How Good is the Model in Model-in-the-loop Event Coreference Resolution Annotation?

Shafiuddin Rehan Ahmed$^1$ Abhijnan Nath$^2$ Michael Regan$^3$
Adam Pollins$^1$ Nikhil Krishnaswamy$^2$ James H. Martin$^1$

$^1$University of Colorado, Boulder, CO, USA  $^3$University of Washington, Seattle, WA, USA
$^2$Colorado State University, Fort Collins, CO, USA

{shah7567,james.martin,adpo0729}@colorado.edu
{abhijnan.nath,nkrishna}@colostate.edu
mregan@cs.washington.edu

Abstract

Annotating cross-document event coreference links is a time-consuming and cognitively demanding task that can compromise annotation quality and efficiency. To address this, we propose a model-in-the-loop annotation approach for event coreference resolution, where a machine learning model suggests likely corefering event pairs only. We evaluate the effectiveness of this approach by first simulating the annotation process and then, using a novel annotator-centric Recall-Annotation effort trade-off metric, we compare the results of various underlying models and datasets. We finally present a method for obtaining 97% recall while substantially reducing the workload required by a fully manual annotation process.

1 Introduction

Event Coreference Resolution (ECR) is the task of identifying mentions of the same event either within or across documents. Consider the following excerpts from three related documents:

$e_1$: 55 year old star will $\text{replace}^{m_1}$ Matt Smith, who announced in June that he was leaving the sci-fi show.

$e_2$: Matt Smith, 26, will make his debut in 2010, $\text{replacing}^{m_2}$ David Tennant, who leaves at the end of this year.

$e_3$: Peter Capaldi $\text{takes over}^{m_3}$ Doctor Who . . . Peter Capaldi $\text{stepped into}^{m_4}$ Matt Smith’s soon to be vacant Doctor Who shoes.

$e_1$, $e_2$, and $e_3$ are example sentences from three documents where the event mentions are highlighted and sub-scripted by their respective identifiers ($m_1$ through $m_4$). The task of ECR is to automatically form the two clusters $\{m_1, m_3, m_4\}$, and $\{m_2\}$. We refer to any pair between the mentions of a cluster, e.g., $(m_1, m_3)$ as an ECR link. Any pair formed across two clusters, e.g., $(m_1, m_2)$ is referred to as non-ECR link.

Annotating ECR links can be challenging due to the large volume of mention pairs that must be compared. The annotating task becomes increasingly time-consuming as the number of events in the corpus increases. As a result, this task requires a lot of mental effort from the annotator and can lead to poor quality annotations (Song et al., 2018; Wright-Bettner et al., 2019). Indeed, an annotator has to examine multiple documents simultaneously often relying on memory to identify all the links which can be an error-prone process.

To reduce the cognitive burden of annotating ECR links, annotation tools can provide integrated model-in-the-loop for sampling likely coreferent mention pairs (Pianta et al., 2008; Yimam et al., 2014; Klie et al., 2018). These systems typically store a knowledge base (KB) of annotated documents and then use this KB to suggest relevant candidates. The annotator can then inspect the candidates and choose a coreferent event if present.

The model’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; Bornstein et al., 2020; Yuan et al., 2022) or by using batches of annotations (Yimam et al., 2014). While there have been advances in suggestion-based annotations, there is little to no work in evaluating the effectiveness of these systems, particularly in the use case of ECR.

Specifically, both the overall coverage, or recall, of the annotation process as well as the degree of annotator effort needed depend on the performance of the model. In order to address this shortcoming, we offer the following contributions:

1. We introduce a method of model-in-the-loop annotations for ECR.

2. We compare three existing methods for ECR (differing widely in their computational costs), by adapting them as the underlying ML mod-
3. We introduce a novel methodology for assessing the workflow by simulating the annotations and then 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. Most recent work uses a transformer-based language model (LM) like BERT (Devlin et al., 2018; Liu et al., 2019) to generate joint representations of mention pairs, a method known as cross-encoding. The cross-encoder is fine-tuned using a coreference scoring objective (Barhoom 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 scorer to then agglomeratively cluster coreferent events.

