END-TO-END NEURAL COREFERENCE RESOLUTION REVISITED: A SIMPLE YET EFFECTIVE BASELINE

Tuan Manh Lai [1] *Trung Bui* [2] *Doo Soon Kim* [3]

[1] University of Illinois at Urbana-Champaign, USA
[2] Adobe Research, San Jose, CA
[3] Roku Inc., San Jose, CA

ABSTRACT

Since the first end-to-end neural coreference resolution model was introduced, many extensions to the model have been proposed, ranging from using higher-order inference to directly optimizing evaluation metrics using reinforcement learning. Despite improving the coreference resolution performance by a large margin, these extensions add substantial extra complexity to the original model. Motivated by this observation and the recent advances in pre-trained Transformer language models, we propose a simple yet effective baseline for coreference resolution. Even though our model is a simplified version of the original neural coreference resolution model, it achieves impressive performance, outperforming all recent extended works on the public English OntoNotes benchmark. Our work provides evidence for the necessity of carefully justifying the complexity of existing or newly proposed models, as introducing a conceptual or practical simplification to an existing model can still yield competitive results.

Index Terms— Natural Language Processing, Coreference Resolution, Transformer Language Models

1. INTRODUCTION

Coreference resolution is the task of clustering mentions in text that refer to the same entities [1] (Figure 1). As a fundamental task of natural language processing, coreference resolution can be an essential component for many downstream applications. Many traditional coreference resolution systems are pipelined systems, each consists of two separate components: (1) a mention detector for identifying entity mentions from text (2) a coreference resolver for clustering the extracted mentions [2, 3, 4, 5, 6]. These models typically rely heavily on syntatic parsers and use highly engineered mention proposal algorithms.

In 2017, the first end-to-end coreference resolution model named `e2e-coref` was proposed [7]. It outperforms previous pipelined systems without using any syntactic parser or complicated hand-engineered features. Since then, many extensions to the `e2e-coref` model have been introduced, ranging from using higher-order inference to directly optimizing evaluation metrics using reinforcement learning [8, 9, 10, 11, 12, 13, 14, 15] (Figure 2). Despite improving the coreference resolution performance by a large margin, these extensions add a lot of extra complexity to the original model. Motivated by this observation and the recent advances in pretrained Transformer language models, we propose a simple yet effective baseline for coreference resolution. We introduce simplifications to the original `e2e-coref` model, creating a conceptually simpler model for coreference resolution. Despite its simplicity, our model outperforms all aforementioned methods on the public English OntoNotes benchmark. Our work provides evidence for the necessity of carefully justifying the complexity of existing or newly proposed models,

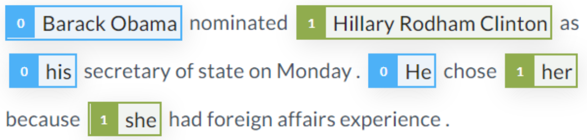


Fig. 1. An example of coreference resolution. There are two coreference chains in this example.

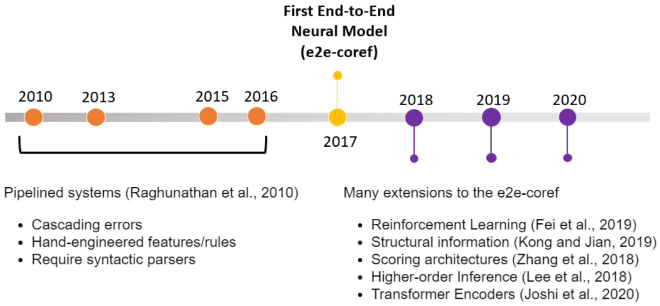

First End-to-End Neural Model (e2e-coref)
2010
2013
2015
2016
2017
2018
2019
2020
Pipelined systems (Raghunathan et al., 2010)
• Cascading errors
• Hand-engineered features/rules
• Require syntactic parsers
Many extensions to the e2e-coref
• Reinforcement Learning (Fei et al., 2019)
• Structural information (Kong and Jian, 2019)
• Scoring architectures (Zhang et al., 2018)
• Higher-order Inference (Lee et al., 2018)
• Transformer Encoders (Joshi et al., 2020)

Fig. 2. An overview of coreference resolution research in the last decade. Pipelined systems were heavily used before the introduction of `e2e-coref`. Since 2017, various extensions to the model have been proposed.

