How Good is a Recommender in Machine-Assisted Cross Document Event Coreference Resolution Annotation?

Anonymous ACL submission

Abstract

Annotating cross document event coreference links is a tedious task that requires annotators to have near-oracle knowledge of a document collection. The heavy cognitive load of this task decreases overall annotation quality while inevitably increasing latency. To support annotation efforts, machine-assisted *recommenders* can sample likely coreferent events for a given target event, thus eliminating the burden of examining large numbers of true negative pairs. However, there has been little to no work in evaluating the effectiveness of recommender approaches, particularly for the task of event coreference. To this end, we first create a simulated version of recommender based annotation for cross document event coreference resolution. Then, we adapt an existing method as the model governing recommendations. And finally, we introduce a novel method to assess the simulated recommender by evaluating an annotator-centric Recall-Annotation effort tradeoff.

1 Introduction

Event Coreference Resolution (ECR) is the task of identifying mentions of the same event either within or across documents. We refer to the task of event coreference for a single document as Within-Document Event Coreference Resolution (WDCR), with the task involving multiple documents referred to as Cross Document Event Coreference Resolution (CDCR).

Consider the following excerpts from three related documents (document name in **bold**):

> **39_11ecbplus**: [Peter Capaldi]$_{ARG0}$ will *replace*$_{evt1}$ [Matt Smith]$_{ARG1}$, who announced in June that he was leaving the sci-fi show.

> **39_1ecb**: [Matt Smith]$_{ARG0}$, 26 , will make his debut in 2010, *replacing*$_{evt2}$ [David Tennant]$_{ARG1}$, who leaves at the end of this year.

> **39_5ecbplus**: [Peter Capaldi]$_{ARG0}$ *takes over*$_{evt3}$ [Doctor Who]$_{ARG1}$ . . . [Peter Capaldi]$_{ARG0}$ *stepped into*$_{evt4}$ [Matt Smith's]$_{ARG1}$ soon to be vacant Doctor Who shoes.

The task of WDCR is to determine that event mentions *evt3* and *evt4* are coreferent within document **39_5ecbplus**. The more challenging task of CDCR is to form the two clusters, {*evt1*, *evt3*, *evt4*} and {*evt2*}, by disambiguating events from the three closely related documents.

While manually annotating WDCR links can be difficult, the far greater challenge of CDCR arises from the large number of pairs that need to be examined as a collection grows, as well as to the cognitive load of assessing if two events are actually coreferent (Song et al., 2018; Wright-Bettner et al., 2019). Indeed, an annotator has to examine multiple documents often relying on memory to identify all CDCR links, leading to errors.

To reduce the cognitive burden of CDCR, annotation tools can provide integrated *recommenders* for coreferent links (Pianta et al., 2008; Yimam et al., 2014; Klie et al., 2018). Recommender systems typically store a knowledge base (KB) of annotated documents and then use this KB to suggest likely coreferent candidates for a target event by querying and ranking the candidates. The annotator can then inspect the candidates and choose a coreferent event if present. Figure 1 illustrates a typical workflow for this process.

A recommender's querying and ranking operations are typically driven by machine learning (ML) systems that are trained either actively (Pianta et al., 2008; Klie et al., 2018) or by using batches of annotations (Yimam et al., 2014). While there have been advances in recommendation-based annotations, there is little to no work in evaluating the effectiveness of these systems, particularly in the use case of event coreference. Specifically, both the overall coverage, or recall, of the annotation