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). An ECR system is evaluated using these metrics to determine how effectively it can find event clusters (recall) and how cleanly separated the clusters are (precision). From the perspective of annotation, it may only be necessary to focus on the system’s recall or its effectiveness in finding ECR links. However, an annotator might still want to know how much effort is required to identify these links in a corpus to estimate their budget. In the remainder of the paper, we attempt to answer this question by first quantifying annotation effort and analyzing its relation with recall of the system.

We implement an iterative model-in-the-loop methodology\(^3\) for annotating ECR links in a corpus containing annotated event triggers. This approach has two main components – (1) the storage and retrieval of annotated event clusters, which are then compared with each new target event, and (2), an ML model that ranks and prunes the sampled candidate clusters by evaluating their semantic similarity to the target mention.

As illustrated in Figure 1, our annotation workflow queries the Annotated Event Store for the target event (m\(_1\)), retrieving three potential coreferent candidates (m\(_2\), m\(_4\) and m\(_*\)). The ranking module (an ECR scorer) then ranks them based on their semantic similarity to m\(_1\). The annotator reviews each candidate one-at-a-time and makes decisions on coreference. m\(_*\) is skipped after finding m\(_4\) as coreferent. The cluster store is then updated based on these decisions.

3 Annotation Methodology

We use the Event Coreference Bank Plus (ECB+; Cybulska and Vossen (2014)) and the Gun Violence Corpus (GVC; Vossen et al. (2018)) for our experiments. The ECB+ is a common choice for assessing ECR, as well as the experimental setup of Cybulska and Vossen (2015) and gold topic clustering of documents and gold mention annotations for both training and testing\(^2\). On the other hand, the GVC offers a more challenging set of exclusively event-specific coreference decisions that require resolving gun violence-related events.

\(^2\)The ECB+ test set has 1,780 event mentions with 5K ECR links among 100K pairwise mentions, while the GVC test set has 1,008 mentions with 2K ECR links in 20K pairs. Full statistics in Table 1 in Appendix A

\(^3\)Utilizing the prodi.gy annotation tool. See Appendix D
use these scores to rank candidates. We use a single RTX 3090 (24 GB) for running our experiments.

**Cross-encoder (CDLM):** In this method, we use the fine-tuned cross-encoder ECR system of Caciularu et al. (2021) to generate pairwise mention scores. Their state of the art system uses a modified Longformer (Beltagy et al., 2020) as the underlying LM to generate document-level representations of the mention pairs (detailed in §B.1). More specifically, we generate a unified representation (Eq. 1) of the mention pair \((m_i, m_j)\) by concatenating the pooled output of the transformer \(E_{CLS}\), the outputs of the individual event triggers \(E_{t_i}, E_{t_j}\), and their element-wise product. Thereafter, pairwise-wise scores are generated for each mention-pair after passing the above representations through a Multi-Layer Perceptron (mlp) (Eq. 2) that was trained using the gold-standard labels for supervision.

\[
LF(m_i, m_j) = \left< E_{CLS}, E_{t_i}, E_{t_j}, E_{m_i} \odot E_{m_j} \right> \quad (1)
\]

\[
CDLM(m_i, m_j) = mlp(LF(m_i, m_j)) \quad (2)
\]

**BERTScore (BERT):** (Zhang et al., 2019) BERT-Score (BS) is an NLP metric that measures pairwise text similarity by exploiting pretrained BERT models. It calculates cosine similarity of token embeddings with inverse document frequency weights to rate token importance and aggregates them into precision, recall, and F1 scores. This method emphasizes semantically significant tokens, resulting in a more accurate similarity score (details in §B.2).

\[
S_{bert}(m) = \left< t_m, [SEP], S_m \right> \quad (3)
\]

\[
BERT(m_i, m_j) = \lambda \ BS(t_{m_i}, t_{m_j}) + (1 - \lambda) \ BS(S_{m_i}, S_{m_j}) \quad (4)
\]

To calculate the BERTScore between the mentions, we first construct a combined sentence \((S_{bert}(m); \text{Shi and Lin} (2019))\) for a mention \(m\) by concatenating the mention text \(t_m\) and its corresponding sentence \(S_m\), as depicted in Equation 3. Subsequently, we compute the BS for each mention pair using \(S_{bert}(m)\) and \(t_m\) separately, then extract the F1 from each. We then take the weighted average of the two scores as shown in Equation 4 as our ranking metric. This process, carried out using the distilbert –base – uncased (Sanh et al., 2019) model, requires approximately seven seconds to complete on each test set.

