Graph Refinement for Coreference Resolution

Lesly Miculicich [*]
Microsoft
leslym@microsoft.com

James Henderson
Idiap Research Institute
james.henderson@idiap.ch

Abstract

The state-of-the-art models for coreference resolution are based on independent mention pair-wise decisions. We propose a modelling approach that learns coreference at the document-level and takes global decisions. For this purpose, we model coreference links in a graph structure where the nodes are tokens in the text, and the edges represent the relationship between them. Our model predicts the graph in a non-autoregressive manner, then iteratively refines it based on previous predictions, allowing global dependencies between decisions. The experimental results show improvements over various baselines, reinforcing the hypothesis that document-level information improves conference resolution.

1 Introduction

Current state-of-the-art (SOTA) solutions for coreference resolution such as (Toshniwal et al., 2020; Xu and Choi, 2020; Wu et al., 2020) formulate the problem in an end-to-end manner where the models jointly learn to detect mentions and link coreferent mentions. The objective is to predict the antecedent of each mention-span in a document, so the model performs pair-wise decisions of all mentions. After having the model predictions, related mentions are grouped into clusters. Under this scenario, each decision (i.e., whether two mentions are related to the same entity or not) is independent. Lee et al. (2018) proposed an iterative method to update the representation of a mention with information of its probable antecedents. However, the final decisions are still made locally.

We propose a modeling approach that learns coreference at the document-level and takes global decisions. We propose to model mentions and coreference links in a graph structure where the nodes are tokens in the text, and the edges represent the relationships between them. Figures 1 and 2 show a short example taken from the CoNLL 2012 dataset (Pradhan et al., 2012) showing the graph in two perspectives. Figure 1 shows how the token nodes in a text are connected with edges drawn with arrows. We differentiate the connections between words in a coreference mention, '*mention links*', and the ones among mentions in a cluster, '*coreference links*' (see Sec. 4). Figure 2 shows the same graph in a matrix representation, where the number in a cell indicates the type of relation between the row and the column. Our model receives a document as input then predicts and iteratively refines the graph of mentions and coreference links.

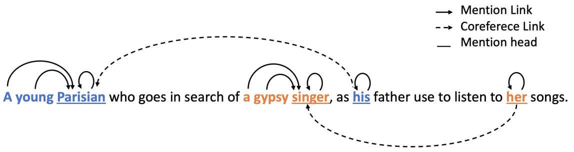
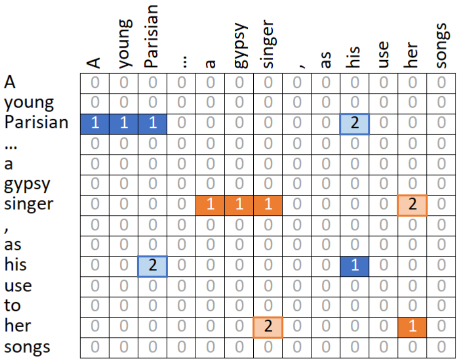

Mention Link
Coreferece Link
Mention head
A young Parisian who goes in search of a gypsy singer, as his father use to listen to her songs.

Figure 1: Example of a graph structure for coreference. Mention spans are shown in bold, and colors represent entity clusters. The mention heads are underlined.



	A	young	Parisian	...	a	gypsy	singer	,	as	his	use	her	songs
A	0	0	0	0	0	0	0	0	0	0	0	0	0
young	0	0	0	0	0	0	0	0	0	0	0	0	0
Parisian	1	1	1	0	0	0	0	0	0	2	0	0	0
...	0	0	0	0	0	0	0	0	0	0	0	0	0
a	0	0	0	0	0	0	0	0	0	0	0	0	0
gypsy	0	0	0	0	0	0	0	0	0	0	0	0	0
singer	0	0	0	0	1	1	1	0	0	0	0	2	0
,	0	0	0	0	0	0	0	0	0	0	0	0	0
as	0	0	0	0	0	0	0	0	0	0	0	0	0
his	0	0	2	0	0	0	0	0	0	1	0	0	0
use	0	0	0	0	0	0	0	0	0	0	0	0	0
to	0	0	0	0	0	0	0	0	0	0	0	0	0
her	0	0	0	0	0	0	2	0	0	0	0	1	0
songs	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2: Example of a graph in matrix representation. The connection types are encoded as, 0: no links, 1: mention links, 2: coreference links.

