Real-Time Adaptation of Lexical Embeddings for Enhanced Part-of-Speech Tagging

Abstract

This research introduces a method for real-time unsupervised domain adaptation (DA) that can be applied incrementally as new information arrives. This method is especially useful when conventional batch DA is unfeasible. Through evaluations focused on part-of-speech (POS) tagging, we observe that real-time unsupervised DA achieves accuracy levels on par with those of batch DA.

1 Introduction

Unsupervised domain adaptation is a frequently encountered challenge for developers aiming to create robust natural language processing (NLP) systems. This situation typically arises when labeled data is available for a source domain, but there is a need to enhance performance in a target domain using only unlabeled data. A majority of the current NLP research on unsupervised domain adaptation employs batch learning, which presumes the availability of a substantial corpus of unlabeled data from the target domain before the testing phase. However, batch learning is impractical in numerous real-world situations where data from a new target domain must be processed without delay. Further, in many practical scenarios, data may not be neatly categorized by domain, making it difficult to immediately discern when an input stream begins providing data from a new domain.

For instance, consider an NLP system within a company that is tasked with analyzing a continuous stream of emails. This stream evolves over time without any explicit signals indicating that the current models should be adjusted to the new data distribution. Given that the system is expected to operate in real-time, it would be beneficial for any system adaptation to be done in an online manner, as opposed to the batch method, which involves halting the system, modifying it, and then restarting it.

This paper introduces real-time unsupervised domain adaptation as an enhancement to conventional unsupervised DA. In this approach, domain adaptation is carried out incrementally as data is received. Specifically, our implementation involves a type of representation learning, where the focus is on updating word representations in our experiments. Every instance a word appears in the data stream during testing, its representation is refined.

To our understanding, the research presented here is the first to examine real-time unsupervised DA. In particular, we assess this method for POS tagging tasks. We analyze POS tagging outcomes using three different methods: a static baseline, batch learning, and real-time unsupervised DA. Our findings indicate that real-time unsupervised DA performs comparably to batch learning, yet it does not require retraining or pre-existing data from the target domain.

2 Experimental setup

Tagger. We have adapted the FLORS tagger, which is recognized for its speed and simplicity, and is particularly effective in DA scenarios. This tagger approaches POS tagging as a multi-label classification problem within a window-based framework, rather than a sequence classification one. FLORS is well-suited for real-time unsupervised DA because its word representations include

distributional vectors, which can be updated during both batch learning and real-time unsupervised DA. Each word's representation in FLORS consists of four feature vectors: one for its suffix, one for its shape, and one each for its left and right distributional neighbors. Suffix and shape features are standard in the literature, and we utilize them as described previously.

Distributional features. The ith element xi of the left distributional vector for a word w is the weighted count of times the indicator word ci appears immediately to the left of w:

$$xi = tf(freq(bigram(ci, w))) \tag{1}$$

where ci is the word with frequency rank i in the corpus, freq(bigram(ci, w)) is the occurrence count of the bigram "ci w", and non-zero frequencies are weighted logarithmically: tf(x) = 1 + log(x). The right distributional vector is defined similarly. We limit the set of indicator words to the 500 most frequent. To avoid zero vectors, an additional element xn+1 is added to each vector to account for omitted contexts:

$$xn + 1 = tf(\sum .5freq(bigram(c_i, w))) \tag{2}$$

Let f(w) be the concatenation of the two distributional, suffix, and shape vectors of word w. Then FLORS represents token vi as follows:

$$f(vi\Phi\check{2}2122)\Phi\check{2}295f(vi\Phi\check{2}2121)\Phi\check{2}295f(vi)\Phi\check{2}295f(vi+1)\Phi\check{2}295f(vi+2) \tag{3}$$

where ŏ2295 is vector concatenation. FLORS then tags token vi based on this representation.

FLORS operates under the assumption that the fundamental relationship between distributional features and labels remains consistent when transitioning from the source to the target domain. This contrasts with other studies that select "stable" distributional features and discard "unstable" ones. The central hypothesis of FLORS is that fundamental distributional POS characteristics are relatively stable across different domains, unlike semantic or more intricate tasks. The effectiveness of FLORS suggests the validity of this hypothesis.

