Maverick:
Efficient and Accurate Coreference Resolution Defying Recent Trends

Giuliano Martinelli, Edoardo Barba, and **Roberto Navigli**
Sapienza NLP Group, Sapienza University of Rome
{martinelli, barba, navigli}@diag.uniroma1.it

Abstract

Large autoregressive generative models have emerged as the cornerstone for achieving the highest performance across several Natural Language Processing tasks. However, the urge to attain superior results has, at times, led to the premature replacement of carefully designed task-specific approaches without exhaustive experimentation. The Coreference Resolution task is no exception; all recent state-of-the-art solutions adopt large generative autoregressive models that outperform encoder-based discriminative systems. In this work, we challenge this recent trend by introducing Maverick, a carefully designed – yet simple – pipeline, which enables running a state-of-the-art Coreference Resolution system within the constraints of an academic budget, outperforming models with up to 13 billion parameters with as few as 500 million parameters. Maverick achieves state-of-the-art performance on the CoNLL-2012 benchmark, training with up to 0.006x the memory resources and obtaining a 170x faster inference compared to previous state-of-the-art systems. We extensively validate the robustness of the Maverick framework with an array of diverse experiments, reporting improvements over prior systems in data-scarce, long-document, and out-of-domain settings. We release our code and models for research purposes at https://github.com/SapienzaNLP/maverick-coref.

1 Introduction

As one of the core tasks in Natural Language Processing, Coreference Resolution aims to identify and group expressions (called mentions) that refer to the same entity (Karttunen, 1969). Given its crucial role in various downstream tasks, such as Knowledge Graph Construction (Li et al., 2020), Entity Linking (Kundu et al., 2018; Agarwal et al., 2022), Question Answering (Dhingra et al., 2018; Dasigi et al., 2019; Bhattacharjee et al., 2020; Chen and Durrett, 2021), Machine Translation (Stojanovski and Fraser, 2018; Voita et al., 2018; Ohtani et al., 2019; Yehudai et al., 2023) and Text Summarization (Falke et al., 2017; Pasunuru et al., 2021; Liu et al., 2021), *inter alia*, there is a pressing need for both high performance and efficiency. However, recent works in Coreference Resolution either explore methods to obtain reasonable performance optimizing time and memory efficiency (Kirstain et al., 2021; Dobrovolskii, 2021; Otmazgin et al., 2022), or strive to improve benchmark scores regardless of the increased computational demand (Bohnet et al., 2023; Zhang et al., 2023).

Efficient solutions usually rely on discriminative formulations, frequently employing the mention-antecedent classification method proposed by Lee et al. (2017). These approaches leverage relatively small encoder-only transformer architectures (Joshi et al., 2020; Beltagy et al., 2020) to encode documents and build on top of them task-specific networks that ensure high speed and efficiency. On the other hand, performance-centered solutions are nowadays dominated by general-purpose large Sequence-to-Sequence models (Liu et al., 2022; Zhang et al., 2023). A notable example of this formulation, and currently the state of the art in Coreference Resolution, is Bohnet et al. (2023), which proposes a transition-based system that incrementally builds clusters of mentions by generating coreference links sentence by sentence in an autoregressive fashion. Although Sequence-to-Sequence solutions achieve remarkable performance, their autoregressive nature and the size of the underlying language models (up to 13B parameters) make them dramatically slower and memory-demanding compared to traditional encoder-only approaches. This not only makes their usage for downstream applications impractical, but also poses a significant barrier to their accessibility for a large number of users operating within an academic budget.

In this work we argue that discriminative encoder-only approaches for Coreference Reso-

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 13380–13394

lution have not yet expressed their full potential and have been discarded too early in the urge to achieve state-of-the-art performance. In proposing Maverick, we strike an optimal balance between high performance and efficiency, a combination that was missing in previous systems. Our framework enables an encoder-only model to achieve top-tier performance while keeping the overall model size less than one-twentieth of the current state-of-the-art system, and training it with academic resources. Moreover, when further reducing the size of the underlying transformer encoder, Maverick performs in the same ballpark as encoder-only efficiency-driven solutions while improving speed and memory consumption. Finally, we propose a novel incremental Coreference Resolution method that, integrated into the Maverick framework, results in a robust architecture for out-of-domain, data-scarce, and long-document settings.

2 Related Work

We now introduce well-established approaches to neural Coreference Resolution. Specifically, we first delve into the details of traditional discriminative solutions, including their incremental variations, and then present the recent paradigm shift for approaches based on large generative architectures.

2.1 Discriminative models

Discriminative approaches tackle the Coreference Resolution task as a classification problem, usually employing encoder-only architectures. The pioneering works of Lee et al. (2017, 2018) introduced the first end-to-end discriminative system for Coreference Resolution, the Coarse-to-Fine model. First, it involves a mention extraction step, in which the spans most likely to be coreference mentions are identified. This is followed by a mention-antecedent classification step where, for each extracted mention, the model searches for its most probable antecedent (i.e., the extracted span that appears before in the text). This pipeline, composed of mention extraction and mention-antecedent classification steps, has been adopted with minor modifications in many subsequent works, that we refer to as *Coarse-to-Fine* models.

Coarse-to-Fine Models Among the works that build upon the Coarse-to-Fine formulation, Lee et al. (2018), Joshi et al. (2019) and Joshi et al. (2020) experimented with changing the underlying document encoder, utilizing ELMo (Peters et al., 2018), BERT (Devlin et al., 2019) and SpanBERT (Joshi et al., 2020), respectively, achieving remarkable score improvements on the English OntoNotes (Pradhan et al., 2012). Similarly, Kirstain et al. (2021) introduced s2e-coref that reduces the high memory footprint of SpanBERT by leveraging the LongFormer (Beltagy et al., 2020) sparse-attention mechanism. Based on the same architecture, Otmazgin et al. (2023) analyzed the impact of having multiple experts score different linguistically motivated categories (e.g., pronouns-nouns, nouns-nouns, etc.). While the foregoing works have been able to modernize the original Coarse-to-Fine formulation, training their architectures on the OntoNotes dataset still requires a considerable amount of memory.[1] This occurs because they rely on the traditional Coarse-to-Fine pipeline that, as we cover in Section 3.1, has a large memory overhead and is based on manually-set thresholds to regulate memory usage.

Incremental Models Discriminative systems also include incremental techniques. Incremental Coreference Resolution has a strong cognitive grounding: research on the "garden-path" effect shows that humans resolve referring expressions incrementally (Altmann and Steedman, 1988).

A seminal work that proposed an automatic incremental system was that of Webster and Curran (2014), which introduced a clustering approach based on the shift-reduce paradigm. In this formulation, for each mention, a classifier decides whether to SHIFT it into a singleton (i.e., single mention cluster) or to REDUCE it within an existing cluster. The same approach has recently been reintroduced in ICoref (Xia et al., 2020) and longdoc (Toshniwal et al., 2021), which adopted SpanBERT and LongFormer, respectively. In these works the mention extraction step is identical to that of Coarse-to-Fine models. On the other hand, the mention clustering step is performed by using a linear classifier that scores each mention against a vector representation of previously built clusters, in an incremental fashion. This method ensures constant memory usage since cluster representations are updated with a learnable function. In Section 3.2 we present a novel performance-driven incremental method that obtains superior performance and generalization capabilities, in which we adopt a lightweight transformer architecture that retains the mention representations.

[1] Training those models requires at least 32G of VRAM.

