of completeness, we provide tagging accuracies for BIO in Table 6, "standard setup". The results are in line with SANCL results: FLORS beats the baselines on ALL and OOV accuracies. However, if we build the NN bias into our model by simply replacing all NNP tags with NN tags, then accuracy goes up by 4% on ALL and by almost 20% on OOV. Even TnT, the most basic tagger, achieves ALL/OOV accuracy of 91.75/78.33, better than any method in the standard setup. These accuracies are well above those in (Blitzer et al., 2006) and (Huang and Yates, 2010).

Since simply replacing NNPs with NNs has such a large effect, BIO cannot be used sensibly for evaluating DA methods. In practice, it is not possible to separate "true" improvements due to generic better DA from elements of the proposed method that simply introduce a negative bias for NNP.

In summary, when comparing different DA methods caution should be exercised in the choice of domains. In particular, the effect of unknown tags should be made transparent and the gold standards should be analyzed to determine whether the task addressed in the TD differs significantly in some aspects from that addressed in the source domain.

5 Comparison of word representations

Our approach to DA is an instance of representation learning: we aim to find representations that are robust across domains. In this section, we compare FLORS with two other widely used representation learning methods: (i) Brown clusters (Brown et al., 1992) and (ii) C&W embeddings, the word embeddings of Collobert et al. (2011). We use $f_{\text{dist}}(w) = f_{\text{left}}(w) \oplus f_{\text{right}}(w)$ to refer to our own distributional word representations (see Section 2).

The perhaps oldest and most frequently used low-dimensional representation of words is based on Brown clusters. Typically, prefixes of Brown clusters (Brown et al., 1992) are added to increase the robustness of POS taggers (e.g., Toutanova et al. (2003)). Computational costs are high (quadratic in the vocabulary size) although the computation can be parallelized (Uszkoreit and Brants, 2008).

More recently, general word representations (Collobert et al., 2011; Turian et al., 2010) have been used for robust POS tagging. These word representations are typically trained on a large amount of unlabeled text and fine-tuned for specific NLP tasks. Similar to Brown clusters, they are low-dimensional and can be used as features in many NLP tasks, ei-

ther alone or in combination with other features.

To compare $f_{\text{dist}}(w)$ (our distributional representations) with Brown clusters, we induced 1000 Brown clusters on the joint corpus data $\mathcal{D}_{\text{ALL}}$ (see Section 2) using the publicly available implementation of Liang (2005). We padded sentences with ⟨BOUNDARY⟩ tokens on each side and used path prefixes of length 4, 6, 10 and 20 as features for each word (cf. Ratinov and Roth (2009), Turian et al. (2010)).

C&W embeddings are provided by Collobert et al. (2011): 50-dimensional vectors for 130,000 words from WSJ, trained on Wikipedia. Similar to our distributional representations $f_{\text{dist}}(w)$, the embeddings also contain a ⟨BOUNDARY⟩ token (which they call PADDING). Moreover, they have a special embedding for unknown words (called UNKNOWN) which we use whenever we encounter a word that is not in their lookup table. We preprocess our raw tokens the same way they do (lowercase and replace sequences of digits by "0") before we look up a representation during training and testing.

We replaced the distributional features in our basic setup by either Brown cluster features or C&W embeddings. Table 7 repeats lines 5 and 7 of Table 2 and gives results of the modified FLORS setup.

All three representations improve both ALL and OOV accuracies in all domains. f_{dist} outperforms Brown in all cases except for OOV on emails. Brown may suffer from noisy data; cleaning methods have been used in the literature (Liang, 2005; Turian et al., 2010), but they are not unproblematic since a large part of the data available is lost, which results in more unknown words.