Data. *Test set.* Our evaluation utilizes the development sets from six different target domains (TDs): five SANCL domains (newsgroups, weblogs, reviews, answers, emails) and sections 22-23 of the Wall Street Journal (WSJ) for in-domain testing.

Two training sets of varying sizes are employed. In the l:big condition (large labeled data set), FLORS is trained on sections 2-21 of the WSJ. The l:small condition uses 10% of the l:big data set.

Data for word representations. We also adjust the size of the datasets used for computing word representations before training the FLORS model. In the u:big condition, distributional vectors are computed on the combined corpus of all labeled and unlabeled text from both source and target domains (excluding test sets), along with 100,000 WSJ sentences from 1988 and 500,000 sentences from a large external corpus. In the u:0 condition, only labeled training data is utilized.

Methods. We implemented a modification from the original setup: distributional vectors are stored in memory as count vectors, enabling count increases during online tagging.

Experiments are conducted with three versions of FLORS: STATIC, BATCH, and ONLINE. All three methods compute word representations on "data for word representations" before model training on one of the two "training sets".

STATIC. Word representations remain unchanged during testing.

BATCH. Before testing, count vectors are updated by freq(bigram(ci, w)) += freq*(bigram(ci, w)), where freq*(ŏ0b7) denotes the bigram "ci w" occurrences in the entire test set.

ONLINE. Before tagging a test sentence, both left and right distributional vectors are updated via freq(bigram(ci, w)) += 1 for each "ci w" bigram appearance in the sentence. The sentence is then tagged using the updated word representations. As tagging progresses, distributional representations become increasingly specific to the target domain (TD), converging to the representations that BATCH uses at the end of the tagging process.

In all three modes, suffix and shape features are always fully specified, for both known and unknown words.

3 Experimental results

Table 1 shows that the performance levels of BATCH and ONLINE are on par with each other and represent the current state-of-the-art. The highest accuracy in each column is highlighted in bold.

Table 1: BATCH and ONLINE accuracies are comparable and state-of-the-art. Best number in each column is bold.

	wsj	newsgroups		reviews		weblogs		answers		emails		
	ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV	ALL	OOV
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
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
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
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
S&S	90.86	66.42	92.95	75.29	94.71	83.64	90.30	62.16	89.44	62.61	96.59	90.37
S&S (reimpl.)	90.68	65.52	93.00	75.50	94.64	82.91	90.18	61.98	89.53	62.46	96.60	89.70
BATCH	90.87	71.18	93.07	79.03	94.86	86.53	90.70	65.29	89.84	65.44	96.63	91.86
ONLINE	90.85	71.00	93.07	79.03	94.86	86.53	90.68	65.16	89.85	65.48	96.62	91.69

Table 2 shows that the accuracy rates for ONLINE and BATCH methods are generally superior to those of the STATIC method, as indicated by the numbers in bold. It also demonstrates that performance improves with an increase in both training data and unlabeled data.

The performance of ONLINE is similar to that of BATCH. It is slightly lower than BATCH in the u:0 condition, with the most significant difference in accuracy being 0.29, and it is at most 0.02 different from BATCH in terms of overall accuracy in the u:big condition. The reasons for ONLINE occasionally outperforming BATCH, particularly in certain conditions, are discussed subsequently.

3.1 Time course of tagging accuracy

The ONLINE model introduced here has a unique characteristic not commonly found in other statistical NLP research: its predictive accuracy evolves as it processes text due to the modification of its representations.

To analyze the progression of these changes over time, a substantial application domain is necessary because subtle changes might be too inconsistent in the smaller test sets of the SANCL TDs. The WSJ corpus is the only labeled domain that is sufficiently large for this purpose. Consequently, we invert the usual setup by training the model on the development sets of the five SANCL domains (l:big) or on the initial 5000 labeled words of reviews (l:small). In this reversed setup, u:big utilizes the five unlabeled SANCL datasets along with a large external corpus as before. Given the importance of performance variability, we conduct 100 trials on randomly selected 50% samples of WSJ and report both the average and standard deviation of tagging errors across these trials.

The results presented in Table 3 indicate that ONLINE's error rates are only marginally higher than, or comparable to, those of BATCH. Specifically, in the l:small/u:0 condition, the error rate for known words is lower for ONLINE (0.1186) than for BATCH, similar to observations in Table 2.

