Contextualized Word Representations from Distant Supervision with and for NER

Abbas Ghaddar
RALI-DIRO
Université de Montréal
Montréal, Canada
abbas.ghaddar@umontreal.ca

Philippe Langlais
RALI-DIRO
Université de Montréal
Montréal, Canada
felipe@iro.umontreal.ca

Abstract

We describe a special type of deep contextualized word representation that is *learned from* distant supervision annotations and *dedicated to* named entity recognition. Our extensive experiments on 7 datasets show systematic gains across all domains over strong baselines, and demonstrate that our representation is complementary to previously proposed embeddings. We report new state-of-the-art results on CoNLL and OntoNotes datasets.

1 Introduction

Contextualized word representations are nowadays a resource of choice for most NLP tasks (Peters et al., 2018). These representations are trained with unsupervised language modelling (Jozefowicz et al., 2016), masked-word prediction (Devlin et al., 2018), or supervised objectives like machine translation (McCann et al., 2017). Despite their strength, best performances on downstream tasks (Akbik et al., 2018; Lee et al., 2018; He et al., 2018) are always obtained when these representations are stacked with traditional (classic) word embeddings (Mikolov et al., 2013; Pennington et al., 2014).

Our main contribution in this work is to revisit the work of Ghaddar and Langlais (2018a) that explores distant supervision for learning *classic* word representations, used later as features for Named Entity Recognition (NER). Motivated by the recent success of pre-trained language model embeddings, we propose a contextualized word representation trained on the distant supervision material made available by the authors. We do so by training a model to predict the entity type of each word in a given sequence (e.g. paragraph).

We run extensive experiments feeding our representation, along side with previously proposed traditional and contextualized ones, as features to a vanilla Bi-LSTM-CRF (Ma and Hovy, 2016). Results shows that our contextualized representation leads to significant boost in performances on 7 NER datasets of various sizes and domains. The proposed representation surpasses the one of Ghaddar and Langlais (2018a) and is complementary to popular contextualized embeddings like ELMo (Peters et al., 2018).

By simply stacking various representations, we report new state-of the-art performances on CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003) and OntoNotes 5.0 (Pradhan et al., 2013) with a F1 score of 93.22 and 89.95 respectively.

2 Related Work

Pre-trained contextualized word-embeddings have shown great success in NLP due to their ability to capture both syntactic and semantic properties. ELMo representations (Peters et al., 2018) are built from internal states of forward and backward word-level language models. Akbik et al. (2018) showed that pure character-level language models can also be used. Also, McCann et al. (2017) used the encoder of a machine translation model to compute contextualized representations. Recently, (Devlin et al., 2018) proposed BERT, an encoder based on the Transformer architecture (Vaswani et al., 2017). To overcome the unidirectionality of the language model objective, the authors propose two novel tasks for unsupervised learning: masked words and next sentence prediction.

Ghaddar and Langlais (2018a) applied distant supervision (Mintz et al., 2009) in order to induce traditional word representations. They used WiFiNE[1] (Ghaddar and Langlais, 2018b, 2017), a Wikipedia dump with massive amount of automatically annotated entities, using the fine-grained

[1] http://rali.iro.umontreal.ca/rali/en/wikipedia-lex-sim

Proceedings of the 2019 EMNLP Workshop W-NUT: The 5th Workshop on Noisy User-generated Text, pages 101–108
Hong Kong, Nov 4, 2019. ©2019 Association for Computational Linguistics

tagset proposed in (Ling and Weld, 2012). Making use of Fasttext (Bojanowski et al., 2016), they embedded words and (noisy) entity types in this resource into the same space from which they induced a 120-dimensional word-representation, where each dimension encodes the similarity of a word with one of the 120 types they considered. While the authors claim the resulting representation captures contextual information, they do not specifically train it to do so. Our work revisits precisely this.

3 Data and Preprocessing

We leverage the entity type annotations in WiFiNE which consists of 1.3B tokens annotated with 159.4M mentions, which cover 15% of the tokens. A significant amount of named entities such as person names and countries can actually be resolved via their mention tokens only (Ghaddar and Langlais, 2016a,b). With the hope to enforce context, we use the fine-grained type annotation available in the resource (e.g. `/person/politician`). Also, inspired by the recent success of masked-word prediction (Devlin et al., 2018), we further apply preprocessing to the original annotations by **(a)** replacing an entity by a special token [MASK] with a probability of 0.2, and **(b)** replacing primary entity mentions, e.g. all mentions of *Barack Obama* within its dedicated article, by the special mask token with a probability of 0.5. In WiFiNE, named-entities that do not have a Wikipedia article (e.g. *Malia Ann* in Figure 2) are left unannotated, which introduces false negatives. Therefore, we mask non-entity words when we calculate the loss.