We follow a similar approach to the Graph-to-Graph Transformer (G2GT) proposed in (Mohammadshahi and Henderson, 2021, 2020) for syntactic parsing, but instead of encoding sentences, we en-

[*] Work done as a PhD student at EPFL/Idiap

Findings of the Association for Computational Linguistics: ACL 2022, pages 2732 - 2742
May 22-27, 2022 ©2022 Association for Computational Linguistics

code documents. Our model predicts the graph in a non-autoregressive manner, then iteratively refines it based on previous predictions. This recursive process introduces global dependencies between decisions. Unlike (Mohammadshahi and Henderson, 2021), we define different structures for input and output graphs, to reflect the different roles of these graphs. To ensure that locality in the input graph reflects all the relevant relationships, the input graph encodes relations for all mention tokens. This makes the encoding process easier. To provide a unique specification of the target graph, the output only encodes a minimal set of connections. This facilitates prediction. We initialize the Transformer with pre-trained language models, either BERT (Devlin et al., 2019), or SpanBERT (Joshi et al., 2020).

Another difference with (Mohammadshahi and Henderson, 2021) is that our model predicts two levels of representation. While they predict the whole graph at each iteration, during the first iteration our model only predicts edges that identify mention-spans. This is because mention detection is a sentence-level phenomenon whose outputs are required as inputs to coreference resolution, which is a discourse-level phenomenon. But we do not organise these two tasks in a pipeline. Starting at the second iteration, the model predicts the complete graph. This allows the model to refine mention decisions given coreference decisions, and vice versa. In this way, we propose to use iterative graph refinement as an alternative to pipeline architectures for multi-level deep learning models. The iterative process finishes when there are no more changes in the graph or when a maximum number of iterations is reached.

Ideally, the whole document should be encoded at once, but in practice there is a limit on the maximum length. In order to deal with this issue, we propose two strategies: overlapping windows and reduced document. In the first strategy, we split documents into overlapping windows of the maximum allowed size K. The segments overlap for a length $K/2$. At decoding time, segments are input in order, and we construct the final graph by joining all graphs from different segments. In the second strategy, we use two networks. The mention-span network is the previously described overlapping model, and we use it for predicting the first graph. For the second network, we reduce the document by including only the tokens of candidate mention-spans, separated by a special token. This network refines the initial graph for the following iterations.

The experiments show improvements over the relevant baselines and state-of-the-art. They also indicate that the models reach the best solution in a maximum of three iterations. Given that we predict the graph at once for each iteration, our model's complexity is lower than the baselines. Our contributions are the following:

- We propose a novel modeling approach to coreference resolution using a graph structure and multi-level iterative refinement.
- We propose two iterative graph refinement models that can predict the complete entity coreference structure of a document.
- We show improvements over baseline models and the relevant state-of-the-art.

The rest of the paper is organized as follows. Section 2 presents a summary of coreference resolution approaches related to this paper. Section 3 briefly describes the fundamentals of state-of-the-art approaches. In Section 4, we define entity mentions and their coreference links as a graph, and fomulate the task as a sequence-to-graph problem. In Section 5, we present our iterative refinement solution to global modelling of the coreference graph, and in Section 6, we present two proposed architectures to address the resulting computational issues. Sections 7, 8 and 9 contain the experimental setup, results and discussion, respectively. Finally, Section 10 draws the conclusions of this paper.

2 Related Work

The first approaches to coreference resolution (CR) were rule-based systems (Lappin and Leass, 1994; Manning et al., 2014), but eventually, they were outperformed by machine learning approaches (Aone and William, 1995; McCarthy, 1995; Mitkov, 2002) due to annotated corpora's creation. In genral, there are three coreference approaches : mention-pair, entity-mention, and ranking models. Mention-pair models set coreference as a binary classification problem. The initial stage is the mention detection, where the input is raw text, and the output is the locations of each entity mention in the text. Mention detection is done as an independent task in a pipeline model (Soon et al., 2001) or as part of an end-to-end model (Lee et al., 2017). The next stage is the classification of mention pairs. At first, the best classifiers were decision trees (Soon et al., 2001; McCarthy, 1995; Aone and William, 1995), but later, neural networks became the SOTA. The final stage is reconciling the pair-wise decisions to create entity chains, usually by utilizing greedy algorithms or clustering approaches. Entity-mention models focus on maintaining single underlying entity representation for each cluster, contrasting the independent pair-wise decisions of mention-pair approaches (Clark and Manning, 2015, 2016). Ranking models aim at ranking the possibles antecedent of each mention instead of making binary decisions (Wiseman et al., 2016). An alternative modeling approach is to perform clustering instead of classification (Fernandes et al., 2012).