Table 2: ONLINE / BATCH accuracies are generally better than STATIC (see bold numbers) and improve with both more training data and more unlabeled data.

			u:0			u:big				
		ALL	KN	SHFT	OOV	ALL	KN	SHFT	OOV	
	l:small	STATIC	87.02	90.87	71.12	57.16	89.02	91.48	81.53	58.30
		ONLINE	87.99	90.87	76.10	65.64	89.84	92.38	82.58	67.09
newsgroups	l:big	BATCH	88.28	91.08	77.01	66.37	89.82	92.37	82.65	67.03
		STATIC	89.69	93.00	82.65	57.82	89.93	92.41	84.94	58.97
		ONLINE	**90.51**	**93.13**	82.51	**67.57**	**90.85**	**93.04**	84.94	**71.00**
		BATCH	90.69	93.12	**83.24**	69.43	90.87	93.03	**85.20**	71.18
	l:small	STATIC	89.08	91.96	66.55	65.90	91.45	92.47	80.11	70.81
		ONLINE	89.67	92.14	70.14	69.67	92.11	93.62	81.46	78.42
reviews	l:big	BATCH	89.79	92.23	69.86	71.27	92.10	93.60	81.51	78.42
		STATIC	91.96	93.94	82.30	67.97	92.42	93.53	84.65	69.97
		ONLINE	92.33	94.03	83.59	72.50	**93.07**	**94.36**	**85.71**	**79.03**
		BATCH	**92.42**	**94.09**	**83.53**	**73.35**	**93.07**	**94.36**	**85.71**	**79.03**
	l:small	STATIC	91.58	94.29	79.95	72.74	93.42	94.77	89.80	77.42
		ONLINE	92.51	94.52	81.76	80.46	94.21	95.40	91.08	84.03
weblogs	l:big	BATCH	92.68	94.60	82.34	81.20	94.20	95.42	91.03	83.87
		STATIC	93.45	95.64	90.15	72.68	94.09	95.54	91.90	76.94
		ONLINE	94.18	95.82	89.80	80.35	**94.86**	**95.81**	**92.60**	**86.53**
!		BATCH	**94.34**	**95.85**	**90.03**	**81.84**	**94.86**	95.82	**92.60**	**86.53**
	l:small	STATIC	86.93	90.89	66.51	53.43	88.98	91.09	77.63	57.36
		ONLINE	87.48	91.18	68.07	56.47	89.71	92.42	78.11	64.21
answers	l:big	BATCH	87.56	91.11	68.25	58.44	89.71	92.43	78.23	64.09
		STATIC	89.54	92.76	78.65	56.22	90.06	92.18	80.70	58.25
		ONLINE	89.98	92.97	79.07	59.77	**90.68**	**93.21**	**81.48**	**65.16**
		BATCH	**90.14**	**93.10**	**79.01**	**60.72**	90.70	93.22	81.54	65.29
	l:small	STATIC	85.43	90.85	57.85	51.65	87.76	90.35	70.86	56.76
		ONLINE	86.30	91.26	60.56	55.83	88.45	92.31	71.67	61.57
emails	l:big	BATCH	86.42	91.31	61.03	56.32	88.46	92.32	71.71	61.65
		STATIC	88.31	92.98	71.38	52.71	89.21	91.74	73.80	58.99
		ONLINE	88.86	93.08	72.38	57.78	**89.85**	**93.30**	**75.32**	**65.48**
		BATCH	**88.96**	**93.11**	**72.28**	**58.85**	89.84	93.30	75.27	65.44
	l:small	STATIC	94.64	95.44	83.38	82.72	95.73	95.88	90.36	87.87
		ONLINE	94.86	95.53	85.37	85.22	95.80	96.21	89.89	89.70
wsj	l:big	BATCH	94.80	95.46	85.51	85.38	95.80	96.22	89.89	89.70
		STATIC	96.44	96.85	92.75	85.38	96.56	96.72	93.35	88.04
		ONLINE	96.50	**96.85**	**93.55**	**86.38**	**96.62**	**96.89**	**93.35**	**91.69**
		BATCH	**96.57**	96.82	93.48	86.54	96.63	**96.89**	**93.42**	91.86

Table 3 also includes data on "unseens" along with unknowns, as prior research indicates that unseens lead to at least as many errors as unknowns. Unseens are defined as words with tags not present in the training data, and error rates for unseens are calculated across all their occurrences, including those with both seen and unseen tags. As shown in Table 3, the error rate for unknowns is higher than that for unseens, which in turn is higher than the error rate for known words.

