Vol.2011-NL-201 No.13
Vol.2011-SLP-86 No.13
2011/5/17

部分的アノテーションから学習可能な係り受け解析器

森　信介[†1]　FLANNERY Daniel [†2]
宮尾祐介[†3]　NEUBIG Graham[†2]

本論文では、部分的アノテーションコーパスから学習することができる最大全域木に基づく係り受け解析を提案する。この枠組みにより、様々な言語資源が利用可能になる。この枠組みは、分野適応などの現実的な状況においてとりわけ重要である。日本語を対象とした係り受け解析の実験の結果、フルアノテーションコーパスのみから学習可能である従来手法と同程度の解析精度が得られることと部分的アノテーションコーパスの利用による分野適応が可能であることを確認した。

Training MST Parsers from Partially Annotated Corpora

Shinsuke MORI ,[†1] Daniel FLANNERY ,[†2]
Yusuke MIYAO [†3] and Graham NEUBIG[†2]

We introduce a maximum spanning tree (MST) dependency parser that can be trained from partially annotated corpora, allowing us to maximize the use of available linguistic resources and reduce the costs of preparing new training data. This is especially important for domain adaptation in a real-world situation. Experiments on Japanese dependency parsing show that this approach allows for rapid training and achieves accuracy comparable to parsers trained on fully annotated data.

†1 京都大学学術情報メディアセンター
Academic Center for Computing and Media Studies, Kyoto University
†2 京都大学情報学研究科
Graduate School of Infomatics, Kyoto University
†3 国立情報学研究所
National Institute of Informatics

1. Introduction

Parsing is one of the fundamental building blocks of natural language processing, with applications ranging from machine translation[1)] to information extraction[2)]. However, while statistical parsers achieve higher and higher accuracies on in-domain text, the creation of data to train these parsers is labor-intensive, which becomes a bottleneck for smaller languages. In addition, it is also a well known fact that accuracy plummets when tested on sentences of a different domain than the training corpus[3),4)], so in-domain data must be annotated to make up for this weakness.

In this paper, we propose a maximum spanning tree (MST) parser that helps ameliorate these problems by allowing for the efficient development of training data. This is done through a combination of a novel parsing method and an efficient corpus annotation strategy. For corpus construction, we use partial annotation[5),6)], which allows an annotator to skip annotation of unnecessary edges, focusing their efforts only on the ones that will provide the maximal gains of accuracy. In the parser, we make the assumption that the score of each edge is independent of the other edges in the dependency tree, which allows for simple training using either fully or partially annotated data.

We perform an evaluation of the proposed method on a Japanese dependency parsing task. First, we compare the proposed method to both a traditional MST parser[7)] (which cannot be trained on partially annotated data), and a deterministic parser[8)]. We find that the proposed method is able to achieve accuracy similar to that of the traditional MST parser, and training and testing speeds similar to that of the deterministic parser. Second, we perform experiments in both domain adaptation and small-data settings, and find that partial annotation allows for greater performance gains with comparable amounts of annotated data.

2. Pointwise estimation for dependency parsing

This work follows the standard setting of recent work on dependency parsing[9)]. Given as input a sequence of words, $\boldsymbol{w} = \langle w_1, w_2, \ldots, w_n \rangle$, the goal is to output a dependency

Vol.2011-NL-201 No.13
Vol.2011-SLP-86 No.13
2011/5/17

tree $\boldsymbol{d} = \langle d_1, d_2, \ldots, d_n \rangle$, where $d_i \equiv j$ when the head of w_i is w_j.[⋆1] We assume that $d_i = 0$ for some word w_i in a sentence, which indicates that w_i is the head of the sentence.

The parsing model we pursue in this paper is McDonald et al.[7]'s edge-factored model. A score, $\sigma(d_i)$, is assigned to each edge (i.e. dependency) d_i, and the parsing is to find a dependency tree, $\hat{\boldsymbol{d}}$, that maximizes the sum of the scores of all the edges.

$$\hat{\boldsymbol{d}} = \mathbf{argmax}_{\boldsymbol{d}} \sum_{d \in \boldsymbol{d}} \sigma(d).$$

It has been known that, given $\sigma(d)$ for all possible dependencies in a sentence, $\hat{\boldsymbol{d}}$ can be computed by the maximum spanning tree algorithm such as Chu-Liu/Edmonds' algorithm.

