INVESTIGATION OF ENSEMBLE MODELS FOR SEQUENCE LEARNING

Asli Celikyilmaz and Dilek Hakkani-Tur

Microsoft

ABSTRACT

While ensemble models have proven useful for sequence learning tasks there is relatively fewer work that provide insights into what makes them powerful. In this paper, we investigate the empirical behavior of the ensemble approaches on sequence modeling, specifically for the semantic tagging task. We explore this by comparing the performance of commonly used and easy to implement ensemble methods such as majority voting, linear combination and stacking to a learning based and rather complex ensemble method. Next, we ask the question: when models of different learning methods such as predictive and representation learning (e.g., deep learning) are aggregated, do we get performance gains over the individual baseline models. We explore these questions on a range of datasets on syntactic and semantic tagging tasks such as slot filling. Our findings show that a ranking based ensemble model outperforms all other well-known ensemble models.

Index Terms— ensemble learning, conditional random fields, slot tagging, spoken language understanding

1. INTRODUCTION

Ensemble learning typically refers to combining a collection of diverse and accurate models into a single one, which is more powerful than its base models. While ensemble learning has been successfully employed to improve sequence learning models for many speech and language processing tasks, less attention has been paid to laying out the characteristics of a reasonably good performing ensemble model for sequence tagging. In this paper, we provide insights into *the complexity of the ensemble learning methods* that relatively few papers have investigated for sequence learning tasks, but can affect their performance. Various research has shown that ensemble approaches based on scoring [1], linear combination [2] and stacking [3, 4] perform well over individual models. On the other hand, a recent study in information retrieval [5] has shown superior performance using an arbitrary structure Conditional Random Fields (CRF) method (which is different than a linear-chain CRF [6]) to aggregate the predicted rankings of the base models. In this paper, we investigate whether a more complex and smart ensemble learning method such as the one in [5] is more beneficial than commonly used easy to implement and simpler methods. Because [5]'s approach is not specifically designed for sequence learning tasks, we present a new approach to tailor it for sequence models. In experiments we show up to 2% relative improvement in F-score in semantic tagging and up to 1% in syntactic tagging compared to the best performing voting, scoring and stacking baseline embedding models.

Among several popular ensemble methods are "bagged" ensembles [7], boosting [8, 9], random forests [10], etc. However, a broader term of *multiple classifier systems* (referred as ensemble learning in NLP research), covers multiple hypothesis that are not induced by the same base learner. The majority voting, linear combination and stacking are examples of such ensemble approaches. In this paper, we focus only on the latter ensembles and use a variety of learning algorithms to build the base models. We first summarize the earlier CRF based ensemble method and provide details of our approach for tailoring it for sequence tagging. In the experiments, we empirically investigate the ensemble models from different aspects and finally draw conclusions.

2. SUPERVISED SEQUENCE LEARNING

Research on sequence learning can be categorized into two: the *predictive* and *representation learning*. It is nearly standard to stage sequence learning tasks of NLP as a *predictive learning* (PL) problem, where we are interested in predicting some aspect of a given observed data. We model the conditional distribution $p(t|w)$ of a tag sequence t given the input word sequence w. On the other hand, *representation learning* (RL) (also known as *feature learning*) is based on single to multiple levels of representation of the observed data to extract useful information when later building classifiers or sequence learners [11]. Deep learning is the most common RL method, which is formed by the composition of multiple non-linear transformations of the data, with the goal of yielding more abstract - and more useful - representations [11, 12].

Evidence shows that the RL methods are consistently better than the PL counterparts on several sequence learning tasks including syntactic POS tagging, NER recognition task, chunking [13], semantic parsing [14], slot filling [15, 16], etc. However, a recent benchmark for comparing the strengths of PL and RL methods have shown that RL methods, such as deep learning, is effective with low dimensional continuous features, whereas not as much as the PL counterparts with high dimensional discrete features [17]. In this work, we are asking: would aggregating the predictions from PL and RL-based base sequence models with an ensemble approach perform even better ? We start with two different *predictive learning* methods to train the base models:

(1) CRF++: A linear chain CRF that captures the linear relation between input and output sequences and uses quasi Newton Methods for optimization [6, 18].

(2) CRFSGD: A linear chain CRF that uses stochastic gradient descent, an on-line learning algorithm [19] and known for its fast convergence [20]. We use L_2 regularizer, $R(\theta)=\frac{\lambda}{2}||\theta||_2^2$, which can be numerically optimized.