SOTA models for CR are mostly based on Lee et al. (2017). They introduced the first end-to-end model that jointly optimizes mention detection and coreference resolution tasks. These neural network-based models also simplify the mention input representation to be word embedding vectors, instead of the traditional pipeline of different linguistic feature extraction tools such as part-of-speech (POS) tagging and dependency parsing. The following models proposed improvements over this work. (Lee et al., 2018) improved the previous model by introducing higher order inference so the entity's mention representation will get iteratively updated with the weighted average of antecedent representations, where the weights are the predictions from the model at the previous iteration. This contrasts with our approach in that we iterate over the whole coreference link graph and we perform discrete decisions at each iteration. Fei et al. (2019) use reinforcement learning to directly optimize the model on the evaluation metrics. Joshi et al. (2019) uses BERT embeddings (Devlin et al., 2019) as input. Joshi et al. (2020) introduced a new SpanBERT embedding model, which is shown to outperform BERT for the CR task. Xu and Choi (2020) showed that higher order inference has low impact on strong models such as SpanBERT. Toshniwal et al. (2020) proposed a bounded memory model trained to manage limited memory by learning to forget entities. Finally, Wu et al. (2020) formulated the problem of coreference resolution as question-answering and trained a model for span prediction. This model has the advantage of being pretrained with larger data-sets from the question-answering task.

3 Baseline: Neural Coreference Resolution

Neural coreference resolution, as formulated in (Lee et al., 2017, 2018), is a mention-pair approach. It uses an exhaustive method defining mentions as any text span of any size in a document. There, a document D represents a sequence of tokens of size N. The objective is to assign an antecedent y_i to each of the M text spans m_i in D. The set of possible antecedents of the span m_i is denoted as $\mathrm{Y}(i)$. This set contains all text spans with index less than i, plus a null antecedent ϵ, $\mathrm{Y}(i) = \{\epsilon, m_1, ..., m_{i-1}\}$. The null antecedent is assigned when: (a) the span is not an entity mention, (b) the span is the first mention of an entity in the document. The final mention clusters are constructed greedily by grouping connected spans based on the model predictions during decoding time.

The model is trained to learn a conditional probability distribution over documents $p(y_1, ..., y_n|D)$, assuming independence among each decision of antecedent assignment y_i, as follows:

$$p(y_1, ..., y_M|D) = \prod_{i=1}^{M} p(y_i|D) \tag{1}$$

In (Lee et al., 2018), the probability distribution $p(y_i|D)$ is inferred over T iterations of the model over the same input document. At each iteration t, the span representations are updated with the weighted average of all possible antecedents at time $t-1$ where the weights are given by the probability distribution of the model at time $t-1$. They called this model high-order coreference resolution since each mention representation considers information from its probable antecedents.

The training optimization is done using cross-entropy. Given that a mention-span m_i can have more than one true antecedent, the loss considers the sum of probabilities of all true antecedents in the annotated data:

$$log \prod_{i=1}^{M} \sum_{y_i \in \mathrm{Y}(i) \cap \mathrm{C}(i)} p(y_i|D) \tag{2}$$

where $\mathrm{C}(i)$ indicates the cluster of mention-spans that includes m_i in the annotated data. If the span does not belong to any cluster or all its antecedents have been pruned, then the span is assigned to the null cluster $\mathrm{C}(i) = \{\epsilon\}$.

This model's complexity is of the order $\mathcal{O}(N^4)$, where N is the document length. The complexity is computed by considering all possible text spans M of the document, so $\mathcal{O}(M) = \mathcal{O}(N^2)$. Then, it considers all possible combinations of span-antecedents $\mathcal{O}(M^2)$. The model prunes spans and candidate antecedents to predetermined maximum numbers in order to maintain computational efficiency.

4 Graph Modeling

We propose to model the set of coreference links of a document in a graph structure where the nodes are tokens[1] and the edges are links of different types. Given a document $D = [x_1, ..., x_N]$ of size N, the coreference graph is defined as the matrix $G \subset \mathbb{N}^{N \times N}$ of links between tokens. Here, the relation type between two tokens, x_i and x_j, is encoded with integers and is denoted as $g_{i,j} \in \{0, 1, 2\}$. We define three relation types: (0) no link, (1) mention link, and (2) coreference link, as illustrated in Figure 2.

Mention links This type of link serves to identify mentions. We define mention links in two different manners depending on whether the graph is an input or output of the model, for functional reasons. When the graph is an input G^{in}, there is a directed link from each mention's token to the mention head, including the head to itself. When the graph is the model's output G^{out}, there is only one directed link from the last token of the mention-span to the first token. Both encoding methods define a mention-span uniquely, even when having nested mentions; every mention has a unique start-end combination and a unique head. The model utilizes the output for prediction, so it is simpler to predict one single link, whereas, in the input, the model uses links to all tokens to provide a more direct representation of the role of every token in the mention.

Mention heads We simplified the head identification process by considering the first token of a mention span as the head. Although this method is naive, experiments show that this approximation works well enough in practice. However, as some spans can potentially have the same first token in case of nested mentions, we fix this issue by assigning the next token as the head if the first is already the head of any other mention. Investigating alternative approaches to mention head identification is future work.

