Multi-Task Learning with Contextualized Word Representations for Extended Named Entity Recognition

Thai-Hoang Pham\textsuperscript{1,2}, Khai Mai\textsuperscript{2}, Nguyen Minh Trung\textsuperscript{2}, Nguyen Tuan Duc\textsuperscript{2}, Danushka Bolegala\textsuperscript{3}, Ryohei Sasano\textsuperscript{4} and Satoshi Sekine\textsuperscript{5}

\textsuperscript{1}Ohio State University
\textsuperscript{2}Alt Inc
\textsuperscript{3}University of Liverpool
\textsuperscript{4}Nagoya University
\textsuperscript{5}Riken AIP

pham.375@osu.edu, \{mai.tien.khai, nguyen.minh.trung, nguyen.tuan.duc\}@alt.ai, danushka.bollegala@liverpool.ac.uk, sasano@i.nagoya-u.ac.jp, satoshi.sekine@riken.jp

Abstract

Fine-Grained Named Entity Recognition (FG-NER) is critical for many NLP applications. While classical named entity recognition (NER) has attracted a substantial amount of research, FG-NER is still an open research domain. The current state-of-the-art (SOTA) model for FG-NER relies heavily on manual efforts for building a dictionary and designing hand-crafted features. The end-to-end framework which achieved the SOTA result for NER did not get the competitive result compared to SOTA model for FG-NER. In this paper, we investigate how effective multi-task learning approaches are in an end-to-end framework for FG-NER at different aspects. Our experiments show that using multi-task learning approaches with contextualized word representations can help an end-to-end neural network model achieve SOTA results without using any additional manual effort for creating data and designing features.

1 Introduction

Fine-grained named entity recognition (FG-NER) is a special kind of named entity recognition (NER) that focuses on identifying and classifying a large number of entity categories. In traditional NER task, often less than eleven named entity (NE) categories are defined. For example, in two shared tasks, CoNLL 2002 and CoNLL 2003 [Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003], there were only four NE types considered: Person, Location, Organization, and Miscellaneous. From these shared tasks, ten NE categories were defined for Twitter texts [Ritter et al., 2011]. The FG-NER, on the other hand, handles hundreds NE categories which are the fine-grained classification of coarse-grained categories. In particular, [Sekine et al., 2002; Sekine, 2008] proposed the entity hierarchy which contains 200 NE categories designed manually. Meanwhile, [Ling and Weld, 2012; Yosef et al., 2012; Gillick et al., 2014] used unsupervised methods for creating FG-NER category from knowledge bases such as Freebase [Bollacker et al., 2008] and YAGO [Suchanek et al., 2007]. Figure 1 shows an example when identifying and classifying NE by traditional NER and FG-NER systems.

While there have been many methods proposed for classical NER [Zhou and Su, 2002; McCallum and Li, 2003; Ma and Hovy, 2016; Pham and Le-Hong, 2017], FG-NER is still an open research domain. Unlike NER, the current SOTA model for FG-NER [Mai et al., 2018] requires significant manual effort for building a dictionary and designing features. The end-to-end neural network architectures have not achieved competitive results for this task. It is because the data sparseness problem of some NE categories when the size of FG-NER dataset are comparable with NER dataset while the number of NE categories is much larger. Moreover, identifying NE for FG-NER is more difficult compared to NER because these NE types are more complex and longer.

Recently, multi-task learning approaches have been proposed for improving the performances of NER systems [Yang et al., 2017; Lin and Lu, 2018; Lin et al., 2018; Changpinyo et al., 2018]. It can be seen as a form of inductive transfer that introduces an auxiliary task as an inductive bias to help a model prefer some hypotheses over the others. Another way to improve NER systems is using contextualized word representations to learn the dependencies among words in a sentence [Peters et al., 2018]. From these motivations,
we investigate the effectiveness of multi-task learning for the end-to-end neural network architecture in both cases uncontextualized and contextualized word representations for FG-NER task.