An important difference from McDonald et al.[7] is in the estimation of $\sigma(d)$. McDonald et al.[7] applied a perceptron-like algorithm that optimizes a score of entire dependency trees. However, we stick to pointwise estimation: $\sigma(d_i)$ is estimated for each w_i independently. A variety of machine learning-based classifiers can be applied to the estimation of $\sigma(d)$, because it is essentially a n-class classification problem. In the experiments, we estimate a log-linear model $p(d_i = j) \equiv p(j|\boldsymbol{w}, i)$, and $\sigma(d_i)$ is defined as log probability $\log p(d_i)$. It should be noted that the probability depends only on i, j, and the input $\boldsymbol{w}$, which assures that $p(d_i)$ is estimated independently for each w_i. Because parameter estimation does not involve computing $\hat{\boldsymbol{d}}$, we do not apply the maximum spanning tree algorithm in training.

Our current implementation uses features on the distance between a dependent word and its candidate head, the surface forms of the dependent/head words and their surrounding words (up to three words before/after the dependent/head words), and the parts-of-speech of the dependent/head words.

Pointwise estimation rather than structured estimation might hurt parsing accuracy. However, our method can enjoy greater flexibility, which allows for training from partially annotated corpora as will be described in Section 3.

In the experiments, we target Japanese parsing. Because Japanese is a head-final language, we assume $d_i > i$ for all $i \neq n$ and $d_n = 0$. This assumption reduces the maximum spanning tree algorithm to a simpler algorithm: for each word we select the dependency with the maximum score. This never creates a loop of dependencies, and a recursive process as in Chu-Liu/Edmonds' algorithm is not necessary.

⋆1 While we describe unlabeled dependency parsing for simplicity, it is trivial to extended it to labeled dependency parsing.

i	1	2	3	4	5	6	7	8	9
w_i	政府	は	投資	に	つなが	る	と	歓迎	し
Eng.	Gov.	*subj.*	investment	to	leads	*ending*	that	welcomes	do
d_i		8							

The second word, case marker は (*subj.*), has two grammatically possible heads: the verb つなが (leads) and the verb 歓迎 (welcomes). In our framework, only this word needs to be annotated with its head.

図 1 部分的アノテーションコーパスの例
Fig. 1 An example of a partially annotated sentence.

3. Domain Adaptation for MST Parsing

Assuming that the cost of annotation corresponds roughly to the number of annotations performed, out of all possible annotations to have annotators perform for a target domain corpus we want to select the ones which provide the greatest benefit to accuracy when training. The high cost of annotation work is the primary motivation for this approach.

3.1 Partial Annotation for a Parser

Before text can be annotated with dependencies for use in our system, it must first be tokenized and labeled with POS tags. [⋆2] We assume that the results of this tokenization and POS tagging are accurate enough that we need to manually annotate only the dependencies between the tokenized words.

In the context of dependency parsing, partial annotation refers to annotating only certain dependencies between words in a sentence. Dependencies which are assumed to

⋆2 We take a language-independent approach that does not make any assumptions about the unit of tokenization or the meaning of tags used.

Vol.2011-NL-201 No.13
Vol.2011-SLP-86 No.13
2011/5/17

have little to no value for training are left unannotated. Figure 1 shows an example of a partially annotated sentence that can be used as training data by our system.

3.2 Estimating Edge Score from Partial Annotations

As explained in Section 2, edge scores, $\sigma(d_i)$, are estimated for each w_i independently. This means that the estimation of $\sigma(d_i)$ requires only a gold dependency of w_i, and the other dependencies in a sentence are not necessary. This allows us to learn $\sigma(d_i)$ from partially annotated corpora. When training data includes a gold dependency that w_i depends on w_j, a discriminative classifier like a log-linear model can be trained by regarding $d_i = j$ as a positive sample and $d_i = j'$ s.t. $j' \neq j$ as negative samples.