Coreference links This type of link defines the relationship between a mention and each of its antecedents. We also define coreference links in two different manners depending on whether the graph is an input or output of the model. When the graph is input, there is a link from a mention head token to the head of each mention in the same cluster. When the graph is a model's output, the mention should be connected to at least one of its antecedents. If the mention has no antecedent, or corresponds to the first mention of an entity in the text, then it is connected to a null antecedent ϵ. We use all possible connections between mentions in an entity cluster for the input so that the model receives a direct input for each coreference relationship. On the other hand, we consider that predicting at least one connection of the mention to its cluster is sufficient to specify the output graph.

The objective is to learn the conditional probability distribution $p(G|D)$. This distribution is initially approximated by assuming independence among each relation $g_{i,j}$ as:

$$p(G|D) = \prod_{i=1}^{N} \prod_{j=1}^{i} p(g_{i,j}|D) \tag{3}$$

The probability $p(g_{i,j}|D)$ is split in two cases: one for mention links p_m and the other for coreference links p_c. The mention link probability is defined as:

$$p_m(g_{i,j}{=}1|D) = \sigma(W_m \cdot [h_i, h_j]) \tag{4}$$

where W_m is a parameter matrix, and h_i and h_j are the hidden state representations of the tokens x_i and x_j respectively. This probability indicates whether there is a mention starting at position j and ending at position i of the document D. The optimization is done using binary-cross-entropy $loss_m$.

The coreference link probability is defined as:

$$p_c(g_{i,j}{=}2|D) = \frac{exp(W_c \cdot [h_i, h_j])}{\sum_{j' \in \mathrm{A}(i)} exp(W_c \cdot [h_i, h_{j'}])} \tag{5}$$

where W_c is a parameter matrix, and h_i and h_j are the hidden state representations of the tokens x_i and x_j respectively. Similar to the baseline, we denote $\mathrm{A}(i)$ as the set of all candidate antecedents of x_i. This set contains all mention heads with

[1] The tokenization of the words in the document, and thus the nodes of the graph, are defined by the input format of the relevant pre-trained Transformer model.

an index less than i, plus a null head ϵ, $\mathrm{A}(i) = \{\epsilon, x_k \mid k < i \text{ and } x_k \in \mathrm{H(D)}\}$, and H(D) is the set of all candidate mention heads in the document. The optimization is done with cross-entropy loss. Given that a mention-span m_i can have more than one true antecedent, the loss considers the sum of probabilities of all true antecedents in the annotated data (as in Equation(2)):

$$loss_c = log \prod_{i \in \mathrm{H}(D)} \sum_{j \in \mathrm{Y}(i) \cap \hat{\mathrm{C}}(i)} p_c(g_{i,j}|D) \quad (6)$$

where $\hat{\mathrm{C}}(i)$ indicates the annotated cluster of mention-spans that includes m_i in the annotated data. If the mention does not belong to any cluster, then the span is assigned to the null cluster $\hat{\mathrm{C}}(i) = \{\epsilon\}$. The final loss is the sum of $loss_m$ and $loss_c$.

The token's hidden state representations $\{h_1, .., h_N\}$ are the last hidden layer of a Transformer model. We use various pre-trained Transformer models to initialize the weight parameters, then fine-tune for the coreference task.

5 Iterative Refinement

The strong independence assumption made in Equation (3) does not reflect the real scenario and could lead to poor performance. Therefore, we use an iterative refinement approach to model interdependencies between relations, similar to G2GT (Mohammadshahi and Henderson, 2021). Under this approach, the model makes T iterations over the same document D. At each iteration t, the predicted coreference graph G_t is conditioned on the previously predicted one G_{t-1}. The model's conditional probability distribution is now defined as follows:

$$p(G^t|D, G^{t-1}) = \prod_{i=1}^{N} \prod_{j=1}^{i} p(g_{i,j}|D, G^{t-1}) \quad (7)$$

This means that the graph should be input to the Transformer model (Vaswani et al., 2017). Following (Mohammadshahi and Henderson, 2021), the graph is encoded by inputting an embedding for the type of each relation into the self-attention function of the Transformer :

$$\begin{aligned} &\mathsf{Attention}(Q, K, V, L_k, L_v) = \\ &\quad \text{softmax}(\frac{Q \cdot (K + L_k)^\intercal}{\sqrt{d}}) \cdot (V + L_v) \quad (8) \end{aligned}$$

$$\text{where} \quad \begin{aligned} L_v &= E(G_{t-1}) \cdot W_v \\ L_k &= E(G_{t-1}) \cdot W_k \end{aligned}$$

where E is a matrix of embeddings which encode the types of links in the graph, as illustrated in Figure 2. Thus, the relationship between a pair of tokens is encoded as an embedding vector which is input when computing the attention function for that pair of tokens. W_k, W_v are weight matrices that serve to specialize $E(G_{t-1})$ to be either *key* or *value* vectors. The complexity of our model is of the order of $\mathcal{O}(N^2 \times T)$, where N is the document length, and T is the number of refinement iterations of the model.

