UPGRADING CRFS TO JRFS AND ITS BENEFITS TO SEQUENCE MODELING AND LABELING

Yunfu Song[1], *Zhijian Ou*[1†], *Zitao Liu*[2], *Songfan Yang*[2]

[1]Speech Processing and Machine Intelligence (SPMI) Lab, Tsinghua University, Beijing, China.
[2]TAL AI Lab, Beijing, China.
songyf17@mails.tsinghua.edu.cn, ozj@tsinghua.edu.cn

ABSTRACT

Two important sequence tasks are sequence modeling and labeling. Sequence modeling involves determining the probabilities of sequences, e.g. language modeling. It is still difficult to improve language modeling with additional relevant tags, e.g. part-of-speech (POS) tags. For sequence labeling, it is worthwhile to explore task-dependent semi-supervised learning to leverage a mix of labeled and unlabeled data, besides pre-training. In this paper, we propose to upgrade condtional random fields (CRFs) and obtain a joint generative model of observation and label sequences, called joint random fields (JRFs). Specifically, we propose to use the potential function in the original CRF as the potential function that defines the joint distribution. This development from CRFs to JRFs benefits both modeling and labeling of sequence data, as shown in our experiments. For example, the JRF model (using POS tags) outperforms traditional language models and avoids the need to produce hypothesized labels by a standalone POS tagger. For sequence labeling, task-dependent semi-supervised learning by JRFs consistently outperform the CRF baseline and self-training, on POS tagging, chunking and NER.

Index Terms— conditional random fields, joint random fields, sequence modeling, sequence labeling

1. INTRODUCTION

Probabilistic generative modeling is a principled methodology that promisingly can learn from data of various forms (whether labeled or unlabeled) to benefit downstream tasks, which, however, is particularly challenging for sequence data. Two important sequence tasks are sequence modeling and labeling.

A basic problem of *sequence modeling* is to determine the probabilities of sequences, e.g. language modeling [1] which is a crucial component in speech recognition. For sequences of length l, $x^l \triangleq x_1, x_2, ..., x_l$, this amounts to calculate $p(l, x^l)$, where we make explicit the role of the length l. Ideally, this density modeling can be improved with additional relevant labels. e.g. incorporating part-of-speech (POS) tags for language modeling. There are some previous studies in [2, 3]. The difficulty is that the labels (e.g. POS) usually are not available in testing, so a standalone POS tagger is needed to provide hypothesized labels in testing.

The task of *sequence labeling* is, given observation sequence x^l, to predict the label sequence $y^l \triangleq y_1, y_2, ..., y_l$, with one label for one observation at each position. Sequence labeling has been widely applied in various tasks, e.g. POS labeling [4, 5], named entity recognition (NER) [6, 7, 8], and chunking [6, 9]. It is desirable for the labeling model to leverage both labeled data (namely pairs of x^l and y^l) and unlabeled data (namely x^l without labels). Pre-training has proved to be effective [4, 10], which, however, is task-independent followed by task-dependent fine-tuning. Besides pre-training, it is worthwhile to explore task-dependent semi-supervised learning (SSL), which learns for a task on a mix of labeled and unlabeled data. Self-training is such a method with limited success [11].

Conditional random fields (CRFs) [12] have been shown to be one of the most successful approaches to sequence labeling. A CRF is a discriminative model, which directly defines a conditional distribution $p(y^l|x^l)$, and thus mainly depends on supervised learning with abundant labeled data. In this paper, we propose to upgrade CRFs and obtain a joint generative model of x^l and y^l, $p(l, x^l, y^l)$, called joint random fields (JRFs). Specifically, we propose to use the potential function $u(x^l, y^l)$ in the original CRF $p(y^l|x^l)$ as the potential function that defines a joint distribution $p(x^l, y^l)$. The resulting $p(x^l, y^l)$ defines a random field [13], also known as energy-based model [14]. And interestingly, the conditional distribution of y^l given x^l induced from this joint distribution is exactly the original CRF $p(y^l|x^l)$[1]; and the marginal distribution of $p(l, x^l)$ induced from the joint distribution is a trans-dimensional random field (TRF) language model [15, 16].