2.2 Sequence-to-Sequence models

Recent state-of-the-art Coreference Resolution systems all employ autoregressive generative approaches. However, an early example of Sequence-to-Sequence model, TANL (Paolini et al., 2021), failed to achieve competitive performance on OntoNotes. The first system to show that the autoregressive formulation was competitive was ASP (Liu et al., 2022), which outperformed encoder-only discriminative approaches. ASP is an autoregressive pointer-based model that generates actions for mention extraction (bracket pairing) and then conditions the next step to generate coreference links. Notably, the breakthrough achieved by ASP is not only due to its formulation but also to its usage of large generative models. Indeed, the success of their approach is strictly correlated with the underlying model size, since, when using models with a comparable number of parameters, the performance is significantly lower than encoder-only approaches. The same occurs in Zhang et al. (2023), a fully-seq2seq approach where a model learns to generate a formatted sequence encoding coreference notation, in which they report a strong positive correlation between performance and model sizes.

Finally, the current state-of-the-art system on the OntoNotes benchmark is held by Link-Append (Bohnet et al., 2023), a transition-based system that incrementally builds clusters exploiting a multi-pass Sequence-to-Sequence architecture. This approach incrementally maps the mentions in previously coreference-annotated sentences to system actions for the current sentence, using the same shift-reduce incremental paradigm presented in Section 2.1. This method obtains state-of-the-art performance at the cost of using a 13B-parameter model and processing one sentence at a time, drastically increasing the need for computational power. While the foregoing models ensure superior performance compared to previous discriminative approaches, using them for inference is out of reach for many users, not to mention the exorbitant cost of training them from scratch.

3 Methodology

In this section, we present the Maverick framework: we propose replacing the preprocessing and training strategy of Coarse-to-Fine models with a novel pipeline that improves the training and inference efficiency of Coreference Resolution systems. Furthermore, with the Maverick Pipeline, we eliminate the dependency on long-standing manually-set hyperparameters that regulate memory usage. Finally, building on top of our pipeline, we propose three models that adopt a mention-antecedent classification technique, namely Maverick$_{\text{s2e}}$ and Maverick$_{\text{mes}}$, and a system that is based upon a novel incremental formulation, Maverick$_{\text{incr}}$.

3.1 Maverick Pipeline

The Maverick Pipeline combines i) a novel mention extraction method, ii) an efficient mention regularization technique, and iii) a new mention pruning strategy.

Mention Extraction When it comes to extracting mentions from a document D, there are different strategies to model the probability that a span contains a mention. Several previous works follow the Coarse-to-Fine formulation presented in Section 2.1, which consists of scoring all the possible spans in D. This entails a quadratic computational cost in relation to the input length, which they mitigate by introducing several pruning techniques.

In this work, we employ a different strategy. We extract coreference mentions by first identifying all the possible starts of a mention, and then, for each start, extracting its possible end. To extract start indices, we first compute the hidden representation $(x_1, \ldots, x_n)$ of the tokens $(t_1, \ldots, t_n) \in D$ using a transformer encoder, and then use a fully-connected layer F to compute the probability for each t_i being the start of a mention as:

$$F_{start}(x) = W'_{start}(GeLU(W_{start}x))$$

$$p_{start}(t_i) = \sigma(F_{start}(x_i))$$

with W'_{start}, W_{start} being the learnable parameters, and σ the sigmoid function. For each start of a mention t_s, i.e., those tokens having $p_{start}(t_s) > 0.5$, we then compute the probability of its subsequent tokens t_j, with $s \leq j$, to be the end of a mention that starts with t_s. We follow the same process as that of the mention start classification, but we condition the prediction on the starting token by concatenating the start, x_s, and end, x_j, hidden representations before the linear classifier:

$$F_{end}(x, x') = W'_{end}(GeLU(W_{end}[x, x']))$$

$$p_{end}(t_j|t_s) = \sigma(F_{end}(x_s, x_j))$$

with W'_{end}, W_{end} being learnable parameters. This formulation handles overlapping mentions since,

for each start t_s, we can find multiple ends t_e (i.e., those that have $p_{end}(t_j|t_s) > 0.5$).

Previous works already adopted a linear layer to compute start and end mention scores for each possible mention, i.e., s2e-coref (Kirstain et al., 2021), and LingMess (Otmazgin et al., 2023). However, our mention extraction technique differs from previous approaches since i) we produce two probabilities $(0 < p < 1)$ instead of two unbounded scores and ii) we use the computed start probability to filter out possible mentions, which reduces by a factor of 9 the number of mentions considered compared to existing Coarse-to-Fine systems (Table 1, first row).

Mention Regularization To further reduce the computation demand of this process, in the Maverick Pipeline we use the end-of-sentence (EOS) mention regularization strategy: after extracting the span start, we consider only the tokens up to the nearest EOS as possible mention end candidates.[2] Since annotated mentions never span across sentences, EOS mention regularization prunes the number of mentions considered without any loss of information. While this heuristic was initially introduced in the implementation of Lee et al. (2018), all the recent Coarse-to-Fine have abandoned it in favor of the maximum span-length regularization, which is a manually-set hyperparameter that regulates a threshold to filter out spans that exceed a certain length. This implies a large overhead of unnecessary computations and introduces a structural bias that does not consider long mentions that exceed a fixed length.[3] In our work, we not only reintroduce the EOS mention regularization, but we also study its contribution in terms of efficiency, as reported in Table 1, second row.

Mention Pruning After the mention extraction step, as a result of the Maverick Pipeline, we consider an 18x lower number of candidate mentions for the successive mention clustering phase (Table 1). This step consists of computing, for each mention, the probability of all its antecedents being in the same cluster, incurring a quadratic computational cost. Within the Coarse-to-Fine formulation, this high computational cost is mitigated by considering only the top k mentions according to their probability score, where k is a manually set hyper-

[2]We note that all the well-established Coreference Resolution datasets are sentence-split.

[3]The max-length regularization filters out 196 correctly annotated spans when training on OntoNotes.

	Coarse-to-Fine	Maverick	Δ
Ment. Extraction	Enumeration	(i) Start-End	
	183,577	20,565	-8,92x
(+) Regularization	(+) Span-length	(ii) (+) EOS	
	14,265	777	-18,3x
Ment. Clustering	Top-k	(iii) Pred-only	
	29,334	2,713	-10,81x

Table 1: Comparison between the Coarse-to-Fine pipeline and the Maverick Pipeline in terms of the average number of mentions considered in the mention extraction step (top) and the average number of mention pairs considered in the mention clustering step (bottom). The statistics are computed on the OntoNotes devset, and refer to the hyperparameters proposed in Lee et al. (2018), which were unchanged by subsequent Coarse-to-Fine works, i.e., span-len = 30, top-k = 0.4.

parameter. Since after our mention extraction step we obtain probabilities for a very concise number of mentions, we consider only mentions classified as probable candidates (i.e., those with $p_{end} > 0.5$ and $p_{start} > 0.5$), reducing the number of mention pairs considered by a factor of 10. In Table 1, we compare the previous Coarse-to-Fine formulation with the new Maverick Pipeline.

3.2 Mention Clustering

As a result of the Maverick Pipeline, we obtain a set of candidate mentions $M = (m_1, m_2, \ldots, m_l)$, for which we propose three different clustering techniques: Maverick$_{\text{s2e}}$ and Maverick$_{\text{mes}}$, which use two well-established Coarse-to-Fine mention-antecedent techniques, and Maverick$_{\text{incr}}$, which adopts a novel incremental technique that leverages a light transformer architecture.