Although contextualized representation learning has access to arbitrary large contexts (e.g. the document), in practice representations mainly depend on sentence level context (Chang et al., 2019). To overcome this limitation to some extent, we use the Wikipedia layout provided in WiFiNE to concatenate sentences of the same paragraphs, sections and document up to a maximum size of 512 tokens.

An illustration of the preprocessing is depicted in Figure 2 where for the sake of space, a single sentence is being showed. Masked entities encourage the model to learn good representations for non-entity words even if they do not participate in the final loss. Because our examples are sections and paragraphs, the model will be forced to encode sentence- as well as document-based context. In addition, training on (longer) paragraphs is much faster and memory efficient than batching sentences.

4 Learning our Representation

We use a model (Figure 1) composed of a multi-layer bidirectional encoder that produces hidden states for each token in the input sequence. At the output layer, the last hidden states are fed into a softmax layer for predicting entity types. Following (Strubell et al., 2017), we used as our encoder the Dilated Convolutional Neural Network (DCNN) with an exponential increasing dilated width. DCNN was first proposed by (Yu and Koltun, 2015) for image segmentation, and was successfully deployed for NER by (Strubell et al., 2017). The authors show that stacked layers of DCNN that incorporate document context have comparable performance to Bi-LSTM while being 8 times faster. DCNN with a size 3 convolution window needs 8 stacked layers to incorporate the entire input context of a sequence of 512 tokens, compared to 255 layers using a regular CNN. This greatly reduces the number of parameters and makes training more scalable and efficient. Because our examples are paragraphs rather than sentences, we employ a self-attention mechanism on top of DCNN output with the aim to encourage the model to focus on salient global information. In this paper, we adopt the multi-head self-attention formulation by Vaswani et al. (2017). Comparatively, Transformer-based architectures (Devlin et al., 2018) require a much larger[2] amount of resources and computations. To improve the handling of rare and unknown words, our input sequence consists of WordPiece embeddings (Wu et al., 2016) as used by Devlin et al. (2018); Radford et al. (2018). We use the same vocabulary distributed by the authors, as it was originally learned on Wikipedia. Model parameters and training details are provided in Appendix A.1.

5 Experiments on NER

5.1 Datasets

To compare with state-of-the-art models, we consider two well-established NER benchmarks: CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003) and OntoNotes 5.0 (Pradhan et al., 2012).

[2] Actually prohibitive with our single GPU computer.

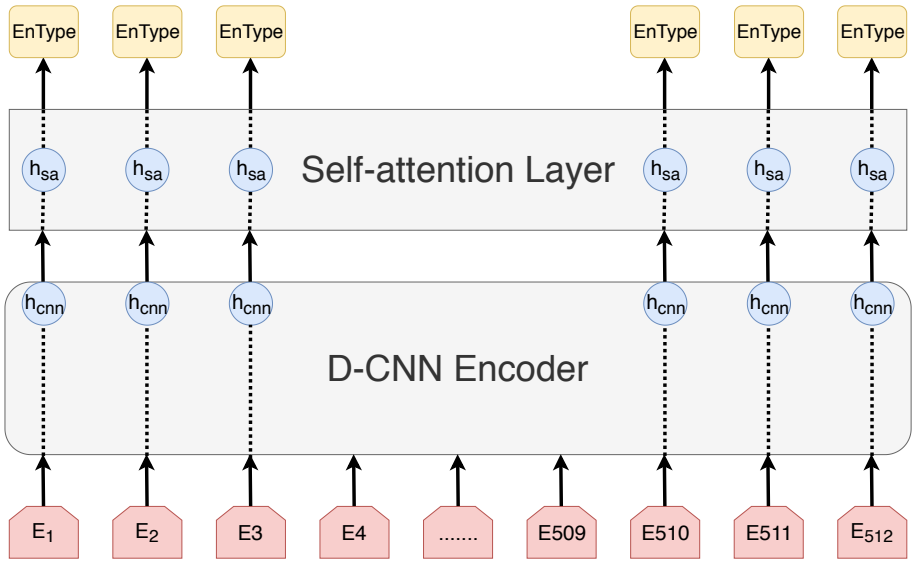

Figure 1: Illustration of the architecture of the model used for learning our representation. It consists of stacked layers of dilated convolutional neural network followed by a self-attention layer. The input is a sequence of tokens with a maximum length of 512, where the output is the associated entity type sequence. We use the hidden state of the last DCNN layer and the self-attention layer as our representation.

before	*[Obama]* first daughter, Malia Ann, was born in *[July 1998]* at *[Chicago]*, *[Illinois]*.
after	***[MASK]*** first daughter, Malia Ann, was born in *[July 1998]* at *[Chicago]*, *[Illinois]*.
tags	`/person/politician` X X X X X X X X `/date` `/date` X `/location/city` X `/location/province` X

Figure 2: Sequence before and after masking, along with output tags. X indicates that no prediction is made for the corresponding token.