Brown and f_{dist} can be directly compared since they were trained on exactly the same data. f_{dist} and C&W are harder to compare directly because there are many differences. (i) C&W is trained on a much larger dataset. One consequence of this is that OOV accuracy on WSJ may be higher because some words that are unknown for other methods are actually known to C&W. (ii) C&W vectors are not trained on the SANCL TD data sets – this gives f_{dist} an advantage. (iii) C&W vectors are not trained on the WSJ. Again, this could give f_{dist} an advantage. (iv) C&W and f_{dist} are fundamentally different in the way they handle unknown words. C&W has a limited vocabulary and must replace all words not in this vocabulary by the token UNKNOWN. In contrast, f_{dist} can create a meaningful individual representation for any OOV word it encounters.

Our FLORS tagger provides best ALL accuracies in all domains but WSJ, where C&W has best results. The good performance of C&W is rather unsurprising since the embeddings were created for the 130,000 most frequent words of the WSJ and thus cover the WSJ domain much better. Also, WSJ was used to tune parameters during development. As with our previous experiments, OOV results on emails seem slightly more sensitive to parameter choices than on other domains (recall the discussion of this issue in Section 4).

In summary, we have shown that f_{dist} representations work better for POS DA than Brown clusters. Furthermore, the evidence we have presented suggests that f_{dist} are comparable in performance to C&W embeddings if not better for POS DA.

The most important difference between f_{dist} and Brown / C&W is that f_{dist} are much simpler and much faster to compute. They are simpler because they are just slightly transformed counts in contrast to the other two approaches, which solve complex optimization problems. f_{dist} can be computed efficiently through simple incrementation in one pass through the corpus. In contrast, the other two approaches are an order of magnitude slower.

6 Related work

Unsupervised DA methods can be broadly put into four categories: representation learning and constraint-based frameworks – which require some tailoring to a task – and instance weighting and bootstrapping – which can be more generally applied to a wide range of problems. Since many approaches are application-specific, we focus on the ones that have been applied to POS tagging.

Representation learning. We already discussed two important approaches to representation learning in Section 5: C&W embeddings and Brown clusters.

Blitzer et al.'s (2006) structural correspondence learning (SCL) supports DA by creating similar representations for correlated features in the *pivot feature space*. This is a potentially powerful method. FLORS is simpler in that correlations are made directly accessible to the supervised learner.

		newsgroups		reviews		weblogs		answers		emails		wsj	
		ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV
1	FLORS f_{dist}(w), n=500	**90.86**	**66.42**	**92.95**	**75.29**	**94.71**	**83.64**	**90.30**	**62.15**	**89.44**	62.61	96.59	90.37
2	FLORS f_{dist}(w), n=0	89.14*	55.59*	91.80*	66.31*	93.40*	72.55*	89.47*	55.82*	88.21*	57.83*	96.29*	85.55*
3	FLORS C&W for f_{dist}(w)	90.57	64.57	92.54*	72.48*	94.51	80.58*	90.23	60.99	**89.44**	63.13	**96.72**	**90.48**
4	FLORS Brown for f_{dist}(w)	90.34*	62.41*	92.23*	71.47*	94.45	81.76	89.71*	56.28*	89.02*	**63.20**	96.48*	87.50

Table 7: Tagging accuracy of different word representations on the dev sets. Line 1 corresponds to FLORS basic. n: number of indicator words. A column's best result is bold.

Moreover, FLORS representations consist of simple counts whereas SCL solves a separate optimization problem for each pivot feature.

Umansky-Pesin et al. (2010) derive distributional information for OOVs by running web queries. This approach is slow since it depends on a search engine.

Ganchev et al. (2012) successfully use search logs. This is a promising enhancement for FLORS.

Huang and Yates (2009) evaluate CRFs with distributional features. They examine lower dimensional feature representations using SVD or the latent states of an unsupervised HMM. They find better accuracies for their HMM method than Blitzer et al. (2006); however, they do not compare them against a CRF baseline using distributional features.

In later work, Huang and Yates (2010) add the latent states of multiple, differently trained HMMs as features to their CRF. Huang and Yates (2012) argue that finding an optimal feature representation is computationally intractable and propose a new framework that allows prior knowledge to be integrated into representation learning.