In the case of Japanese parsing, because $j > i$ for all $d_i = j$, negative samples are $d_i = j'$ s.t. $j' \neq j \wedge j' > i$. For example, from the partial annotation given in Figure 1, we can create a training instance for w_2, は (*subj.*), where the positive sample is $d_2 = 8$ and the negative samples are $d_2 = 3, 4, \ldots, 7, 9$.

3.3 Dependency Selection Criterion

The criterion we use to select words to annotate with their heads is based on the idea of additive smoothing[10]. The motivation for this idea is that it yields better performance than a simple maximum likelihood estimate when the size of the training corpus is small. The following is the procedure to annotate a corpus with k dependencies.

(1) count the frequency of each word $f(w)$ in the training corpus.

(2) select k words according to the following probability

$$\frac{\alpha + f(w)}{|\mathcal{W}|\alpha + \sum_i f(w_i)},$$

where $\mathcal{W}$ is the set of words appearing in the training corpus.

(3) annotated the selected words with their heads.

This criterion is very naive but is expected to work better than randomly selecting words to annotate. Theoretically speaking, it may be a good idea to annotate all dependencies in which the selected word appears, but for an annotator finding all of a word's dependents is much more difficult than finding only its head. This is because a word has only one head, which for Japanese we assume always occurs to the right of that word in the sentence. In contrast, there could be multiple dependents.

表 1 コーパスサイズ
Table 1 Sizes of Corpora.

ID	source	usage	#sentences	#words	#chars
EHJ-train	example sentences	learning	11,700	145,925	197,941
EHJ-test	from a dictionary	test	1,300	16,348	22,207
NKN-train	newspaper	PA pool	9,023	263,427	398,570
NKN-test	articles	test	1,002	29,038	43,695

NKN-train is used as a partial annotation (PA) pool.

3.4 Related Work

There has been a significant amount of work on how to utilize in-domain data to improve the accuracy of parsing. The majority of this work has focused on using unlabeled data in combination with self-training[11),12)] or other semi-supervised learning methods[13)–15)]. Roark and Bacchiani[11)] also presents work on supervised domain adaptation, although this focuses on the utilization of an already-existing in-domain corpus.

There has also been some work on efficient annotation of data for parsing[16)–18)]. In particular Sassano and Kurohashi[18)] present a method for using partially annotated data with deterministic dependency parsers. In contrast, we present results for MST parsers, and demonstrate effectiveness in a domain adaptation scenario, where large amounts of labeled out-of-domain data are available.

4. Evaluation

As an evaluation of our parser, we measured parsing accuracies of several systems on test corpora in two domains: one is a general domain in which a fully annotated corpus annotated with word boundary and dependency information is available, and the other is a target domain assuming an adaptation situation in which only a partially annotated corpus is available for quick and low-cost domain adaptation.

4.1 Experimental Settings

In the experiments we used example sentences from a dictionary[19)] as the general domain data, and business newspaper articles (Nikkei), similar to the Wall Street Journal, as the target domain data. Their usages and specifications are shown in Table 1. All the sentences are segmented into words manually and all the words are annotated with

Vol.2011-NL-201 No.13
Vol.2011-SLP-86 No.13
2011/5/17

表 2　EHJ コーパスに対する解析精度
Table 2　Parsing Accuracy on EHJ-test.

method	EHJ-test
Malt	96.63%
MST	96.67%
PW	96.83%

All systems were trained on EHJ-train.

their heads manually ⋆1. The dependencies have no labels because almost all nouns are connected to a verb with a case marker and many important labels are obvious. The words are not annotated with part of speech (POS) tags, so we used a Japanese POS tagger, KyTea[20], trained on about 40k sentences from the BCCWJ[21].

For the general domain experiments we compared the following systems, using projective parsing algorithms for training because of the assumptions about Japanese parsing outlined in Section 2.

(1) **Malt**: Nivre et al.[22]'s MaltParser, using Nivre's arc-eager algorithm and the option for strict root handling.

(2) **MST**: McDonald et al.[7]'s MST Parser, using k-best parse size with k=5.