978-1-6654-0540-9/22/$31.00 ©2022 IEEE

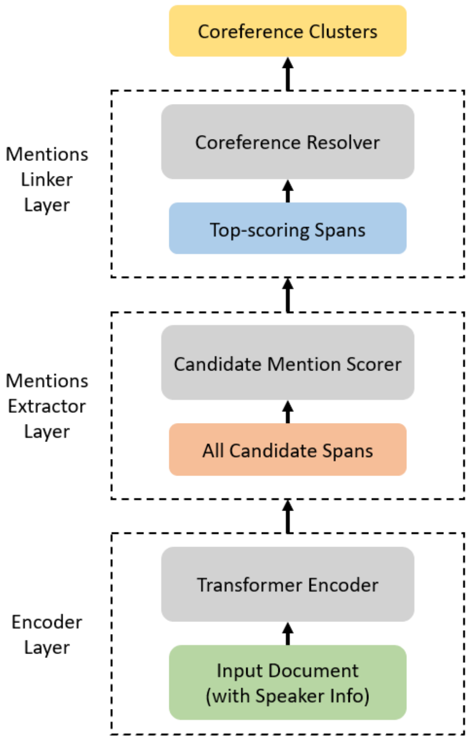


Fig. 3. A high level overview of our proposed model for coreference resolution.

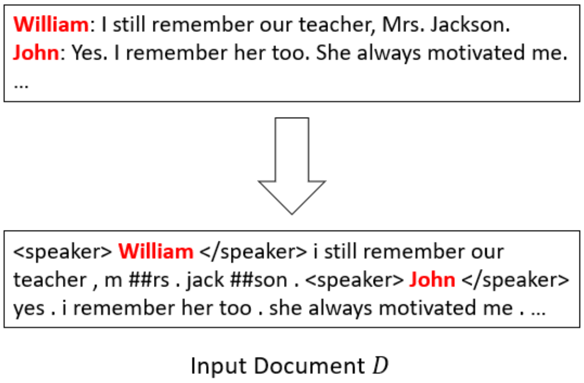
as introducing a conceptual or practical simplification to an existing model can still yield competitive results. The findings of our work agree with the results of several recent studies [16, 17, 18].

2. METHOD

At a high level, our coreference resolution model is similar to the `e2e-coref` model (Figure 3). Given a sequence of tokens from an input document, the model first forms a contextualized representation for each token using a Transformer-based encoder. After that, all the spans (up to a certain length) in the document are enumerated. The model then assigns a score to each candidate span indicating whether the span is an entity mention. A portion of top-scoring spans is extracted and fed to the next stage where the model predicts distributions over possible antecedents for each extracted span. The final coreference clusters can be naturally constructed from the antecedent predictions. In the following subsections, we go into more specific details.

2.1. Notations and Preliminaries

Given an input document $D = (t_1, t_2, ..., t_n)$ consisting of n tokens, the total number of possible text spans is $N = n(n+1)/2$. For each span i, we denote the start and end indices of the span by $\texttt{START}(i)$ and $\texttt{END}(i)$ respectively. We also assume an ordering of the spans based on $\texttt{START}(i)$; spans with the same start index are ordered by $\texttt{END}(i)$. Furthermore, we only consider spans that are entirely within a sentence and limit spans to a max length of L.


William: I still remember our teacher, Mrs. Jackson.
John: Yes. I remember her too. She always motivated me.
...

<speaker> William </speaker> i still remember our teacher , m ##rs . jack ##son . <speaker> John </speaker> yes . i remember her too . she always motivated me . ...

