Word-Level Coreference Resolution

Vladimir Dobrovolskii
ABBYY / Moscow, Russia
v.dobrovolskii@abbyy.com

Abstract

Recent coreference resolution models rely heavily on span representations to find coreference links between word spans. As the number of spans is $O(n^2)$ in the length of text and the number of potential links is $O(n^4)$, various pruning techniques are necessary to make this approach computationally feasible. We propose instead to consider coreference links between individual words rather than word spans and then reconstruct the word spans. This reduces the complexity of the coreference model to $O(n^2)$ and allows it to consider all potential mentions without pruning any of them out. We also demonstrate that, with these changes, SpanBERT for coreference resolution will be significantly outperformed by RoBERTa. While being highly efficient, our model performs competitively with recent coreference resolution systems on the OntoNotes benchmark.

1 Introduction

Coreference resolution is used in various natural language processing pipelines, such as machine translation (Ohtani et al., 2019; Miculicich Werlen and Popescu-Belis, 2017), question answering (Dhingra et al., 2018) and text simplification (Wilkens et al., 2020). As a building block of larger systems itself, it is crucial that coreference resolution models aim to not only achieve higher benchmark scores but be time and memory efficient as well.

Most recent coreference resolution models for English use the mention ranking approach: the highest ranking antecedent of each span is predicted to be coreferent to that span. As this approach presents a serious computational challenge of $O(n^4)$ complexity in the document length, Lee et al. (2017) keep only M top-scoring spans, while Lee et al. (2018) extend this idea by keeping only top K antecedents for each span based on an easily computable score. Their C2F-COREF model, which showed 72.6 F1 on the OntoNotes 5.0 shared task (Pradhan et al., 2012), was later improved by Joshi et al. (2020), who introduced SpanBERT to obtain better span representations (79.6 F1), and by Xu and Choi (2020), who proposed a novel way of higher-order coreference resolution (80.2 F1).

The current state-of-the-art system built by Wu et al. (2020) uses a different approach, formulating the task as a machine reading comprehension problem. While they were able to achieve an impressive 83.1 F1 on the OntoNotes 5.0 shared task, their model is particularly computationally expensive as it requires a full transformer pass to score each span's antecedents.

To make coreference resolution models more compact and allow for their easier incorporation into larger pipelines, we propose to separate the task of coreference resolution from span extraction and to solve it on the word-level, lowering the complexity of the model to $O(n^2)$. The span extraction is to be performed separately only for those words that are found to be coreferent to some other words. An additional benefit of this approach is that there is no need for a complex span representation, which is usually concatenation-based. Overall this allows us to build a lightweight model that performs competitively with other mention ranking approaches and achieves 81.0 F1 on the OntoNotes benchmark.[1]

2 Related work

2.1 End-to-end coreference resolution

The model proposed by Lee et al. (2018) aims to learn a probability distribution P over all possible antecedent spans Y for each span i:

$$P(y_j) = \frac{e^{s(i,y_j)}}{\sum_{y' \in Y(i)} e^{s(i,y')}} \quad (1)$$

[1] Source code and models are available at https://github.com/vdobrovolskii/wl-coref

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 7670–7675
November 7–11, 2021.

Here $s(i, j)$ is the pairwise coreference score of spans i and j, while $Y(i)$ contains spans to the left of i and a special dummy antecedent ϵ with a fixed score $s(i, \epsilon) = 0$ for all i.

The pairwise coreference score $s(i, j)$ of spans i and j is a sum of the following scores:

1. $s_m(i)$, whether i is a mention;
2. $s_m(j)$, whether j is a mention;
3. $s_c(i, j)$, coarse coreference score of i and j;
4. $s_a(i, j)$, fine coreference score of i and j.

They are calculated as follows:

$$s_m(i) = \text{FFNN}_m(\mathbf{g}_i) \quad (2)$$
$$s_c(i, j) = \mathbf{g}_i^\intercal \, \mathbf{W}_c \, \mathbf{g}_j \quad (3)$$
$$s_a(i, j) = \text{FFNN}_a([\mathbf{g}_i, \mathbf{g}_j, \mathbf{g}_i \odot \mathbf{g}_j, \phi]) \quad (4)$$

where $\mathbf{g}_i$ is the vector representation of span i and ϕ is a vector of pairwise features, such as the distance between spans, whether they are from the same speaker, etc. The span representation $\mathbf{g}_i$ is initialized as a concatenation of contextual embeddings of the start and end tokens, the weighted sum of all the tokens in the span and a feature vector (learnable width and genre embeddings):

$$\mathbf{g}_i = [\mathbf{x}_{\text{START}(i)}, \mathbf{x}_{\text{END}(i)}, \hat{\mathbf{x}}_i, \phi(i)] \quad (5)$$

The weights to calculate $\hat{\mathbf{x}}_i$ are obtained using an attention mechanism (Bahdanau et al., 2014).

