Entity Identification as Multitasking*

Karl Stratos
Toyota Technological Institute at Chicago
stratos@ttic.edu

Abstract

Standard approaches in entity identification hard-code boundary detection and type prediction into labels and perform Viterbi. This has two disadvantages: 1. the runtime complexity grows quadratically in the number of types, and 2. there is no natural segment-level representation. In this paper, we propose a neural architecture that addresses these disadvantages. We frame the problem as multitasking, separating boundary detection and type prediction but optimizing them jointly. Despite its simplicity, this architecture performs competitively with fully structured models such as BiLSTM-CRFs while scaling linearly in the number of types. Furthermore, by construction, the model induces type-disambiguating embeddings of predicted mentions.

1 Introduction

A popular convention in segmentation tasks such as named-entity recognition (NER) and chunking is the so-called "BIO"-label scheme. It hard-codes boundary detection and type prediction into labels using the indicators "B" (Beginning), "I" (Inside), and "O" (Outside). For instance, the sentence `Where is John Smith` is tagged as `Where/O is/O John/B-PER Smith/I-PER`. In this way, we can treat the problem as sequence labeling and apply standard structured models such as CRFs.

But this approach has certain disadvantages. First, the runtime complexity grows quadratically in the number of types (assuming exact decoding with first-order label dependency). We emphasize that the asymptotic runtime remains quadratic even if we heuristically prune previous labels based on the BIO scheme. This is not an issue when the number of types is small but quickly becomes problematic as the number grows. Second, there is no segment-level prediction: every prediction happens at the word-level. As a consequence, models do not induce representations corresponding to multi-word mentions, which can be useful for downstream tasks such as named-entity disambiguation (NED).

In this paper, we propose a neural architecture that addresses these disadvantages. Given a sentence, the model uses bidirectional LSTMs (BiLSTMs) to induce features and separately predicts:

1. Boundaries of mentions in the sentence.

2. Entity types of the boundaries.

Crucially, during training, the errors of these two predictions are minimized jointly.

One might suspect that the separation could degrade performance; neither prediction accounts for the correlation between entity types. But we find that this is not the case due to joint optimization. In fact, our model performs competitively with fully structured models such as BiLSTM-CRFs (Lample et al., 2016), implying that the model is able to capture the entity correlation indirectly by multitasking. On the other hand, the model scales linearly in the number of types and induces segment-level embeddings of predicted mentions that are type-disambiguating by construction.

*Part of the work was done while the author was at Bloomberg L. P.

Proceedings of the 2nd Workshop on Structured Prediction for Natural Language Processing, pages 7–11
Copenhagen, Denmark, September 7–11, 2017.

2 Related Work

Our work is directly inspired by Lample et al. (2016) who demonstrate that a simple neural architecture based on BiLSTMs achieves state-of-the-art performance on NER with no external features. They propose two models. The first makes structured prediction of NER labels with a CRF loss (LSTM-CRF) using the conventional BIO-label scheme. The second, which performs slightly worse, uses a shift-reduce framework mirroring tansition-based dependency parsing (Yamada and Matsumoto, 2003). While the latter also scales linearly in the number of types and produces embeddings of predicted mentions, our approach is quite different. We frame the problem as multitasking and do not need the stack/buffer data structure. Semi-Markov models (Kong et al., 2015; Sarawagi et al., 2004) explicitly incorporate the segment structure but are computationally intensive (quadratic in the sentence length).

Multitasking has been shown to be effective in numerous previous works (Collobert et al., 2011; Yang et al., 2016; Kiperwasser and Goldberg, 2016). This is especially true with neural networks which greatly simplify joint optimization across multiple objectives. Most of these works consider multitasking across different problems. In contrast, we decompose a single problem (NER) into two natural subtasks and perform them jointly. Particularly relevant in this regard is the parsing model of Kiperwasser and Goldberg (2016) which multitasks edge prediction and classification.

LSTMs (Hochreiter and Schmidhuber, 1997), and other variants of recurrent neural networks such as GRUs (Chung et al., 2014), have recently been wildly successful in various NLP tasks (Lample et al., 2016; Kiperwasser and Goldberg, 2016; Chung et al., 2014). Since there are many detailed descriptions of LSTMs available, we omit a precise definition. For our purposes, it is sufficient to treat an LSTM as a mapping $\phi : \mathbb{R}^d \times \mathbb{R}^{d'} \rightarrow \mathbb{R}^{d'}$ that takes an input vector x and a state vector h to output a new state vector $h' = \phi(x, h)$.