We have novel contributions in two folds. First, to the best of our knowledge, our work is the first study that concentrates on multi-task learning approach for sequence labeling problem in general and for FG-NER task in particular at different aspects including different parameter sharing schemes for multi-task sequence labeling, learning with neural language model, and learning at different word representation settings. We also give empirical analysis to understand the effectiveness of contextualized word representations for FG-NER task. Second, we propose an end-to-end neural network architecture which achieves SOTA result compared to the previous systems that require significant manual effort for building a dictionary and designing features. This neural network system, despite focusing on FG-NER task, can still be applied to any other sequence labeling problems.

The remainder of this paper is structured as follows. Section 2 describes multi-task learning architectures and contextual word representations used in our system. Section 3 gives experimental results and discussions. Finally, Section 4 concludes the paper.

2 Approach

2.1 Single-Task Sequence Labeling Model

With a recent resurgence of the deep learning approaches, there have been several neural network models proposed for sequence labeling problem. Most of these models shared the same abstract architecture. In particular, each input sentence is fed to these models as a sequence of words and is transformed into a sequence of distributed representations by the word embedding layer. These distributed representations can be improved by incorporating character-level information from Convolutional Neural Network (CNN) or Long Short-Term Memory (LSTM) layer into word embedding layer. That distributed representation sequence is then passed to the recurrent neural network layer (LSTM or a Gated Recurrent Unit (GRU)), and then a Conditional Random Field (CRF) layer takes as input the output of the recurrent neural network layer to predict the best output sequence [Huang et al., 2015; Lample et al., 2016; Ma and Hovy, 2016].

In our work, we re-implement the neural network architecture in [Ma and Hovy, 2016] which is the combination of CNN, bi-directional LSTM (BLSTM), and CRF models as our base model. For training, we minimize the negative log-likelihood function:

\[
E = -\sum_{t=1}^{T} \log(P(y_t|h_t))
\]

(1)

where \(h_t\) is the output of BLSTM and \(y_t\) is the label at time step \(t\). Decoding can be solved effectively by Viterbi algorithm to find the sequence with the highest conditional probability.

2.2 Multi-Task Learning with Sequence Labeling Model

Recently, multi-task sequence learning approach has been used successfully in sequence labeling problem [Yang et al., 2017; Ruder, 2017; Peng and Dredze, 2017; Hashimoto et al., 2017; Changpinyo et al., 2018; Clark et al., 2018]. In particular, the main sequence labeling task is learned with auxiliary sequence labeling tasks during training to improve the performance of the main task. These multi-task sequence labeling models are extensions to the base model discussed above with different parameter sharing schemes.

In our work, we investigate two kinds of multi-task sequence labeling models for FG-NER task including same-level-shared model and hierarchical-shared model. To train multi-task sequence labeling models, we minimize both of auxiliary and main objective function. In particular, for an input sequence \(x = (x_1, x_2, ..., x_T)\), we minimize:

\[
E_{\text{auxiliary}} = -\sum_{t=1}^{T} \log(P(y_{t}^{\text{auxiliary}}|h_{t}^{\text{auxiliary}}))
\]

(2)

if \(x\) belongs to auxiliary data, or

\[
E_{\text{main}} = -\sum_{t=1}^{T} \log(P(y_{t}^{\text{main}}|h_{t}^{\text{main}}))
\]

(3)

if \(x\) belongs to main data. \(h_{t}^{\text{auxiliary}}, h_{t}^{\text{main}}\) are the outputs of BLSTM and \(y_{t}^{\text{auxiliary}}, y_{t}^{\text{main}}\) are the labels at time step \(t\).

Same-level-Shared Model For same-level-shared model, both main and auxiliary tasks are trained and predicted at the same-level layer. Specifically, we experiment with two kinds of same-level-shared model including embedding-shared model (Figure 2b) which uses the same embedding layer for both main and auxiliary tasks, and separate LSTM and CRF layers for each task, and RNN-shared model (Figure 2c) which uses the same embedding and LSTM layers for both main and auxiliary task, and separate CRF layers for each task. In RNN-shared model, \(h_{t}^{\text{auxiliary}}, h_{t}^{\text{main}}\) are the same and are computed from one BLSTM layer:

\[
h_{t}^{\text{auxiliary}} = h_{t}^{\text{main}} = \text{BLSTM}(x)
\]

(4)

while in embedding-shared model, \(h_{t}^{\text{auxiliary}}, h_{t}^{\text{main}}\) are computed from separate BLSTM layers:

\[
h_{t}^{\text{auxiliary}} = \text{BLSTM}^{\text{auxiliary}}(x)
\]