The model also updated the span representations with the weighted sums of their antecedent representations to allow for cluster information to be available for a second iteration of calculating $\mathbf{S}_a$. However, Xu and Choi (2020) have demonstrated that its influence on the performance is negative to marginal.

2.2 Coreference resolution without span representations

Recently, Kirstain et al. (2021) have proposed a modification of the mention ranking model which does not use span representations. Instead, they compute the start and end representations of each subtoken in the sequence:

$$\mathbf{X}^s = \text{RELU}(\mathbf{W}^s \cdot \mathbf{X}) \quad (6)$$
$$\mathbf{X}^e = \text{RELU}(\mathbf{W}^e \cdot \mathbf{X}) \quad (7)$$

The resulting vectors are used to compute mention and antecedent scores for each span without the need to construct explicit span representations. This approach performs competitively with other mention ranking approaches, while being more memory efficient. However, the theoretical size of the antecedent score matrix is still $O(n^4)$, so the authors need to prune the resulting mentions. In our paper, we present an approach that does not exceed the quadratic complexity without the need for mention pruning, while retaining the comparable performance.

3 Model

3.1 Token representation

After obtaining contextual embeddings of all the subtokens of a text, we compute token representations $\mathbf{T}$ as weighted sums of their respective subtokens. The weights are obtained by applying the softmax function to the raw scores of subtokens of each token. The scores are calculated by multiplying the matrix of subtoken embeddings $\mathbf{X}$ by a matrix of learnable weights $\mathbf{W}_a$:

$$\mathbf{A} = \mathbf{W}_a \cdot \mathbf{X} \quad (8)$$

No mention score is computed for the resulting token representations and none of them is subsequently pruned out.

3.2 Coarse-to-fine antecedent pruning

Following Lee et al. (2018) we first use a bilinear scoring function to compute k most likely antecedents for each token. This is useful to further reduce the computational complexity of the model.

$$\mathbf{S}_c = \mathbf{T} \cdot \mathbf{W}_c \cdot \mathbf{T}^\intercal \quad (9)$$

Then we build a pair matrix of $n \times k$ pairs, where each pair is represented as a concatenation of the two token embeddings, their element-wise product and feature embeddings (distance, genre and same/different speaker embeddings). The fine antecedent score is obtained with a feed-forward neural network:

$$s_a(i, j) = \text{FFNN}_a([\mathbf{T}_i, \mathbf{T}_j, \mathbf{T}_i \odot \mathbf{T}_j, \phi]) \quad (10)$$

The resulting coreference score is defined as the sum of the two scores:

$$s(i, j) = s_c(i, j) + s_a(i, j) \quad (11)$$

The candidate antecedent with the highest positive score is assumed to be the predicted antecedent of

each token. If there are no candidates with positive coreference scores, the token is concluded to have no antecedents.

Our model does not use higher-order inference, as Xu and Choi (2020) have shown that it has a marginal impact on the performance. It is also computationally expensive since it repeats the most memory intensive part of the computation during each iteration.

3.3 Span extraction

The tokens that are found to be coreferent to some other tokens are further passed to the span extraction module. For each token, the module reconstructs the span by predicting the most probable start and end tokens in the same sentence. To reconstruct a span headed by a token, all tokens in the same sentence are concatenated with this head token and then passed through a feed-forward neural network followed by a convolution block with two output channels (for start and end scores) and a kernel size of three. The intuition captured by this approach is that the best span boundary is located between a token that is likely to be in the span and a token that is unlikely to be in the span. During inference, tokens to the right of the head token are not considered as potential start tokens and tokens to the left are not considered as potential end tokens.

3.4 Training

To train our model, we first need to transform the OntoNotes 5.0 training dataset to link individual words rather than word spans. To do that, we use the syntax information available in the dataset to reduce each span to its syntactic head. We define a span's head as the only word in the span that depends on a word outside the span or is the head of the sentence. If the number of such words is not one, we choose the rightmost word as the span's head. This allows us to build two training datasets: 1) word coreference dataset to train the coreference module and 2) word-to-span dataset to train the span extraction module.