3 Model

Let $\mathcal{C}$ denote the set of character types, $\mathcal{W}$ the set of word types, and $\mathcal{E}$ the set of entity types. Let $\oplus$ denote the vector concatenation operation. Our model first constructs a network over a sentence closely following Lample et al. (2016); we describe it here for completeness. The model parameters Θ associated with this base network are

- Character embedding $e_c \in \mathbb{R}^{25}$ for $c \in \mathcal{C}$
- Character LSTMs $\phi^{\mathcal{C}}_f, \phi^{\mathcal{C}}_b : \mathbb{R}^{25} \times \mathbb{R}^{25} \rightarrow \mathbb{R}^{25}$
- Word embedding $e_w \in \mathbb{R}^{100}$ for $w \in \mathcal{W}$
- Word LSTMs $\phi^{\mathcal{W}}_f, \phi^{\mathcal{W}}_b : \mathbb{R}^{150} \times \mathbb{R}^{100} \rightarrow \mathbb{R}^{100}$

Let $w_1 \dots w_n \in \mathcal{W}$ denote a word sequence where word w_i has character $w_i(j) \in \mathcal{C}$ at position j. First, the model computes a character-sensitive word representation $v_i \in \mathbb{R}^{150}$ as

$$\begin{aligned} f^{\mathcal{C}}_j &= \phi^{\mathcal{C}}_f\left(e_{w_i(j)}, f^{\mathcal{C}}_{j-1}\right) & \forall j = 1 \dots |w_i| \\ b^{\mathcal{C}}_j &= \phi^{\mathcal{C}}_b\left(e_{w_i(j)}, b^{\mathcal{C}}_{j+1}\right) & \forall j = |w_i| \dots 1 \\ v_i &= f^{\mathcal{C}}_{|w_i|} \oplus b^{\mathcal{C}}_1 \oplus e_{w_i} \end{aligned}$$

for each $i = 1 \dots n$.[1] Next, the model computes

$$\begin{aligned} f^{\mathcal{W}}_i &= \phi^{\mathcal{W}}_f\left(v_i, f^{\mathcal{W}}_{i-1}\right) & \forall i = 1 \dots n \\ b^{\mathcal{W}}_i &= \phi^{\mathcal{W}}_b\left(v_i, b^{\mathcal{W}}_{i+1}\right) & \forall i = n \dots 1 \end{aligned}$$

and induces a character- and context-sensitive word representation $h_i \in \mathbb{R}^{200}$ as

$$h_i = f^{\mathcal{W}}_i \oplus b^{\mathcal{W}}_i \tag{1}$$

for each $i = 1 \dots n$. These vectors are used to define the boundary detection loss and the type classification loss described below.

Boundary detection loss We frame boundary detection as predicting BIO tags without types. A natural approach is to optimize the conditional probability of the correct tags $y_1 \dots y_n \in \{\texttt{B}, \texttt{I}, \texttt{O}\}$:

$$p(y_1 \dots y_n | h_1 \dots h_n) \propto \exp\left(\sum_{i=1}^{n} T_{y_{i-1}, y_i} \times g_{y_i}(h_i)\right) \tag{2}$$

where $g : \mathbb{R}^{200} \rightarrow \mathbb{R}^3$ is a function that adjusts the length of the LSTM output to the number of targets. We use a feedforward network $g(h) = W^2 \text{relu}(W^1 h + b^1) + b^2$. We write Θ_1 to refer to $T \in \mathbb{R}^{3 \times 3}$ and the parameters in g. The boundary detection loss is given by the negative log likelihood:

$$L_1(\Theta, \Theta_1) = -\sum_l \log p\left(y^{(l)} | h^{(l)}\right)$$

[1] For simplicity, we assume some random initial state vectors such as $f^{\mathcal{C}}_0$ and $b^{\mathcal{C}}_{|w_i|+1}$ when we describe LSTMs.

where l iterates over tagged sentences in the data.

The global normalizer for (2) can be computed using dynamic programming; see Collobert et al. (2011). Note that the runtime complexity of boundary detection is constant despite dynamic programming since the number of tags is fixed (three).

Type classification loss Given a mention boundary $1 \leq s \leq t \leq n$, we predict its type using (1) as follows. We introduce an additional pair of LSTMs $\phi_f^{\mathcal{E}}, \phi_b^{\mathcal{E}} : \mathbb{R}^{200} \times \mathbb{R}^{200} \rightarrow \mathbb{R}^{200}$ and compute a corresponding mention representation $\mu \in \mathbb{R}^{|\mathcal{E}|}$ as

$$\begin{aligned} f_j^{\mathcal{E}} &= \phi_f^{\mathcal{E}}\left(h_j, f_{j-1}^{\mathcal{E}}\right) & \forall j = s \ldots t \\ b_j^{\mathcal{E}} &= \phi_b^{\mathcal{E}}\left(h_j, b_{j+1}^{\mathcal{E}}\right) & \forall j = t \ldots s \\ \mu &= q\left(f_t^{\mathcal{E}} \oplus b_s^{\mathcal{E}}\right) \end{aligned} \tag{3}$$