(5)

\[
h_{t}^{\text{main}} = \text{BLSTM}^{\text{main}}(x)
\]

(6)

Hierarchical-Shared Model For hierarchical-shared model, we train and predict different supervised tasks at different-level layers. The auxiliary and main tasks are predict by the low-level and high-level layers respectively. To avoid catastrophic interference between main and auxiliary tasks, the word representations are fed into both both low-level and high-level layers. In particular, \(h_{t}^{\text{auxiliary}}, h_{t}^{\text{main}}\) are computed as follows:

\[
h_{t}^{\text{auxiliary}} = \text{BLSTM}^{\text{auxiliary}}(x)
\]

(7)

\[
h_{t}^{\text{main}} = \text{BLSTM}^{\text{main}}([x; h_{t}^{\text{auxiliary}}])
\]

(8)
2.3 Multi-Task Learning with Neural Language Model

Learning with auxiliary sequence labeling task requires additional data which may not be available for some languages. For this reason, several models have been proposed for training sequence labeling task with other unsupervised learning tasks. In particular, [Cheng et al., 2015; Rei, 2017] trained single-task sequence labeling models with neural language model simultaneously.

In our work, we incorporate a word-level neural language model into both single and multi-task sequence labeling models to improve the performances. Specifically, we put the hidden state from BLSTM at each time step into softmax layer to predict the next and previous words. Note that we use two separate language models for each forward and backward passes of BLSTM. The objective function now is the combination of sequence labeling and language model objective functions and is computed as follows:

$$E_{joint} = E + \lambda (\overrightarrow{E}_{LM} + \overleftarrow{E}_{LM})$$  \hspace{1cm} (9)$$

where $\lambda$ is a parameter controlled the impact of the language modeling task to the sequence labeling task and $\overrightarrow{E}_{LM}$, $\overleftarrow{E}_{LM}$ are the objective functions of forward and backward language models. These objective functions are computed as follows:

$$\overrightarrow{E}_{LM} = -\sum_{t=1}^{T} \log(P(w_{t-1} | \overrightarrow{h_t}))$$  \hspace{1cm} (10)$$

$$\overleftarrow{E}_{LM} = -\sum_{t=1}^{T} \log(P(w_{t+1} | \overleftarrow{h_t}))$$  \hspace{1cm} (11)$$

where $\overrightarrow{h_t}$, $\overleftarrow{h_t}$ are hidden states of forward and backward LSTM and $w_0$, $w_{T+1}$ are special tokens START, END. We investigate two kinds of incorporating neural language model into our multi-task sequence labeling model: shared-LM
which shares neural language model for both auxiliary and main sequence labeling task and unshared-LM which uses separate neural language model for each task. Figure 2d and Figure 2e show the difference between these two kinds of incorporating neural language model.

### 2.4 Deep Contextualized Word Representations

Uncontextualized word embeddings such as Word2Vec [Mikolov et al., 2013], GloVe [Pennington et al., 2014] have been used widely in neural natural language processing models and have improved their performances. However, these word embeddings still have some drawbacks. In particular, it is difficult for them to represent the complex characteristics of a word and its meaning at different contexts. Recently, deep contextualized word representations have been proposed to solve these problems. [Peters et al., 2018] introduces the word representations which are computed from multi-layer bidirectional language model with character convolutions. Unlike [Peters et al., 2018], [Radford et al., ] use Transformer instead of BLSTM to calculate the language model. [Devlin et al., 2018] improves [Radford et al., ] work by jointly learning both left and right context in Transformer.

Our proposed multi-task models can be trained with any kind of these contextualized word representations but in the scope of this paper, we only experiment with contextualized word representations described in [Peters et al., 2018] which are called Embeddings from Language Models (ELMo) and leave other contextualized word representations in our future works. ELMo are functions of the entire input sentence and are computed as follows:

$$\text{ELMo}_t = \gamma \sum_{l=1}^{L} s_l h_{t,l}^{LM}$$  \hspace{1cm} (12)

where $h_{t,0}^{LM}$ is the input representation and $h_{t,l}^{LM}$ is the output at $t^{th}$ layer of L-layer bidirectional language model at time step $t$, $s = (s_1, s_2, ..., s_L)$ are the softmax-normalized weights and $\gamma$ are the scalar parameter which allows the model to scale the ELMo vector. In our work, we incorporate a 2-layer bidirectional language model pre-trained on 1 Billion Word Language Model Benchmark dataset to our system. We set $s_1 = 0$ and $s_2 = 1$ which means we only use the output of 2nd layer of the bidirectional language model as an input for the next layer in our system.