Fig. 4. An example illustrating the strategy of concatenating the speaker's name with the corresponding utterance (assuming the model utilizes WordPiece for tokenization).

Since the speaker information is known to contain useful information for coreference resolution, it has been extensively used in previous works [3, 7, 9, 13, 15]. For example, the original `e2e-coref` model converts speaker information into binary features indicating whether two candidate mentions are from the same speaker. In this work, we employ a more intuitive strategy that directly concatenates the speaker's name with the corresponding utterance [19]. This straightforward strategy is simple to implement and has been shown to be more effective than the feature-based method [19]. Figure 4 illustrates the concatenation strategy.

2.2. Encoder Layer

Given the input document $D = (t_1, t_2, ..., t_n)$, the model simply forms a contextualized representation for each input token, using a Transformer-based encoder such as BERT [20] or SpanBERT [15]. These pretrained language models typically can only run on sequences with at most 512 tokens. Therefore, to encode a long document (i.e., $n > 512$), we split the document into overlapping segments by creating a n-sized segment after every $n/2$ tokens. These segments are then passed on to the Transformer-based encoder independently. The final token representations are derived by taking the token representations with maximum context. Let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)$ be the output of the Transformer encoder.

Note that the `e2e-coref` model uses the GloVe and Turian embeddings [21, 22] and character embeddings produced by 1-dimensional convolution neural networks. From an implementation point of view, it is easier to use a Transformer encoder than combining these traditional embeddings.

For example, the `Transformers` library[1] allows users to experiment with various state-of-the-art Transformer-based models by simply writing few lines of code.

Now, for each span i, its span representation $\mathbf{g}_i$ is defined as:

$$\mathbf{g}_i = \left[\mathbf{x}_{\texttt{START}(i)}, \mathbf{x}_{\texttt{END}(i)}, \hat{\mathbf{x}}_i\right] \tag{1}$$

where $\mathbf{x}_{\texttt{START}(i)}$ and $\mathbf{x}_{\texttt{END}(i)}$ are the boundary representations, consisting of the first and the last token representations of the span i. And $\hat{\mathbf{x}}_i$ is computed using an attention mechanism [23] as follows:

$$\begin{aligned}
\alpha_t &= \text{FFNN}_\alpha(\mathbf{x}_t) \\
\beta_{i,t} &= \frac{\exp(\alpha_t)}{\sum\limits_{j=\texttt{START}(i)}^{\texttt{END}(i)} \exp(\alpha_j)} \\
\hat{\mathbf{x}}_i &= \sum_{j=\texttt{START}(i)}^{\texttt{END}(i)} \beta_{i,j}\, \mathbf{x}_j
\end{aligned} \tag{2}$$

where FFNN_α is a multi-layer feedforward neural network that maps each token-level representation $\mathbf{x}_t$ into an unnormalized attention score. $\hat{\mathbf{x}}_i$ is a weighted sum of token vectors in the span i. Our span representation generation process closely follows that of `e2e-coref`. However, a simplification we make is that we do not include any additional features such as the size of span i in its representation $\mathbf{g}_i$.

2.3. Mentions Extractor Layer

In this layer, we first enumerate all the spans (up to a certain length L) in the document. For each span i, we simply use a feedforward neural network FFNN_m to compute its mention score:

$$s_m(i) = \text{FFNN}_\text{m}(\mathbf{g}_i) \tag{3}$$

After this step, we only keep up to λn spans with the highest mention scores. In previous works, to maintain a high recall of gold mentions, λ is typically set to be 0.4 [7, 9]. These works do not directly train the mention extractor: The mention extractor and the mention linker are jointly trained to only maximize the marginal likelihood of gold antecedent spans.