where $q : \mathbb{R}^{400} \rightarrow \mathbb{R}^{|\mathcal{E}|}$ is again a feedforward network that adjusts the vector length to $|\mathcal{E}|$.[2] We write Θ_2 to refer to the parameters in $\phi_f^{\mathcal{E}}, \phi_b^{\mathcal{E}}, q$. Now we can optimize the conditional probability of the correct type τ:

$$p(\tau | h_s \ldots h_t) \propto \exp\left(\mu_\tau\right) \tag{4}$$

The type classification loss is given by the negative log likelihood:

$$L_2\left(\Theta, \Theta_2\right) = -\sum_l \log p\left(\tau^{(l)} | h_s^{(l)} \ldots h_t^{(l)}\right)$$

where l iterates over typed mentions in the data.

Joint loss The final training objective is to minimize the sum of the boundary detection loss and the type classification loss:

$$L(\Theta, \Theta_1, \Theta_2) = L_1\left(\Theta, \Theta_1\right) + L_2\left(\Theta, \Theta_2\right) \tag{5}$$

In stochastic gradient descent (SGD), this amounts to computing the tagging loss l_1 and the classification loss l_2 (summed over all mentions) at each annotated sentence, and then taking a gradient step on $l_1 + l_2$. Observe that the base network Θ is optimized to handle both tasks. During training, we use gold boundaries and types to optimize $L_2\left(\Theta, \Theta_2\right)$. At test time, we predict boundaries from the tagging layer (2) and classify them using the classification layer (4).

[2] Clearly, one can consider different networks over the boundary, for instance simple bag-of-words or convolutional neural networks. We leave the exploration as future work.

CoNLL 2003 (4 types)	F1	# words/sec
BiLSTM-CRF	90.22	3889
Mention2Vec	90.90	4825
OntoNotes (18 types)	**F1**	**# words/sec**
BiLSTM-CRF	90.77	495
Mention2Vec	89.37	4949

Table 1: Test F1 scores on CoNLL 2003 and OntoNotes newswire portion.

Model	F1
McCallum and Li (2003)	84.04
Collobert et al. (2011)	89.59
Lample et al. (2016)–Greedy	89.15
Lample et al. (2016)–Stack	90.33
Lample et al. (2016)–CRF	90.94
Mention2Vec	90.90

Table 2: Test F1 scores on CoNLL 2003.

4 Experiments

Data We use two NER datasets: CoNLL 2003 which has four entity types `PER`, `LOC`, `ORG` and `MISC` (Tjong Kim Sang and De Meulder, 2003), and the newswire portion of OntoNotes Release 5.0 which has 18 entity types (Weischedel et al., 2013).

Implementation and baseline We denote our model Mention2Vec and implement it using the DyNet library.[3] We use the same pre-trained word embeddings in Lample et al. (2016). We use the Adam optimizer (Kingma and Ba, 2014) and apply dropout at all LSTM layers (Hinton et al., 2012). We perform minimal tuning over development data. Specifically, we perform a 5×5 grid search over learning rates $0.0001 \ldots 0.0005$ and dropout rates $0.1 \ldots 0.5$ and choose the configuration that gives the best performance on the dev set.

We also re-implement the BiLSTM-CRF model of Lample et al. (2016); this is equivalent to optimizing just $L_1(\Theta, \Theta_1)$ but using typed BIO tags. Lample et al. (2016) use different details in optimization (SGD with gradient clipping), data preprocessing (replacing every digit with a zero), and the dropout scheme (droptout at BiLSTM input (1)). As a result, our re-implementation is not directly comparable and obtains different (slightly lower) results. But we emphasize that the main goal of this paper is to demonstrate the utility the