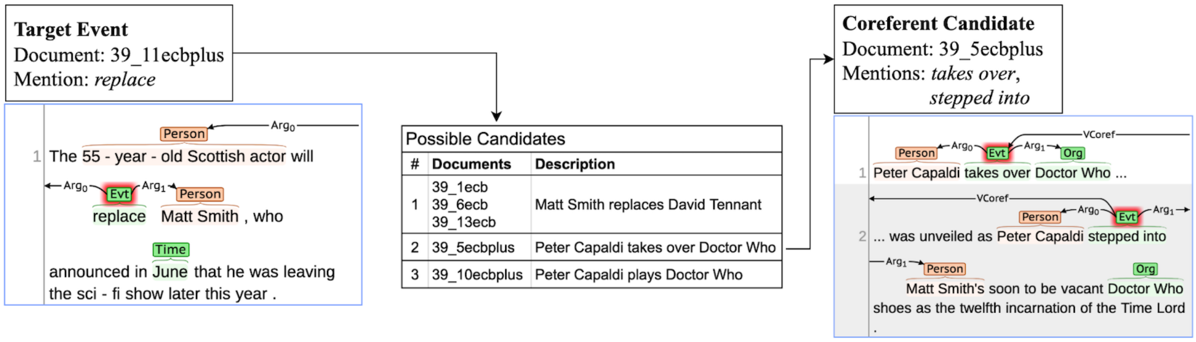

Figure 1: Typical Workflow of Machine-Assisted Annotation of CDCR Links[1]. While annotating document **39_11ecbplus**, the annotator comes across $replace_{evt1}$. The recommender queries and ranks candidates from a KB built over previously annotated documents, then presents them to the annotator in rank order for the annotator to choose from. In this example, the second candidate is the coreferent event in **39_5ecbplus**.

process as well as the degree of annotator effort needed depend on the performance of the recommender. In order to address this shortcoming, we offer the following contributions:

1. We introduce a novel method of recommender-based annotation for CDCR.
2. We compare two existing methods for CDCR (differing widely in their computational costs and portability), by adapting them as the underlying ML models guiding the recommendations.
3. We introduce a novel methodology for assessing the simulated recommender by evaluating an annotator-centric Recall-Annotation effort tradeoff.

2 Related Work

Previous work for ECR is largely based on modeling the probability of coreference between mention pairs. These models are built on supervised classifiers trained using features extracted from the pairs. Earlier work on feature representation uses the broader context of the event mentions to create symbolic linguistic similarities (Lee et al., 2012; Liu et al., 2014; Yang et al., 2015; Araki and Mitamura, 2015). While these models fall short in their performance when compared to current methods, they still are useful in terms of application with limited computational resources.

Most recent work uses a transformer-based language model (LM) like BERT (Devlin et al., 2018) or RoBERTa (Liu et al., 2019) to generate contextualized pair representations of mentions, followed by LM fine-tuning using a coreference scoring objective (Barhom et al., 2019; Cattan et al., 2020; Meged et al., 2020; Zeng et al., 2020; Yu et al., 2020; Caciularu et al., 2021). These methods use scores generated from the coreference scorer to agglomeratively cluster coreferent events. Caciularu et al. (2021) use a modified Longformer (Beltagy et al., 2020) as the underlying LM to generate a document level representation of the event mention pairs. Following the work of Kenyon-Dean et al. (2018), they fine-tune the corresponding CDCR system by training over sampled coreferent and non-coreferent mention pairs. To our knowledge, it is the state of the art system for CDCR.

Over the years, a number of metrics have been proposed to evaluate ECR (Vilain et al., 1995; Bagga and Baldwin, 1998; Luo, 2005; Recasens and Hovy, 2011; Luo et al., 2014; Pradhan et al., 2014). While these metrics do help in assessing the quality of the underlying ML model, an annotator might still want to have an estimate of how much effort is required to identify CDCR links using a recommender. In the remainder of the paper, we attempt to answer this question by quantifying annotation effort and analyzing its relation in terms of finding CDCR links.

3 Dataset

For our experiments, we use the corpus Event Coreference Bank Plus (ECB+; Cybulska and Vossen (2014)), a common choice for assessing CDCR, as well as the experimental setup of Cybulska and Vossen (2015) and gold topic clustering of documents and gold mentions annotations for both training and testing.

We use gold-standard within-document corefer-

[1]Only a subset of possible annotations is shown here.

ence annotations to merge coreferent mentions into within-document event instances. The goal is to group these event instances into what we refer to as cross-document event clusters. We include dataset statistics in the appendix.

4 Recommender Methodology

To simulate a typical human annotation process and isolate the performance of the recommender, we employ incremental clustering where a target event is either merged or added to a store of event clusters. The main steps of the recommender are (1) retrieve candidate clusters for the target event from the existing set of event clusters, (2) rank each candidate based on how similar it is to the target event, and (3) prune lower ranked candidates. Following previous work, we choose a simple retrieval strategy in which we query all the existing event clusters that come from the same topic. For ranking, we adapt methods that work well in an agglomerative clustering setting to a streaming approach.

4.1 Ranking

We investigate two separate methods to drive the ranking of candidates distinguished by their computational cost and likely portability to new domains. We use these methods to generate the average pairwise coreference scores between mentions of the candidate and target events, then use these scores to rank candidates.