In coreference resolution datasets such as the OntoNotes benchmark [24], singleton mentions are not explicitly labeled, because the annotations contain only mentions that belong to a coreference chain. However, these annotations of non-singleton mentions can still provide useful signals for training an efficient mention extractor [8]. Thus, we also propose to pre-train our mention extractor using these annotations. In Section 3, we will empirically demonstrate that this pretraining step greatly improves the performance of our mention extractor layer. As a result, we only need to set the parameter λ to be 0.25 in order to maintain a high recall of gold mentions. To this end, the pretraining loss is calculated as follows:

$$\begin{aligned}
\mathcal{L}_\text{detect}(i) &= y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \\
\mathcal{L}_\text{detect} &= -\sum_{i \in \text{S}} \mathcal{L}_\text{detect}(i)
\end{aligned} \tag{4}$$

where $\hat{y}_i = \text{sigmoid}(s_m(i))$, and $y_i = 1$ if and only if the span i is a mention in one of the coreference chains. S is the set of the top scoring spans (and so $|S| \leq \lambda n$).

2.4. Mentions Linker Layer

For each span i extracted by the mention extractor, the mention linker needs to assign an antecedent a_i from all preceding spans or a dummy antecedent ϵ: $a_i \in Y(i) = \{\epsilon, 1, \dots, i{-}1\}$ (the ordering of spans was discussed in Subsection 2.1). The dummy antecedent ϵ represents two possible cases. One case is the span itself is not an entity mention. The other case is the span is an entity mention but it is not coreferent with any previous span extracted by the mention extractor.

The coreference score $s(i,j)$ of two spans i and j is computed as follows:

$$\begin{aligned}
s_a(i,j) &= \text{FFNN}_\text{a}\left(\left[\mathbf{g}_i, \mathbf{g}_j, \mathbf{g}_i \circ \mathbf{g}_j\right]\right) \\
s(i,j) &= s_m(i) + s_m(j) + s_a(i,j)
\end{aligned} \tag{5}$$

where FFNN_a is a feedforward network. $s_m(i)$ and $s_m(j)$ are calculated using Equation 3. The score $s(i,j)$ is affected by three factors: (1) $s_m(i)$, whether span i is a mention, (2) $s_m(j)$, whether span j is a mention, and (3) $s_a(i,j)$ whether j is an antecedent of i. In the special case of the dummy antecedent, $s(i,\epsilon)$ is fixed to 0. In the `e2e-coref` model, when computing $s_a(i,j)$, a vector encoding additional features such as genre information and the distance between the two spans is also used. We do not use such a feature vector when computing $s_a(i,j)$ to simplify the implementation.

We want to maximize the marginal log-likelihood of all antecedents in the correct coreference chain for each mention:

$$\log \prod_{i \in S} \sum_{\hat{y} \in Y(i) \cap \text{GOLD}(i)} P(\hat{y}) \tag{6}$$

where S is the set of the top scoring spans extracted by the mention extractor (i.e., the set of unpruned spans). $\text{GOLD}(i)$ is the set of spans in the gold cluster containing span i. If span i does not belong to any coreference chain or all gold antecedents have been pruned, then $\text{GOLD}(i) = \{\epsilon\}$.

To summarize, we first pre-train the mention extractor to minimize the loss function defined in Eq. 4. We then jointly train the mention extractor and the mention linker to optimize the objective defined in Eq. 6 in an end-to-end manner.

3. EXPERIMENTS AND RESULTS

Dataset and Experiments Setup To evaluate the effectiveness of the proposed approach, we use the CoNLL-