To further determine how useful our learned representation is on other domains, we also considered three additional datasets: WNUT17 (Derczynski et al., 2017) (social media), I2B2 (Stubbs and Uzuner, 2015) (biomedical), and FIN (Alvarado et al., 2015) (financial). In addition, we perform an out-domain evaluation for models trained on CONLL-2003 and tested on WIKIGOLD (Balasuriya et al., 2009) (wikipedia) and WEBPAGES (Ratinov and Roth, 2009) (web pages). Statistics of the datasets are provided in Appendix A.2.

5.2 Input Representations

Our NER model is a vanilla Bi-LSTM-CRF (Ma and Hovy, 2016) that we feed with various representations (hereafter described) at the input layer. Model parameters and training details are provided in Appendix A.3.

5.2.1 Word-Shape Features

We use 7 word-shape features: `allUpper`, `allLower`, `upperFirst`, `upperNotFirst`, `numeric`, `punctuation` or `noAlphaNum`. We randomly allocate a 25-dimensional vector for each feature, and learn them during training.

5.2.2 Traditional Word Embeddings

We use the 100-dimensional case sensitive SSKIP (Ling et al., 2015) word embeddings. We also compare with the previously described 120-dimensional vector representation of (Ghaddar and Langlais, 2018a), they call it LS.

5.2.3 Contextualized Word Embeddings

We tested 3 publicly available contextualized word representations: ELMo (Peters et al., 2018): $dim = 1024$, $layers = 3$; FLAIR (Akbik et al., 2018): $d = 2048$, $l = 1$; and BERT (Devlin et al., 2018): $d = 1024$, $l = 4$. For the latter, we use the hidden state of the 4 last layers of the `Large` model. For the proposed representation, we use the hidden state of the last DCNN layer and the self-attention layer as feature input ($d = 384$, $l = 2$). Following Peters et al. (2018), each representation (including ours) is the weighted sum of the hidden layers, where weights are learned

	Conll			Ontonotes		
	$\mathcal{X}$	LS	ours	$\mathcal{X}$	LS	ours
ws+sskip	90.37	91.23 (+0.9)	91.76 (+1.4)	86.44	87.95 (+0.9)	88.13 (+0.9)
ws+sskip+elmo	92.47	92.49 (+0.0)	92.82 (+0.4)	89.37	89.44 (+0.1)	89.68 (+0.3)
ws+sskip+elmo+flair	92.69	92.75 (+0.1)	**93.22 (+0.5)**	89.55	89.59 (+0.0)	89.73 (+0.2)
ws+sskip+elmo+flair+bert	92.91	92.87 (+0.0)	93.01 (+0.1)	89.66	89.70 (+0.0)	**89.95 (+0.3)**
(Peters et al., 2018)		92.20			-	
(Clark et al., 2018)		92.61			88.81	
(Devlin et al., 2018)		92.80			-	

Table 1: F1 scores over five runs on CoNLL and OntoNotes test set of ablation experiments. We evaluate 4 baselines without additional embeddings (column $\mathcal{X}$) and with LS embeddings (Ghaddar and Langlais, 2018a) or ours. Figures in parenthesis indicate the gain over the baselines.

during training. We use concatenation to stack the resulting representations in the input layer of our vanilla Bi-LSTM-CRF model, since Coates and Bollegala (2018) show that concatenation performs reasonably well in many NLP tasks.

6 Experiments

6.1 Comparison to LS embeddings

Since we used the very same distant supervision material for training our contextual representation, we compare it to the one of Ghaddar and Langlais (2018a). We concentrate on CoNLL-2003 and OntoNotes 5.0, the datasets most often used for benchmarking NER systems.

Table 1 reports results of 4 strong baselines that use popular embeddings (column $\mathcal{X}$), further adding either the LS representation (Ghaddar and Langlais, 2018a) or ours. In all experiments, we report the results on the test portion of models performing the best on the official development set of each dataset. As a point of comparison, we also report 2018 state-of-the-art systems.