Ranking directly with Caciularu et al. (2021) (`CDLM`): In this method, we use the pretrained LM and the fine-tuned CDCR system of Caciularu et al. (2021) to generate pairwise mention scores[2]. This method is expensive as it runs a large LM over all the pairs of mentions (over 100,000) within each topic during prediction.

Ranking with Features (`Regressor`): In the second method, we use a two-layer neural regressor trained over similarity features mostly adopted from Lee et al. (2012). We add one more feature by taking the cosine similarity of contextualized representations of the mentions from the frozen CDLM. To sample for and train the `Regressor`, we follow the methods of Caciularu et al. (2021). Considering the generation of the contextualized representation using `CDLM` to be a simple preprocessing step, the `Regressor` represents a computationally inexpensive method which can be run without dedicated GPUs.

[2]Can be downloaded here

4.2 Pruning

To limit the number of candidates an annotator would have to inspect for each target, we only pick the top k candidates. If k is not an integer (e.g., $k = 2.5$) and the coreferent candidate is not among the top $\lfloor k \rfloor$ (i.e., 2) candidates, we add one more candidate to the top $\lfloor k \rfloor$ with a probability of $k - \lfloor k \rfloor$ (i.e., 0.5). We further prune the candidates by applying a threshold on the coreference score. Section 5.2 describes the threshold tuning process.

Pruning comes at the cost of recall but is a necessary step to reduce annotation effort. Pruning may create the artifact of multiple recommended coreferent candidates for a target event. We detect these cases and merge all coreferent candidates and the target event.

4.3 Simulation

We run the incremental clustering pipeline on the events of the ECB+ development and test sets. For each target event, the recommender retrieves the candidates from the existing clusters and, using each of the methods described earlier, ranks and filters the candidates. We then identify coreferent candidate(s) using ground-truth annotation and merge the target accordingly.

5 Evaluation Methodology

We evaluate the performance of the recommendation methods on three aspects: how well it finds the coreferent links, how "good" the recommendations are, and how much effort it would take to annotate the links using it.

5.1 Recall-Annotation Effort Tradeoff

Recall: To assess the recommender's performance in finding the CDCR links, we use the recall measure of MUC score ($\texttt{MUC}_\texttt{R}$; Vilain et al. (1995)). Since MUC assesses equivalence classes with minimum links between the members, and an incremental clustering pipeline always produces clusters of that kind, $\texttt{MUC}_\texttt{R}$ is a suitable metric for recall here.

Precision: In order to assess the quality of the recommendations, we need a measure of precision. A recommendation is said to be correct if the coreferent candidate is among the candidates and faulty otherwise. We get the ratio of the correct recommendations and present this score as `P`.

Effort: To quantify annotation effort, we count the number of recommended candidates presented by the recommender. A unit effort represents the comparison between a candidate and target that an annotator would have to make in the annotation process. We represent this number as `Comparisons`.

5.2 Analysis

For our analysis, we run the simulation with pruning by varying the k in top k candidates as 2, 2.5, 3, … 5. For pruning with a threshold score, we tune it using the development set by first fixing the k to be 10, and then finding a threshold that achieves 97% recall. The tuned threshold for `CDLM` is 10^{-4} while for the `Regressor`, it is 0.508.

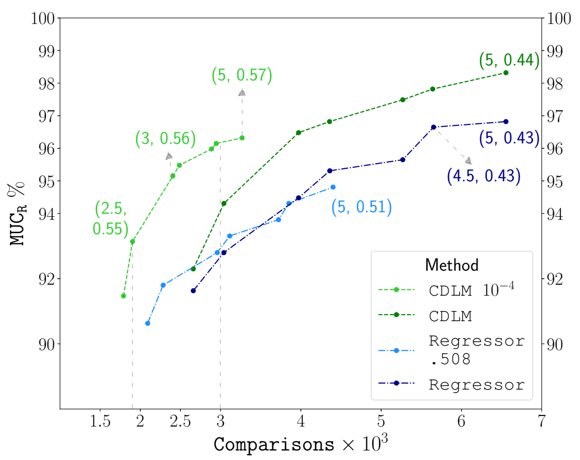

Figure 2: Plot of `Comparisons` vs $\mathtt{MUC}_\mathtt{R}$ analysing simulated Annotation effort using the methods on the Test set of ECB+ Corpus containing 1780 event mentions. The plot is an interpolation over the two measures calculated at various values of k. Select points are labeled in the form (k, `P`).