**Lemma Similarity (Lemma):** The lemma similarity method emulates the annotation process carried out by human annotators when determining coreference based on keyword comparisons between two mentions. To estimate this similarity, we compute the token overlap (Jaccard similarity; JS) between the triggers and sentences containing the respective mentions and take a weighted average of the two similarities (like Eq 4) as shown in Eq 5.

\[
\text{Lemma}(m_i, m_j) = \lambda \ JS(t_{m_i}, t_{m_j}) + (1 - \lambda) \ JS(S_{m_i}, S_{m_j}) \quad (5)
\]

**No Ranking (Random):** For our baseline approach, we employ a method that directly picks the candidate-mention pairs through random sampling and without ranking, providing a reference point for evaluating the effectiveness of the above three ranking techniques.

**3.2 Pruning**

To control the comparisons between candidate and target events, we restrict our selection to the top-\(k\) ranked candidates. To refine our analysis, we employ non-integer \(k\) values, allowing for the inclusion of an additional candidate with a probability equal to the decimal part of \(k\). We vary the values of \(k\) from 2 to 20 on increments of 0.5 and then investigate its relation to recall and effort in §4.

**3.3 Simulation**

To evaluate the ranking methods, we conduct annotation simulations on the events in the ECB+ and GVC development and test sets. These simulations follow the same annotation methodology of retrieving and ranking candidate events for each target but utilize ground-truth for clustering. By executing simulations on different ranking methods and analyzing their performance, we effectively isolate and assess each approach.

**4 Evaluation Methodology**

We evaluate the performance of the model-in-the-loop annotation with the ranking methods through simulation on two aspects: (1) how well it finds the coreferent links, and (2) how much effort it would take to annotate the links using the ranking method. 

\[\text{Lemma}^5 \text{ similarity method emulates the annotation process carried out by human annotators when determining coreference based on keyword comparisons between two mentions. To estimate this similarity, we compute the token overlap (Jaccard similarity; JS) between the triggers and sentences containing the respective mentions and take a weighted average of the two similarities (like Eq 4) as shown in Eq } (5).\]

\[\text{No Ranking (Random): For our baseline approach, we employ a method that directly picks the candidate-mention pairs through random sampling and without ranking, providing a reference point for evaluating the effectiveness of the above three ranking techniques.}\]

\[\text{3.2 Pruning}\]

To control the comparisons between candidate and target events, we restrict our selection to the top-\(k\) ranked candidates. To refine our analysis, we employ non-integer \(k\) values, allowing for the inclusion of an additional candidate with a probability equal to the decimal part of \(k\). We vary the values of \(k\) from 2 to 20 on increments of 0.5 and then investigate its relation to recall and effort in §4.

\[\text{3.3 Simulation}\]

To evaluate the ranking methods, we conduct annotation simulations on the events in the ECB+ and GVC development and test sets. These simulations follow the same annotation methodology of retrieving and ranking candidate events for each target but utilize ground-truth for clustering. By executing simulations on different ranking methods and analyzing their performance, we effectively isolate and assess each approach.

\[\text{4 Evaluation Methodology}\]

We evaluate the performance of the model-in-the-loop annotation with the ranking methods through simulation on two aspects: (1) how well it finds the coreferent links, and (2) how much effort it would take to annotate the links using the ranking method. 

\[\text{5 We use spaCy 3.4 en_core_web_md lemmatizer}\]

\[\text{6} \lambda \text{ is a hyper-parameter to control the weightage of the trigger and sentence similarities in Equations 4 and 5, which we tune using the development set. See Appendix C.}\]
Comparisons × 10^3 (log)

Recall %

Figure 2: Recall and Comparisons achieved upon varying the k for each ranking method in the ECR annotation simulation. The three methods result in significantly fewer comparisons than the no-ranking Random baseline.