To illustrate the iterative refinement of a graph, Figure 3 shows an example of two iterations of the model. The mention links are indicated with solid line arrows and the coreference links with dotted arrows. The initial graph matrix G_0^{in} is full of zeros, so no connections are drawn. The first predicted graph G_1^{out} only has mention-links because initially there were no mention heads to be connected. This graph is transformed to serve as input G_1^{in} for the next iteration. Finally, during the second iteration, the model predicts the coreference graph G_2^{out}. The model can continue iterating for a maximum of T times.

6 Architectures

There exists in practice a maximum length for encoding a document due to limited hardware memory. In this section, we describe two strategies to manage this issue: overlapping windows and reduced document. In the experiments we also report results for a naive strategy of truncating the documents at the maximum segment length of K for both training and testing.

6.1 Overlapping Windows

Here, we split the documents into overlapping segments of the maximum size K, with an overlap of $K/2$ tokens. The segments are encoded individually in our G2GT model. During training, each segment is treated as an independent sample. However, during decoding, the segments are decoded in order. The subgraph corresponding to the overlapping part is input to the next segment. The union of the segmented graphs forms the final graph.

6.2 Reduced Document

This model has two parts; one to detect mentions and the other to perform coreference resolution.

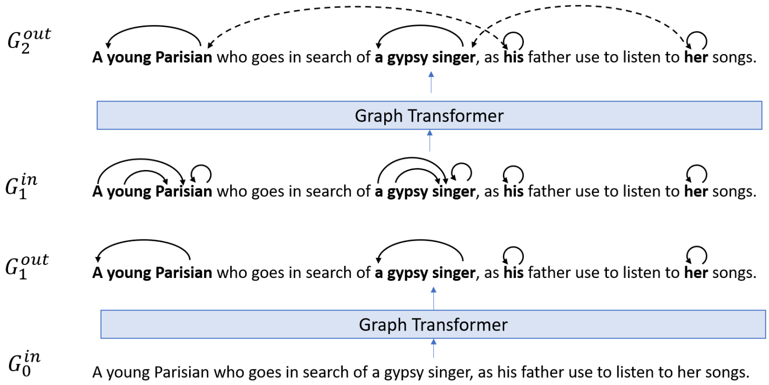

G_2^{out}
A young Parisian who goes in search of a gypsy singer, as his father use to listen to her songs.
Graph Transformer
G_1^{in}
A young Parisian who goes in search of a gypsy singer, as his father use to listen to her songs.
G_1^{out}
A young Parisian who goes in search of a gypsy singer, as his father use to listen to her songs.
Graph Transformer
G_0^{in}
A young Parisian who goes in search of a gypsy singer, as his father use to listen to her songs.

Figure 3: Example of iterations with G2GT.

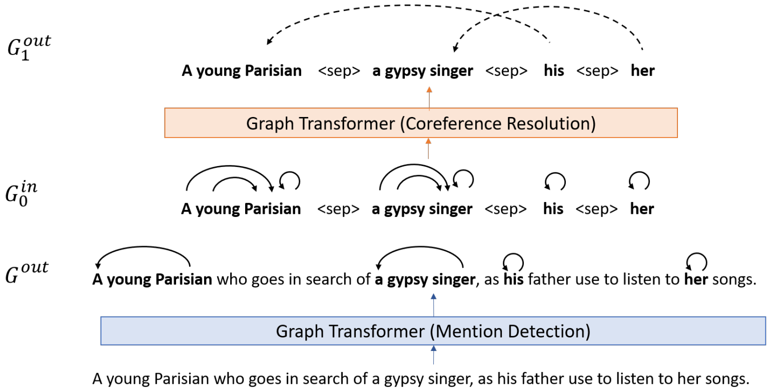

G_1^{out}
A young Parisian <sep> a gypsy singer <sep> his <sep> her
Graph Transformer (Coreference Resolution)
G_0^{in}
A young Parisian <sep> a gypsy singer <sep> his <sep> her
G^{out}
A young Parisian who goes in search of a gypsy singer, as his father use to listen to her songs.
Graph Transformer (Mention Detection)
A young Parisian who goes in search of a gypsy singer, as his father use to listen to her songs.

Figure 4: Example of iterations with G2GT in two stages.

The mention detection is similar to the previously described model. The coreference resolution part receives a shorter version of the document as input. The complete model is described in the following:

Mention Detection This Transformer is non-iterative so it corresponds to the definition in Equation (3). To encode the document, we apply overlapping windows, as in the previous section. For prediction, we used the *soft-target* method proposed in (Miculicich and Henderson, 2020). This method enables the model to increase the recall of detection. Given that the candidate mentions will be fixed for the coreference resolution part, we need to detect most of them here.