This development from CRFs to JRFs benefits both modeling and labeling of sequence data. For sequence modeling, the marginal likelihood $p(l, x^l)$ can be efficiently calculated by JRFs, without the step of producing hypothesized labels by a standalone POS tagger. For sequence labeling, JRFs admit not only supervised learning from labeled data by maximizing the conditional likelihood $p(y^l|x^l)$ (which is like the training of a CRF), but also unsupervised learning from unlabeled data by maximizing the marginal likelihood $p(l, x^l)$ (which is like the training of a TRF LM), thereby achieving task-dependent semi-supervised learning.

The benefits of JRFs to sequence modeling and labeling are demonstrated through two sets of experiments. First, various traditional language models (LMs) such as Kneser-Ney (KN) smoothed n-gram LM [1], LSTM LM [17] and TRF LM [16] are trained on Wall Street Journal (WSJ) portion of Penn Treebank (PTB) English dataset (without using POS tags). JRF LMs are trained by using POS tags. These models are then used to rescore the 1000-best list generated from the WSJ'92 test set, with similar experimental setup as in [16]. The JRF model is effective in incorporating POS tags and performs the best with the lowest rescoring word error rate (WER). Second, we conduct experiments on three sequence labeling tasks - POS tagging, chunking and NER, with Google one-billion-word dataset [18] as the unlabeled data resource. It is found that the JRF based SSL consistently outperform the CRF basline and self-training.

This work is supported by NSFC 61976122. † Corresponding author.

[1]So writing the JRF as $p(l, x^l, y^l)$ and the CRF as $p(y^l|x^l)$ is correct, not an abuse of notation.

2. JOINT RANDOM FIELDS

2.1. Background

We first briefly review CRFs, especially linear-chain CRFs, which define a conditional distribution with parameters θ for label sequence y^l given observation sequence x^l of length l:

$$p_\theta(y^l|x^l) = \frac{1}{Z_\theta(x^l)} \exp(u_\theta(x^l, y^l)) \quad (1)$$

Here the potential function

$$u_\theta(x^l, y^l) = \sum_{i=1}^{l} \phi_i(y_i, x^l) + \sum_{i=1}^{l} \psi_i(y_{i-1}, y_i, x^l), \quad (2)$$

is defined as a sum of node potentials and edge potentials, and $Z_\theta(x^l) = \sum_{y^l} \exp(u_\theta(x^l, y^l))$ is the normalizing constant. $\phi_i(y_i, x^l)$ is the node potential defined at position i, which, in recently developed neural CRFs [4, 6, 7, 8] is implemented by using features generated from a neural network (NN) of different network architectures. $\psi_i(y_{i-1}, y_i, x^l)$ is the edge potential defined on the edge connecting y_{i-1} and y_i, often implemented as a matrix A, $\psi_i(y_{i-1} = j, y_i = k, x^l) = A_{j,k}$, thereby defines a linear-chain CRF. In linear-chain CRFs, there are efficient algorithms for training (the forward-backward algorithm) and decoding (the Viterbi algorithm).

2.2. Model

Inspired from the idea of jointly modeling fixed-dimensional observations (e.g. images) and labels via a neural random field in [19], we propose to upgrade CRFs and obtain a joint distribution over sequential observations and labels, called JRFs. The keypoint is that we can use $u_\theta(x^l, y^l)$ in Eq. (2) from the original CRF to define a joint distribution $p_\theta(x^l, y^l)$:

$$p_\theta(l, x^l, y^l) = \pi_l p_\theta(x^l, y^l; l) = \frac{\pi_l}{Z_\theta(l)} \exp(u_\theta(x^l, y^l)) \quad (3)$$

where π_l is the empirical prior probability for length l. Notably, a CRF is a conditional distribution, normalizing over label sequences. In contrast, a JRF is a joint distribution, normalizing over both observation and label sequences, with the normalizing constant for length l defined as $Z_\theta(l) = \sum_{x^l, y^l} \exp(u_\theta(x^l, y^l))$.