4.1 Recall-Annotation Effort Tradeoff

Recall: The recall metric evaluates the percentage of ECR links that are correctly identified by the suggestion model. It is calculated as the ratio of the number of times the true coreferent candidate is among the suggested candidates. The recall error is introduced when the coreferent candidate is erroneously removed based on the top-k value.

Comparisons: A unit effort represents the comparison between a candidate and target mentions that an annotator would have to make in the annotation process. We count the sampled candidates for each target and stop counting when the coreferent candidate is found. For example, the number of comparisons for the target m₁, in Figure 1, is 2 (m₂ and m₄). We count this number for each target event and present the sum as Comparisons.

4.2 Analysis and Discussion

We present an analysis of the various ranking methods employed in our study, highlighting the performance and viability of each approach. We employ the ranking methods on the test sets of ECB+ and GVC. Then, estimate the Recall and Comparisons measures for different k values, and collate them into the plots as shown in Figure 2.

Performance Comparison: The performance improvement of CDLM over BERT and BERT over Lemma can be quantified by examining the graph for the ECB+ and GVC datasets. For example, when targeting a 95% recall for the ECB+ corpus, CDLM provides an almost 100 percent improvement over BERT reducing the number of comparisons to almost half of the latter. However, both CDLM and BERT outperform Lemma by a significant margin while being drastically better than the Random baseline (See Fig. 2). Interestingly, for GVC, the performance gap between CDLM and BERT is quite close, both needing at least three-fourths as many comparisons as the Lemma and crucially outperforming the Random baseline. CDLM’s inconsistent performance on GVC suggests that a corpus-fine-tuned model such as itself is more effective when applied to a dataset similar to the one it was trained on.

Efficiency and Generalizability of BERT: BERT offers a compelling advantage in terms of efficiency, as it can be run on low-compute settings. Moreover, BERT exhibits greater generalizability out-of-the-box when comparing its performance on both the ECB+ and GVC datasets. This makes it an attractive option for ECR annotation task especially when compute resources are limited or when working with diverse corpora.

5 Conclusion

We introduced a model-in-the-loop annotation method for annotating ECR links. We compared three ranking models through a novel evaluation methodology that answers key questions regarding the quality of the model in the annotation loop (namely, recall and effort). Overall, our analysis demonstrates the viability of the models, with CDLM exhibiting the best performance on the ECB+ dataset, followed by BERT and Lemma. The choice of ranking method depends on the specific use case, dataset, and resource constraints, but all three methods offer valuable solutions for different scenarios.

Note that recall is always 100% if no candidates are ever pruned.
Limitations

It is important to note that the approaches presented in this paper have several constraints. Firstly, the methods presented are restricted to English language only, as Lemma necessitates a lemmatizer and, BERT and CDLM rely on models trained exclusively on English corpora. Secondly, the utilization of the CDLM model demands at least a single GPU, posing potential accessibility issues. Thirdly, ECR annotation is susceptible to errors and severe disagreements amongst annotators, which could entail multiple iterations before achieving a gold-standard quality. Lastly, the generated corpora may be biased to the model used during the annotation process, particularly for smaller values of $k$.

Ethics Statement

We use publicly-available datasets, meaning any bias or offensive content in those datasets risks being reflected in our results. By its nature, the Gun Violence Corpus contains violent content that may be troubling for some.

Acknowledgements

We would like to express our sincere gratitude to the anonymous reviewers whose insightful comments and constructive feedback helped to greatly improve the quality of this paper. We gratefully acknowledge the support of U.S. Defense Advanced Research Projects Agency (DARPA) FA8750-18-2-0016-AIDA – RAMFIS: Representations of vectors and Abstract Meanings For Information Synthesis. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA or the U.S. government. We would also like to thank ExplosionAI GmbH for partly funding this work. Finally, we extend our thanks to the BoulderNLP group and the SIGNAL Lab at Colorado State for their valuable input and collaboration throughout the development of this work.