### 3 Experiments

#### 3.1 Datasets

We conduct our experiments with FG-NER as our main task and POS tagging, chunking, NER, and language model as our auxiliary tasks. For FG-NER task, we use the English part of the dataset described in [Nguyen et al., 2017; Mai et al., 2018]. For chunking task, we use CoNLL 2000 dataset [Tjong Kim Sang and Buchholz, 2000]. This dataset has only training and testing sets so we used one part of the training set for validation. For NER task, we use CoNLL 2003 and OntoNotes 5.0 datasets [Tjong Kim Sang and

| Datasets | #Sentence | #Word | #Label |
|----------|-----------|-------|-------|
| FG-NER   | 14176     | 32052 | 208   |
| POS      | 58891     | 68241 | 51    |
| Chunk    | 8000      | 21589 | 23    |
| NER (CoNLL) | 14987  | 30290 | 8     |
| NER (OntoNotes) | 58891 | 68241 | 30    |

Table 1: Statistics of datasets used in our experiments.

| Hyper-parameter | Value |
|-----------------|-------|
| LSTM hidden size| 256   |
| CNN window size | 3     |
| #filter         | 30    |
| Dropout input dropout | 0.33 |
| BLSTM dropout    | 0.5   |
| Embedding GloVe dimension | 300 |
| ELMo dimension   | 1024  |
| $\gamma$        | 1     |
| Language Model $\lambda$ | 0.05 |
| Training batch size | 16 |
| initial learning rate | 0.01 |
| decay rate       | 0.05  |

Table 2: Hyper-parameters used in our systems.

De Meulder, 2003; Pradhan et al., 2012]. OntoNotes 5.0 dataset is also used for POS tagging task. The details of each dataset are described in Table 1.

#### 3.2 Training and Evaluation Method

The training procedure for multi-task sequence labeling models is as follows. For same-level-shared models, at each iteration, we first sample a task (main or auxiliary tasks) by Bernoulli trial based on sizes of datasets. Next, we sample a batch of training examples from the given task and then update gradients for both the shared parameters and the task-specific parameters according to the loss function of the given task. For hierarchical-shared models, at each iteration, we train the auxiliary (low-level) task first and then move to the main (high-level) task because selecting the task randomly hampers the effectiveness of hierarchical-shared models [Hashimoto et al., 2017]. We use stochastic gradient descent algorithm with decay rate 0.05. Table 2 shows the hyper-parameters we used in our models.

We evaluate the performance of our system with $F_1$ score:

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

Precision and recall are the percentage of correct named entities identified by the system and the percentage of identified named entities present in the corpus respectively. To compare fairly with previous systems, we use an available evaluation script provided by the CoNLL 2003 shared task1 to calculate $F_1$ score of our FG-NER system.
3.3 Results

**Base Model** Our base model is similar to LSTM + CNN + CRF model in [Mai et al., 2018], but in contrast to their model, we implement by PyTorch instead of Theano and train sentences with same length at each batch to make the training process faster. It achieves $F_1$ score of 81.51% compared to 80.93% reported in their paper.

**Deep Contextualized Word Representations** In the first experiment, we investigate the effectiveness of contextualized word representations (ELMo) compared to uncontextualized word representations (GloVe) when incorporating in our FG-NER systems (Base Model (GloVe) vs. Base Model (ELMo)). From Table 3, we see that using ELMo significantly improves the $F_1$ score of our system compared to using GloVe (from 81.51% to 82.74%).