Interestingly, it can be easily seen that the conditional distribution of y^l given x^l induced from the JRF's joint distribution Eq. (3) is exactly the original CRF Eq. (1). Further, by marginalizing out y^l, the marginal distribution of $p(l, x^l)$ induced from the joint distribution is:

$$p_\theta(l, x^l) = \frac{\pi_l}{Z_\theta(l)} \sum_{y^l} \exp(u_\theta(x^l, y^l)) = \frac{\pi_l}{Z_\theta(l)} \exp(u_\theta(x^l))$$

which acts like a trans-dimensional random field (TRF) language model [15, 16], with the potential defined by

$$u_\theta(x^l) = \log \sum_{y^l} \exp(u_\theta(x^l, y^l)).$$

Notably $u_\theta(x^l)$, which is $\log Z_\theta(x^l)$ from the CRF, can be calculated via the forward algorithm from the linear-chain potential $u_\theta(x^l, y^l)$.

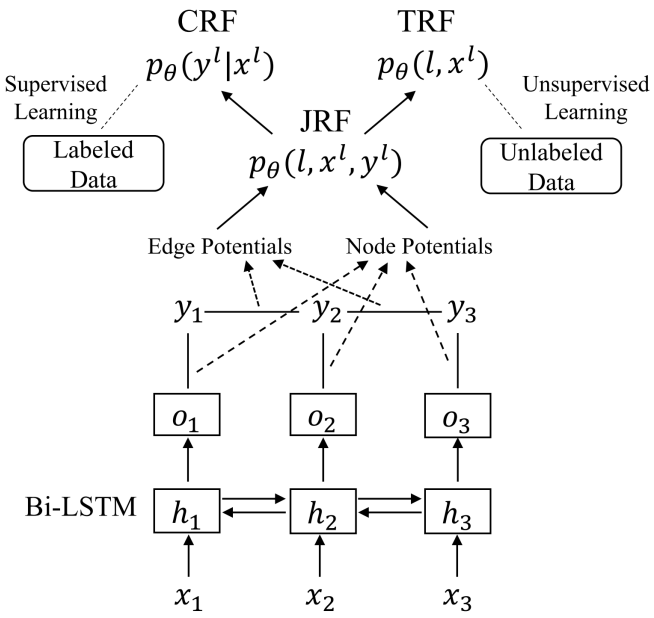

Fig. 1. Overview of the JRF model. The node and edge potentials define a JRF (a joint distribution over x^l and y^l). Inducing the conditional and the marginal from the joint yields a CRF and a TRF respectively. A JRF can be trained from labeled data (acting like a CRF) and also from unlabeled data (acting like a TRF). In practice, the node potentials are calculated from the logits $o_i, i = 1, \cdots, l$ from the NN, and the edge potential follows linear-chain definition.

2.3. Learning

Given different data resources (labeled or unlabeled), JRF can be trained under different settings (supervised, unsupervised, or semi-supervised) and applied in different downstream tasks (sequence modeling or labeling), as illustrated in Fig. 1. Note that the embedded CRF and TRF inside a JRF share all parameters θ, which is different from multi-task learning where only bottom-level parameters are shared [20].

Supervised learning of JRFs amounts to training of the embedded CRF with the following supervised objective, given labeled data in the form of empirical distribution $p_L(x^l, y^l)$,

$$\max_\theta L_s(\theta) = E_{(x^l, y^l) \sim p_L(x^l, y^l)}[\log p_\theta(y^l|x^l)] \quad (4)$$

which can be solved by applying minibatch-based stochastic gradient descent (SGD). At each iteration, a minibatch of sentences and labels is sampled from $p_L(x^l, y^l)$, denoted by D_L, and the stochastic gradients are:

$$\widehat{\frac{\partial L_s(\theta)}{\partial \theta}} = \frac{1}{|D_L|} \sum_{(x^l, y^l) \in D_L} \left[\frac{\partial u_\theta(x^l, y^l)}{\partial \theta} - \frac{\partial u_\theta(x^l)}{\partial \theta} \right]$$

Unsupervised learning of JRFs amounts to training the embedded TRF, by applying the dynamic noise-contrastive estimation (DNCE) algorithm developed in [16]. Given unlabeled data (e.g. sentences) in the form of empirical distribution $p_U(l, x^l)$, DNCE jointly optimizes over a JRF and a noise distribution $p_\phi(l, x^l)$ (gen-

Table 1. Speech recognition results of different models, trained on WSJ portion of PTB dataset. "WER" is the rescoring word error rate on WSJ'92 test data.