[1] `https://github.com/huggingface/transformers`

	MUC			B-CUBED			CEAF$_{\phi_4}$			
	P	R	F1	P	R	F1	P	R	F1	Avg. F1
e2e-coref [7]	78.4	73.4	75.8	68.6	61.8	65.0	62.7	59.0	60.8	67.2
e2e-coref + Structural info [14]	80.5	73.9	77.1	71.2	61.5	66.0	64.3	61.1	62.7	68.6
c2f-coref + ELMo [9]	81.4	79.5	80.4	72.2	69.5	70.8	68.2	67.1	67.6	73.0
EE + BERT-large [12]	82.6	84.1	83.4	73.3	76.2	74.7	72.4	71.1	71.8	76.6
c2f-coref + BERT-large [13]	84.7	82.4	83.5	76.5	74.0	75.3	74.1	69.8	71.9	76.9
c2f-coref + SpanBERT-large [15]	**85.8**	84.8	85.3	78.3	77.9	78.1	**76.4**	**74.2**	**75.3**	79.6
Simplified e2e-coref (Ours)	85.4	**85.4**	**85.4**	**78.4**	**78.9**	**78.7**	76.1	73.9	75.0	**79.7**

Table 1. Performance on the OntoNotes coreference resolution benchmark.

	Avg. Nb Spans Proposed	Gold Mention Recall
e2e-coref [7]	$\sim$ 200.43 spans / docs	92.7%
Simplified e2e-coref (Ours)	**$\sim$ 141.79 spans / docs**	**95.7%**

Table 2. Proportion of gold mentions covered in the development data by the mention extractor of e2e-coref and our mention extractor.

2012 Shared Task English data [24] which is based on the OntoNotes corpus. This dataset has 2802/343/348 documents for the train/dev/test split. Similar to previous works, we report precision, recall, and F1 of the MUC, B^3, and CEAF$_{\phi_4}$ metrics, and also average the F1 score of all three metrics. We used SpanBERT (spanbert-large-cased) [15] as the encoder. Two different learning rates are used, one for the lower pretrained SpanBERT encoder (5e-05) and one for the upper layers (1e-4). We also use learning rate decay. The number of training epochs is set to be 100. The batch size is set to be 32. We did hyper-parameter tuning using the provided dev set. To train our model, we use two 16GB V100 GPUs and use techniques such as gradient checkpointing and gradient accumulation to avoid running out of GPUs' memory.

Comparison with Previous Methods Table 1 compares our model with several state-of-the-art coreference resolution systems. Overall, our model outperforms the original `e2e-coref` model and also all recent extended works. For example, compared to the variant [c2f-coref + SpanBERT-large], our model achieves higher F1-scores for the MUC and B^3 metrics. Even though our model achieves a slightly lower F1-score for the CEAF$_{\phi_4}$ metric, the overall averaged F1 score of our model is still better. The variant [c2f-coref + SpanBERT-large] is more complex than our method, because it has some other additional components such as coarse-to-fine antecedent pruning and higher-order inference [9, 15].

Recently, a model named `CorefQA` has been proposed [19], and it achieves an averaged F1 score of 83.1 on the English OntoNotes benchmark. The work takes a complete departure from the paradigm used by the `e2e-coref` model, and instead, proposes to formulate the coreference resolution problem as a span prediction task, like in question answering. To achieve its impressive performance, the `CorefQA` model is very computationally expensive. In order to predict coreference clusters for a single document, `CorefQA` needs to run a Transformer-based model on the same document many times (each time a different query is appended to the document).

Analysis on the Performance of the Mention Extractor In our work, the value of the parameter λ for pruning is set to be 0.25. On the other hand, it is set to be 0.4 in the `e2e-coref` model. Table 2 shows the comparison in more detail. Our mention extractor extracts 95.7% of all the gold mentions in the dev set, while the mention extractor of `e2e-coref` extracts only 92.7% of them. By proposing fewer candidate spans, the workload of our mention linker is also reduced.

4. CONCLUSIONS

In this work, we propose a simple yet effective baseline for the task of coreference resolution. Despite its simplicity, our model still outperforms all recent extended works on the English OntoNotes benchmark. In future work, we plan to reduce the computational complexity of our baseline model using compression techniques [25, 26, 27]. We also plan to address the task of event coreference resolution [28, 29].

5. REFERENCES

[1] Vincent Ng, "Supervised noun phrase coreference research: The first fifteen years," in *ACL*, 2010.