Mention-Antecedent models The first proposed model, Maverick$_{\text{s2e}}$, adopts an equivalent mention clustering strategy to Kirstain et al. (2021): given a mention $m_i = (x_s, x_e)$ and its antecedent $m_j = (x_{s'}, x_{e'})$, with their start and end token hidden states, we use two fully-connected layers to model their corresponding representations:

$$F_s(x) = W'_s(GeLU(W_s x))$$

$$F_e(x) = W'_e(GeLU(W_e x))$$

we then calculate their probability to be in the same cluster as:

$$\begin{aligned} p_c(m_i, m_j) = \sigma(&F_s(x_s) \cdot W_{ss} \cdot F_s(x_{s'}) + \\ &F_e(x_e) \cdot W_{ee} \cdot F_e(x_{e'}) + \\ &F_s(x_s) \cdot W_{se} \cdot F_e(x_{e'}) + \\ &F_e(x_e) \cdot W_{es} \cdot F_s(x_{s'})) \end{aligned}$$

with $W_{ss}, W_{ee}, W_{se}, W_{es}$ being four learnable matrices and W_s, W'_s, W_e, W'_e the learnable parameters of the two fully-connected layers.

A similar formulation is adopted in Maverick$_{\text{mes}}$, where, instead of using only one generic mention-pair scorer, we use 6 different scorers that handle linguistically motivated categories, as introduced by Otmazgin et al. (2023). We detect which category k a pair of mentions m_i and m_j belongs to (e.g., if m_i is a pronoun and m_j is a proper noun, the category will be PRONOUN-ENTITY) and use a category-specific scorer to compute p_c. A complete description of the process along with the list of categories can be found in Appendix A.

Incremental model Finally, we introduce a novel incremental approach to tackle the mention clustering step, namely Maverick$_{\text{incr}}$, which follows the standard shift-reduce paradigm introduced in Section 2.1. Differently from the previous neural incremental techniques (i.e., ICoref (Xia et al., 2020) and longdoc (Toshniwal et al., 2021)) which use a linear classifier to obtain the clustering probability between each mention and a fixed length vector representation of previously built clusters, Maverick$_{\text{incr}}$ leverages a lightweight transformer model to attend to previous clusters, for which we retain the mentions' hidden representations. Specifically, we compute the hidden representations $(h_1, \ldots, h_l)$ for all the candidate mentions in M using a fully-connected layer on top of the concatenation of their start and end token representations. We first assign the first mention m_1 to the first cluster $c_1 = (m_1)$. Then, for each mention $m_i \in M$ at step i we obtain the probability of m_i being in a certain cluster c_j by encoding h_i with all the representations of the mentions contained in the cluster c_j using a transformer architecture. We use the first special token ([CLS]) of a single-layer transformer architecture T to obtain the score $S(m_i, c_j)$ of m_i being in the cluster $c_j = (m_f, \ldots, m_g)$ with $f \leq g < i$ as:

$$S(m_i, c_j) = W_c \cdot (ReLU(T_{CLS}(h_i, h_f, \ldots, h_g)))$$

Finally, we compute the probability of m_i belonging to c_j as:

$$p_c(m_i \in c_j | c_j = (m_f, \ldots, m_g)) = \sigma(S(m_i, c_j))$$

We calculate this probability for each cluster c_j up to step i. We assign the mention m_i to the most probable cluster c_j having $p_c(m_i \in c_j) > 0.5$ if one exists, or we create a new singleton cluster containing m_i.

As we show in Sections 5.3 and 5.5, this formulation obtains better results than previous incremental methods, and is beneficial when dealing with long-document and out-of-domain settings.

3.3 Training

To train a Maverick model, we optimize the sum of three binary cross-entropy losses:

$$L_{coref} = L_{start} + L_{end} + L_{clust}$$

Our loss formulation differs from previous transformer-based Coarse-to-Fine approaches, which adopt the marginal log-likelihood to optimize the mention to antecedent score (Lee et al., 2018; Kirstain et al., 2021). Since their formulation "makes learning slow and ineffective, especially for mention detection" (Zhang et al., 2018), we directly optimize both mention extraction and mention clustering with a multitask approach. L_{start} and L_{end} are the start loss and end loss, respectively, of the mention extraction step, and are defined as:

$$L_{start} = \sum_{i=1}^{N} -(y_i \log(p_{start}(t_i)) + (1 - y_i) \log(1 - p_{start}(t_i)))$$

$$L_{end} = \sum_{s=1}^{S} \sum_{j=1}^{E_s} -(y_i \log(p_{end}(t_j | t_s)) + (1 - y_i) \log(1 - p_{end}(t_j | t_s)))$$

where N is the sequence length, S is the number of starts, E_s is the number of possible ends for a start s and $p_{start}(t_i)$ and $p_{end}(t_j | t_s)$ are those defined in Section 3.1.

Finally, L_{clust} is the loss for the mention clustering step. Since we experiment with two different mention clustering formulations, we use a different loss for each clustering technique, namely L^{ant}_{clust} for the mention-antecedent models, i.e., Maverick$_{\text{s2e}}$ and Maverick$_{\text{mes}}$, and L^{incr}_{clust} for the incremental model, i.e., Maverick$_{\text{incr}}$:

$$L^{ant}_{clust} = \sum_{i=1}^{|M|} \sum_{j=1}^{|M|} -(y_i \log(p_c(m_i | m_j)) + (1 - y_i) \log(1 - p_c(m_i | m_j)))$$

$$L^{incr}_{clust} = \sum_{i=1}^{|M|} (\sum_{j=1}^{C_i} -(y_i \log(p_c(m_i \in c_j)) + (1 - y_i) \log(1 - p_c(m_i \in c_j))))$$

Dataset	# Train	# Dev	# Test	Tokens	Mentions	% Sing
OntoNotes	2802	343	348	467	56	0
LitBank	80	10	10	2105	291	19.8
PreCo	36120	500	500	337	105	52.0
GAP	-	-	2000	95	3	-
WikiCoref	-	-	30	1996	230	0

Table 2: Dataset statistics: number of documents in each dataset split, average number of words and mentions per document, and singletons percentage.

where $|M|$ is the number of extracted mentions, C_i is the set of clusters created up to step i, and $p_c(m_i|m_j)$ and $p_c(m_i \in c_j)$ are defined in Section 3.2.

All the models we introduce are trained using teacher forcing. In particular, in the mention token end classification step, we use gold start indices to condition the end tokens prediction, and, for the mention clustering step, we consider only gold mention indices. For Maverick$_{\text{incr}}$, at each iteration, we compare each mention only to previous gold clusters.

4 Experiments Setup

4.1 Datasets

We train and evaluate all the comparison systems on three Coreference Resolution datasets:

OntoNotes (Pradhan et al., 2012), proposed in the CoNLL-2012 shared task, is the de facto standard dataset used to benchmark Coreference Resolution systems. It consists of documents that span seven distinct genres, including full-length documents (broadcast news, newswire, magazines, weblogs, and Testaments) and multiple speaker transcripts (broadcast and telephone conversations).

LitBank (Bamman et al., 2020) contains 100 literary documents typically used to evaluate long-document Coreference Resolution.

PreCo (Chen et al., 2018) is a large-scale dataset that includes reading comprehension tests for middle school and high school students.

Notably, both LitBank and PreCo have different annotation guidelines compared to OntoNotes, and provide annotation for singletons (i.e., single-mention clusters). Furthermore, we evaluate models trained on OntoNotes on three out-of-domain datasets:

- **GAP** (Webster et al., 2018) contains sentences in which, given a pronoun, the model has to choose between two candidate mentions.
- **LitBank$_{\text{ns}}$** and **PreCo$_{\text{ns}}$**, the datasets' test-set where we filter out singleton annotations.
- **WikiCoref** (Ghaddar and Langlais, 2016), which contains Wikipedia texts, including documents with up to 9,869 tokens.