Latent sequence states are a form of word representation. Thus, it would be interesting to compare them to the non-sequence-based distributional representation that FLORS uses.

Constraint-based methods. Rush et al. (2012) use global constraints on OOVs to improve out-of-domain tagging. Although constraints ensure consistency, they require careful manual engineering. Distributional features can also be seen as a form of constraint since feature weights will be shared among all words.

Subramanya et al. (2010) construct a graph to encourage similar n-grams to be tagged similarly, resulting in moderate gains in one domain, but no gains on BIO when compared to self-training. The reason could be an insufficient amount of unsupervised data for BIO (100,000 sentences). Our approach does not seem to suffer from this problem.

Bootstrapping. Both self-training (McClosky et al., 2006) – which uses one classification model – and co-training (Blum and Mitchell, 1998) – which uses $\geq$2 models – have been applied to POS tagging.

Self-training usually improves a POS baseline only slightly if at all (Huang et al., 2009; Huang and Yates, 2010). Devising features based on labeled instances (instead of training on them) has been more successful (Florian et al., 2004; Søgaard, 2011).

Chen et al. (2011) use co-training for DA. In each round of their algorithm, both new training instances from the unlabeled data and new features are added. Their model is limited to binary classification. The co-training method of Kübler and Baucom (2011) trains several taggers and adds sentences from the TD to the training set on which they agree. They report slight, but statistically significant increases in accuracy for POS tagging of dialogue data.

Instance weighting. Instance weighting formalizes DA as the problem of having data from different probability distributions in each domain. The goal is to make these two distributions align by using instance-specific weights during training. Jiang and Zhai (2007) propose a framework that integrates prior knowledge from different data sets into the learning objective by weights.

In related work, C&P train *generalized* and *domain-specific* models. An input sentence is tagged by the model that is most similar to the sentence. FLORS could be easily extended along these lines, an experiment we plan for the future.

In terms of the basic classification setup, our POS tagger is most similar to the SVM-based approaches of Giménez and Màrquez (2004) and C&P. However, we do not use a left-to-right approach when tagging sentences. Moreover, SVMTool trains two separate models, one for OOVs and one for known words. FLORS only has a single model. In addition,

we do not make use of ambiguity classes, token-tag dictionaries and rare feature thresholds. Instead, we rely only on three types of features: distributional representations, suffixes and word shapes.

The local-context-only approach of SVMTool, C&P and FLORS is different from standard sequence classification such as MEMMs (e.g., Ratnaparkhi (1996), Toutanova et al. (2003), Tsuruoka and Tsujii (2005)) and CRFs (e.g., Collins (2002)). Sequence models are more powerful in theory, but this may not be an advantage in DA because the subtle dependencies they exploit may not hold across domains.

7 Conclusion

We have presented FLORS, a new POS tagger for DA. FLORS uses robust representations that work especially well for unknown words and for known words with unseen tags. FLORS is simpler and faster than previous DA methods, yet we were able to demonstrate that it has significantly better accuracy than several baselines.

Acknowledgments. This work was supported by DFG (Deutsche Forschungsgemeinschaft).

References

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *EMNLP*, pages 120–128.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *COLT*, pages 92–100.

Thorsten Brants. 2000. TnT: A statistical part-of-speech tagger. In *ANLP*, pages 224–231.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

Minmin Chen, Kilian Q. Weinberger, and John Blitzer. 2011. Co-training for domain adaptation. In *NIPS*, pages 1–9.

Jinho D. Choi and Martha Palmer. 2012. Fast and robust part-of-speech tagging using dynamic model selection. In *ACL: Short Papers*, pages 363–367.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *EMNLP*, pages 1–8.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874.

Steven Finch and Nick Chater. 1992. Bootstrapping syntactic categories using statistical methods. In *Background and Experiments in Machine Learning of Natural Language*, pages 229–235.

Radu Florian, Hany Hassan, Abraham Ittycheriah, Hongyan Jing, Nanda Kambhatla, Xiaoqiang Luo, Nicolas Nicolov, and Salim Roukos. 2004. A statistical model for multilingual entity detection and tracking. In *HLT-NAACL*, pages 1–8.

