Accelerated Estimation of Conditional Random Fields using a Pseudo-Likelihood-inspired Perceptron Variant

Teemu Ruokolainen[a] **Miikka Silfverberg**[b] **Mikko Kurimo**[a] **Krister Lindén**[b]
[a] Department of Signal Processing and Acoustics, Aalto University, firstname.lastname@aalto.fi
[b] Department of Modern Languages, University of Helsinki, firstname.lastname@helsinki.fi

Abstract

We discuss a simple estimation approach for conditional random fields (CRFs). The approach is derived heuristically by defining a variant of the classic perceptron algorithm in spirit of pseudo-likelihood for maximum likelihood estimation. The resulting approximative algorithm has a linear time complexity in the size of the label set and contains a minimal amount of tunable hyper-parameters. Consequently, the algorithm is suitable for learning CRF-based part-of-speech (POS) taggers in presence of large POS label sets. We present experiments on five languages. Despite its heuristic nature, the algorithm provides surprisingly competetive accuracies and running times against reference methods.

1 Introduction

The conditional random field (CRF) model (Lafferty et al., 2001) has been successfully applied to several sequence labeling tasks in natural language processing, including part-of-speech (POS) tagging. In this work, we discuss accelerating the CRF model estimation in presence of a large number of labels, say, hundreds or thousands. Large label sets occur in POS tagging of morphologically rich languages (Erjavec, 2010; Haverinen et al., 2013).

CRF training is most commonly associated with the (conditional) maximum likelihood (ML) criterion employed in the original work of Lafferty et al. (2001). In this work, we focus on an alternative training approach using the averaged perceptron algorithm of Collins (2002). While yielding competitive accuracy (Collins, 2002; Zhang and Clark, 2011), the perceptron algorithm avoids extensive tuning of hyper-parameters and regularization required by the stochastic gradient descent algorithm employed in ML estimation (Vishwanathan et al., 2006). Additionally, while ML and perceptron training share an identical time complexity, the perceptron is in practice faster due to sparser parameter updates.

Despite its simplicity, running the perceptron algorithm can be tedious in case the data contains a large number of labels. Previously, this problem has been addressed using, for example, k-best beam search (Collins and Roark, 2004; Zhang and Clark, 2011; Huang et al., 2012) and parallelization (McDonald et al., 2010). In this work, we explore an alternative strategy, in which we modify the perceptron algorithm in spirit of the classic *pseudo-likelihood* approximation for ML estimation (Besag, 1975). The resulting novel algorithm has linear complexity w.r.t. the label set size and contains only a single hyper-parameter, namely, the number of passes taken over the training data set.

We evaluate the algorithm, referred to as the *pseudo-perceptron*, empirically in POS tagging on five languages. The results suggest that the approach can yield competitive accuracy compared to perceptron training accelerated using a violation-fixed 1-best beam search (Collins and Roark, 2004; Huang et al., 2012) which also provides a linear time complexity in label set size.

The rest of the paper is as follows. In Section 2, we describe the pseudo-perceptron algorithm and discuss related work. In Sections 3 and 4, we describe our experiment setup and the results, respectively. Conclusions on the work are presented in Section 5.

2 Methods

2.1 Pseudo-Perceptron Algorithm

The (unnormalized) CRF model for input and output sequences $x = (x_1, x_2, \ldots, x_{|x|})$ and

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 74–78,
Gothenburg, Sweden, April 26-30 2014.

$y = (y_1, y_2, \ldots, y_{|x|})$, respectively, is written as

$$p\left(y \mid x; \boldsymbol{w}\right) \propto \exp\Big(\boldsymbol{w} \cdot \boldsymbol{\Phi}(y, x)\Big) = \prod_{i=n}^{|x|} \exp\Big(\boldsymbol{w} \cdot \boldsymbol{\phi}(y_{i-n}, \ldots, y_i, x, i)\Big)\,, \tag{1}$$

where $\boldsymbol{w}$ denotes the model parameter vector, $\boldsymbol{\Phi}$ the vector-valued global feature extracting function, $\boldsymbol{\phi}$ the vector-valued local feature extracting function, and n the model order. We denote the tag set as $\mathcal{Y}$. The model parameters $\boldsymbol{w}$ are estimated based on training data, and test instances are decoded using the Viterbi search (Lafferty et al., 2001).