We calculate $\mathtt{MUC}_\mathtt{R}$ and `Comparisons` for each of the k values and methods with and without using the threshold, collated for visualization in Figure 2, and label some informative points with their respective `P` score. All methods achieve a $\mathtt{MUC}_\mathtt{R}$ greater than 95% when $k = 5$, showing the scores from the two methods are reliable for ranking the candidates.

The `P` score is better for methods that use an additional threshold for pruning, as expected. The `CDLM` method with a threshold clearly performs better than the rest with a score almost reaching 0.6. This means, using this method about 60% of the recommendations lead to finding a coreferent link in the dataset when targeting 97% recall.

From the figure, we can see that some methods are better than others in terms of effort required to achieve a particular recall. For a fixed amount of effort, `CDLM` is better than `Regressor` by 2-4% with or without the use of a threshold. The `CDLM` method greatly cuts the effort using a threshold, but the difference in results for the `Regressor` with and without the threshold is not apparent. The benefits of using non-integer probabilistic k values is clear from the sharp increases in the $\mathtt{MUC}_\mathtt{R}$ with little increase in `Comparisons` at those points for all the methods.

The plot also shows the tremendous effort required to annotate the last 5% of the links. We hypothesize the additional comparisons are in part due to the vast number of singleton clusters in the dataset and also because certain topics have many closely related documents. We leave the analysis of these faulty comparisons for future work.

6 Discussion

Annotating CDCR links has a high cost. While the `Regressor` does not have any additional computing cost, the `CDLM` method incurs the cost associated with high-performance GPUs. Just running the simulation required four hours of computation on a machine with four A100 GPUs at a total cost of about 55 USD. This cost will be much greater if the annotator's own machine needs GPUs. Another issue of using `CDLM` to annotate a new dataset is the generalizability of the model. CDCR annotation guidelines are an evolving research area. The `Regressor` can be easily adapted according to the guidelines through inclusion of additional rules, but it might be difficult for the `CDLM` to adapt without additional annotated data. The ease of application and results similar to those of the `CDLM` method motivates further research into better similarity feature-based models for CDCR annotation recommenders.

7 Conclusion

We introduced a methodology in which a state-of-the-art coreference system can be converted into a recommender system for annotating the same task. We compared two recommenders through a novel evaluation method that answers key questions regarding the quality of the recommender before an annotator uses it. Next steps include testing the transferability of the recommenders for annotating documents of a different domain, and assessing active learning approaches for the task. We also plan to integrate the methodology into an annotation tool like BRAT (Stenetorp et al., 2012), or Inception (Klie et al., 2018) for carrying out annotation on new datasets.

References

Jun Araki and Teruko Mitamura. 2015. Joint event trigger identification and event coreference resolution with structured perceptron. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2074–2080, Lisbon, Portugal. Association for Computational Linguistics.

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.

Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. 2019. Revisiting joint modeling of cross-document entity and event coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4179–4189, Florence, Italy. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Avi Caciularu, Arman Cohan, Iz Beltagy, Matthew E Peters, Arie Cattan, and Ido Dagan. 2021. Cross-document language modeling. *Findings of the Association for Computational Linguistics: EMNLP 2021*.

Arie Cattan, Alon Eirew, Gabriel Stanovsky, Mandar Joshi, and Ido Dagan. 2020. Streamlining cross-document coreference resolution: Evaluation and modeling.

Agata Cybulska and Piek Vossen. 2014. Using a sledgehammer to crack a nut? lexical diversity and event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, May 26-31, 2014*, pages 4545–4552. European Language Resources Association (ELRA).

Agata Cybulska and Piek Vossen. 2015. Translating granularity of event slots into features for event coreference resolution. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 1–10, Denver, Colorado. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Kian Kenyon-Dean, Jackie Chi Kit Cheung, and Doina Precup. 2018. Resolving event coreference with supervised representation learning and clustering-oriented regularization. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 1–10, New Orleans, Louisiana. Association for Computational Linguistics.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9, Santa Fe, New Mexico. Association for Computational Linguistics.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 489–500, Jeju Island, Korea. 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.

Zhengzhong Liu, Jun Araki, Eduard Hovy, and Teruko Mitamura. 2014. Supervised within-document event coreference using information propagation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4539–4544, Reykjavik, Iceland. European Language Resources Association (ELRA).

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, page 25–32, USA. Association for Computational Linguistics.