Kuzman Ganchev, Keith Hall, Ryan McDonald, and Slav Petrov. 2012. Using search-logs to improve query tagging. In *ACL: Short Papers*, pages 238–242.

Jesús Giménez and Lluís Màrquez. 2004. SVMTool: A general pos tagger generator based on support vector machines. In *LREC*, pages 43–46.

Fei Huang and Alexander Yates. 2009. Distributional representations for handling sparsity in supervised sequence-labeling. In *ACL-IJCNLP*, pages 495–503.

Fei Huang and Alexander Yates. 2010. Exploring representation-learning approaches to domain adaptation. In *DANLP*, pages 23–30.

Fei Huang and Alexander Yates. 2012. Biased representation learning for domain adaptation. In *EMNLP-CoNLL*, pages 1313–1323.

Zhongqiang Huang, Vladimir Eidelman, and Mary Harper. 2009. Improving a simple bigram HMM part-of-speech tagger by latent annotation and self-training. In *NAACL-HLT: Short Papers*, pages 213–216.

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *ACL*, pages 264–271.

S. Sathiya Keerthi and Chih-Jen Lin. 2003. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural computation*, 15(7):1667–1689.

Sandra Kübler and Eric Baucom. 2011. Fast domain adaptation for part of speech tagging for dialogues. In *RANLP*, pages 41–48.

Percy Liang, Hal Daumé III, and Dan Klein. 2008. Structure compilation: trading structure for features. In *ICML*, pages 592–599.

Percy Liang. 2005. Semi-supervised learning for natural language processing. Master's thesis, Massachusetts Institute of Technology.

Christopher D. Manning. 2011. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In *CICLing*, pages 171–189.

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

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Reranking and self-training for parser adaptation. In *ACL*, pages 337–344.

John Miller, Manabu Torii, and Vijay K. Shanker. 2007. Building domain-specific taggers without annotated (domain) data. In *EMNLP-CoNLL*, pages 1103–1111.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *HLT-NAACL*, pages 404–411.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 Shared Task on Parsing the Web. Notes of the 1st SANCL Workshop.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *CoNLL*, pages 147–155.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *EMNLP*, pages 133–142.

Alexander M. Rush, Roi Reichart, Michael Collins, and Amir Globerson. 2012. Improved parsing and POS tagging using inter-sentence consistency constraints. In *EMNLP-CoNLL*, pages 1434–1444.

Beatrice Santorini. 1990. Part-of-speech tagging guidelines for the Penn Treebank project (3rd revision, 2nd printing). Technical report, Department of Linguistics, University of Pennsylvania.

Tobias Schnabel and Hinrich Schütze. 2013. Towards robust cross-domain domain adaptation for part-of-speech tagging. In *IJCNLP*, pages 198–206.

Hinrich Schütze. 1993. Part-of-speech induction from scratch. In *ACL*, pages 251–258.

Hinrich Schütze. 1995. Distributional part-of-speech tagging. In *EACL*, pages 141–148.

Anders Søgaard. 2011. Semisupervised condensed nearest neighbor for part-of-speech tagging. In *ACL: Short papers*, pages 48–52.

Amarnag Subramanya, Slav Petrov, and Fernando Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *EMNLP*, pages 167–176.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL-HLT*, pages 173–180.

Yoshimasa Tsuruoka and Jun'ichi Tsujii. 2005. Bidirectional inference with the easiest-first strategy for tagging sequence data. In *EMNLP-HLT*, pages 467–474.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *ACL*, pages 384–394.

Shulamit Umansky-Pesin, Roi Reichart, and Ari Rappoport. 2010. A multi-domain web-based algorithm for POS tagging of unknown words. In *COLING*, pages 1274–1282.

Jakob Uszkoreit and Thorsten Brants. 2008. Distributed word clustering for large scale class-based language modeling in machine translation. In *ACL*, pages 755–762.