The statistics of the datasets used are shown in Table 2.

4.2 Comparison Systems

Discriminative Among the discriminative systems, we consider c2f-coref (Joshi et al., 2020) and s2e-coref (Kirstain et al., 2021), which build upon the Coarse-to-Fine formulation and adopt different document encoders. We also report the results of LingMess (Otmazgin et al., 2023), which is the previous best encoder-only solution, and f-coref (Otmazgin et al., 2022), which is a distilled version of LingMess. Furthermore, we include CorefQA (Wu et al., 2020), which casts Coreference as extractive Question Answering, and wl-coref (Dobrovolskii, 2021), which first predicts coreference links between words, then extracts mentions spans. Finally, we report the results of incremental systems, such as ICoref (Xia et al., 2020) and longdoc (Toshniwal et al., 2021).

Sequence-to-Sequence We compare our models with TANL (Paolini et al., 2021) and ASP (Liu et al., 2022), which frame Coreference Resolution as an autoregressive structured prediction. We also include Link-Append (Bohnet et al., 2023), a transition-based system that builds clusters with a multi-pass Sequence-to-Sequence architecture. Finally, we report the results of seq2seq (Zhang et al., 2023), a model that learns to generate a sequence with Coreference Resolution labels.

4.3 Maverick Setup

All Maverick models use DeBERTa-v3 (He et al., 2023) as the document encoder. We use DeBERTa because it can model very long input texts effectively (He et al., 2021).[4] Moreover, compared to the LongFormer, which was previously adopted by several token-level systems, DeBERTa ensures a larger input max sequence length (e.g., DeBERTa$_{\text{large}}$ can handle sequences up to 24,528 tokens while LongFormer only 4096) and has shown better performances empirically in our experiments on the OntoNotes dataset. On the other hand, using

[4]This is because its attention mechanism enables its input length to grow linearly with the number of its layers.

DeBERTa to encode long documents is computationally expensive because its attention mechanism incurs a quadratic computational complexity. Whereas this further increases the computational cost of traditional Coarse-to-Fine systems, the Maverick Pipeline enables us to train models that leverage DeBERTa$_{\text{large}}$ on the OntoNotes dataset, without any performance-lowering pruning heuristic. To train our models we use Adafactor (Shazeer and Stern, 2018) as our optimizer, with a learning rate of 3e-4 for the linear layers, and 2e-5 for the pre-trained encoder. We perform all our experiments within an academic budget, i.e., a single RTX 4090, which has 24GB of VRAM. We report more training details in Appendix B.

5 Results

5.1 English OntoNotes

We report in Table 3 the average CoNLL-F1 score of the comparison systems trained on the English OntoNotes, along with their underlying pre-trained language models and total parameters. Compared to previous discriminative systems, we report gains of +2.2 CoNLL-F1 points over LingMess, the best encoder-only model. Interestingly, we even outperform CorefQA, which uses additional Question Answering training data.

Concerning Sequence-to-Sequence approaches, we report extensive improvements over systems with a similar amount of parameters compared to our large models (500M): we obtain +3.4 points compared to ASP (770M), and the gap is even wider when taking into consideration Link-Append (3B) and seq2seq (770M), with +6.4 and +5.6, respectively. Most importantly, Maverick models surpass the performance of all Sequence-to-Sequence transformers even when they have several billions of parameters. Among our proposed methods, Maverick$_{\text{mes}}$ shows the best performance, setting a new state of the art with a score of 83.6 CoNLL-F1 points on the OntoNotes benchmark. More detailed results, including a table with MUC, B^3, and CEAFϕ_4 scores and a qualitative error analysis, can be found in Appendix C.

5.2 PreCo and LitBank

We further validate the robustness of the Maverick framework by training and evaluating systems on the PreCo and LitBank datasets. As reported in Table 4, our models show superior performance when dealing with long documents in a data-scarce setting such as the one LitBank poses. On this dataset, Maverick$_{\text{incr}}$ achieves a new state-of-the-art score of 78.3, and gains +1.0 CoNLL-F1 points compared with seq2seq. On PreCo, Maverick$_{\text{incr}}$ outperforms longdoc, but seq2seq still shows slightly better performance. This is mainly due to the high presence of singletons in PreCo (52% of all the clusters). Our systems, using a mention extraction technique that favors precision rather than recall, are penalized compared to high recall systems such as seq2seq.[5] Among our systems, Maverick$_{\text{incr}}$, leveraging its hybrid architecture, performs better on both PreCo and LitBank.

5.3 Out-of-Domain Evaluation

In Table 5, we report the performance of Maverick systems along with LingMess, the best encoder-only model, when dealing with out-of-domain texts, that is, when they are trained on OntoNotes and tested on other datasets. First of all, we report considerable improvements on the GAP test set, obtaining a +1.2 F1 score compared to the previous state of the art. We also test models on WikiCoref, PreCo$_{\text{ns}}$ and LitBank$_{\text{ns}}$ (Section 4.1). However, since the span annotation guidelines of these corpora differ from the ones used in OntoNotes, in Table 5 we also report the performance using gold mentions, i.e., skipping the mention extraction step (gold column).[6] On the WikiCoref benchmark, we achieve a new state-of-the-art score of 67.2 CoNLL-F1, with an improvement of +4.2 points over the previous best score obtained by LingMess. On the same dataset, when using pre-identified mentions the gap increases to +5.8 CoNLL-F1 points (76.6 vs 82.4). In the same setting, our models obtain up to +7.3 and +10.1 CoNLL-F1 points on Preco$_{\text{ns}}$ and LitBank$_{\text{ns}}$, respectively, compared to LingMess. These results suggest that the Maverick training strategy makes this model more suitable when dealing with pre-identified mentions and out-of-domain texts. This further increases the potential benefits that Maverick systems can bring to many downstream applications that exploit coreference as an intermediate layer, such as Entity Linking (Rosales-Méndez et al., 2020) and Relation Extraction (Xiong et al., 2023; Zeng

[5] Precision and Recall scores are reported in Appendix C.

[6] We do not include autoregressive models because none of the original articles report scores on out-of-domain datasets. We also could not test those models because they do not provide the code to perform mention clustering alone, and performing it with such approaches is not as straightforward as in encoder-only models.