To further investigate this phenomenon, we give an analysis to see which NE types are improved when using ELMo. Table 4 shows $F_1$ scores of 5 NE types which are most improved and their average token lengths. While the average token length of NEs in our dataset is 1.9, the average token lengths of these NE types are much longer. It shows that ELMo helps to improve the performance of our system when identifying NEs which are long sequences. This result is understandable because Base Model (GloVe) relies on only BLSTM layer to learn the dependencies among words in sequence to predict NE labels while Base Model (ELMo) learns these dependencies by both embedding and BLSTM layers. Unlike NER, NE types in FG-NER are often more complex and longer so using only BLSTM layer is not sufficient to capture these dependencies.

### Table 3: Results in $F_1$ scores for FG-NER (We run each setting five times and report the average $F_1$ scores.)

| Model                                | FG-NER | +Chunk | +NER (CoNLL) | +POS | +NER (Ontonotes) |
|--------------------------------------|--------|--------|--------------|------|-----------------|
| Base Model (GloVe)                   | 81.51  | -      | -            | -    | -               |
| RNN-Shared Model (GloVe)             |        | 80.53  | 81.38        | 80.55| 81.13           |
| Embedding-Shared Model (GloVe)       |        | 81.49  | 81.21        | 81.59| 81.24           |
| Hierarchical-Shared Model (GloVe)    |        | 81.65  | **82.14**    | 81.27| 81.67           |
| Base Model (ELMo)                    | 82.74  | -      | -            | -    | -               |
| RNN-Shared Model (ELMo)              |        | 82.60  | 82.09        | 81.77| 82.12           |
| Embedding-Shared Model (ELMo)        |        | 82.75  | 82.45        | 82.34| 81.94           |
| Hierarchical-Shared Model (ELMo)     |        | **83.04**| 82.72        | 82.76| 82.96           |
| Base Model (GloVe) + LM [Rei, 2017]  | 81.77  | -      | -            | -    | -               |
| RNN-Shared Model (GloVe) + Shared-LM |        | 80.83  | 81.34        | 80.69| 81.45           |
| Embedding-Shared Model (GloVe) + Shared-LM |        | 81.54  | 81.95        | 81.86| 81.34           |
| Hierarchical-Shared Model (GloVe) + Shared-LM |        | 81.69  | **81.96**    | 81.42| 81.78           |
| Base Model (ELMo) + LM               | 82.91  | -      | -            | -    | -               |
| RNN-Shared Model (ELMo) + Shared-LM  |        | 82.68  | 82.64        | 81.61| 82.36           |
| Embedding-Shared Model (ELMo) + Shared-LM |        | 82.61  | 82.32        | 82.46| 82.45           |
| Hierarchical-Shared Model (ELMo) + Shared-LM |        | 82.87  | 82.82        | 82.85| **82.99**       |
| Hierarchical-Shared Model (GloVe) + Unshared-LM |        | 81.77  | 81.80        | 81.72| 81.88           |
| Hierarchical-Shared Model (ELMo) + Unshared-LM |        | **83.35**| 83.14        | 83.06| 82.82           |
| [Mai et al., 2018]                   | 83.14  | -      | -            | -    | -               |

Table 4: 5 most improved NE types when using ELMo.

| Named Entity     | GloVe | ELMo | Token Length |
|------------------|-------|------|--------------|
| Book             | 48.65 | 76.92| 3.2          |
| Printing Other   | 60.38 | 83.33| 3.5          |
| Spaceship        | 61.90 | 80.00| 2.7          |
| Earthquake       | 75.00 | 90.20| 3.8          |
| Public Institution| 80.00 | 95.00| 4.2          |

**Parameter Sharing Schemes** In the second experiment, we investigate the impact of training FG-NER with other auxiliary sequence labeling tasks including POS tagging, chunking, and NER by our multi-task sequence labeling models at different parameter sharing schemes. In particular, we compare three kinds of multi-task sequence labeling architectures including embedding-shared, RNN-shared, and hierarchical-shared models. The size of the original OntoNotes dataset is much larger than FG-NER dataset so it is difficult for our system to focus on learning FG-NER task. Thus, we sample 10,000 sentences from OntoNotes for training POS tagging and NER.