(3) **PW**: Our system, where pointwise estimation is used to estimate dependencies. Stochastic gradient descent training for log-linear models is used.

4.2 With a Fully Annotated Training Corpus

For the first experiment, we measured the accuracy of each system on an in-domain test set when training on a fully annotated corpus. The results are shown in Table 2. Malt and MST have similar accuracy, but PW outperforms both of these systems. We also measured the training time and the parsing speed of each system. Table 3 shows the results. From this table, first we see that MST is much slower than Malt, as is well known. Our method, however, is much faster than MST and the parsing speed is approximately the same as the shift-reduce-based Malt.

Active learning has been shown to effective at identifying informative examples to annotate for domain adaptation[23),24)], but since the system must be retrained after every set of annotations training time can become a bottleneck. Theoretically the training time of our method is proportional to the number of annotated dependencies. On this size of training corpus the training speed is fast enough that we can adopt an active learning framework in a real domain adaptation situation. This is one of the main advantages of our framework.

We performed a second experiment in the general domain to measure the impact of the training corpus size on parsing accuracy. To make smaller training corpora, we set a

⋆1 The Japanese data provided by the CoNLL organizers[9] are the result of an automatic conversion from phrase (*bunsetsu*) dependencies. For a more appropriate evaluation we have prepared a word-based dependency data set.

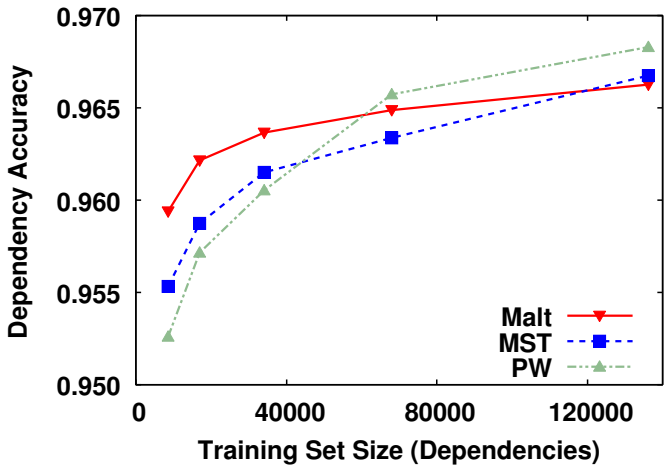

図 2　各解析器の精度
Fig. 2　Comparison of parsing accuracy for different parsers.

表 3　学習に要した時間と解析速度
Table 3　Training Time and Parsing Speed.

method	training time	parsing speed
Malt	14[s]	1.3[ms/sent.]
MST	1901[s]	32.7[ms/sent.]
PW	125[s]	2.8[ms/sent.]

All systems were trained on EHJ-train and tested on EHJ-test. The machine used had a 3.33GHz processor and 12GB of RAM.

Vol.2011-NL-201 No.13
Vol.2011-SLP-86 No.13
2011/5/17

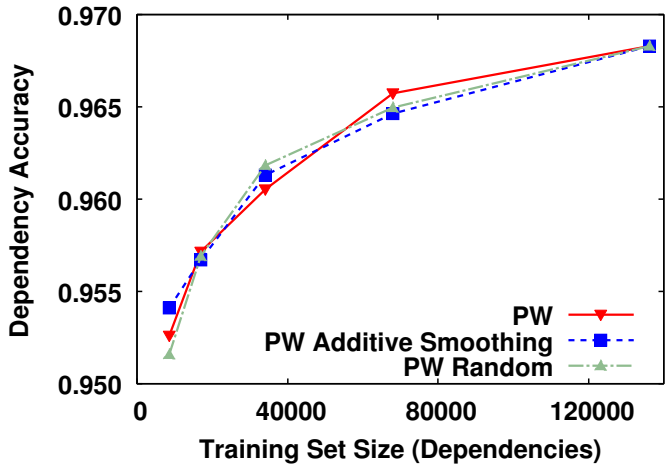

図 3　各アノテーション方法の解析精度
Fig. 3　Comparison of parsing accuracy for different annotation methods.