Xiaoqiang Luo, Sameer Pradhan, Marta Recasens, and Eduard Hovy. 2014. An extension of BLANC to system mentions. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 24–29, Baltimore, Maryland. Association for Computational Linguistics.

Yehudit Meged, Avi Caciularu, Vered Shwartz, and Ido Dagan. 2020. Paraphrasing vs coreferring: Two sides of the same coin. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4897–4907, Online. 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. Pytorch: An imperative style, high-performance deep learning library.

Emanuele Pianta, Christian Girardi, and Roberto Zanoli. 2008. The TextPro tool suite. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

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.

M. Recasens and Eduard Hovy. 2011. Blanc: Implementing the rand index for coreference evaluation. *Natural Language Engineering*, 17:485 – 510.

Zhiyi Song, Ann Bies, Justin Mott, Xuansong Li, Stephanie Strassel, and Christopher Caruso. 2018. Cross-document, cross-language event coreference annotation using event hoppers. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France. Association for Computational Linguistics.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the 6th Conference on Message Understanding*, MUC6 '95, page 45–52, USA. Association for Computational Linguistics.

Kristin Wright-Bettner, Martha Palmer, Guergana Savova, Piet de Groen, and Timothy Miller. 2019. Cross-document coreference: An approach to capturing coreference without context. In *Proceedings of the Tenth International Workshop on Health Text Mining and Information Analysis (LOUHI 2019)*, pages 1–10, Hong Kong. Association for Computational Linguistics.

Bishan Yang, Claire Cardie, and Peter Frazier. 2015. A hierarchical distance-dependent Bayesian model for event coreference resolution. *Transactions of the Association for Computational Linguistics*, 3:517–528.

Seid Muhie Yimam, Chris Biemann, Richard Eckart de Castilho, and Iryna Gurevych. 2014. Automatic annotation suggestions and custom annotation layers in WebAnno. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 91–96, Baltimore, Maryland. Association for Computational Linguistics.

Xiaodong Yu, Wenpeng Yin, and Dan Roth. 2020. Paired representation learning for event and entity coreference.

Yutao Zeng, Xiaolong Jin, Saiping Guan, Jiafeng Guo, and Xueqi Cheng. 2020. Event coreference resolution with their paraphrases and argument-aware embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3084–3094, Barcelona, Spain (Online). International Committee on Computational Linguistics.

A ECB+ Corpus Event Statistics

Table 1 contains the stats for the ECB+ corpus.

	Topics		
	Train	Dev	Test
Topics	25	8	10
Documents	594	196	206
Mentions	3808	1245	1780
Within-doc Event Instances	3102	991	1403
Cross-doc Event Instances	1464	409	805
Cross-doc Event Clusters	411	129	182
Singletons	1053	280	623

Table 1: ECB+ Corpus Statistics for Event Mentions. The Within-doc Event Instances are counted after merging coreferent mentions within documents. Singleton Event Instances are event clusters with only a single event.

B Regressor Model

B.1 Model

The classifier in the `Regressor` method is a 2-layered neural network with four hidden units in the first layer. We use Stocastic Gradient Descent to train the weights with a Binary Cross Entropy loss function and a learning rate of 10^{-5}. We train the model for 100 epochs and use the saved model to run predictions on the development and test set. All the models were implemented using PyTorch (Paszke et al.) and the code is attached with the submission for reproducing the results.

B.2 Feature List

We use a total of 9 features for the method:

lemma match: Binary feature, True if the lemmas of the two mentions are the same.

lemma n-gram overlap: The ratio of overlapping lemma n-grams between mention pairs.

Entities in the sentence overlap: Ratio of overlapping named entities in the sentence. We use gold standard coreference annotations here.

Entities in the Document overlap: Ratio of overlapping named entities in the document. We use gold standard coreference annotations here.

Tf-idf cosine similarity of the documents: The cosine similarity between tf-idf vectors of the documents in which the mentions appear.

Cosine similarity of contextualized representation using `CDLM`: We encode the representation of the mention individually using the entire document as context using the implementation of Caciularu et al. (2021). We then calculate the cosine similarity between the representations of mention pairs.

Word relatedness using Lin Thesaurus for lemmas: 3 features. a) ratio of overlap between the top-50 synonyms from Lin Thesaurus of the lemmas of the pairs. b) binary feature when lemma of the target is among the synonyms of candidate c) binary feature when lemma of candidate is among the synonyms of target.