Multilingual coreference resolution as text generation

Natalia Skachkova
DFKI / Saarland Informatics Campus, Saarbrücken, Germany
natalia.skachkova@dfki.de

Abstract

This paper presents a multilingual coreference resolution system *DFKI-CorefGen* submitted for the CRAC Shared Task 2024. We cast the task as text generation and use *mT5-base* as the pre-trained model. Our system takes the sixth place out of seven in the competition. We analyze the reasons for poor performance and suggest possible improvements.

1 Introduction

Coreference resolution is an important part of many natural language processing (NLP) tasks like question answering, information extraction, text summarization, etc. CRAC 2024 focuses on multilingual coreference resolution, which is less researched than the English one. It is also more challenging than monolingual coreference resolution, as the training data typically come from different sources and may be characterized by large variability in size, domain, the definition of markables, annotation consistency, completeness and quality. Ideally, a good multilingual coreference resolution system should be able to deal with these challenges without a significant performance loss.

Currently, many state-of-the-art (multilingual) coreference resolution systems are modifications of the model first introduced by Lee et al. (2017). They are typically characterized by rather complex architectures based on pre-trained large language models and require careful data preprocessing. One needs to have not only novel ideas, but also very good programming skills and mathematical knowledge to modify such architectures. Additionally, the approach has some inherent limitations, e.g., it is tricky to use to identify discontinuous mentions or split antecedents.

On the other hand, one is always searching for easier ways to solve a task. Such possibility is offered nowadays by large language models [1] (LLMs). They are generative models, which demonstrate an excellent performance in many NLP tasks (e.g., see Zhao et al., 2023; Minaee et al., 2024; Chang et al., 2024) and are relatively easy to use for inference. However, they have their shortcomings too, the most important being a huge number of parameters, so that one needs a lot of computational resources to use them.

The aim of this work is to check if we can cast multilingual coreference resolution as a text generation task using a much smaller model, like mT5-base (Xue et al., 2021).We try to keep the task as simple as possible. No careful pre-processing is required – the input is the raw text and the output is the same text marked with coreference clusters. To summarize, our contributions are as follows.

- We investigate how multilingual coreference resolution can be represented as a purely generative end-to-end task, and discuss challenges and limitations of the approach.
- We show that mT5-base is to certain extent capable of the task, but obviously not large enough to achieve good scores and compete with the baseline.

2 Related Work

One of the seminal and most successful coreference resolution models is the one by Lee et al. (2017). It is a span-based mention-ranking model. Namely, all spans in a document are treated as potential mentions and represented as context-depending embeddings. These spans are ranked and paired with the most likely antecedent spans.

A lot of the state-of-the-art coreference resolution models, no matter multilingual or not, inherit this architecture with some modifications. E.g., it is the case for all the systems whose descriptions

[1] We use this term to refer to all models that have $\geq$ 13B parameters.

Proceedings of the Seventh Workshop on Computational Models of Reference, Anaphora and Coreference (CRAC 2024), pages 114–122

were submitted for CRAC 2023 (Žabokrtský et al., 2023).

There also exist works casting coreference resolution as a sequence-to-sequence problem. Some early experiments are conducted by Raffel et al. (2020), who apply the T5 model to resolve ambiguous pronouns in the WNLI, WSC (Levesque et al., 2012) and DPR (Rahman and Ng, 2012) data. They focus on separate pronouns and do not build any coreference chains or clusters, as the main goal is to evaluate the model's commonsense reasoning ability. Similar experiments (often on the same data), but with LLMs and few-shot prompting are presented by Perez et al. (2021), Min et al. (2022) and Lin et al. (2022).

Some researchers cast coreference resolution as a question answering task and use LLMs to generate answers. E.g., Wu et al. (2020) generate a list of coreferent mentions, given a question about an entity, Yang et al. (2022) generate "yes/no" answers, given a mention pair, Agrawal et al. (2022) generate the most likely antecedent, given an anaphor, and Le et al. (2022) - a chain of antecedents.

Another generative coreference resolution model is presented by Bohnet et al. (2023). It is a "link-append" transition system based on mT5-xl. It is multilingual and was successfully tested on English, Arabic, Chinese, Dutch, Catalan, German, Italian and Spanish data. As input it takes an encoding of the previous sentences annotated with coreference clusters, followed by the new sentence. As output, the system produces links from mentions in the new sentence to either previously created coreference clusters or to previous singleton mentions.