We also use two different neural network based *representation learning* methods:

(3) CNF: Conditional Neural Fields extends linear chain CRF's by a single layer of hidden units between input and output layers to model the non-linear relationship between them as well as learn a representation from observed data [21]. It was shown that the CNF's outperform CRF methods on handwriting recognition and gene sequence learning tasks.

(4) RNNSEQ: Recurrent Neural Network (RNN) for sequence learning represents utterances as observed input node sequences connected to multiple layers of hidden nodes, a fully connected set of recurrent connections amongst the hidden nodes, and a set of output nodes, which are the target labels. [16] shows that RNNSEQ significantly improves the performance over linear CRFs in semantic slot filling task.

3. ENSEMBLE LEARNING APPROACHES

Ensemble models combine a collection of baseline models into a single one. Here, we provide background on most common ensemble methods used in the experiments.

Scoring Based Ensembles: The *voting schema* is the most commonly used scoring method to combine a collection of diverse and accurate models into a more powerful one. It assigns scores to candidate sequences produced by base models based on the number of votes it receives from each model. *Linear combination* is another commonly used scoring based approach, where K base model predictions are combined as features into a secondary regression or log-linear model.

Stacking Based Ensembles: They lie somewhere between scoring and learning algorithms [22]. Typically, the estimations from base models at level-0 are augmented as the base features of a level-1 model, which is considered the ensemble model. Level-1 ensemble model is another learner, which learns the errors of the base learners and corrects them.

Learning Based Ensembles: Because the majority preference may often be wrong, aggregation methods that aim to satisfy the majority [1] may lead to suboptimal results. In a recent work, [5] presents a supervised arbitrary structured CRF based ensemble approach that learns to aggregate the rankings of documents obtained from different search engines and show superior performance over other ensemble models. We adapt the CRF-based ensemble method for our task below.

$\mathbf{s}^{(i)}$	$\rightarrow$	"*time*	*flies*	*like*	*an*	*arrow*"		
		$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$		
Truth $\mathbf{t}^{(i)}\rightarrow$		NN	VBZ	IN	DT	NN	$p_n^{(i)}$	Accuracy
$M(1)$	$\hat{\mathbf{t}}_1$	NN	VBZ	IN	DT	NN	0.98	1.00
	$\hat{\mathbf{t}}_2$	NN	NN	IN	DT	NN	0.87	0.80
	$\hat{\mathbf{t}}_3$	NN	JJ	IN	DT	NN	0.65	0.80
$M(2)$	$\hat{\mathbf{t}}_1$	NN	VBZ	IN	DT	NN	0.98	1.00
	$\hat{\mathbf{t}}_2$	NN	NN	IN	DT	NN	0.75	0.80
	$\hat{\mathbf{t}}_4$	NN	NN	VBZ	DT	NN	0.46	0.60

Table 1. N-best tag sequences of sentence $\mathbf{s}^{(i)}$ obtained from each model $M(k)$. $p_n^{(i)}$ is the posterior probability of the *n*th sequence produced by each model. Accuracy shows how similar is the generated tag sequence $\hat{t}_n$ to the truth $\mathbf{t}^{(i)}$.

4. CRF BASED ENSEMBLE METHOD

Preference aggregation is the task of learning to aggregate the rankings of each document returned by different search engines based on a user query to generate a more comprehensive ranking result [15]. The supervised CRF-based preference aggregation method (CRF-ESB) [5] is designed for this task. On top of the document ranks, it uses categorical valued document-query relevance labels from annotators as supervision at training time. Similarly, our task is to rank the n-best sequences produced by base models, so we can pose our ensemble model as a posterior aggregation task similar to CRF-ESB. But, we have the *N*-best tag sequence posteriors instead of rankings, and do not have the sequence relevance labels. We present below our method for tailoring the CRF-ESB for sequence learning task. We map the posteriors into rankings and obtain the accuracy of the n-best sequences, which are used as the relevance labels of the *N*-best tag sequences.