Table 3 shows the performances of our multi-task sequence labeling models with GloVe and ELMo representations. In both cases, hierarchical-shared model gives the best performances. In particular, it achieves an $F_1$ score of 82.14% when learning with NER (CoNLL) and an $F_1$ score of 83.04% when learning with NER (Ontonotes) compared to $F_1$ scores of 81.51% and 82.74% of base model in GloVe and ELMo settings respectively. For same-level-shared models, they also achieve better results compared to base model but the differences are not very large. These results indicate that learning
FG-NER with other sequence labeling tasks at different parameter sharing schemes helps to improve the performances of FG-NER system. Also, in most cases, it is more beneficial when learning the auxiliary and the main tasks at different levels (hierarchical-shared model) compared to learning at the same level (RNN-shared and embedding-shared models).

For same-level sharing scheme, we also see that embedding-shared model achieves better performances than RNN-shared model in most cases. The gap between these two models is larger when the auxiliary task is more different from the main task (POS tagging, chunking are more different from FG-NER compared to NER).

**Neural Language Model** In the third experiment, we incorporate our systems including both single and multi-task sequence labeling models with neural language model. We experiment with two kinds of incorporating neural language model: shared-LM which shares neural language model for both auxiliary and main sequence labeling tasks and unshared-LM which uses separate neural language model for each task. For single-task model, incorporating neural language model helps to improve performance from 81.51% to 81.77% and from 82.74% to 82.91% in GloVe and ELMo settings respectively. For multi-task models, with shared-LM, our best result is an F1 score of 82.99% when learning hierarchical-shared FG-NER model with NER (Ontonotes), and with unshared-LM, our best result is an F1 score of 83.35% when learning hierarchical-shared FG-NER model with chunking. We also see that using unshared-LM helps our multi-task models achieves better performances compared to using shared-LM in most cases.

**Comparison with SOTA System** Our best system achieves the SOTA result for FG-NER. In particular, our hierarchical-shared model with chunking as an auxiliary sequence labeling task and unshared-LM achieves an F1 score of 83.35% compared to 83.14% of the previous SOTA model for FG-NER [Mai et al., 2018]. While that model requires significant manual effort for building a dictionary and designing hand-crafted features, our best model is truly end-to-end framework without using any additional information.

4 Conclusion

We present an experimental study on the effectiveness of using multi-task learning with contextualized word representations in FG-NER task. In particular, we examine the multi-task approach at different aspects including different parameter sharing schemes for multi-task sequence labeling, learning with neural language model, and learning at different word representation settings. Our best model, while does not use any additional manual effort for creating data and designing features, achieves an F1 score of 83.35% which is the SOTA result compared to the previous FG-NER model.

References

[Bollacker et al., 2008] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pages 1247–1250. ACM, 2008.

[Changpinyo et al., 2018] Soravit Changpinyo, Hexiang Hu, and Fei Sha. Multi-task learning for sequence tagging: An empirical study. In Proceedings of the 27th International Conference on Computational Linguistics, pages 2965–2977, 2018.

[Cheng et al., 2015] Hao Cheng, Hao Fang, and Mari Ostendorf. Open-domain name error detection using a multi-task rnn. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 737–746, 2015.

[Clark et al., 2018] Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc Le. Semi-supervised sequence modeling with cross-view training. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 1914–1925, 2018.

[Devlin et al., 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[Gillick et al., 2014] Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huyhn. Context-dependent fine-grained entity type tagging. arXiv preprint arXiv:1412.1820, 2014.

[Hashimoto et al., 2017] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. A joint many-task model: Growing a neural network for multiple nlp tasks. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1923–1933, 2017.

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

[Lample et al., 2016] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 260–270, 2016.

[Lin and Lu, 2018] Bill Yuchen Lin and Wei Lu. Neural adaptation layers for cross-domain named entity recognition. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 2012–2022, 2018.

[Lin et al., 2018] Ying Lin, Shengqi Yang, Veselin Stoyanov, and Heng Ji. A multi-lingual multi-task architecture for low-resource sequence labeling. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 799–809, 2018.

[Ling and Weld, 2012] Xiao Ling and Daniel S Weld. Fine-grained entity recognition. In n Proceedings of the Twenty-
Hyena: Hierarchical type classification for entity names. *Proceedings of the 24th International Conference on Computational Linguistics*, pages 1361–1370, 2012.

[Zhou and Su, 2002] GuoDong Zhou and Jian Su. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 473–480. Association for Computational Linguistics, 2002.