Other recent sequence-to-sequence approaches are introduced, e.g., by Urbizu et al. (2020), Paolini et al. (2021), Liu et al. (2022) and Zhang et al. (2023), who focus on English and generate coreference annotation, i.e. mentions and clusters they belong to, within the given text, typically using a fine-tuned encoder-decoder model.

Our approach DFKI-CorefGen falls into the latter category, but has the following differences. First, it is multilingual. Second, we keep the pretrained model frozen, and do prefix tuning (Li and Liang, 2021) instead. Third, we process the input text incrementally and teach our model to correct clustering mistakes in the previous sentences as well. Fourth, we create training data by corrupting the coreference annotations.

3 Method

We perform multilingual mention identification [2] and coreference resolution jointly and treat the task as text generation. Thus, given a piece of text, we want to find all mentions and group them into clusters by marking them in this text with square brackets and cluster identifiers. Example 3.1 demonstrates the idea on a short text sequence from the *en_parcorfull* corpus.

Example 3.1. Gold model output
[0 [1 The victim 1] 's brother 0] , [0 Louis Galicia 0] , told ABC station KGO in San Francisco that [1 Frank 1] , previously a line cook in Boston , had landed [1 his 1] dream job as line chef at [2 San Francisco 's Sons & Daughters 2] restaurant six months ago . [3 A spokesperson for [2 Sons & Daughters 2] 3] said [2 they 2] were " shocked and devastated " by [1 his 1] death .

The approach is implemented as a prefix tuning using OpenPrompt (Ding et al., 2022) with mT5-base as the core model. We apply prefix tuning, because mT5-base is relatively small (580M parameters) and thus not designed for inference in a zero- or few-shot manner. To save computational resources, we keep mT5-base frozen and tune only the prefix of 100 randomly initialized tokens. The input for the model, as shown in Example 3.2, contains a *[TEXT]* sequence, a task tag "coreference", and a *[MASK]* token, instead of which the model is to generate the *[TEXT]* with coreference clusters. No instructions or demonstrations are given to the model.

Example 3.2. Model input
[TEXT] Task: "coreference" [MASK]

We train one model for all the languages, using the official training data only. It is done on one NVIDIA GeForce GTX TITAN X GPU with 12 GB memory for five epochs with the batch size 1, the *AdamW* optimizer, learning rate of 5e-5 and a linear schedule with warm-up.

3.1 Input data

As the input length of mT5-base is limited by 1024 sub-tokens, we have to split each document into several pieces. In addition, our initial experiments showed that the model struggles finding the correct clusters, if it receives the whole raw piece of text

[2] Discontinuous mentions are discarded. Empty tokens (zero anaphora), represented as an underscore "_" in the data, are treated like all other tokens.

as input, especially if this piece is long. We deal with this challenge as follows.

First, we limit the length of each input piece by five sentences that can have various lengths but are no longer than 512 sub-tokens. Second, the task becomes easier, if some clusters (not necessarily always correctly marked) are already identified. Therefore, we proceed with the task incrementally, i.e., we start with giving the model the very first sentence and asking to find the clusters there, then we add the second sentence and ask the model to do the same task, revising its initial predictions, and so on until the five-sentence text piece is over.

To teach our model to do that, we create input data by splitting the five-sentence pieces into overlapping sub-pieces of 1-5 sentences long and corrupting the gold annotations in them. Now, if a sub-piece consists of a single sentence only, we remove all the clusters' annotations from it, if there are any. If the sub-piece is longer, we completely remove the annotations from the very last sentence, and either keep or (partially) corrupt the annotations in the previous ones. Keeping sub-pieces with correct clusters is needed to create examples which help the model differentiate between "good" and "bad" cluster annotations. If a gold text piece does not contain any clusters at all, we keep it as it is and consider it a negative example, as the model does not need to annotate anything there at all.

Theoretically, we can create an infinite number of training examples by corrupting the gold annotations in all possible ways. However, as we are limited by time and computational resources, we want to pick out only the most useful ones. To do so, we first conduct some experiments, where our model has to deal with the raw pieces without any clusters (wrong or correct) marked in them. Based on these experiments' results, we collect the most frequent generation error types and come up with the following modifications of the gold clusters.

First, we discard the annotations of half of the clusters in the sub-piece. Second, we merge half of the clusters together. Namely, we first divide all the clusters in the sub-piece in two groups, then merge them pairwise randomly. Third, we split half of the non-singleton clusters. Each one is picked out randomly and split in two. Fourth, we mix non-singleton clusters so that the number of mentions in each cluster stays the same, but half of the mentions in them is wrong. Fifth, we violate some mentions' boundaries.