Model	LM	Avg. F1	Params	Training		Inference	
				Time	Hardware	Time	Mem.
Discriminative							
c2f-coref (Joshi et al., 2020)	SpanBERT$_{large}$	79.6	370M	-	1x32G	50s	11.9
ICoref (Xia et al., 2020)	SpanBERT$_{large}$	79.4	377M	40h	1x1080TI-12G	38s	2.9
CorefQA (Wu et al., 2020)	SpanBERT$_{large}$	83.1*	740M	-	1xTPUv3-128G	-	-
s2e-coref (Kirstain et al., 2021)	LongFormer$_{large}$	80.3	494M	-	1x32G	17s	3.9
longdoc (Toshniwal et al., 2021)	LongFormer$_{large}$	79.6	471M	16h	1xA6000-48G	25s	2.1
wl-coref (Dobrovolskii, 2021)	RoBERTa$_{large}$	81.0	360M	5h	1xRTX8000-48G	11s	2.3
f-coref (Otmazgin et al., 2022)	DistilRoBERTa	78.5*	91M	-	1xV100-32G	3s	1.0
LingMess (Otmazgin et al., 2023)	LongFormer$_{large}$	81.4	590M	23h	1xV100-32G	20s	4.8
Sequence-to-Sequence							
ASP (Liu et al., 2022)	FLAN-T5L	80.2	770M	-	1xA100-40G	-	-
	FLAN-T5xxl	82.5	11B	45h	6xA100-80G	20m	-
Link-Append (Bohnet et al., 2023)	mT5xl	78.0^{d}	3B	-	128xTPUv4-32G	-	-
	mT5xxl	83.3	13B	48h	128xTPUv4-32G	30m	-
seq2seq (Zhang et al., 2023)	T5-large	77.2^{d}	770M	-	8xA100-40G	-	-
	T0-11B	83.2	11B	-	8xA100-80G	40m	-
Ours (Discriminative)							
Maverick$_{s2e}$	DeBERTa$_{base}$	81.1	192M	7h	1xRTX4090-24G	6s	1.8
	DeBERTa$_{large}$	83.4	449M	14h	1xRTX4090-24G	13s	4.0
Maverick$_{incr}$	DeBERTa$_{base}$	81.0	197M	21h	1xRTX4090-24G	22s	1.8
	DeBERTa$_{large}$	83.5	452M	29h	1xRTX4090-24G	29s	3.4
Maverick$_{mes}$	DeBERTa$_{base}$	81.4	223M	7h	1xRTX4090-24G	6s	1.9
	DeBERTa$_{large}$	**83.6**	504M	14h	1xRTX4090-24G	14s	4.0

Table 3: Results on the OntoNotes benchmark. We report the Avg. CoNLL-F1 score, the number of parameters, the training time, and the hardware used to train each model. Inference time (sec) and memory (GiB) were calculated on an RTX4090. For Sequence-to-Sequence models we include statistics that are reported in the original papers, since we could not run models locally. (*) indicates models trained on additional resources. (^{d}) indicates scores obtained on the dev set, however, Maverick systems always perform better on the dev than on the test sets. Missing values (-) are not reported in the original paper, and it is not feasible to reproduce them using our limited hardware resources.

Model	PreCo	LitBank
longdoc (Toshniwal et al., 2021)	87.8	77.2
seq2seq (Zhang et al., 2023)	**88.5**	77.3
Maverick$_{s2e}$	87.2	77.6
Maverick$_{incr}$	88.0	**78.3**
Maverick$_{mes}$	87.4	78.0

Table 4: Results of the compared systems on the PreCo and LitBank test-sets in terms of CoNLL-F1 score.

Model	GAP	WikiCoref		PreCo$_{ns}$		LitBank$_{ns}$	
		sys.	gold	sys.	gold	sys.	gold
LingMess	89.6	63.0	76.6	65.1	80.6	64.4	73.9
Maverick$_{s2e}$	91.1	**67.2**	81.5	**67.2**	**87.9**	64.8	83.1
Maverick$_{incr}$	**91.2**	66.8	**82.4**	66.1	86.5	**65.4**	**84.0**
Maverick$_{mes}$	91.1	66.8	82.1	66.1	86.9	65.1	82.8

Table 5: Comparison between LingMess and Maverick systems on GAP, WikiCoref, PreCo$_{ns}$ LitBank$_{ns}$. We report scores using systems prediction (sys.) or passing gold mentions (gold).

et al., 2023), where the mentions are already identified. Among our models, on LitBank$_{ns}$ and WikiCoref, Maverick$_{incr}$ outperforms Maverick$_{mes}$ and Maverick$_{s2e}$, confirming the superior capabilities of the incremental formulation in the long-document setting. Finally, we highlight that the performance gap between using gold mentions and performing full Coreference Resolution is wider when tested on out-of-domain datasets (on average +17%) compared to testing it directly on OntoNotes (83.6 vs 93.6, +10%).[7] This result, obtained on three different out-of-domain datasets, suggests that the difference in annotation guidelines considerably contribute to lower the OOD performances (-7%).

[7] An evaluation of the proposed Maverick models in terms of mention extraction and mention clustering using gold mentions scores can be found in Appendix C.

5.4 Speed and Memory Usage

In Table 3, we include details regarding the training time and the hardware used by each comparison system, along with the measurement of the inference time and peak memory usage on OntoNotes the validation set. Compared to Coarse-to-Fine models, which require 32GB of VRAM, we can train Maverick systems under 18GB. At inference time both Maverick$_{mes}$ and Maverick$_{s2e}$, exploiting DeBERTa$_{large}$, achieve competitive speed and memory consumption compared to wl-coref and s2e-coref. Furthermore, when adopting DeBERTa$_{base}$, Maverick$_{mes}$ proves to be the most efficient approach[8] among those directly trained

[8] In terms of inference peak memory usage and speed.

Model	LM	Score
Maverick$_{\text{s2e}}$		
Maverick$_{\text{s2e}}$	DeBERTa$_{\text{base}}$	81.0
s2e-coref$_{\text{t}}$	DeBERTa$_{\text{base}}$	78.3
Maverick$_{\text{s2e}}$	LongFormer$_{\text{large}}$	80.6
s2e-coref	LongFormer$_{\text{large}}$	80.3
Maverick$_{\text{mes}}$		
Maverick$_{\text{mes}}$	DeBERTa$_{\text{base}}$	81.4
LingMess$_{\text{t}}$	DeBERTa$_{\text{base}}$	78.6
Maverick$_{\text{mes}}$	LongFormer$_{\text{large}}$	81.0
LingMess	LongFormer$_{\text{large}}$	81.4
Maverick$_{\text{incr}}$		
Maverick$_{\text{incr}}$	DeBERTa$_{\text{large}}$	83.5
Maverick$_{\text{prev-incr}}$	DeBERTa$_{\text{large}}$	79.6

Table 6: Comparison between Maverick models and previous techniques. LingMess$_{\text{t}}$ and s2e-coref$_{\text{t}}$ are trained using their official scripts. We use DeBERTa$_{\text{base}}$ because the DeBERTa$_{\text{large}}$ could not fit our hardware when training comparison systems.

on OntoNotes, while, at the same time, attaining performances that are equal to the previous best encoder-only system, LingMess. The only system that shows better inference speed is f-coref, but at the cost of lower performance (-3.0).

Compared to the previous Sequence-to-Sequence state-of-the-art approach, Link-Append, we train our models with 175x less memory requirements. Comparing inference time is more complicated, since we could not run models on our memory-constrained budget. For this reason, we report the inference times from the original articles, and hence times achieved with their high-resource settings. Interestingly, we report as much as 170x faster inference compared to seq2seq, which exploits parallel inference on multiple GPUs, and 85x faster when compared to the more efficient ASP. Among Maverick models, Maverick$_{\text{incr}}$ is notably slower both in inference and training time, as it incrementally builds clusters using multiple steps.

5.5 Maverick Ablation

In Table 6, we compare Maverick$_{\text{s2e}}$ and Maverick$_{\text{mes}}$ models with s2e-coref and LingMess, respectively, using different pre-trained encoders. Interestingly, when using DeBERTa, Maverick systems not only achieve better speed and memory efficiency, but also obtain higher performance compared to the previous systems. When using the LongFormer, instead, their scores are in the same ballpark, showing empirically that the Maverick training procedure better exploits the capabilities of DeBERTa. To test the benefits of our novel incremental formulation, Maverick$_{\text{incr}}$, we also implement a Maverick model with the previously adopted incremental method used in longdoc and ICoref (Section 2.1), which we call Maverick$_{\text{prev-incr}}$. Compared to the previous formulation we report an increase in score of +3.9 CoNLL-F1 points. The improvement demonstrates that exploiting a transformer architecture to attend to all the previously clustered mentions is beneficial, and enables the future usage of hybrid architectures when needed.