Method	KN5	LSTM	TRF	JRF
WER (%)	8.78	7.36	6.99	**6.77**

erally a LSTM language model) parameterized by ϕ:

$$\begin{cases} \max_{\theta} E_{(l,x^l)\sim\frac{p_U(l,x^l)+p_\phi(l,x^l)}{2}}[\log\frac{p_\theta(l,x^l)}{p_\theta(l,x^l)+p_\phi(l,x^l)}]+ \\ \quad E_{(l,x^l)\sim p_\phi(l,x^l)}[\log\frac{p_\phi(l,x^l)}{p_\theta(l,x^l)+p_\phi(l,x^l)}] \triangleq L_u(\theta) \\ \min_{\phi} KL(p_U(l,x^l)||p_\phi(l,x^l)) \end{cases} \quad (5)$$

Thanks to optimization of DNCE, the annoying normalizing constants $Z_\theta(l)$ in JRFs can be jointly estimated along with the parameter estimation. Specifically, we introduce parameters ζ_l for $\log Z_\theta(l)$ and $\zeta = (\zeta_1, \zeta_2, ..., \zeta_L)$ where L is a pre-defined maximum length. At the rest of this paper, we denote by $\xi = (\theta, \zeta)$ all the parameters in the JRF and rewrite $p_\theta(\cdot)$ as $p_\xi(\cdot)$.

At each iteration, a minibatch of sentences are sampled from $p_U(l, x^l)$, denoted by D_U, two minibatches of sentences sampled from $p_\phi(l, x^l)$, denoted by B_1, B_2 ($|B_2| = 2|B_1| = 2|D_U|$), and the stochastic gradients are:

$$\begin{cases} \widehat{\frac{\partial L_u(\xi)}{\partial \xi}} = \frac{1}{|B_2|}\sum_{(l,x^l)\in B_2}\frac{p_\xi(l,x^l)}{p_\xi(l,x^l)+p_\phi(l,x^l)}g(l,x^l;\xi) \\ \quad + \frac{1}{|D_U|+|B_1|}\sum_{(l,x^l)\in D_U\cup B_1}\frac{p_\phi(l,x^l)}{p_\xi(l,x^l)+p_\phi(l,x^l)}g(l,x^l;\xi) \\ \widehat{\frac{\partial KL(p_U||p_\phi)}{\partial \phi}} = -\frac{1}{|D_U|}\sum_{(l,x^l)\in D_U}\frac{\partial \log p_\phi(l,x^l)}{\partial \phi} \end{cases}$$

where $g(l, x^l; \xi)$ denotes the gradient of $\log p_\xi(l, x^l)$ w.r.t. $\xi = (\theta, \zeta)$, and the two gradient components w.r.t. θ and ζ are $\partial u_\theta(x^l)/\partial\theta$ and $-(\delta(l = 1), ..., \delta(l = L))$ respectively.

Semi-supervised learning of JRFs over a mix of labeled and unlabeled data amounts to combining the above supervised and unsupervised training with the following semi-supervised objective:

$$\begin{cases} \max_{\xi} L(\xi) = L_s(\xi) + \alpha L_u(\xi) \\ \min_{\phi} KL(p_U(l,x^l)||p_\phi(l,x^l)) \end{cases} \quad (6)$$

where α is the trade-off weight between supervised and unsupervised learning, and $\xi = (\theta, \zeta)$ is defined before.

3. EXPERIMENTS

3.1. Benefits to sequence modeling

Here we evaluate the performance of various models for sequence modeling, particularly language modeling, through rescoring experiments in speech recognition.

Setup. The WSJ portion of PTB dataset is used for language model training. The trained models are applied to rescore the 1000-best lists from recognizing WSJ'92 test data (330 utterances). The oracle WER of the 1000-best lists is 0.93%. This setting is the same as that used in [16].