First we observe that adding our representation to all baseline models leads to systematic improvements, even for the very strong baseline which exploits all three contextual representations (fourth line). The LS representation does not deliver such gains, which demonstrates that our way of exploiting the very same distant supervision material is more efficient. Second, we see that adding our representation to the weakest baseline (line 1), while giving a significant boost, does not deliver as good performance as when adding other contextual embeddings. Nevertheless, combining all embeddings yields state-of-the-art on both CoNLL and OntoNotes.

6.2 Comparing Contextualized Embeddings

Table 2 reports F1 scores on the test portion of the 7 datasets we considered, for models trained with different embedding combinations. Our baseline is composed of word-shape and traditional (Sskip) embeddings. Then, contextualized word representations are added greedily, that is, the representation that yields the largest gain when considered is added first and so forth.

Expectedly, ELMo is the best representation to add to the baseline configuration, with significant F1 gains for all test sets. We are pleased to observe that the next best representation to consider is ours, significantly outperforming Flair. This is likely due to the fact that both Flair and ELMo embeddings are obtained by training a language model, therefore encoding similar information.

Continuously aggregating other contextual embeddings (Flair and Bert) leads to some improvements on some datasets, and degradations on others. In particular, stacking all representations leads to the best performance on 2 datasets only: OntoNotes and i2b2. Those datasets are large, domain diversified, and have more tags than other ones. In any case, stacking word-shapes, Sskip, ELMo and our representation leads to a strong configuration across all datasets. Adding our representation to ELMo, actually brings noticeable gains (over 2 absolute F1 points) in out-domain settings, a very positive outcome.

Surprisingly, Bert did not perform as we expected, since they bring minor (OntoNotes) or no (CoNLL) improvement. We tried to reproduce the results of fine-tuned and feature-based approaches reported by the authors on CoNLL,

	In Domain					Out Domain	
	Conll	**Onto**	**WNUT**	**FIN**	**I2B2**	**WikiGold**	**WebPage**
WS+SSKIP	90.73	86.44	32.30	81.82	86.41	66.03	45.13
+ELMo	92.47	89.37	44.15	82.03	94.47	76.34	54.45
+Ours	92.96	89.68	**47.40**	83.00	94.75	**78.51**	**57.23**
+FLAIR	**93.22**	89.73	46.80	**83.11**	94.79	77.77	56.20
+BERT	93.02	**89.97**	46.47	81.94	**94.92**	78.06	56.84

Table 2: Mention-level F1 scores. The baseline (first line) uses word shape and traditional (classic) embeddings. Variants stacking various representations are presented in decreasing order of F1 return. So for instance, ELMo is the best representation to add to the baseline one.

but as many others,[3] our results were disappointing.

6.3 Analysis

We suspect one reason for the success of our representation is that it captures document wise context. We inspected the words the most attended according to the the self-attention layer of some documents, an excerpt of which is reported in Figure 3. We observe that attended words in the document are often related to the topic of the document.

84	economic	*Stock, mark, Wall, Treasury, bond*
148	sport	*World, team, record, game, win*
201	news	*truck, Fire, store, hospital, arms*

Figure 3: top 5 attended words for some randomly picked documents in the dev set of CONLL. Column 1 indicate document number, while column 2 is our appreciation of the document topic.

We further checked whether the gain could be imputable to the fact that WiFiNE contains the mentions that appear in the test sets we considered. While this of course happens (for instance 38% of the test mentions in ONTONOTES are in the resource), the performance on those mentions with our representation is no better than the performance on other mentions.

7 Conclusion and Future Work

We have explored the idea of generating a contextualized word representation from distant supervision annotations coming from Wikipedia, improving over the static representation of Ghaddar and Langlais (2018a). When combined with popular contextual ones, our representation leads to state-of-the-art performance on both CONLL and ONTONOTES. We are currently analyzing the complementarity of our representation to others.

We plan to investigate tasks such as coreference resolution and non-extractive machine reading comprehension, where document level context and entity type information is crucial. The source code and the pre-trained models we used in this work are publicly available at http://rali.iro.umontreal.ca/rali/en/wikipedia-ds-cont-emb

Acknowledgments

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp and Titan V GPUs used for this research. We thank the anonymous reviewers for their insightful comments.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649.

Julio Cesar Salinas Alvarado, Karin Verspoor, and Timothy Baldwin. 2015. Domain adaption of named entity recognition to support credit risk assessment. In *Proceedings of the Australasian Language Technology Association Workshop 2015*, pages 84–90.

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R Curran. 2009. Named

[3] https://github.com/google-research/bert/issues?utf8=%E2%9C%93&q=NER

Entity Recognition in Wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, pages 10–18. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606*.