Data: Let $\mathcal{D}$=$\{\mathcal{R}^{(i)}$,$\mathbf{y}^{(i)}\}_{i=1}^{|D|}$ represent $|D|$ sentences in dev. data. Below we explain how we construct $\mathcal{D}$ using POS task as shown in Table 1:

The $\mathbf{t}^{(i)}$ is the ground truth tag sequence of the *i*th sentence $\mathbf{s}^{(i)}$ in dev. data. Using each base model $M(k)$, we decode *N*-best (*n*=1...*N*) tag sequences $\hat{\mathbf{t}}_n^{(i)}$ of each sentence and obtain sentence level posteriors $p_n^{(i)}(\hat{\mathbf{t}}_n^{(i)}|\mathbf{s}^{(i)},k)$, later to construct $N\times K$ score matrix $\mathcal{P}^{(i)}$, where each cell $\mathcal{P}^{(i)}(n,k)$=$p_n(\hat{\mathbf{t}}_n^{(i)}|\mathbf{s}^{(i)},k)$ is the *n*th sequence tag posterior from *k*th base model. (A combination of the N-best sequences from base models reveals more sequences than N, but we only take the top N total of sequences). The rank matrix $\mathcal{R}^{(i)}$ is the ranked order of *N*-best decoded sequences from each model. Thus, we derive the ranking of each generated sequence $\mathcal{R}^{(i)}(n,k)$ directly from score matrix by sorting the scores $\{\mathcal{P}^{(i)}(n,k)\}_{n=1...N}$ of each base model that generated *N* sequences and map to a rank. Because not all the models generate the same *N*-best tag-sequences, we set posterior scores $\mathcal{P}^{(i)}(n,k)$=0, so as rank-scores $\mathcal{R}^{(i)}(n,k)$=0, when model *k* does not predict a particular tag-sequence.

The $\mathbf{y}^{(i)}$ in $\mathcal{D}$ are the relevance values characterizing how relevant the generated tag sequence is to the ground truth. Similar to categorical valued document-query relevance labels in [5], we construct relevance labels $\mathbf{y}_n^{(i)}$ for each *n*th predicted sequence as follows:

$$\mathbf{y}_n^{(i)} = \begin{cases} 2 & \text{if } Acc(\mathbf{t}^{(i)}, \hat{\mathbf{t}}_n^{(i)}) = 1.0 \\ 1 & \text{if } Acc(\mathbf{t}^{(i)}, \hat{\mathbf{t}}_n^{(i)}) > \bar{A} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

To set the relevance values to predicted tag sequences, we use accuracy (*Acc*) which is taken as the ratio of the correctly predicted tags to all tags in sequence (see Table 1). Specifically, a relevance value of '2' indicates that the predicted tag sequence $\hat{\mathbf{t}}_n^{(i)}$ matches the true tag sequence $\mathbf{t}^{(i)}$, whereas a value of '1' indicates that for some tokens in the sequence, the predicted tags do not match the true tags. A threshold $\bar{A}$ sets the confidence of accepting a predicted tag sequence, and in the experiments we learn its value by grid search.

CRF Ensemble Learning Method: Our goal is to use pairwise preferences from training examples for predicting a ranking for all possible sequences for a new test example. Now that we converted the sequence posteriors into preference rank matrices that the CRF-ESB can use as training data $\mathcal{D}$=$\{\mathcal{R}^{(i)}$,$\mathbf{y}^{(i)}\}_{i=1}^{|D|}$, we are ready to learn a mapping from the constructed rank matrix $\mathcal{R}^{(i)}$ to the acceptance values $\mathbf{y}^{(i)}$. To do that, the CRF-ESB defines a conditional distribution $p(\mathbf{y}|\mathcal{R})$ through an energy $E(\mathbf{y}, \mathcal{R}; \beta)$:

$$p(\mathbf{y}|\mathcal{R}) = 1/Z(\mathcal{R}) \exp\left(-E(\mathbf{y}, \mathcal{R}; \beta)\right) \quad (2)$$

and optimizes it for the target metric between predicted ranks $\hat{\mathbf{y}}_n^{(i)}$ and the truth $\mathbf{y}_n^{(i)}$. The partition function $Z(\mathcal{R})$ sums over $M_k!$ valid rankings of $\mathbf{y}$. We try to learn the model parameters β, that minimize the average training loss $\frac{1}{|D|}\sum_i^{|D|}\mathcal{L}(\mathbf{y}_i,\hat{\mathbf{y}}_i)$[1]. One of the characteristics of the CRF-ESB method is to consider disagreements between the $\hat{\mathbf{y}}_n^{(i)}$ and $\mathbf{y}_n^{(i)}$. Thus, we derive unary $\varphi_k(j)$ and pairwise potentials $\phi_k(j,l)$ from the rank matrix and use these potentials to define a smooth energy function over the rankings. The unary potential for a sequence *j* is defined as, $\varphi_k(j) = I[\mathcal{R}^{(i)}(j,k) = 0]$, where $I[\cdot]$ is indicator function that is turned on when the potential is active only when sequence *j* is not ranked by the model *k*. Given the $N\times K$ ranking matrix $\mathcal{R}$, we convert it into K $N\times N$ pairwise potentials $\phi_k(j,l)$, to emphasize the importance of the relative position of each candidate sequence. We use the log-rank difference function to define pairwise potentials, which was identified as the most effective function in [5]:

$$\phi_k(j,l) = I[\mathcal{R}^{(i)}(j,k) < \mathcal{R}^{(i)}(l,k)] \cdot LR_k(j,l) \quad (3)$$