Following Lee et al. (2017), the coreference module uses negative log marginal likelihood as its base loss function, since the exact antecedents of gold spans are unknown and only the final gold clustering is available:

$$L_{\text{NLML}} = -\log \prod_{i=0}^{N} \sum_{y' \in Y(i) \cap \text{GOLD}(i)} P(y') \quad (12)$$

We propose to use binary cross-entropy as an additional regularization factor:

$$L_{\text{COREF}} = L_{\text{NLML}} + \alpha L_{\text{BCE}} \quad (13)$$

This encourages the model to output higher coreference scores for all coreferent mentions, as the pairs are classified independently from each other. We use the value of $\alpha = 0.5$ to prioritize NLML loss over BCE loss.

The span extraction module is trained using cross-entropy loss over the start and end scores of all words in the same sentence.

We jointly train the coreference and span extraction modules by summing their losses.

4 Experiments

We use the OntoNotes 5.0 test dataset to evaluate our model and compare its performance with the results reported by authors of other mention ranking models. The development portion is used to reason about the importance of the decisions in our model's architecture.

4.1 Experimental setup

We implement our model with PyTorch framework (Paszke et al., 2019). We use the Hugging Face Transformers (Wolf et al., 2020) implementations of BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), SpanBERT (Joshi et al., 2020) and Longformer (Beltagy et al., 2020). All the models were used in their *large* variants. We also replicate the span-level model by Joshi et al. (2020) to use it as a baseline for comparison.

We do not perform hyperparameter tuning and mostly use the same hyperparameters as the ones used in the *independent* model by Joshi et al. (2020), except that we also linearly decay the learning rates.

Our models were trained for 20 epochs on a 48GB nVidia Quadro RTX 8000. The training took 5 hours (except for the Longformer-based model, which needed 17 hours). To evaluate the final model one will need 4 GB of GPU memory.

4.2 Selecting the best model

The results in Table 2 demonstrate that the word-level SpanBERT-based model performs better than its span-level counterpart (JOSHI-REPLICA), despite being much more computationally efficient. The replacement of SpanBERT by RoBERTa or

	MUC			B^3			$CEAF_{\phi4}$			
	P	R	F1	P	R	F1	P	R	F1	Mean F1
Joshi et al. (2020)	85.8	84.8	85.3	78.3	77.9	78.1	76.4	74.2	75.3	79.6
JOSHI-REPLICA	85.2	85.5	85.4	78.2	78.7	78.4	75.4	75.2	75.3	79.7
Xu and Choi (2020)	85.9	85.5	85.7	79.0	78.9	79.0	76.7	75.2	75.9	80.2
Kirstain et al. (2021)	**86.5**	85.1	85.8	**80.3**	77.9	79.1	**76.8**	75.4	76.1	80.3
wl-coref + RoBERTa	84.9	**87.9**	**86.3**	77.4	**82.6**	**79.9**	76.1	**77.1**	**76.6**	**81.0**

Table 1: Best results on the OntoNotes 5.0 test dataset. All the scores have been obtained using the official CoNLL-2012 scorer (Pradhan et al., 2014) and rounded to 1 decimal place. MUC is the metric proposed by Vilain et al. (1995), B^3 was introduced by Bagga and Baldwin (1998) and $CEAF_{\phi4}$ was designed by Luo (2005). The usage of mean F1 score as the aggregate metric was suggested by Pradhan et al. (2012).

	WL F1	SA	SL F1
wl + RoBERTa	83.11	97.16	80.72
-BCE	83.05	97.11	80.60
wl + SpanBERT	82.52	97.13	80.14
-BCE	82.32	97.10	79.99
wl + BERT	77.55	96.20	74.80
wl + Longformer	82.98	97.14	80.56
JOSHI-REPLICA	n/a	n/a	79.74
+RoBERTa	n/a	n/a	78.65

Table 2: Model comparisons on the OntoNotes 5.0 development dataset (best out of 20 epochs). **WL F1** means word-level CoNLL-2012 F1 score, i.e. the coreference metric on the word-level dataset; **SA** is the span extraction accuracy or the percentage of correctly predicted spans; **SL F1** is the span-level CoNLL-2012 F1 score, the basic coreference metric.

	Mentions	Mention pairs	SBC
WL	163,104	62,803,841	475,208
SL	2,263,299	13,970,813,822	n/a

Table 3: The total number of mentions and mention pairs under the word-level and span-level approaches on the OntoNotes 5.0 development dataset. The spans do not cross sentence boundaries. For each mention, only mentions to its left are counted as mention pairs. The total number of span boundary candidates (**SBC**) that need to be considered to predict spans is negligible compared to the number of mention pairs.