References


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


A ECB+ Corpus Event Statistics

Table 1 contains the detailed statistics for both the ECB+ and the GVC corpora.

B Model Details

B.1 CDLM

The CDLM model, based on the Longformer architecture, cleverly uses a combination of global and local attention for event trigger words and the rest of the document containing those events respectively. More specifically, the Longformer’s increased input capacity of 4096 tokens is utilized
Figure 3: Illustration of Cross-encoding with CDLM from Caciularu et al. (2021).

Table 1: ECB+ and GVC Corpus statistics for event mentions. T/ST = topics/sub-topics, D = documents, M = event mentions, C = clusters, S = singletons. P = unique mention pairs by topic. P+ = mention pairs that are coreferent.

B.2 BERTScore

BERT-Score is an easy-to-use, low-compute scoring metric that can be used to evaluate NLP tasks that require semantic-similarity matching. This task-agnostic metric uses a base language model like BERT to generate token embeddings and leverages the entire sub-word tokenized reference and candidate sentences ($x$ and $\hat{x}$ in Fig. 4) to calculate the pairwise cosine similarity between the sentence pair. It uses a combination of a greedy-matching subroutine to maximize the similarity scores while considering the importance weights.

Figure 4: Illustration of the Recall Measure of BERTScore from Zhang et al. (2019).
normalizing the generated scores based on the IDF (Inverse Document Frequency) of the sub-tokens thereby resulting in more human-readable scores. The latter weighting parameter takes care of rare-word occurrences in sentence pairs that are usually more indicative of how semantically similar such pairs are. In our experiments, we use the distilbert-base-uncased model to get the pairwise coreference scores, consistent with our goal of deploying an annotation workflow suitable for resource-constrained settings. Such lighter and 'distilled' encoders allow us to optimize resources at inference with minimal loss in performance.

C Hyper-parameter Tuning

We employ the evaluation methodology detailed in §4 to determine the optimal value of \( \lambda \) (the weight for trigger similarity and sentence similarity) for both BERT and Lemma approaches. By conducting incremental annotation simulations on the development sets of ECB+ and GVC, we assess \( \lambda \) values ranging from 0 to 1. The recall-effort curve is plotted for each \( \lambda \) value, as shown in Figure 5, allowing us to identify the one that consistently achieves the highest recall with the fewest comparisons. Remarkably, the optimal value for both methods is found to be 0.7, and this value remains consistent across the two datasets and approaches.

D Annotation Interface using Prodigy

Figure 6 illustrates the interface design of the annotation methodology on the popular model-in-the-loop annotation tool - Prodigy (prodi.gy). We use this tool for the simplicity it offers in plugging in the various ranking methods we explained. The recipe for plugging it in to the tool along with other experiment code: github.com/ahmeshaf/model_in_coref.
Figure 5: Trigger and Sentence Similarity weight ($\lambda$) Hyper-parameter tuning on the development sets of ECB+ and GVC. We deduce $\lambda = 0.7$ is optimal for both methods for both datasets.
The strong 6.1 magnitude quake left hundreds more injured as it rocked a region that was devastated by the triggered tsunami of 2004.

The strong 6.1 magnitude quake left hundreds more injured as it rocked a region that was devastated by the quake-triggered tsunami of 2004. The earthquake reduced houses in parts of Aceh to rubble, set off several landslides and badly damaged roads. Rescuers were struggling to find the children still trapped after the mosque collapse in Blang Mancung village, Central Aceh district. "Our search and rescue teams are struggling to evacuate an estimated 14 children still trapped under the rubble," said Subhan Sahara, the head of the local disaster management agency. "I hope they can be found alive but the chances are very slim," he added, explaining they were reading the Koran together when the quake struck. The quake, which hit at a shallow depth of just 10 kilometres, injured more than 200 people and damaged more than 300 houses in Aceh, said national disaster agency spokesman Sutopo Purwo Nugroho. As police and military personnel struggled to reach affected areas, two more aftershocks were recorded.

Figure 6: The model-in-the-loop ECR annotation using the Prodigy Annotation Tool. The target event is on the left and the Candidate cluster is on the right.