When examining individual conditions, ONLINE generally outperforms STATIC, showing better results in 10 out of 12 cases and only slightly underperforming in the l:small/u:big condition for unseens and known words (0.1086 vs. 0.1084, 0.0802 vs. 0.0801). In four conditions, ONLINE is significantly better, with improvements ranging from 0.005 to over 0.06. The differences between ONLINE and STATIC in the remaining eight conditions are minimal. For the six u:big conditions, this is expected as the large unlabeled dataset is from the news domain, similar to WSJ. Therefore, if large unlabeled datasets similar to the target domain are available, using STATIC tagging may suffice since the additional effort for ONLINE/BATCH may not be justified.

Table 3: Error rates (err) and standard deviations (std) for tagging. ˇ2020 (resp. ˇ2217): significantly different from ONLINE error rate above&below (resp. from "u:0" error rate to the left).

		unknowns				unseens				
		u:0		u:big		u:0		u:big		
		err	std	err	std	err	std	err	std	err
l:small	STATIC	.3670ˇ2020	.00085	.3094	.00160	.1659ˇ2020	.00076	.1467	.00120	.1309ˇ202
	ONLINE	.3050ˇ2020	.00143	.2104	.00081	.1646ˇ2020	.00145	.1084	.00056	.1251ˇ202
	BATCH	.3094	.00160	.2102ˇ2217	.00093	.1404	.00125	.1037ˇ2217	.00098	.1186
l:big	STATIC	.1451ˇ2020	.00114	.1042	.00100	.0732	.00052	.0690	.00042	.0534
	ONLINE	.1404	.00125	.1037ˇ2217	.00098	.0727	.00051	.0689ˇ2217	.00051	.0529
	BATCH	.1382ˇ2020	.00140	.1033	.00112	.0723	.00065	.0680	.00062	.0528

Increasing the amount of labeled data consistently reduces error rates, as does increasing unlabeled data. The differences are significant for ONLINE tagging in all six cases, marked by ˇ2217 in the table.

There is no significant difference in variability between ONLINE and BATCH, suggesting that ONLINE is preferable due to its equal variability and higher performance, without requiring a dataset available before tagging begins.

The progression of tagging accuracy over time is illustrated in Figure 1. BATCH and STATIC maintain constant error rates as they do not adjust representations during tagging. ONLINE's error rate for unknown words decreases, approaching BATCH's error rate, as more is learned with each occurrence of an unknown word.

4 Related Work

Online learning typically refers to supervised learning algorithms that update the model after processing a few training examples. Many supervised learning algorithms are online or have online versions. Active learning is another supervised learning framework that processes training examples ˇ2014 usually obtained interactively ˇ2014 in small batches. All of this work on supervised online learning is not directly relevant to this paper since we address the problem of unsupervised domain adaptation. Unlike online supervised learners, we keep the statistical model unchanged during domain adaptation and adopt a representation learning approach: each unlabeled context of a word is used to update its representation.

There is much work on unsupervised domain adaptation for part-of-speech tagging, including work using constraint-based methods, instance weighting, self-training, and co-training. All of this work uses batch learning. For space reasons, we do not discuss supervised domain adaptation.

5 Conclusion

This study introduces a method for real-time updating of word representations, a new form of domain adaptation designed for scenarios where target domain data are processed in a stream, making BATCH processing unfeasible. We demonstrate that real-time unsupervised domain adaptation achieves performance levels comparable to batch learning. Moreover, it significantly reduces error rates compared to STATIC methods, which do not employ domain adaptation.

Acknowledgments. This research was supported by a scholarship from Baidu awarded to Wenpeng Yin and by the Deutsche Forschungsgemeinschaft (grant DFG SCHU 2246/10-1 FADeBaC).