As a further analysis of whether the efficiency improvements of our systems stem from using DeBERTa or are attributable to the Maverick Pipeline, we compared the speed and memory occupation of a Maverick system using as underlying encoder either DeBERTa$_{\text{large}}$ or LongFormer$_{\text{large}}$. Our experiments show that using DeBERTa leads to an increase of +77% of memory space and +23% of time to complete an epoch when training on OntoNotes. An equivalent measurement, attributable to the quadratic memory attention mechanism of DeBERTa, was observed for the inference time and memory occupation on the OntoNotes test set. These results highlight the efficiency contribution of the Maverick Pipeline, which is agnostic to the document encoder and can be applied to future coreference systems to ensure higher efficiency.

6 Conclusion

In this work, we challenged the recent trends of adopting large autoregressive generative models to solve the Coreference Resolution task. To do so, we proposed Maverick, a new framework that enables fast and memory-efficient Coreference Resolution while obtaining state-of-the-art results. This demonstrates that the large computational overhead required by Sequence-to-Sequence approaches is unnecessary. Indeed, in our experiments Maverick systems demonstrated that they can outperform large generative models and improve the speed and memory usage of previous best-performing encoder-only approaches. Furthermore, we introduced Maverick$_{\text{incr}}$, a robust multi-step incremental technique that obtains higher performance in the out-of-domain and long-document setting. By releasing our systems, we make state-of-the-art models usable by a larger portion of users in different scenarios and potentially improve downstream applications.

7 Limitations

Our experiments were limited by our resource setting i.e., a single RTX 4090. For this reason, we could not run Maverick using larger encoders, and could not properly test Sequence-to-Sequence models as we did with encoder-only models. Nevertheless, we believe this limitation is a common scenario in many real-world applications that would benefit substantially from our system. We also did not test our formulation on multiple languages, but note that both the methodology behind Maverick and our novel incremental formulation are language agnostic, and thus could be applied to any language.

Acknowledgements


We gratefully acknowledge the support of the PNRR MUR project PE0000013-FAIR.


Roberto Navigli also gratefully acknowledges the support of the CREATIVE project (CRoss-modal understanding and gEnerATIon of Visual and tExtual content), which is funded by the MUR Progetti di Rilevante Interesse Nazionale programme (PRIN 2020). This work has been carried out while Giuliano Martinelli was enrolled in the Italian National Doctorate on Artificial Intelligence run by Sapienza University of Rome.

References

Dhruv Agarwal, Rico Angell, Nicholas Monath, and Andrew McCallum. 2022. Entity linking via explicit mention-mention coreference modeling. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4644–4658, Seattle, United States. Association for Computational Linguistics.

Gerry Altmann and Mark Steedman. 1988. Interaction with context during human sentence processing. *Cognition*, 30(3):191–238.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 79–85, Montreal, Quebec, Canada. Association for Computational Linguistics.

David Bamman, Olivia Lewke, and Anya Mansoor. 2020. An annotated dataset of coreference in English literature. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 44–54, Marseille, France. European Language Resources Association.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Santanu Bhattacharjee, Rejwanul Haque, Gideon Maillette de Buy Wenniger, and Andy Way. 2020. Investigating query expansion and coreference resolution in question answering on bert. *Natural Language Processing and Information Systems*, 12089:47 – 59.

Bernd Bohnet, Chris Alberti, and Michael Collins. 2023. Coreference resolution through a seq2seq transition-based system. *Transactions of the Association for Computational Linguistics*, 11:212–226.

Hong Chen, Zhenhua Fan, Hao Lu, Alan Yuille, and Shu Rong. 2018. PreCo: A large-scale dataset in preschool vocabulary for coreference resolution. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 172–181, Brussels, Belgium. Association for Computational Linguistics.

Jifan Chen and Greg Durrett. 2021. Robust question answering through sub-part alignment. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1251–1263, Online. Association for Computational Linguistics.

Pradeep Dasigi, Nelson F. Liu, Ana Marasović, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5925–5932, Hong Kong, China. Association for Computational Linguistics.

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.

Vladimir Dobrovolskii. 2021. Word-level coreference resolution. In *Proceedings of the 2021 Conference*

on Empirical Methods in Natural Language Processing, pages 7670–7675, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tobias Falke, Christian M. Meyer, and Iryna Gurevych. 2017. Concept-map-based multi-document summarization using concept coreference resolution and global importance optimization. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 801–811, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Abbas Ghaddar and Phillippe Langlais. 2016. WikiCoref: An English coreference-annotated corpus of Wikipedia articles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 136–142, Portorož, Slovenia. European Language Resources Association (ELRA).

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention.

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.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 2019. BERT for coreference resolution: Baselines and analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5803–5808, Hong Kong, China. Association for Computational Linguistics.

Lauri Karttunen. 1969. Discourse referents. In *International Conference on Computational Linguistics COLING 1969: Preprint No. 70*, Sånga Säby, Sweden.

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.

Gourab Kundu, Avi Sil, Radu Florian, and Wael Hamza. 2018. Neural cross-lingual coreference resolution and its application to entity linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 395–400, Melbourne, Australia. 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.

Manling Li, Alireza Zareian, Ying Lin, Xiaoman Pan, Spencer Whitehead, Brian Chen, Bo Wu, Heng Ji, Shih-Fu Chang, Clare Voss, Daniel Napierski, and Marjorie Freedman. 2020. GAIA: A fine-grained multimedia knowledge extraction system. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 77–86, Online. Association for Computational Linguistics.

Tianyu Liu, Yuchen Eleanor Jiang, Nicholas Monath, Ryan Cotterell, and Mrinmaya Sachan. 2022. Autoregressive structured prediction with language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 993–1005, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zhengyuan Liu, Ke Shi, and Nancy Chen. 2021. Coreference-aware dialogue summarization. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 509–519, Singapore and Online. Association for Computational Linguistics.

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.

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.

Shon Otmazgin, Arie Cattan, and Yoav Goldberg. 2022. F-coref: Fast, accurate and easy to use coreference resolution. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 48–56, Taipei, Taiwan. Association for Computational Linguistics.

Shon Otmazgin, Arie Cattan, and Yoav Goldberg. 2023. LingMess: Linguistically informed multi expert scorers for coreference resolution. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2752–2760, Dubrovnik, Croatia. Association for Computational Linguistics.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. *arXiv preprint arXiv:2101.05779*.

Ramakanth Pasunuru, Mengwen Liu, Mohit Bansal, Sujith Ravi, and Markus Dreyer. 2021. Efficiently summarizing text and graph encodings of multi-document clusters. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4768–4779, Online. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. 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.

Henry Rosales-Méndez, Aidan Hogan, and Barbara Poblete. 2020. Fine-grained entity linking. *Journal of Web Semantics*, 65:100600.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4596–4604. PMLR.

Dario Stojanovski and Alexander Fraser. 2018. Coreference and coherence in neural machine translation: A study using oracle experiments. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 49–60, Brussels, Belgium. Association for Computational Linguistics.

Shubham Toshniwal, Patrick Xia, Sam Wiseman, Karen Livescu, and Kevin Gimpel. 2021. On generalization in coreference resolution. In *Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 111–120, Punta Cana, Dominican Republic. 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*.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1264–1274, Melbourne, Australia. Association for Computational Linguistics.

Kellie Webster and James R. Curran. 2014. Limited memory incremental coreference resolution. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2129–2139, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics*, 6:605–617.

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.

Patrick Xia, João Sedoc, and Benjamin Van Durme. 2020. Incremental neural coreference resolution in constant memory. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8617–8624, Online. Association for Computational Linguistics.