[2] Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nate Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher D. Manning, "A multi-pass sieve for coreference resolution," in *EMNLP*, 2010.

[3] Greg Durrett and Dan Klein, "Easy victories and uphill battles in coreference resolution," in *EMNLP*, 2013.

[4] Kevin Clark and Christopher D. Manning, "Entity-centric coreference resolution with model stacking," in *ACL*, 2015.

[5] Sam Wiseman, Alexander M. Rush, and Stuart M. Shieber, "Learning global features for coreference resolution," *ArXiv*, vol. abs/1604.03035, 2016.

[6] Kevin Clark and Christopher D. Manning, "Deep reinforcement learning for mention-ranking coreference models," in *EMNLP*, 2016.

[7] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer, "End-to-end neural coreference resolution," in *EMNLP*, 2017.

[8] Rui Zhang, Cícero Nogueira dos Santos, Michihiro Yasunaga, Bing Xiang, and Dragomir R. Radev, "Neural coreference resolution with deep biaffine attention by joint mention detection and mention clustering," in *ACL*, 2018.

[9] Kenton Lee, Luheng He, and Luke Zettlemoyer, "Higher-order coreference resolution with coarse-to-fine inference," in *NAACL-HLT*, 2018.

[10] Jia-Chen Gu, Zhen-Hua Ling, and Nitin Indurkhya, "A study on improving end-to-end neural coreference resolution," in *CCL*, 2018.

[11] Hongliang Fei, Xu Li, Dingcheng Li, and Ping Li, "End-to-end deep reinforcement learning based coreference resolution," in *ACL*, 2019.

[12] Ben Kantor and Amir Globerson, "Coreference resolution with entity equalization," in *ACL*, 2019.

[13] Mandar Joshi, Omer Levy, Daniel S. Weld, and Luke Zettlemoyer, "Bert for coreference resolution: Baselines and analysis," in *EMNLP/IJCNLP*, 2019.

[14] Fang Kong and Jian Fu, "Incorporating structural information for better coreference resolution," in *IJCAI*, 2019.

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

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

[17] Yuval Kirstain, Ori Ram, and Omer Levy, "Coreference resolution without span representations," *ArXiv*, vol. abs/2101.00434, 2021.

[18] Tuan Lai, Heng Ji, and ChengXiang Zhai, "Bert might be overkill: A tiny but effective biomedical entity linker based on residual convolutional neural networks," *arXiv preprint arXiv:2109.02237*, 2021.

[19] Wenjiong Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li, "Corefqa: Coreference resolution as query-based span prediction," in *ACL*, 2020.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT*, 2019.

[21] Joseph P. Turian, Lev-Arie Ratinov, and Yoshua Bengio, "Word representations: A simple and general method for semi-supervised learning," in *ACL*, 2010.

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

[23] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2015.

[24] Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang, "Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes," in *EMNLP-CoNLL Shared Task*, 2012.

[25] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *ArXiv*, vol. abs/1910.01108, 2019.

[26] Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou, "Mobilebert: a compact task-agnostic bert for resource-limited devices," in *ACL*, 2020.

[27] Tuan Manh Lai, Quan Hung Tran, Trung Bui, and Daisuke Kihara, "A simple but effective bert model for dialog state tracking on resource-limited systems," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 8034–8038.

[28] Tuan Lai, Heng Ji, Trung Bui, Quan Hung Tran, Franck Dernoncourt, and Walter Chang, "A context-dependent gated module for incorporating symbolic semantics into event coreference resolution," *arXiv preprint arXiv:2104.01697*, 2021.

[29] Haoyang Wen, Ying Lin, Tuan Lai, Xiaoman Pan, Sha Li, Xudong Lin, Ben Zhou, Manling Li, Haoyu Wang, Hongming Zhang, et al., "Resin: A dockerized schema-guided cross-document cross-lingual cross-media information extraction and event tracking system," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, 2021, pp. 133–143.