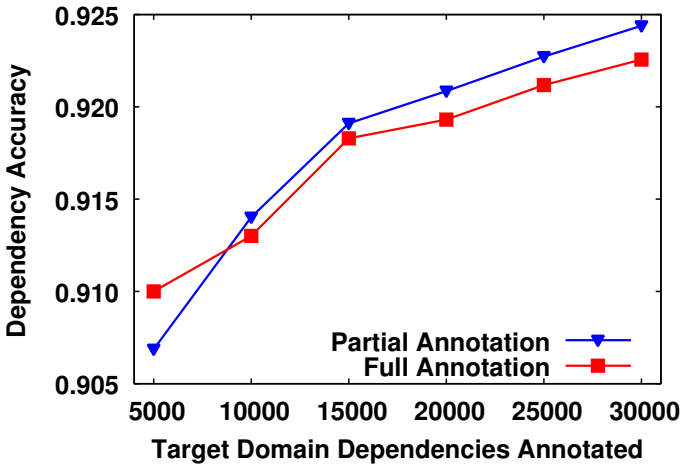

図 4　NKN-train に付与した係り受けの数と NKN-test に対する解析精度の関係
Fig. 4　The relationship between the parsing accuracy on NKN-test and the number of dependencies in NKN-train.

fixed number of dependency annotations and then sequentially selected sentences from EHJ-train until the desired number of dependency annotations were collected. The results are shown in Figure 2. Though PW achieves the highest accuracy when the full training corpus is used, Malt has higher accuracy than both of the MST-based systems when the training corpus is one-third or less the size of the full training corpus. It can also be shown that both MST-based systems improve at a similar rate for all sizes of training corpora.

We also tested two different annotation methods on training corpora of various sizes in preparation for the domain adaptation experiment. We use PW as the baseline for this comparison.

(1) **PW Additive Smoothing**: Our system, using the dependency selection criterion we proposed in Section 3.3 to perform partial annotations. We set $\alpha = 0.5$.

(2) **PW Random**: Our system, using fully annotated sentences selected randomly from the training data.

The results are shown in Figure 3. Comparing the different annotation methods for PW, we see that the accuracy of PW Additive Smoothing is comparable to PW Random and the baseline PW across the different sizes of training corpora. This shows that the dependency selection criterion we proposed can make partial annotation a viable alternative to full annotation methods.

4.3 Domain Adaptation with a Partially Annotated Training Corpus

One of the advantages of our parser is that it can be trained on a partially annotated corpus. Thus for the domain adaptation experiment, we used EHJ-train and a partially annotated corpus built from newspaper data (NKN-train) according to the criterion described in Section 3. For the partial annotation case, we set $\alpha = 0.005$.

The results are shown in Figure 4. When only 5k dependencies are annotated the full annotation method gives higher parsing accuracy, but as the number of annotations is increased the partial annotation method has higher accuracy. After 15k annotations have been performed, the full annotation method requires almost 5k additional annotations to match the accuracy of the partial annotation method.

Vol.2011-NL-201 No.13
Vol.2011-SLP-86 No.13
2011/5/17

5. Conclusion

We introduced an MST parser that evaluates the score for each edge in a dependency tree independently, which allows for the use of partially annotated corpora in training. When combined with a suitable strategy for selecting informative dependencies to annotate, training data can be prepared efficiently even in situations where data is limited, such as domain adaptation.

We evaluated state-of-the-art dependency parsers on a Japanese dependency parsing task, and found that our parser achieves accuracy comparable to that of a traditional MST parser. Additionally, the training and parsing speed of our parser is much faster than the traditional one, which allows it to be used for active learning in a real-world domain adaptation situation.