Coreference Resolution This part is a G2GT with iterative refinement. The input is a shorter version of the document obtained by concatenating the tokens from candidate mention-spans with a separation token in between and removing all other tokens. To maintain coherence in the document, we modify the token input representation to the sum of three vectors: (a) a token embedding, (b) an embedding of the token's position in the original document, so we retain information of distance between mentions, and (c) the token's contextualized representation obtained from the mention detection part where the original document is encoded. This second part predicts only coreference links, but the input graph contains both candidate mentions and coreference links. The set of candidate mentions remains the same across all iterations of this second part, but the mentions are refined in the sense that the final output only includes the mentions which are involved in the final coreference links.

Figure 4 shows an example of this architecture with one iteration over a document. The mention links are indicated with solid line arrows and the coreference links with dotted arrows. The first model predicts the graph of mention-spans G^{out}. This graph is transformed into the input format for the next model G_0^{in}. Then, the second model predicts the graph of coreference G_1^{out}. Note that this coreference resolution model can continue iterating for T times. The final coreference graph is the output after the final iteration of the second model. The final set of mentions is only a subset of the mention candidates output by the first

	Train	Dev.	Test	Total
# documents	2,802	343	348	3,493
# words	1.3 M	160 K	170 K	1.6 M
Avg. length	464	466	488	458
# entity changes/clusters	35 K	4.5 K	4.5 K	44 K
# coreference links	120 K	14 K	15 K	150 K
# mentions	155 K	19 K	19 K	194 K

Table 1: Dataset statistics and splits.

model, namely those mentions which participate in coreference links.

7 Experimental Setting

7.1 Dataset

We use the CoNLL 2012 corpus (Pradhan et al., 2012). It contains data from diverse domains e.g., newswire, magazines, conversations. We experiment only with the English part. Table 1 shows the statistics of the dataset; the average length per document does not exceed 500 words. We pre-process the text to extract sub-word units (Sennrich et al., 2016) with BERT tokenizer (Wu et al., 2016). We map the positional annotation of mentions from words to sub-words and retain this mapping for back transformation during evaluation.

7.2 Model configuration

We use the implementation of Wolf et al. (2019)[2] of 'BERT-base', 'BERT-large' (Joshi et al., 2019) and 'SpanBERT-large' (Joshi et al., 2020). All hyper-parameters follow this implementation unless specified otherwise.

Training The G2GT considers an independent loss for each different refinement iteration. There is no back-propagation between refinement iterations because the model makes discrete decisions when predicting the graph for the next refinement step. There are two stopping criteria for the refinement: (a) when a maximum number of iterations T is reached, or (b) when there are no more changes in the graph, $G_t = G_{t-1}$. This criterion is for both training and testing. Our models are trained with a maximum segment length of $K = 512$ and a batch size of 1 document. We use Adam (Kingma and Ba, 2014; Wolf et al., 2019) optimizer with a base learning rate of $2e{-}3$ and no warm-up. As our graphs are directed, we use only the lower triangle of G for predictions. The components of the reduced models are trained independently. The coreference resolution follows the currently described training schema. The mention detection model has no iterative refinement step and follows the training schema of the *span scoring soft-target* approach described in (Miculicich and Henderson, 2020), with $\rho = 0.1$.

Model	Iter.	MUC	B^3	CEAF$_{\phi_4}$	Avg. F1
G2GT	$T = 2$	75.7	68.4	65.2	69.8
BERT-base	$T = 3$	76.9	69.3	66.0	70.7
truncated	$T = 4$	77.2	69.7	66.3	71.0
	$T = 5$	77.2	69.7	66.3	71.0
G2GT	$T = 2$	80.6	69.8	67.4	72.6
BERT-base	$T = 3$	81.6	71.0	68.6	73.7
overlap	$T = 4$	81.5	70.9	68.7	73.7
	$T = 5$	81.4	70.6	68.7	73.5
G2GT	$T = 2$	79.2	76.1	68.5	71.6
BERT-base	$T = 3$	80.0	69.6	70.2	73.3
reduced	$T = 4$	81.9	70.1	71.2	74.4
	$T = 5$	81.9	70.1	71.2	74.4

Table 2: Refinement iterations T on the development set (CoNLL 2012).

Evaluation At evaluation time, we map back all sub-word units to words and reconstruct the document in CoNLL 2012 format. We use the precision, recall, and F1 score calculated in three different manners: MUC that counts the number of links between mentions, B^3 that counts the number of mentions, and CEAF that counts the entity clusters. We did paired bootstrapping re-sampling for significance test following (Koehn, 2004).

8 Results Analysis

This section describes the results of various baselines and our models. First, we analyze the optimum number of refinement iterations, and then we show results using the best models.

Table 2 shows the performance of our G2GT models when varying the maximum number of refinement iterations T from 2 to 5 (T=1 is mention detection only). The results are in terms of the F1 score of the three coreference metrics and the average. All three implementations shown in the table perform the best when using T=4. There is a significant decrease in performance when the graphs are not refined, T=2, showing the importance of modelling the interdependencies between coreference relations.