Given the model definition (1), the parameters $\boldsymbol{w}$ can be estimated in a straightforward manner using the structured perceptron algorithm (Collins, 2002). The algorithm iterates over the training set a single instance (x, y) at a time and updates the parameters according to the rule $\boldsymbol{w}^{(i)} = \boldsymbol{w}^{(i-1)} + \Delta\boldsymbol{\Phi}(x, y, z)$, where $\Delta\boldsymbol{\Phi}(x, y, z)$ for the ith iteration is written as $\Delta\boldsymbol{\Phi}(x, y, z) = \boldsymbol{\Phi}(x, y) - \boldsymbol{\Phi}(x, z)$. The prediction z is obtained as

$$z = \arg\max_{u \in \mathcal{Y}(x)} \boldsymbol{w} \cdot \boldsymbol{\Phi}(x, u) \tag{2}$$

by performing the Viterbi search over $\mathcal{Y}(x) = \mathcal{Y} \times \cdots \times \mathcal{Y}$, a product of $|x|$ copies of $\mathcal{Y}$. In case the perceptron algorithm yields a small number of incorrect predictions on the training data set, the parameters generalize well to test instances with a high probability (Collins, 2002).

The time complexity of the Viterbi search is $O(|x| \times |\mathcal{Y}|^{n+1})$. Consequently, running the perceptron algorithm can become tedious if the label set cardinality $|\mathcal{Y}|$ and/or the model order n is large. In order to speed up learning, we define a variant of the algorithm in the spirit of pseudo-likelihood (PL) learning (Besag, 1975). In analogy to PL, the key idea of the pseudo-perceptron (PP) algorithm is to obtain the required predictions over single variables y_i while fixing the remaining variables to their true values. In other words, instead of using the Viterbi search to find the z as in (2), we find a z' for each position $i \in 1..|x|$ as

$$z' = \arg\max_{u \in \mathcal{Y}'_i(x)} \boldsymbol{w} \cdot \boldsymbol{\Phi}(x, u)\,, \tag{3}$$

with $\mathcal{Y}'_i(x) = \{y_1\} \times \cdots \times \{y_{i-1}\} \times \mathcal{Y} \times \{y_{i+1}\} \times \cdots \times \{y_{|x|}\}$. Subsequent to training, test instances are decoded in a standard manner using the Viterbi search.

The appeal of PP is that the time complexity of search is reduced to $O(|x| \times |\mathcal{Y}|)$, i.e., linear in the number of labels in the label set. On the other hand, we no longer expect the obtained parameters to necessarily generalize well to test instances.[1] Consequently, we consider PP a heuristic estimation approach motivated by the rather well-established success of PL (Korč and Förstner, 2008; Sutton and McCallum, 2009).[2]

Next, we study yet another heuristic pseudo-variant of the perceptron algorithm referred to as the *piecewise-pseudo-perceptron* (PW-PP). This algorithm is analogous to the piecewise-pseudo-likelihood (PW-PL) approximation presented by Sutton and McCallum (2009). In this variant, the original graph is first split into smaller, possibly overlapping subgraphs (pieces). Subsequently, we apply the PP approximation to the pieces. We employ the approach coined *factor-as-piece* by Sutton and McCallum (2009), in which each piece contains $n + 1$ consecutive variables, where n is the CRF model order.

The PW-PP approach is motivated by the results of Sutton and McCallum (2009) who found PW-PL to increase stability w.r.t. accuracy compared to plain PL across tasks. Note that the piecewise approximation in itself is not interesting in chain-structured CRFs, as it results in same time complexity as standard estimation. Meanwhile, the PW-PP algorithm has same time complexity as PP.

2.2 Related work

Previously, impractical running times of perceptron learning have been addressed most notably using the k-best beam search method (Collins and Roark, 2004; Zhang and Clark, 2011; Huang et al., 2012). Here, we consider the "greedy" 1-best beam search variant most relevant as it shares the time complexity of the pseudo search. Therefore, in the experimental section of this work, we compare the PP and 1-best beam search.

We are aware of at least two other learning approaches inspired by PL, namely, the pseudo-max and piecewise algorithms of Sontag et al. (2010) and Alahari et al. (2010), respectively. Compared to these approaches, the PP algorithm provides a simpler estimation tool as it avoids the

[1] We leave formal treatment to future work.

[2] Meanwhile, note that pseudo-likelihood is a consistent estimator (Gidas, 1988; Hyvärinen, 2006).

hyper-parameters involved in the stochastic gradient descent algorithms as well as the regularization and margin functions inherent to the approaches of Alahari et al. (2010) and Sontag et al. (2010). On the other hand, Sontag et al. (2010) show that the pseudo-max approach achieves consistency given certain assumptions on the data generating function. Meanwhile, as discussed in previous section, we consider PP a heuristic and do not provide any generalization guarantees. To our understanding, Alahari et al. (2010) do not provide generalization guarantees for their algorithm.

3 Experimental Setup

3.1 Data

For a quick overview of the data sets, see Table 1.