$\phi_k(j,l)$ provides the pairwise potential value between sequence *j* and *l* using the *k*th base model, where log-ratio $LR_k(j,l)$ is defined as:

$$LR_k(j,l) = \frac{\log(\mathcal{R}^{(i)}(l,k)) - \log(\mathcal{R}^{(i)}(j,k))}{\log(max(\mathcal{R}^{(i)}(l,k), \mathcal{R}^{(i)}(j,k)))} \quad (4)$$

[1]Please refer to the [5] for details of the learning and inference methods.

	Learning Methods	POS	NER	SLU
PL	1. CRF++	94.70	79.58	85.07
	2. CRFSGD	94.67	81.43	87.74
RL	3. CNF	95.06	78.78	**88.05**
	4. RNNSEQ	**95.17**	**81.79**	85.72
CRF Ensemble	5. CRF++, CRFSGD, CNF	95.75	81.95	87.67
	6. CRF++, CRFSGD, RNNSEQ	95.95	81.98	86.12
	7. CRFSGD, CNF, RNNSEQ	95.98	82.12	87.34
	8. CRF++, CNF, RNNSEQ	96.01	82.03	86.89
	9. All PL {CRF++, CRFSGD}	94.99	81.79	86.72
	10. All RL {CNF, RNNSEQ}	95.97	81.50	87.79
	11. All Base Models Combined	**96.45**	**82.81**	**88.41**

Table 2. The **F-scores** for the POS, NER and SLU models.

Non-zero entries in $\phi_k(j,l)$ represent the strength of the pairwise preference $\{\hat{t}_j \succ \hat{t}_l\}$ expressed by model $M(k)$.

The CRB-ESB has an arbitrary structure and has a structure as Preference Networks [23], which is different than the linear-chain CRF models since it is not used for sequence tagging . The main idea behind CRF-ESB approach is that the pairwise preferences and the rankings translate to pairwise potentials in a CRF model. The algorithm evaluates the compatibility of any ranking $\mathcal{R}^{(i)}(j,k)$ by comparing the order induced by the ranking with the relevance values $y^{(i)}$.

5. EXPERIMENTS AND DISCUSSION

We focus on three sequence learning tasks: syntactic POS tagging, semantic NER tagging and slot filling task for spoken language understanding (SLU). For POS tagging, we use the Wall Steet Journal (WSJ) section of Penn Treebank [24], sections 00-18 for training and dev. data and rest for testing. For NER we use the CoNLL-03 Shared Task [25] dataset, splitting training data into train and dev. sets, and 'testa' for testing. For SLU, we use a dataset of utterances from real-use scenarios of a spoken dialog system. The utterances are from domains of audiovisual media, including movies, music, games, tv shows. The user is expected to interact by voice with a spoken dialog system to perform a variety of tasks in relation to such media, including browsing, searching, etc.

The NER corpus has four output tags (person (PER), organization (ORG), location (LOC), and miscellaneous (MISC; broadly including events, artworks and nationalities)), whereas POS data has 45 part-of-speech tags. The media dataset has 26 semantic tags including movie-genre, release-date, description, actor, game-title, etc. Because we use the IOB (in-out-begin) format, any token with no tags gets an 'O' tag.

All methods in section 2 that we use to build our base models have publicly available code (see References for link to their code), except for the RNNSEQ, which we re-implemented based on [16]. For each model, we also implemented the forward-backward schema just to obtain the

	NNPS	VBG	RP	JJ	VBN	RBS	IN	VBD
🟥 RL	49.0%	20.0%	21.0%	13.4%	8.6%	6.6%	5.1%	3.9%
🟦 PL	51.5%	19.8%	15.4%	8.8%	7.9%	7.2%	6.6%	4.2%
🟩 ESB	**44.0%**	17.6%	12.9%	**7.9%**	6.9%	**2.6%**	5.7%	3.8%

Table 3. **Prediction errors** of three models, 🟥 RL: Representation Learning using RNNSEQ,🟦 PL: Predictive Learning using CRF++, 🟩 ESB: CRF Ensemble model aggregating RL and PL methods (CRF-ESB). A significant % decrease in error (per tag) based on paired t-test (p<0.01) is **bolded**.