References

1) Yamada, K. and Knight, K.: A Syntax-Based Statistical Machine Translation Model, *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pp.523–530 (2001).
2) Miyao, Y., Sagae, K., Saetre, R., Matsuzaki, T. and Tsujii, J.: Evaluating Contributions of Natural Language Parsers to Protein-Protein Interaction Extraction, *Bioinformatics*, Vol.25, No.3, pp.394–400 (2009).
3) Gildea, D.: Corpus Variation and Parser Performance, *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pp. 167–202 (2001).
4) Petrov, S., Chang, P.-C., Ringgaard, M. and Alshawi, H.: Uptraining for Accurate Deterministic Question Parsing, *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, Cambridge, MA, pp.705–713 (2010).
5) Tsuboi, Y., Kashima, H., Mori, S., Oda, H. and Matsumoto, Y.: Training Conditional Random Fields Using Incomplete Annotations, *Proceedings of the 22th International Conference on Computational Linguistics* (2008).
6) Neubig, G. and Mori, S.: Word-based Partial Annotation for Efficient Corpus Construction, *Proceedings of the Seventh International Conference on Language Resources and Evaluation* (2010).
7) McDonald, R., Pereira, F., Ribarov, K. and Hajič, J.: Non-projective Dependency Parsing Using Spanning Tree Algorithms, *Proceedings of the 2005 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp.523–530 (2005).
8) Nivre, J. and Scholz, M.: Deterministic Dependency Parsing of English Text, *Proceedings of the 20th International Conference on Computational Linguistics*, Association for Computational Linguistics, p.64 (2004).
9) Buchholz, S. and Marsi, E.: CoNLL-X Shared Task on Multilingual Dependency Parsing, *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, Association for Computational Linguistics, pp.149–164 (2006).
10) Chen, S.F. and Goodman, J.: An Empirical Study of Smoothing Techniques for Language Modeling, Technical Report Tr-10-98, Center for Research in Computing Technology, Harvard University, Cambridge, Massachusetts (1998).
11) Roark, B. and Bacchiani, M.: Supervised and Unsupervised PCFG Adaptation to Novel Domains, *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pp.126–133 (2003).
12) McClosky, D., Charniak, E. and Johnson, M.: Reranking and Self-training for Parser Adaptation, *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics*, pp.337–344 (2006).
13) Blitzer, J., McDonald, R. and Pereira, F.: Domain Adaptation with Structural Correspondence Learning, *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pp.120–128 (2006).
14) Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S. and Yuret, D.: The CoNLL 2007 Shared Task on Dependency Parsing, *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, Association for Computational Linguistics (2007).
15) Suzuki, J., Isozaki, H., Carreras, X. and Collins, M.: An Empirical Study of Semi-supervised Structured Conditional Models for Dependency Parsing, *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pp. 551–560 (2009).
16) Tang, M., Luo, X. and Roukos, S.: Active Learning for Statistical Natural Language Parsing, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp.120–127 (2002).
17) Osborne, M. and Baldridge, J.: Ensemble-based Active Learning for Parse Selection, *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 89–96 (2004).
18) Sassano, M. and Kurohashi, S.: Using Smaller Constituents Rather Than Sentences in Active Learning for Japanese Dependency Parsing, *Proceedings of the*

Vol.2011-NL-201 No.13
Vol.2011-SLP-86 No.13
2011/5/17

48th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, pp.356–365 (2010).

19) Keene, D., Hatori, H., Yamada, H. and Irabu, S.: *Japanese-English Sentence Equivalents (in Japanese)*, Asahi Press, Electronic book edition (1992).

20) Neubig, G., Nakata, Y. and Mori, S.: Pointwise Prediction for Robust, Adaptable Japanese Morphological Analysis, *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT) Short Paper Track* (2011). (To appear).

21) Maekawa, K.: Balanced Corpus of Contemporary Written Japanese, *Proceedings of the 6th Workshop on Asian Language Resources*, pp.101–102 (2008).

22) Nivre, J., Hall, J. and Nilsson, J.: MaltParser: A Data-driven Parser-generator for Dependency Parsing, *Proceedings of the Fifth International Conference on Language Resources and Evaluation* (2006).

23) Chan, Y.S. and Ng, H.T.: Domain Adaptation with Active Learning for Word Sense Disambiguation, *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, Prague, Czech Republic, Association for Computational Linguistics, pp.49–56 (2007).

24) Rai, P., Saha, A., Daumé III, H. and Venkatasubramanian, S.: Domain Adaptation Meets Active Learning, *Workshop on Active Learning for Natural Language Processing (ALNLP-10)*, Association for Computational Linguistics, pp.27–32 (2010).