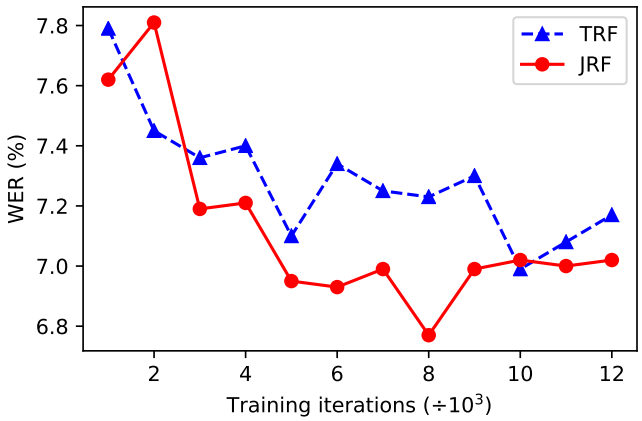

Fig. 2. Rescoring WERs on the WSJ'92 1000-best lists for TRF LM and JRF.

For comparison, traditional LMs - Kneser-Ney smoothed n-gram LM [1], LSTM LM [17] and TRF LM [16] are trained without using POS tags, and JRF trained using POS tags, by optimizing $L_s + L_u$. During testing with JRF, $\log p(l, x^l)$ can be efficiently obtained as the LM score for rescoring, with $u_\theta(x^l)$ calculated by the forward algorithm and the estimated ζ_l for $\log Z_\theta(l)$. In this way, JRFs avoid the need to produce hypothesized labels by a standalone POS tagger in testing, different from [2, 3].

Training configuration. The network structure of JRF is similar to LSTM-CNNs-CRF in [8]. For character embeddings, we use 50-dim embeddings (randomly initialized) and CNNs consisting of 100 filters for $2, 3, 4$ character width respectively. For word embeddings, 300-dim Glove word embeddings [22] are used for initialization. The embedding representations are fed to a single-layer bi-LSTM with 512 hidden size and 512 projection size. The noise distribution is defined by a simple LSTM LM with 1 hidden layer and 200 hidden units. For TRF LM, the neural network structure is similar except that the definition of the potential function is different. TRF LM uses the sum of inner product of the hidden vector and the embedding vector, as detailed in [16].

For optimizers, Adam is used with learning rate decaying as $0.001/(1 + 0.005t^{0.5})$, where t is the number of iterations so far. No dropout is applied in this experiment. For JRF, the labeled minibatch size $|D_L|$ is 64. For each iteration in JRF and TRF LM, the minibatch size of D_U is 50 ($|B_1| = 50, |B_2| = 100$, see Section 2.3). The trade-off weight α in Eq. (6) is gradually increased to 1 during training.

Results. Table 1 shows the WER results. "KN5" denotes the Kneser-Ney 5-gram LM [1] and the LSTM LM consists of two hidden layers with 1500 hidden units (correspond to the large LM in [17]), which contains more parameters than the TRF LM [16] and the JRF. Particularly, Fig. 2 plots the WER results for TRF LM and JRF during training. It can be seen from Table 1 that in this LM rescoring task, JRF using POS tags outperforms traditional LMs without using POS tags, including TRF LM, n-gram LM, and LSTM LM. This reveals the benefit of JRFs to sequence modeling by successfully incorporating additional relevant labels.

3.2. Benefits to sequence labeling

In this experiment, we evaluate the performance of various models for sequence labeling, through three tasks - POS tagging, chunking

Table 2. Sequence labeling results averaged over 5 independent runs with different random seeds. The evaluation metric is accuracy for POS and F_1 for NER and chunking. "10%" and "100%" represent the percentage of the labeled data used in training out of all the available labeled data. Note that in both settings, 50 times as much as the used labeled sentences from the Google one-billion-word dataset is additionally used as unlabeled data in training.

Method	POS (10%)	POS (100%)	NER (10%)	NER (100%)	Chunking (10%)	Chunking (100%)
CRF	96.83	97.45	86.85	90.87	89.98	94.76
Self-training	96.91	97.46	86.92	90.88	90.64	94.84
JRF	**96.96**	**97.47**	**86.99**	**90.90**	**91.12**	**95.10**

and NER, which are widely used for evaluation of sequence labeling methods. During pre-processing step, we replace all digits with zeros, which is a common setting in previous works.