Task	MV	LC	STK	CRF-ESB	Rel. Imp.
POS	95.30	95.23	95.34	**96.45**	+1.2%
NER	81.99	81.94	81.32	**82.81**	+1.0%
SLU	86.45	86.32	87.04	**88.41**	+1.6%

Table 4. The F-scores of CRF-based Ensemble Model (CRF-ESB), Majority Voting (MV), Linear Combination (LC), and Stacking (STK) on POS, NER and SLU tasks. Relative Improvement (Rel.Imp) is the % increase in F-score by CRF-ESB over the best performing scoring and stacking methods (wavy underline)

N-best tag sequence posteriors given a sentence. Each base model is trained using only the n-gram features with 5-gram window centered on the current position. Because the RL methods learn hidden (latent) features from observed data, they have more features than the PL methods. The RNNSEQ uses back-propagation, CRFSGD uses stochastic gradient descent, whereas CRF++ and CNF use L-BFGS for optimization. We compare four types of ensemble methods: majority voting, stacking, linear combination which use linear regression and CRF-ESB ensemble learner which use gradient based procedure for optimization.

Experiment 1: Predictive or Representation or Both? Our first goal is to explore the performance gain from the CRF-ESB model for the three sequence learning tasks. Table 2 shows the results of each base and several CRF-ESB models on POS, NER and SLU tasks.

As expected, in all tasks, we observe larger gains with the ensemble models when all the base models are aggregated (#11). Each ensemble model from #5 through #8 exclude one of the base models; so only three base models are aggregated at training time. Although there is a small difference between their F-scores, when the RNNSEQ is removed (#5), we observe the least performance in POS and NER. Same applies to SLU task that the ensemble without CNF (#6) yields the least performance. It suggests that RNNSEQ and CNF contribute most to the performance, considering RNNSEQ is the best performing base model (#4) for POS and NER and CNF is the best performing base model for SLU (#3). Combining all PL (#9) and all RL methods (#10) also does not yield a significant gain over base models even though the best base models are from RL. Albeit this fact, it is interesting that when all the models are combined in #11 we observe a significant improvement over base models (using paired t-test, p<0.01). This suggests that with the aggregation of each base model we learn a different aspect of the data corresponding to different tags. But, which aspects is CRF-ESB learning better ?

We investigate this fact on the POS task. We take the most frequent POS tags and select the most confusable ones to compare the CRF-ESB model results (#11) against the best PL (#1) and the best RL (#4) base models from Table 2. Note that the ESB (CRF-ESB) model aggregates all four models, whereas PL and RL are only the results from single base model. Although the CRF-ESB model does not directly optimize the token-tag level errors but rather optimizes the ranked order of the *N-best* sequences, its inference predicts the best tag sequence, which we compare against the best tag sequences from base models.

We show the results in Table 3. Not surprisingly, we see the same performance gain (error reduction) per tag that we saw in the overall evaluations in Table 2. The most significant error reductions are observed for the plural nouns (NNPS), adverbs (JJ) and adjectives (RBS). Earlier study shows that the errors between noun phrases (NNP/NN/NNPS/NNS) can be largely attributed to difficulties with unknown words [26]. One conclusion we can derive is that the ensemble model can recover unknown word errors. Another common class of errors of POS tagging models is the RB/RBS/RP/IN ambiguity of words like *up*, *out*, *on*, which require semantic intuition. It appears that the ensemble model learns to make accurate linguistic distinction between ambiguous words.

Experiment 2: Learn, Stack, or Vote for Ensembles? So far, we have provided insights into the effectiveness of a learning based ensemble method. We now provide benchmarks for comparing the performance of the majority voting (MV) schema, stacking (STK) and linear combination (LC) across POS, NER and SLU tasks.

The results as shown in Table 4 confirm our hypothesis that the majority voting as well as linear combination provide suboptimal results, and in some cases does not even improve over base models (e.g., POS, SLU). Although slightly better than voting models, stacking falls short against the CRF-ESB. On the other hand, the CRF-ESB model can pick the correct answer from the crowd even when the majority is incorrect. This is due to the fact that CRF-ESB algorithm learns patterns of the pair-wise rankings between each model, favoring the top ranked ones when base models don't agree.