Yiyun Xiong, Mengwei Dai, Fei Li, Hao Fei, Bobo Li, Shengqiong Wu, Donghong Ji, and Chong Teng. 2023. Dialogre c+: An extension of dialogre to investigate how much coreference helps relation extraction in dialogs. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 222–234. Springer.

Asaf Yehudai, Arie Cattan, Omri Abend, and Gabriel Stanovsky. 2023. Evaluating and improving the coreference capabilities of machine translation models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 980–992, Dubrovnik, Croatia. Association for Computational Linguistics.

Daojian Zeng, Chao Zhao, Chao Jiang, Jianling Zhu, and Jianhua Dai. 2023. Document-level relation extraction with context guided mention integration and inter-pair reasoning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Rui Zhang, Cícero Nogueira dos Santos, Michihiro Yasunaga, Bing Xiang, and Dragomir Radev. 2018. Neural coreference resolution with deep biaffine attention by joint mention detection and mention clustering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 102–107, Melbourne, Australia. Association for Computational Linguistics.

Wenzheng Zhang, Sam Wiseman, and Karl Stratos. 2023. Seq2seq is all you need for coreference resolution. *arXiv preprint arXiv:2310.13774*.

A Multi-Expert Scorers

In Maverick$_{\text{mes}}$, the final coreference score between two spans is calculated using 6 linguistically motivated multi-expert scorers. This approach was introduced by Otmazgin et al. (2023), which demonstrated that linguistic knowledge and symbolic computation can still be used to improve results on the OntoNotes benchmark. In Maverick$_{\text{mes}}$ we adopt this approach on top of the Maverick Pipeline. We use the same set of categories, namely:

1. PRON-PRON-C. Compatible pronouns based on their attributes, such as gender or number (e.g. *(I, I), (I, my) (she, her)*).
2. PRON-PRON-NC, Incompatible pronouns (e.g. *(I, he), (She, my), (his, her)*).
3. ENT-PRON. Pronoun and non-pronoun (e.g. *(George, he), (CNN, it), (Tom Cruise, his)*).
4. MATCH. Non-pronoun spans with the same content words (e.g. *Italy, Italy*).
5. CONTAINS. One contains the other (e.g. *(Barack Obama, Obama)*).
6. OTHER. The Other pairs.

To detect pronouns we use string match with a full list of English pronouns.

To perform mention clustering, we dedicate a mention-pair scorer for each of those categories. Specifically, for the mention $m_i = (x_s, x_e)$ and its antecedent $m_j = (x_{s'}, x_{e'})$, with their start and end token hidden states, we first detect their category k_g using pattern matching on their spans of texts. Then we compute their start and end representations, using the specific fully-connected layers for the category k_g:

$$F_s^{k_g}(x) = W'_{k_{g,s}}(GeLU(W_{k_{g,s}}x))$$

$$F_e^{k_g}(x) = W'_{k_{g,e}}(GeLU(W_{k_{g,e}}x))$$

The probability $p_c^{k_g}$ of m_i and $m_j$is then calculated as:

$$\begin{aligned} p_c^{k_g}(m_i, m_j) = \sigma(&F_s^{k_g}(x_s) \cdot W_{ss} \cdot F_s^{k_g}(x_{s'}) + \\ &F_e^{k_g}(x_e) \cdot W_{ee} \cdot F_e^{k_g}(x_{e'}) + \\ &F_s^{k_g}(x_s) \cdot W_{se} \cdot F_e^{k_g}(x_{e'}) + \\ &F_e^{k_g}(x_e) \cdot W_{es} \cdot F_s^{k_g}(x_{s'})) \end{aligned}$$

With $W_{ss}, W_{ee}, W_{se}, W_{es}$ being four learnable matrices and $W'_{k_{g,e}}, W'_{k_{g,s}}, W_{k_{g,e}}, W'_{k_{g,s}}$ the learnable parameters of the two fully-connected layers. In this way, each mention-pair scorer learns to model the probability for its specific linguistic categories.

B Training details

B.1 Datasets

We report technical details of the adopted datasets.

- **OntoNotes** contains several items of metadata information for each document, such as genre, speakers, and constituent graphs. Following previous works, we incorporate the speaker's name into the text whenever there is a change in speakers for datasets that include this metadata.
- **LitBank** contains 100 literary documents and is available in 10 different cross-validation folds. Our train, dev, and test splits refer to the first cross-validation fold, LB$_0$. We report comparison systems results on the same splits. Since training DeBERTa$_{\text{large}}$ is particularly computationally expensive, as introduced in Section 4.3, we train on LitBank by splitting in half each LitBank training document.
- The authors of **PreCo** have not released their official test set. To evaluate our models consistently with previous approaches, we use the official 'dev' split as our test set and retain the last 500 training examples for model validation.

Model	LM	MUC			B^3			CEAFϕ_4			Avg.
		P	R	F1	P	R	F1	P	R	F1	F1
Discriminative											
e2e-coref (Lee et al., 2017)	-	78.4	73.4	75.8	68.6	61.8	65.0	62.7	59.0	60.8	67.2
c2f-coref (Lee et al., 2018)	ELMo	81.4	79.5	80.4	72.2	69.5	70.8	68.2	67.1	67.6	73.0
c2f-coref (Joshi et al., 2019)	$\text{BERT}_{\text{large}}$	84.7	82.4	83.5	76.5	74.0	75.3	74.1	69.8	71.9	76.9
c2f-coref (Joshi et al., 2020)	$\text{SpanBERT}_{\text{large}}$	85.8	84.8	85.3	78.3	77.9	78.1	76.4	74.2	75.3	79.6
ICoref (Xia et al., 2020)	$\text{SpanBERT}_{\text{large}}$	85.7	84.8	85.3	78.1	77.5	77.8	76.3	74.1	75.2	79.4
CorefQA (Wu et al., 2020)	$\text{SpanBERT}_{\text{large}}$	88.6	87.4	88.0	82.4	82.0	82.2	79.9	78.3	79.1	83.1*
longdoc (Toshniwal et al., 2021)	$\text{LongFormer}_{\text{large}}$	85.5	85.1	85.3	78.7	77.3	78.0	74.2	76.5	75.3	79.6
s2e-coref Kirstain et al. (2021)	$\text{LongFormer}_{\text{large}}$	86.5	85.1	85.8	80.3	77.9	79.1	76.8	75.4	76.1	80.3
wl-coref (Dobrovolskii, 2021)	$\text{RoBERTa}_{\text{large}}$	84.9	87.9	86.3	77.4	82.6	79.9	76.1	77.1	76.6	81.0
f-coref (Otmazgin et al., 2022)	DistilRoberta	85.0	83.9	84.4	77.6	75.5	76.6	74.7	74.3	74.5	78.5*
LingMess (Otmazgin et al., 2023)	$\text{LongFormer}_{\text{large}}$	88.1	85.1	86.6	82.7	78.3	80.5	78.5	76.0	77.3	81.4
Sequence-to-Sequence											
TANL (Paolini et al., 2021)	T5_{base}	-	-	81.0	-	-	69.0	-	-	68.4	72.8
ASP (Liu et al., 2022)	$\text{FLAN-T5}_{\text{XXL}}$	86.1	88.4	87.2	80.2	83.2	81.7	78.9	78.3	78.6	82.5
Link-Append (Bohnet et al., 2023)	mT5_{XXL}	87.4	88.3	87.8	81.8	83.4	82.6	79.1	79.9	79.5	83.3
seq2seq (Zhang et al., 2023)	T0_{XXL}	86.1	89.2	87.6	80.6	84.3	82.4	78.9	80.1	79.5	83.2
Ours (Discriminative)											
$\text{Maverick}_{\text{s2e}}$	$\text{DeBERTa}_{\text{large}}$	87.1	88.6	87.9	81.7	83.8	82.7	80.8	78.7	79.7	83.4
$\text{Maverick}_{\text{incr}}$	$\text{DeBERTa}_{\text{large}}$	87.6	88.1	87.9	82.7	82.6	82.7	80.3	79.3	79.8	83.5
$\text{Maverick}_{\text{mes}}$	$\text{DeBERTa}_{\text{large}}$	87.5	88.5	88.0	82.2	83.5	82.8	80.4	79.3	79.9	**83.6**

Table 7: Results on the OntoNotes test set. The average CoNLL-F1 score of MUC, B^3, and CEAFϕ_4 is the main evaluation criterion. $*$ marks models using additional/different training data.