Datasets. The following benchmark datasets are used - PTB POS tagging, CoNLL-2000 chunking and CoNLL-2003 English NER. For the task of POS tagging, we follow the previous work [8] to split the dataset and report accuracy. For the NER task, we follow the previous work [8] to use the BIOES tagging scheme and report F_1 score. For the chunking task, no standard development set is provided, thus 1500 sentences are randomly sampled from the training set to be the development set, as in [23]. BIOES tagging scheme and F1 score are used.

We use the Google one-billion-word dataset [18] as a pool of unlabeled sentences for semi-supervised learning. Specifically, we examine two settings - 10% (low-resource, randomly drawn) or 100% (rich-resource) out of all the available labeled sentences are used in training. In both settings, we use unlabeled sentences 50 times as much as the used labeled sentences for semi-supervised learning. Thus in the mix of labeled and unlabeled data in training, the labeling percentage is kept as 2% in two settings, while using different amounts of labeled data.

Baselines. To evaluate the benefits of JRFs to sequence labeling, CRFs and self-training are used as baselines. For CRFs, only labeled data is used in training. For self-training, after the supervised CRF is well-trained, it predicts labels for unlabeled data and adds the predicted examples to the training set to fine-tune the CRF, which is a common semi-supervised baseline. Note that all methods employ the same network structures for each task.

Training configuration. The network structures are the same as in Section 3.1. For optimizers, SGD with momentum 0.9 is used in all tasks. During training, the learning rate decays as $0.1/(1 + 0.005t^{0.5})$. For all methods, we apply 0.5 dropout during training. The labeled minibatch size is always 64. For self-training, the unlabeled or predicted minibatch size is 64. For each iteration in JRFs, the minibatch size of D_U is 32.

In experiments of sequence labeling, self-training and JRFs are initialized with the weights from pretrained CRFs, which is empirically found to yield better and faster learning in our experiments. The trade-off weight α in Eq. (6) is gradually increased to 0.1 during training. All methods are trained with 50,000 iterations and the best model according to performance on development set is reloaded to obtain the final results.

Results for all sequence labeling methods and tasks are list in Table 2. The main observations are as follows: 1) With the help of generative modeling over unlabeled data, the performance of JRFs consistently improve over CRFs across all sequence labeling tasks. 2) JRFs outperform self-training in leveraging the same amounts of labeled and unlabeled data. 3) When the amount of labeled sentences is limited (e.g. 10%), the performance superiority of JRFs over to CRFs is more significant.

4. RELATED WORK

For sequence modeling, particularly language modeling, there are directed graphical models (e.g. n-gram [1] and RNN LMs [24, 17]) and undirected graphical models (TRF LMs [21, 15, 16, 25]). Interestingly, marginalizing over y^l in JRF yields a TRF. There are previous efforts to improve LMs with additional relevant linguistic labels but with limited success. Class-based LMs employ automatically-induced word classes [26], POS tag classes [27] or combination of multiple types of tags [3]. Notably most previous methods need a standalone tagger in testing.

For sequence labeling, one of the earliest approaches to semi-supervised learning is self-training [11], which has been successfully applied to NLP tasks such as word-sense disambiguation [28] and parsing [29]. In each round of training, the model predicts labels for unlabeled data to construct labeled examples and then learns from them. Cross-View Training (CVT) is proposed in [23], which is trained by minimizing the distance between a clean prediction distribution and noised prediction distributions for unlabeled data. However, CVT is not based on CRF for sequence labeling, thus can not benefit from the power of CRF modeling (e.g. dependency modeling between labels by edges).

Outside of the area of natural language processing (NLP), a neural random field (NRF) model is presented in [19], which also jointly models observations and labels via a random field. NRFs have been successfully applied to image modeling and semi-supervised learning for image classification. The difference between NRFs and JRFs is that NRFs are for fix-dimensional modeling and classification (e.g. for images), while JRFs are for sequence modeling and labeling and applied to sequence tasks. Interestingly, the implied classifier from NRFs is multi-class logistic regression in the fixed-dimensional setting, while the implied sequential classifier from JRFs is CRFs in the trans-dimensional setting.