Table 3 shows the evaluation results on the test set in terms of precision (P), recall (R), and F1 score for each metric. The last column displays the average F1 of the three metrics. The first section

[2] https://huggingface.co/transformers/

	MUC			B^3			CEAF$_{\phi_4}$			
Model	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**Avg. F1**
Clark and Manning (2015)	76.1	69.4	72.6	65.6	56.0	60.4	59.4	53.0	56.0	63.0
Wiseman et al. (2016)	77.5	69.8	73.4	66.8	57.0	61.5	62.1	53.9	57.7	64.2
Clark and Manning (2016)	79.2	70.4	74.6	69.9	58.0	63.4	63.5	55.5	59.2	65.7
Lee et al. (2017)	78.4	73.4	75.8	68.6	61.8	65.0	62.7	59.0	60.8	67.2
Fei et al. (2019)	85.4	77.9	81.4	77.9	66.4	71.7	70.6	66.3	68.4	73.8
Xu and Choi (2020)	**85.9**	85.5	85.7	79.0	78.9	79.0	**76.7**	75.2	75.9	80.2
Wu et al. (2020)	88.6	87.4	88.0	82.4	82.0	82.2	79.9	78.3	79.1	83.1
Baseline (Lee et al., 2018)	81.4	79.5	80.4	72.2	69.5	70.8	68.2	67.1	67.6	73.0
+ BERT-base (Joshi et al., 2019)	80.4	82.3	81.4	69.6	73.8	71.7	69.0	68.5	68.8	73.9
+ BERT-large (Joshi et al., 2019)	84.7	82.4	83.5	76.5	74.0	75.3	74.1	69.8	71.9	76.9
+ SpanBERT-large (Joshi et al., 2020)	85.8	84.8	85.3	78.3	77.9	78.1	76.4	74.2	75.3	79.6
G2GT BERT-base *truncated*	78.4	77.9	78.1	69.6	71.0	70.3	66.8	67.3	67.0	71.8
G2GT BERT-base *overlap*	81.2	82.8	82.0	69.8	73.6	71.6	69.6	69.3	69.4	74.4
G2GT BERT-base *reduced*	83.4	83.1	83.2	70.1	73.7	71.9	72.1	70.1	71.0	75.4
G2GT BERT-large *truncated*	80.1	79.2	79.6	71.3	71.0	71.1	69.1	68.8	68.9	73.2
G2GT BERT-large *overlap*	83.5	83.2	83.3	74.5	74.1	74.3	75.2	70.1	72.6	76.7
G2GT BERT-large *reduced*	84.7	83.1	83.9	76.8	74.0	75.4	75.3	70.1	72.6	77.3
G2GT SpanBERT-large *overlap*	85.8	84.9	85.3	78.7	78.0	78.3	76.4	74.5	75.4	79.7
G2GT SpanBERT-large *reduced*	**85.9**	**86.0***†	**85.9***	**79.3***	**79.4***†	**79.3***	76.4	**75.9***	**76.1***	**80.5***

Table 3: Evaluation on the test set (CoNLL 2012). $*$ significant at $p < 0.01$ compared to (Joshi et al., 2020), † significant at $p < 0.05$ compared to (Xu and Choi, 2020)

.

of the table exhibits scores of different coreference resolution systems from the literature. The second section shows the result of the 'Baseline' (Lee et al., 2018) system described in Section 3. This model uses ELMo (Peters et al., 2018) instead of BERT to obtain token representations. Baseline plus 'BERT-base', 'BERT-large' (Joshi et al., 2019) and ' SpanBERT-large' (Joshi et al., 2020) correspond to the baseline using those pretrained representations. We copy all these values from the original papers. The last section of the table presents scores of our graph-to-graph models with iterative refinement. '*truncated*' is our model with no special treatment for document length; the documents are truncated at the maximum segment length of K. '*overlap*' and '*reduce*' are the models described in Section 6.

As expected, pre-training with SpanBERT results in better scores than with BERT, and BERT-large is better than BERT-base. Not surprisingly, 'G2GT BERT-base truncated' and 'G2GT BERT-large truncated' perform poorly in comparison to the baseline because their information is incomplete. For BERT-base, both the '*overlap*' and '*reduce*' models have better scores than the comparable baseline. For BERT-large and SpanBERT, the '*overlap*' model has similar scores to the baseline, but the '*reduce*' model consistently improves over the baseline.

Preliminary experiments with G2GT '*overlap*' in a pipeline approach, where mention detection is performed before coreference, showed that it is not better than in a joint approach showed here. Overall, our G2GT '*reduce*' method consistently shows the highest scores across all the models for each pre-trained model. Our models do not surpass SOTA (Wu et al., 2020) (shown in grey), but as mentioned before, this SOTA model is also trained on the much more abundant data from the question-answering task, and so it is not directly comparable to our model. We leave the issue of incorporating additional data into the training of our model to future work.