Ming-Wei Chang, Kristina Toutanova, Kenton Lee, and Jacob Devlin. 2019. Language Model Pre-training for Hierarchical Document Representations. *arXiv preprint arXiv:1901.09128*.

Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc V Le. 2018. Semi-supervised sequence modeling with cross-view training. *arXiv preprint arXiv:1809.08370*.

Joshua Coates and Danushka Bollegala. 2018. Frustratingly easy meta-embedding–computing meta-embeddings by averaging source word embeddings. *arXiv preprint arXiv:1804.05262*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

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

Abbas Ghaddar and Philippe Langlais. 2016a. WikiCoref: An English Coreference-annotated Corpus of Wikipedia Articles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Portorož, Slovenia.

Abbas Ghaddar and Phillippe Langlais. 2016b. Coreference in Wikipedia: Main concept resolution. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 229–238.

Abbas Ghaddar and Phillippe Langlais. 2017. WiNER: A Wikipedia Annotated Corpus for Named Entity Recognition. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 413–422.

Abbas Ghaddar and Phillippe Langlais. 2018a. Robust Lexical Features for Improved Neural Network Named-Entity Recognition. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1896–1907.

Abbas Ghaddar and Phillippe Langlais. 2018b. Transforming Wikipedia into a Large-Scale Fine-Grained Entity Type Corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. *arXiv preprint arXiv:1805.04787*.

Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*.

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

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. *arXiv preprint arXiv:1804.05392*.

Wang Ling, Yulia Tsvetkov, Silvio Amir, Ramon Fermandez, Chris Dyer, Alan W Black, Isabel Trancoso, and Chu-Cheng Lin. 2015. Not all contexts are created equal: Better word representations with variable attention. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1367–1372.

Xiao Ling and Daniel S Weld. 2012. Fine-Grained Entity Recognition. In *AAAI*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6297–6308.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*, volume 14, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards

Robust Linguistic Analysis using OntoNotes. In *CoNLL*, pages 143–152.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958.

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and Accurate Entity Recognition with Iterated Dilated Convolutions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2660–2670.

Amber Stubbs and Özlem Uzuner. 2015. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/UTHealth corpus. *Journal of biomedical informatics*, 58:S20–S29.

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*, pages 142–147. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Fisher Yu and Vladlen Koltun. 2015. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*.

A Appendices

A.1 Training Representation

We use 8 stacked layers of DCNN to encode input sequences of maximum length of 512. WordPiece and position embeddings, number of filters in each dilated layer and self-attention hidden units were all set to 384. For self-attention, we use 6 attention heads and set intermediate hidden unit to 512. We apply a dropout mask (Srivastava et al., 2014) with a probability of 0.3 at the end of each DCNN layer, and at the input and output of the self-attention layer. We adopt the Adam (Kingma and Ba, 2014) optimization algorithm, set the initial learning rate to $1e^{-4}$, and use an exponential decay. We train our model up to 1.5 millions steps with mini-batch size of 64. We implemented our system using the Tensorflow (Abadi et al., 2016) library, and training requires about 5 days on a single TITAN XP GPU.

A.2 Dataset

Table 3 list the dataset used in this study domain, label size, and number of mentions in train/dev/test portions.

			# entities		
Dataset	**Domain**	**Types**	**train**	**dev**	**test**
CoNLL	news	4	23499	5942	5648
OntoNotes	news	18	81828	11066	11257
WNUT17	tweet	6	1975	836	1079
i2b2	bio	23	11791	5453	11360
Fin	finance	4	460	-	120
WikiGold	wikipedia	4	-	-	3558
WebPages	web	4	-	-	783

Table 3: Statistics on the datasets used in our experiments.

We used the last 2 datasets to perform an out-of-domain evaluation of CoNLL models. Those are small datasets extracted from Wikipedia articles and web pages respectively, and manually annotated following CoNLL-2003 annotation scheme.

A.3 NER Model Training

Our system is a single Bi-LSTM layer with a CRF decoder, with 128 hidden units for all datasets except for OntoNotes and i2b2 where we use 256 hidden units. For each learned representations (ours, ELMo, Flair, Bert), we use the weighted sum of all layers as input, where weights are learned during training. For each word, we stack the embeddings by concatenating them to form the input feature of the encoder.

Training is carried out by mini-batch of stochastic gradient descent (SGD) with a momentum of 0.9 and a gradient clipping of 5.0. To mitigate over-fitting, we apply a dropout mask with a probability of 0.7 on the input and output vectors of the Bi-LSTM layer. The mini-batch is 10 and learning rate is 0.011 for all datasets. We trained the models up to 63 epochs and use early stopping based on the official development set. For Fin, we randomly sampled 10% of the train set for development.