5. CONCLUSION

In this paper, we propose to upgrade CRFs to JRFs, obtained as a joint generative model of observation and label sequences. Specifically, we propose to use the potential function in the original CRF to define a joint distribution. This development from CRFs to JRFs enables semi-supervised learning and benefits both sequence modeling and labeling tasks. In language modeling rescoring task, the JRF model outperforms traditional language models and avoids the need to produce hypothesized labels by a standalone POS tagger. For sequence labeling, JRFs achieve consistent improvements over the CRF baseline and self-training on POS tagging, chunking and NER tasks. JRFs are expected to be of broad interest to the community, and can be improved and applied to a wider range of domains for sequence modeling and labeling.

6. REFERENCES

[1] Stanley F Chen and Joshua Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–394, 1999.

[2] Ronald Rosenfeld, Stanley F. Chen, and Xiaojin Zhu, "Whole-sentence exponential language models: a vehicle for linguistic-statistical integration," *Computer Speech & Language*, vol. 15, pp. 55–73, 2001.

[3] Ruhi Sarikaya, Stanley F Chen, Abhinav Sethy, and Bhuvana Ramabhadran, "Impact of word classing on shrinkage-based language models," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[4] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa, "Natural language processing (almost) from scratch," *Journal of machine learning research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[5] Wang Ling, Chris Dyer, Alan W Black, Isabel Trancoso, Ramon Fermandez, Silvio Amir, Luis Marujo, and Tiago Luis, "Finding function in form: Compositional character models for open vocabulary word representation," in *EMNLP*, 2015.

[6] Zhiheng Huang, Wei Xu, and Kai Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[7] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer, "Neural architectures for named entity recognition," in *Proceedings of NAACL-HLT*, 2016, pp. 260–270.

[8] Xuezhe Ma and Eduard Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," in *ACL*, 2016.

[9] Anders Søgaard and Yoav Goldberg, "Deep multi-task learning with low level tasks supervised at lower layers," in *ACL*, 2016, pp. 231–235.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.

[11] H Scudder, "Probability of error of some adaptive pattern-recognition machines," *IEEE Transactions on Information Theory*, vol. 11, no. 3, pp. 363–371, 1965.

[12] Charles Sutton, Andrew McCallum, et al., "An introduction to conditional random fields," *Foundations and Trends® in Machine Learning*, vol. 4, no. 4, pp. 267–373, 2012.

[13] Daphne Koller and Nir Friedman, *Probabilistic graphical models: principles and techniques*, MIT press, 2009.

[14] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang, "A tutorial on energy-based learning," *Predicting structured data*, vol. 1, no. 0, 2006.

[15] Bin Wang, Zhijian Ou, and Zhiqiang Tan, "Learning trans-dimensional random fields with applications to language modeling," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 876–890, 2017.

[16] Bin Wang and Zhijian Ou, "Improved training of neural trans-dimensional random field language models with dynamic noise-contrastive estimation," in *SLT*, 2018.

[17] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[18] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson, "One billion word benchmark for measuring progress in statistical language modeling," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[19] Yunfu Song and Zhijian Ou, "Learning neural random fields with inclusive auxiliary generators," *arXiv preprint arXiv:1806.00271*, 2018.

[20] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil, "Multi-task feature learning," in *NIPS*, 2007, pp. 41–48.

[21] Bin Wang, Zhijian Ou, and Zhiqiang Tan, "Trans-dimensional random fields for language modeling," in *ACL-IJCNLP*, 2015, pp. 785–794.

[22] Jeffrey Pennington, Richard Socher, and Christopher Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.

[23] Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc Le, "Semi-supervised sequence modeling with cross-view training," in *EMNLP*, 2018.

[24] Tomáš Mikolov, Stefan Kombrink, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur, "Extensions of recurrent neural network language model," in *ICASSP*, 2011.

[25] Yinghui Huang, Abhinav Sethy, Kartik Audhkhasi, and Bhuvana Ramabhadran, "Whole sentence neural language models," in *ICASSP*. IEEE, 2018.

[26] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai, "Class-based n-gram models of natural language," *Computational linguistics*, vol. 18, no. 4, pp. 467–479, 1992.

[27] Peter A Heeman, "Pos tagging versus classes in language modeling," in *Sixth Workshop on Very Large Corpora*, 1998.

[28] David Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *ACL*, 1995.

[29] David McClosky, Eugene Charniak, and Mark Johnson, "Effective self-training for parsing," in *Proceedings of the main conference on human language technology conference of the North American Chapter of the Association of Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 152–159.