9 Discussion

These results support our claim that coreference resolution benefits from making global coreference decisions using document-level information. First, refinement of coreference decisions using global information about other coreference decisions clearly improves accuracy, as indicated by the improved scores for models with more than one coreference iteration in Table 2. Second, the model which is able to combine information from the entire document, G2GT '*reduce*', is clearly better than the model which performs the task on large windows of text and then merges the results, G2GT '*overlap*'. We believe that the benefits of full-document iterative refinement will extend to other discourse-level phenomena, and that the G2GT architecture will be an effective way to achieve this benefit.

One issue with our method is the necessity to iteratively pass the input through an expensive encoder model more than once. However, the number of iterations needed is small, and results in significant improvement.

The length management methods would not be necessary if we had more efficient pre-trained Transformer models or larger-memory GPU hardware which could handle longer sequences. However, the computational cost of very large Transformers will always be an issue, so in general there is a need to address the issue of how to reduce the number of inputs when modelling phenomena which require large contexts, such as coreference resolution. This paper contributes towards addressing this general issue.

10 Conclusion

We proposed a G2GT model with iterative refinement for coreference resolution. For this purpose, we define a graph structure to encode coreference links contained in a document. That enables our model to predict the complete coreference graph at once. The graph is then refined in a recursive manner, iterating the model conditioned on the document and the graph prediction from the previous step. This allows global modelling of all coreference decisions using all document-level information, but it introduces computational issues for longer documents. We experimented with two methods to manage long documents and maintain computational efficiency. The first method encodes the document in overlapping segments. The second method reduces the set of tokens which are input.

The evaluation shows that both methods can outperform a comparable baseline, and that the second method has better performance than the first one and than all other comparable models. This experiment shows that global decisions and document-level information are useful to improve coreference and thus should not be ignored. It also shows that the models can benefit from increasingly powerful pre-trained language models, BERT-base (Devlin et al., 2019), BERT-large (Devlin et al., 2019), and SpanBERT (Joshi et al., 2020).

By empirically showing the benefits of making global decisions and using document-level information in coreference resolution, this work motivates further work on this topic. In addition, the model designs developed in this work provide a viable approach to addressing the related issues. Addressing the computational issues with modelling large documents in Transformers is an area of active research, and our proposed methods could be improved in future work.

Acknowledgements

This work was supported in part by the Swiss National Science Foundation, under grants 200021_178862 and CRSII5_180320.

References

Chinatsu Aone and Scott William. 1995. Evaluating automated and manual acquisition of anaphora resolution strategies. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 122–129, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Kevin Clark and Christopher D. Manning. 2015. Entity-centric coreference resolution with model stacking. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1405–1415, Beijing, China. Association for Computational Linguistics.

Kevin Clark and Christopher D. Manning. 2016. Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 643–653, Berlin, Germany. 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.

Hongliang Fei, Xu Li, Dingcheng Li, and Ping Li. 2019. End-to-end deep reinforcement learning based coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 660–665, Florence, Italy. Association for Computational Linguistics.

Eraldo Fernandes, Cícero dos Santos, and Ruy Milidiú. 2012. Latent structure perceptron with feature induction for unrestricted coreference resolution. In *Joint Conference on EMNLP and CoNLL - Shared*

Task, pages 41–48, Jeju Island, Korea. Association for Computational Linguistics.

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

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.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Shalom Lappin and Herbert J. Leass. 1994. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561.

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.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

JF McCarthy. 1995. Using decision trees for coreference resolution. In *Proc. 14th International Joint Conf. on Artificial Intelligence (IJCAI), Quebec, Canada, Aug. 1995*.

Lesly Miculicich and James Henderson. 2020. Partially-supervised mention detection. In *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 91–98, Barcelona, Spain (online). Association for Computational Linguistics.

Ruslan Mitkov. 2002. *Anaphora Resolution*. Longman, London, UK.

Alireza Mohammadshahi and James Henderson. 2020. Graph-to-graph transformer for transition-based dependency parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3278–3289, Online. Association for Computational Linguistics.

Alireza Mohammadshahi and James Henderson. 2021. Recursive non-autoregressive graph-to-graph transformer for dependency parsing with iterative refinement. *Transactions of the Association for Computational Linguistics*, 9:120–138.

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.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

Shubham Toshniwal, Sam Wiseman, Allyson Ettinger, Karen Livescu, and Kevin Gimpel. 2020. Learning to Ignore: Long Document Coreference with Bounded Memory Neural Networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8519–8526, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Sam Wiseman, Alexander M. Rush, and Stuart M. Shieber. 2016. Learning global features for coreference resolution. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 994–1004, San Diego, California. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, pages arXiv–1910.

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.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

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