Penn Treebank. The first data set we consider is the classic Penn Treebank. The complete treebank is divided into 25 sections of newswire text extracted from the Wall Street Journal. We split the data into training, development, and test sets using the sections 0-18, 19-21, and 22-24, according to the standardly applied division introduced by Collins (2002).

Multext-East. The second data we consider is the multilingual Multext-East (Erjavec, 2010) corpus. The corpus contains the novel *1984* by George Orwell. From the available seven languages, we utilize the Czech, Estonian and Romanian sections. Since the data does not have a standard division to training and test sets, we assign the 9th and 10th from each 10 consecutive sentences to the development and test sets, respectively. The remaining sentences are assigned to the training sets.

Turku Dependency Treebank. The third data we consider is the Finnish Turku Dependency Treebank (Haverinen et al., 2013). The treebank contains text from 10 different domains. We use the same data split strategy as for Multext East.

3.2 Reference Methods

We compare the PP and PW-PP algorithms with perceptron learning accelerated using 1-best beam search modified using the early update rule (Huang et al., 2012). While Huang et al. (2012) experimented with several violation-fixing methods (early, latest, maximum, hybrid), they appeared to reach termination at the same rate in POS tagging. Our preliminary experiments using the latest violation updates supported this. Consequently, we employ the early updates.

lang.	train.	dev.	test	tags	train. tags
eng	38,219	5,527	5,462	45	45
rom	5,216	652	652	405	391
est	5,183	648	647	413	408
cze	5,402	675	675	955	908
fin	5,043	630	630	2,355	2,141

Table 1: Overview on data. The training (train.), development (dev.) and test set sizes are given in sentences. The columns titled *tags* and *train. tags* correspond to total number of tags in the data set and number of tags in the training set, respectively.

We also provide results using the CRFsuite toolkit (Okazaki, 2007), which implements a 1st-order CRF model. To best of our knowledge, CRFsuite is currently the fastest freely available CRF implementation.[3] In addition to the averaged perceptron algorithm (Collins, 2002), the toolkit implements several training procedures (Nocedal, 1980; Crammer et al., 2006; Andrew and Gao, 2007; Mejer and Crammer, 2010; Shalev-Shwartz et al., 2011). We run CRFsuite using these algorithms employing their default parameters and the feature extraction scheme and stopping criterion described in Section 3.3. We then report results provided by the most accurate algorithm on each language.

3.3 Details on CRF Training and Decoding

While the methods discussed in this work are applicable for nth-order CRFs, we employ 1st-order CRFs in order to avoid overfitting the relatively small training sets.

We employ a simple feature set including word forms at position $t-2, \ldots, t+2$, suffixes of word at position t up to four letters, and three orthographic features indicating if the word at position t contains a hyphen, capital letter, or a digit.

All the perceptron variants (PP, PW-PP, 1-best beam search) initialize the model parameters with zero vectors and process the training instances in the order they appear in the corpus. At the end of each pass, we apply the CRFs using the latest averaged parameters (Collins, 2002) to the development set. We assume the algorithms have converged when the model accuracy on development

[3] See benchmark results at `http://www.chokkan.org/software/crfsuite/benchmark.html`

has not increased during last three iterations. After termination, we apply the averaged parameters yielding highest performance on the development set to test instances.

Test and development instances are decoded using a combination of Viterbi search and the *tag dictionary* approach of Ratnaparkhi (1996). In this approach, candidate tags for known word forms are limited to those observed in the training data. Meanwhile, word forms that were unseen during training consider the full label set.

3.4 Software and Hardware

The experiments are run on a standard desktop computer. We use our own C++-based implementation of the methods discussed in Section 2.

4 Results

The obtained training times and test set accuracies (measured using accuracy and out-of-vocabulary (OOV) accuracy) are presented in Table 2. The training CPU times include the time (in minutes) consumed by running the perceptron algorithm variants as well as evaluation of the development set accuracy. The column labeled *it.* corresponds to the number of passes over training set made by the algorithms before termination.

We summarize the results as follows. First, PW-PP provided higher accuracies compared to PP on Romanian, Czech, and Finnish. The differences were statistically significant[4] on Czech. Second, while yielding similar running times compared to 1-best beam search, PW-PP provided higher accuracies on all languages apart from Finnish. The differences were significant on Estonian and Czech. Third, while fastest on the Penn Treebank, the CRFsuite toolkit became substantially slower compared to PW-PP when the number of labels were increased (see Czech and Finnish). The differences in accuracies between the best performing CRFsuite algorithm and PP and PW-PP were significant on Czech.