	Time	Memory
JOSHI-REPLICA	95	7.4
-HOI	88	7.4
+wl-coref	32	2.9
Joshi et al. (2020)	85	13.3
Xu and Choi (2020) CM	92	12.4
-HOI	52	12.4
Kirstain et al. (2021)	44	4.0

Table 4: Inference time (sec) and peak GPU memory usage (GiB) on the OntoNotes 5.0 development dataset. The measurements were made using nVidia Quadro RTX 8000.

Longformer yields even higher scores for word-level models, though RoBERTa does not improve the performance of the span-level model.

The addition of binary cross-entropy as an extra regularization factor has a slight positive effect on the performance. As it introduces zero overhead on inference time, we keep it in our final model.

All the underlying transformer models (with BERT being the exception) have reached approximately the same quality of span extraction. While it allows the approach to compete with span-level coreference models, there is still room for improvement, which can be seen from the gap between "pure" word-level coreference scores and span-level scores.

4.3 Efficiency

As one can see from the Table 3, the word-level approach needs to consider 14x fewer mentions and 222x fewer mention pairs than the span-level approach. This allows us to keep all the potential mentions while span-level approaches have to rely on pruning techniques, like scoring the mentions and selecting the top-scoring portion of them for further consideration. Such pruning is a compromise between efficiency and accuracy, so removing the necessity for it positively influences both. Moreover, our representation of a mention does not require concatenating start, end and content vectors, which additionally reduces the memory requirements of the model.

Table 4 contains actual efficiency measurements of our baseline (JOSHI-REPLICA) before and after disabling higher-order inference and switching

to word-level coreference resolution. While the baseline relies on aggressive mention pruning and keeps only top λn mentions, where $\lambda = 0.4$ and n is the number of words in the input text, it still requires more time and memory than its word-level counterpart, which considers all possible mentions.

As a reference, we also provide time and memory measurements for other mention ranking models. However, they should not be directly compared, as there are other factors influencing the running time, such as the choice of a framework, the degree of mention pruning, and code quality.

4.4 Test results

Table 1 demonstrates that our model performs competitively with other mention ranking approaches. It can be seen that it has a higher recall than span-level models, because it does not need any mention pruning techniques to be computationally feasible and considers all the potential antecedents for each token in the text.

5 Conclusion

We introduce a word-level coreference model that, while being more efficient, performs competitively with recent systems. This allows for easier incorporation of coreference models into larger natural language processing pipelines. In addition, the separation of coreference resolution and span extraction tasks makes the model's results more interpretable. If necessary, the span prediction module can be replaced by a syntax-aware system to account for non-continuous spans. Such task separation also makes the coreference module independent from the way coreferent spans are marked up in the training dataset. This will potentially simplify using various data sources to build robust coreference resolution systems.

References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *In The First International Conference on Language Resources and Evaluation Workshop on Linguistics Coreference*, pages 563–566.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *Computing Research Repository*, arXiv:1409.0473. Version 7.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *Computing Research Repository*, abs/2004.05150.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. 2018. Neural models for reasoning over multiple mentions using coreference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 42–48, New Orleans, Louisiana. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Yuval Kirstain, Ori Ram, and Omer Levy. 2021. Coreference resolution without span representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 14–19, Online. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692, New Orleans, Louisiana. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Computing Research Repository*, arXiv:1907.11692.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Lesly Miculicich Werlen and Andrei Popescu-Belis. 2017. Using coreference links to improve Spanish-to-English machine translation. In *Proceedings of the 2nd Workshop on Coreference Resolution Beyond OntoNotes (CORBON 2017)*, pages 30–40, Valencia, Spain. Association for Computational Linguistics.

Takumi Ohtani, Hidetaka Kamigaito, Masaaki Nagata, and Manabu Okumura. 2019. Context-aware neural machine translation with coreference information. In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 45–50, Hong Kong, China. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng, and Michael Strube. 2014. Scoring coreference partitions of predicted mentions: A reference implementation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 30–35, Baltimore, Maryland. Association for Computational Linguistics.

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, Jeju Island, Korea. Association for Computational Linguistics.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

Rodrigo Wilkens, Bruno Oberle, and Amalia Todirascu. 2020. Coreference-based text simplification. In *Proceedings of the 1st Workshop on Tools and Resources to Empower People with REAding DIfficulties (READI)*, pages 93–100, Marseille, France. European Language Resources Association.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6953–6963, Online. Association for Computational Linguistics.

Liyan Xu and Jinho D. Choi. 2020. Revealing the myth of higher-order inference in coreference resolution. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8527–8533, Online. Association for Computational Linguistics.