B.2 Setup

All our experiments are developed using the Pytorch-Lightning framework.[9] For each Maverick model, we load the pre-trained weights for the *base*[10] and *large*[11] version of DeBERTa−v3 from the Huggingface Transformers library (Wolf et al., 2020). We accumulate gradients every 4 steps and use a gradient clipping value of 1.0. We adopt a linear learning rate scheduler a warm-up of 10% of the total steps check validation scores every 50% of the total number of steps per epoch. We select our model upon validation of Avg. CoNLL-f1 score and use patience of 20.

C Additional Results

In Table 7 we report the performance of models according to the standard Coreference Resolution metrics: MUC (Vilain et al., 1995), B^3(Bagga and Baldwin, 1998), CEAFϕ_4 (Luo, 2005) and AVG CoNLL-F1. Scores for Maverick models are computed using the official CoNLL coreference scorer.[12]

System	Ment. Clustering	Ment. Extraction
$\text{Maverick}_{\text{s2e}}$	89.4	93.5
$\text{Maverick}_{\text{incr}}$	89.2	94.2
$\text{Maverick}_{\text{mes}}$	89.6	93.7

Table 8: Mention extraction (F1) and mention clustering (CoNLL-F1) scores on the OntoNotes validation set.

C.1 Error Analysis

To better understand the quality of Maverick predictions, we conduct an error analysis on our best system trained on OntoNotes, $\text{Maverick}_{\text{mes}}$. In Table 8, we report the score of performing only mention extraction (F1) or mention clustering with gold mention (CoNLL-F1) with our systems. Our results highlight that our models have strong capabilities of clustering pre-identified mentions, but limited performance in the identification of correct spans. We investigated this phenomenon by conducting a qualitative evaluation of the outputs of our best system, $\text{Maverick}_{\text{mes}}$, and found out that OntoNotes contains several annotation errors. We report examples of errors in Table 9. The main inconsistency we found in the gold test set is that many documents have incomplete annotations, which directly correlates with the mention extraction error.

[9] https://lightning.ai

[10] https://huggingface.co/microsoft/deberta-v3-base

[11] https://huggingface.co/microsoft/deberta-v3-large

[12] https://conll.github.io/reference-coreference-scorers

Type	Text
Ex. 1	
Gold	Nine people were injured in Gaza when gunmen **[opened]**$_1$**[fire]**$_2$on an Israeli bus. The passengers were off - duty Israeli security workers. Witnesses say **[the shots]**$_2$came from **[the Palestinian international airport]**$_3$. Israeli Prime Minister Ehud Barak **[closed]**$_4$down **[the two - year - old airport]**$_3$in response to **[the incident]**$_1$. **[Palestinians]**$_5$criticized **[the move]**$_4$. **[hey]**$_5$regard **[the airport]**$_3$as a symbol of emerging statehood.
Output	**[Nine people]**$_1$were injured in Gaza when gunmen opened fire on an Israeli bus. **[The passengers]**$_1$were off - duty Israeli security workers. Witnesses say the shots came from **[the Palestinian international airport]**$_2$. Israeli Prime Minister Ehud Barak **[closed]**$_3$down **[the two - year - old airport]**$_2$in response to the incident. **[Palestinians]**$_4$criticized **[the move]**$_3$. **[They]**$_4$regard **[the airport]**$_2$as a symbol of emerging statehood.
Ex. 2	
Gold	**[Mr. Seelenfreund]**$_1$is **[executive vice president and chief financial officer of [McKesson]**$_3$**]**$_2$- and will continue in **[those roles]**$_2$. **[PCS]**$_4$also named Rex R. Malson, 57, executive vice president at McKesson,- as a director, filling the seat vacated by Mr. Field. Messrs. Malson and Seelenfreund are directors of **[McKesson, which has an 86% stake in [PCS]**$_4$**]**$_3$.
Output	**[Mr. Seelenfreund]**$_1$is **[executive vice president and chief financial officer of [McKesson]**$_3$**]**$_2$ and will continue in [those roles]$_2$. **[PCS]**$_4$also named **[Rex R. Malson, 57, executive vice president at [McKesson]**$_3$**,]**$_5$- as a director, filling the seat vacated by Mr. Field. Messrs. **[Malson]**$_5$and **[Seelenfreund]**$_1$are directors of **[McKesson, which has an 86 % stake in [PCS]**$_4$**]**$_3$.
Ex. 3	
Gold	The Second U.S. Circuit Court of Appeals opinion in the Arcadian Phosphate case - did not repudiate the position **[Pennzoil Co.]**$_1$took in **[its]**$_1$dispute with **[Texaco]**$_2$, - contrary to your Sept. 8 article " Court Backs **[Texaco]**$_2$'s View in **[Pennzoil]**$_1$Case – Too Late. " The fundamental rule of contract law applied to **[both cases]**$_3$was that courts will not enforce - **[agreements to [which]**$_4$**the parties did not intend to be bound]**$_4$. In the Pennzoil / Texaco litigation, **[the courts]**$_5$found **[Pennzoil]**$_1$and Getty Oil intended to be bound; in Arcadian Phosphates **[they]**$_5$found there was no intention to be bound.
Output	The Second U.S. Circuit Court of Appeals opinion in **[the Arcadian Phosphate case]**$_1$ - did not repudiate the position **[Pennzoil Co.]**$_2$took in **[[[its]**$_2$**dispute with [Texaco]**$_4$**]**$_3$**, -** **contrary to your Sept. 8 article " Court Backs [Texaco 's]**$_4$**View in [[Pennzoil]**$_2$**Case]**$_3$**]**$_3$– Too Late . " **[[The fundamental rule of contract law]**$_5$**applied to both cases]**$_5$was that courts will not enforce - agreements to which the parties did not intend to be bound. In **[the [Pennzoil]**$_2$ **/ [Texaco]**$_4$**litigation]**$_3$, **[the courts]**$_6$found **[Pennzoil]**$_2$and Getty Oil intended to be bound; in **[Arcadian Phosphates]**$_1$**[they]**$_6$found there was no intention to be bound.
Ex. 4	
Gold	... **[Harry]**$_1$has avoided all that by living in a Long Island suburb with **[his]**$_1$wife, who 's so addicted to soap operas and mystery novels she barely seems to notice when **[her husband[**disappears for drug - seeking forays into Manhattan.
Output	... **[Harry]**$_1$has avoided all that by living in a Long Island suburb with **[[his]**$_1$**wife,** **who 's so addicted to soap operas and mystery novels** **[she]**$_2$**barely seems to notice when [[her]**$_2$**husband]**$_1$**disappears for drug - seeking forays into Manhattan]**$_2$.

Table 9: OntoNotes test set annotation errors examples.