method	it.	time (min)	acc.	OOV
English				
PP	**9**	6	96.99	87.97
PW-PP	10	7	96.98	88.11
1-best beam	17	8	96.91	88.33
Pas.-Agg.	**9**	**1**	**97.01**	**88.68**
Romanian				
PP	9	8	96.81	83.66
PW-PP	**8**	**7**	96.91	84.38
1-best beam	17	10	96.88	**85.32**
Pas.-Agg.	13	9	**97.06**	84.69
Estonian				
PP	10	8	**93.39**	78.10
PW-PP	**8**	**6**	93.35	**78.66**
1-best beam	23	15	92.95	75.65
Pas.-Agg.	15	12	93.27	77.63
Czech				
PP	**11**	26	89.37	70.67
PW-PP	16	41	89.84	72.52
1-best beam	14	**19**	88.95	70.90
Pegasos	15	341	**90.42**	**72.59**
Finnish				
PP	**11**	58	87.09	58.58
PW-PP	**11**	**56**	87.16	58.50
1-best beam	21	94	**87.38**	**59.29**
Pas.-Agg.	16	693	87.17	57.58

Table 2: Results. We report CRFsuite results provided by most accurate algorithm on each language: the *Pas.-Agg.* and *Pegasos* refer to the algorithms of Crammer et al. (2006) and Shalev-Shwartz et al. (2011), respectively.

5 Conclusions

We presented a heuristic perceptron variant for estimation of CRFs in the spirit of the classic pseudo-likelihood estimator. The resulting approximative algorithm has a linear time complexity in the label set cardinality and contains only a single hyper-parameter, namely, the number of passes taken over the training data set. We evaluated the algorithm in POS tagging on five languages. Despite its heuristic nature, the algorithm provided competetive accuracies and running times against reference methods.

Acknowledgements

This work was financially supported by Langnet (Finnish doctoral programme in language studies) and the Academy of Finland under the grant no 251170 (Finnish Centre of Excellence Program (2012-2017)). We would like to thank Dr. Onur Dikmen for the helpful discussions during the work.

[4]We establish significance (with confidence level 0.95) using the standard 1-sided Wilcoxon signed-rank test performed on 10 randomly divided, non-overlapping subsets of the complete test sets.

References

Karteek Alahari, Chris Russell, and Philip H.S. Torr. 2010. Efficient piecewise learning for conditional random fields. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 895–901.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L_1-regularized log-linear models. In *Proceedings of the 24th international conference on Machine learning*, pages 33–40.

Julian Besag. 1975. Statistical analysis of non-lattice data. *The statistician*, pages 179–195.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 111.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, volume 10, pages 1–8.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *The Journal of Machine Learning Research*, 7:551–585.

Tomaž Erjavec. 2010. Multext-east version 4: Multilingual morphosyntactic specifications, lexicons and corpora. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

Basilis Gidas. 1988. Consistency of maximum likelihood and pseudo-likelihood estimators for Gibbs distributions. In *Stochastic differential systems, stochastic control theory and applications*, pages 129–145.

Katri Haverinen, Jenna Nyblom, Timo Viljanen, Veronika Laippala, Samuel Kohonen, Anna Missilä, Stina Ojala, Tapio Salakoski, and Filip Ginter. 2013. Building the essential resources for Finnish: the Turku Dependency Treebank. *Language Resources and Evaluation*.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–151.

Aapo Hyvärinen. 2006. Consistency of pseudolikelihood estimation of fully visible Boltzmann machines. *Neural Computation*, 18(10):2283–2292.

Filip Korč and Wolfgang Förstner. 2008. Approximate parameter learning in conditional random fields: An empirical investigation. *Pattern Recognition*, pages 11–20.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

Ryan McDonald, Keith Hall, and Gideon Mann. 2010. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464.

Avihai Mejer and Koby Crammer. 2010. Confidence in structured-prediction using confidence-weighted models. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 971–981.

Jorge Nocedal. 1980. Updating quasi-Newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782.

Naoaki Okazaki. 2007. CRFsuite: a fast implementation of conditional random fields (CRFs). *URL http://www.chokkan.org/software/crfsuite*.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, volume 1, pages 133–142.

Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. 2011. Pegasos: Primal estimated sub-gradient solver for SVM. *Mathematical Programming*, 127(1):3–30.

David Sontag, Ofer Meshi, Tommi Jaakkola, and Amir Globerson. 2010. More data means less inference: A pseudo-max approach to structured learning. In *Advances in Neural Information Processing Systems 23*, pages 2181–2189.

Charles Sutton and Andrew McCallum. 2009. Piecewise training for structured prediction. *Machine learning*, 77(2):165–194.

S.V.N. Vishwanathan, Nicol Schraudolph, Mark Schmidt, and Kevin Murphy. 2006. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the 23rd international conference on Machine learning*, pages 969–976.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1):105–151.