[3] https://github.com/karlstratos/mention2vec

PER	In another letter dated January 1865, a well-to-do Washington matron wrote to **Lincoln** to plead for . . .
	Chang and **Washington** were the only men's seeds in action on a day that saw two seeded women's . . . "Just one of those things, I was just trying to make contact," said **Bragg**. **Washington**'s win was not comfortable, either.
LOC	Lauck, from **Lincoln**, Nebraska, yelled a tirade of abuse at the court after his conviction for inciting . . .
	. . . warring factions, with the PUK aming to break through to KDP's headquarters in **Saladhuddin**. . . . is not expected to travel to the **West Bank** before Monday," Nabil Abu Rdainah told Reuters. . . . off a bus near his family home in the village of **Donje Ljupce** in the municipality of Podujevo.
ORG	English division three - Swansea v **Lincoln**.
	SOCCER - OUT-OF-SORTS **NEWCASTLE** CRASH 2 1 AT HOME. Moura, who appeared to have elbowed Cyprien in the final minutes of the 3 0 win by **Neuchatel**, was . . . In Sofia: Leviski Sofia (Bulgaria) 1 **Olimpija** (Slovenia) 0
WORK_OF_ART	. . . Bond novels, and "**Treasure Island**," produced by Charlton Heston who also stars in the movie.
	. . . probably started in 1962 with the publication of Rachel Carson's book "**Silent Spring**." . . . Victoria Petrovich) spout philosophic bon mots with the self-concious rat-a-tat pacing of "**Laugh In**." Dennis Farney's Oct. 13 page - one article "**River of Despair**," about the poverty along the . . .
GPE	. . . from a naval station at **Treasure Island** near the Bay Bridge to San Francisco to help fight fires.
	. . . lived in an expensive home on **Lido Isle**, an island in Newport's harbor, according to investigators. . . . Doris Moreno, 37, of **Bell Gardens**; and Ana L. Azucena, 27, of Huntington Park. One group of middle-aged manufacturing men from the company's **Zama** plant outside Tokyo was . . .
ORG	. . . initiative will spur members of **the General Agreement on Tariffs and Trade** to reach . . .
	. . . question of Taiwan's membership in **the General Agreement on Tariffs and Trade** should . . . "He doesn't know himself," Kathy Stanwick of **the Abortion Rights League** says of . . . . . . administrative costs, management and research, **the Office of Technology Assessment** just reported.

Table 3: Nearest neighbors of detected mentions in CoNLL 2003 and OntoNotes using (3).

proposed approach rather than obtaining a new state-of-the-art result on NER.

4.1 NER Performance

Table 1 compares the NER performance and decoding speed between BiLSTM-CRF and Mention2Vec. The F1 scores are obtained on test data. The speed is measured by the average number of words decoded per second.

On CoNLL 2003 in which the number of types is small, our model achieves 90.50 compared to 90.22 of BiLSTM-CRF with minor speed improvement. This shows that despite the separation between boundary detection and type classification, we can achieve good performance through joint optimization. On OntoNotes in which the number of types is much larger, our model still performs well with an F1 score of 89.37 but is behind BiLSTM-CRF which achieves 90.77. We suspect that this is due to strong correlation between mention types that fully structured models can exploit more effectively. However, our model is also an order of magnitude faster: 4949 compared to 495 words/second.

Finally, Table 2 compares our model with other works in the literature on CoNLL 2003. McCallum and Li (2003) use CRFs with manually crafted features; Collobert et al. (2011) use convolutional neural networks; Lample et al. (2016) use BiLSTMs in a greedy tagger (Greedy), a stack-based model (Stack), and a global tagger using a CRF output layer (CRF). Mention2Vec performs competitively.

4.2 Mention Embeddings

Table 3 shows nearest neighbors of detected mentions using the mention representations μ in (3). Since μ_τ represents the score of type τ, the mention embeddings are clustered by entity types *by construction*. The model induces completely different representations even when the mention has the same lexical form. For instance, based on its context `Lincoln` receives a person, location, or organization representation; `Treasure Island` receives a book or location representation. The model also learns representations for long multi-word expressions such as `the General Agreement on Tariffs and Trade`.

5 Conclusion

We have presented a neural architecture for entity identification that multitasks boundary detection and type classification. Joint optimization enables the base BiLSTM network to capture the correlation between entities indirectly via multitasking. As a result, the model is competitive with fully structured models such as BiLSTM-CRFs on CoNLL 2003 while being more scalable and also inducing context-sensitive mention embeddings clustered by entity types. There are

many interesting future directions, such as applying this framework to NED in which type classification is much more fine-grained and finding a better method for optimizing the multitasking objective (e.g., instead of using gold boundaries for training, dynamically use predicted boundaries in a reinforcement learning framework).

Acknowledgments

The author would like to thank Linpeng Kong for his consistent help with DyNet and Miguel Ballesteros for pre-trained word embeddings.

References

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS Deep Learning Workshop*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research* 12:2493–2537.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580* .

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics* 4:313–327.

Lingpeng Kong, Chris Dyer, and Noah A Smith. 2015. Segmental recurrent neural networks. *arXiv preprint arXiv:1511.06018* .

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL*.

Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, pages 188–191.

Barbara Plank. 2016. Keystroke dynamics as signal for shallow syntactic parsing. In *Proceedings of COLING*.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of ACL*.

Sunita Sarawagi, William W Cohen, et al. 2004. Semi-markov conditional random fields for information extraction. In *NIPs*. volume 17, pages 1185–1192.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, pages 142–147.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA* .

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT*. volume 3, pages 195–206.

Zhilin Yang, Ruslan Salakhutdinov, and William Cohen. 2016. Multi-task cross-lingual sequence tagging from scratch. *arXiv preprint arXiv:1603.06270* .