6. CONCLUSION

We investigate ensemble learning for NLP sequence tagging tasks by aggregating different base sequence tagging models. The ensemble models select the most confident predictions of each base model and infer the most likely sequence outperforming the best base models. We empirically analyze the impact of using different learning methods as base taggers. For future work, we will inject the diversity of the base models as an additional feature during learning the ensemble models.

7. REFERENCES

[1] M. Surdeanu and C.D. Manning, "Ensemble models for dependency parsing: Cheap and good," *In Proc. of North American ACL (NAACL)*, 2010.

[2] G. Haffari, M. Razavi, and A. Sarkar, "An ensemble model that combines syntactic and semantic clustering for discriminative dependency parsing," *In Proc. of ACL*, 2011.

[3] J. Nivre and R. McDonald, "Integrating graph-based and transition-based dependency parsers.," *In Proc. ACL*, 2008.

[4] G. Attardi and F. Dell'Orletta, "Reverse revision and linear tree combination for dependency parsing," *In Proc. NAACL-HLT*, 2009.

[5] M. Volkovs and R. Zemel, "Supervised crf framework for preference aggregation.," *In Proc. of CIKM: International Conference on Information and Knowledge Management*, 2013.

[6] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random elds: probabilistic models for segmenting and labeling sequence data," *In Proc. of ICML*, 2001.

[7] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 826, pp. 123–140, 1996.

[8] Y. Freund and R.E. Schapire, "Experiments with a new boosting algorithm," *In Proc. Machine Learning: Proceedings of the Thirteenth International Conference*, pp. 148–156, 1996.

[9] J.H. Friedman, "Greedy function approximation: A gradient boosting machine," *In Proc. of Annals of Statistics*, vol. 29, pp. 1189–1232, 2001.

[10] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[11] A. Paccanaro and G. Hinton, "Learning distributed representations of concepts using linear relational embedding.," *In Proc. of KDE*, 2000.

[12] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *In Proc. of CoRR*, 2012.

[13] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *In Proc. of JMLR*, vol. 12, pp. 2461–2505, 2011.

[14] H. Poon and P. Domingos, "Natural language processing (almost) from scratch," *In Proc. of NIPS Workshop on Deep Learning and Unsupervised Feature Learning, Whistler, Canada*, 2010.

[15] L. Deng, G. Tur, X. He, and D. Hakkani-Tur, "Use of kernel deep convex networks and end-to-end learning for spoken language understanding," *In Proc. of the SLT 2012, IEEE Workshop on Spoken Language Technologies*, 2012.

[16] K. Yao, G. Zweig, M-Y Huang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding," *In Proc. of Interspeech 2013*, 2013.

[17] M. Wang and C. Manning, "Effect of non-linear deep architecture in sequence labeling," *In Proc. of IJCNLP (Short Paper)*, 2013.

[18] T. Kudo, "Crf++: Yet another crf toolkit," in *software:https://code.google.com/p/crfpp/downloads/list*, 2013.

[19] L. Bottou, "Crf stochastic gradient descent," in *leon.bottou.org/projects/sgd*, 2011.

[20] L. Bottou and O. Bousquet, "The tradeoffs of large scale learning," *In Proc. of the Advances in neural information processing systems*, 2008.

[21] L. Bo J. Peng and J. Xu, "Conditional neural fields," in *In Proc. of the 23rd NIPS 2009*, software:https://code.google.com/p/cnf/downloads/list, Ed., 2009.

[22] A.F.T. Martins, D. Das, N.A. Smith, and E.P. Xing, "Stacking dependency parsers," *In Proc. of EMNLP*, 2008.

[23] D.Q. Phung T.T. Truyen and S. Venkatesh, "Preference networks: Probabilistic models for recommendation systems," *In Proc. 6th Australasian Data Mining Conference (AusDM'07), Gold Coast, Australia*, 2007.

[24] M. P. Marcus, B. Santorini, and M.A. Marcinkiewicz, "Building a large annotated corpus of english: The penn treebank," *Computational Linguistics*, vol. 27, pp. 1–30, 1993.

[25] Erik F. T. K. Sang and Fien De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *In Proc. of CoNLL-2003*, Walter Daelemans and Miles Osborne, Eds. 2003, pp. 142–147, Edmonton, Canada.

[26] K. Toutanova and C. D. Manning, "Enriching the knowledge sources used in a maximum entropy part-of-speech tagger," *In Proc. of EMNLP 2000*, 2000.