Additionally, we have to deal with all sorts of repetitions that are a problem of many generative models including mT5 (Holtzman et al., 2020, Fu et al., 2021). Our initial experiments show that mT5-base has a tendency to generate excessively the cluster markers with or without mentions inside, as well as duplicates of marked mentions. To deal with these issues, we adopt two more types of corrupted training examples.

First, we append / prepend excessive cluster identifiers to some mentions. We also insert empty ones, i.e., opening and closing brackets with indices not marking any mentions, like '[4 4]'. Second, given some randomly chosen marked mentions, we extend the original text with their duplicates. The number of duplicates typically varies from two to five.

Finally, we address two more generation problems. Namely, mT5-base tends to excessively generate either empty square brackets, or just sequences of numbers with or without square brackets. And sometimes mT5 refuses to generate any cluster markers at all. We deal with these problems as follows.

Based on the observation that the sequences of '[', ']', '[0]', '[1]', '0', '[0]' and '[1]' are among the most frequent generation errors, we create training examples by randomly inserting such sequences into gold sub-pieces. To make the model learn that it should not just copy the input text, but mark some clusters, we create additional training examples by simply removing all the gold annotations from the sub-pieces. Appendix A.3 gives examples of the main modification types discussed above.

Importantly, we noticed that it is easier for the model to perform the task, if the clusters' identifiers are consecutive, i.e., they should be assigned depending on the order in which the corresponding mentions occur. Therefore, to create each training example we always re-index all the mentions in the given sub-piece.

As a result, given one gold sub-piece, we make from one to twelve training examples, depending on the sub-piece length. Each example contains only a single modification. We first create training and development data from each official dataset. Next, we randomly sample 2,000 training and 70 development examples from the respective parts of each set, regardless of the fact that some languages, e.g., English and German, are represented by several datasets. The distribution of positive and negative examples in the data is shown in Table 1.

Data	Negative (w/o clusters)	Positive		Total
		correct	corrupted	
train	172 (0.4%)	5,220 (12.4%)	36,608 (87.2%)	42,000
dev	4 (0.3%)	173 (11.8%)	1,293 (87.9%)	1,470

Table 1: Distribution of positive and negative examples in the data

3.2 Inference

As mentioned earlier, the main idea is to process the given document incrementally, annotating clusters in each new sentence and correcting the annotations in previous context. During training DFKI-CorefGen learns to deal with sequences up to five sentences long. However, we cannot simply split each document into pieces of five (or less) sentences, because in this case it will be impossible to merge the clusters stretching across several pieces. Therefore, we process the given document using a sliding window of five sentences which moves with a step of two sentences, so that each window contains two new sentences. Because our model expects only one "raw" (i.e., unannotated) sentence, these new sentences are also processed incrementally, one by one. We re-index the clusters in each piece.

Despite having special training examples aimed at dealing with repetitions, hallucinations, or truncation of text, these errors are still very common. Therefore, after having processed a piece, we have to align the generated and gold sequences (see example in Appendix A.4). To avoid cumbersome token level sequence matching, in the future we may switch to generation of dummy tokens instead of the real ones, similarly to Urbizu et al. (2020) and Zhang et al. (2023). Finally, to get clusters for the whole document, we merge clusters found in each piece based on mentions overlap.

4 Results and discussion

DFKI-CorefGen takes the sixth place out of seven with an average 33.38 F1 score. It is far below the 53.16 F1 score achieved by the baseline (Pražák et al., 2021). The results for separate datasets are given in Appendix A.1.

To large extent, bad scores can be explained by the nature of our approach. It resolves coreference incrementally, thus, during inference it is important to (at least partially) correctly identify clusters in the very first sentence. Otherwise, the errors accumulate with each new sentence, so that there are too many of them for the model to correct. We found out that our model is not really good at this task - it achieves the F1 score of only 42.59 when applied on 1,996 single sentences sampled from the gold development data. One possible reason for that is the lack of training examples consisting of one sentence only, as our focus is on clustering and correction of previously assigned clusters in a larger context. In total we only have 1,133 (2.7%) and 33 (2.4%) training and development examples consisting of single sentences that may or may not have gold clusters.

However, we hypothesize that the main reason for such an unsatisfactory performance is that mT5-base is simply not large enough for the task. Small model size also causes difficulties in performing the task for longer inputs, and very persistent hallucinations and repetitions in the output. E.g., currently we limit the sub-piece length by five sentences, which is sub-optimal, as we loose too many clusters by doing so (see Appendix A.2).

Another important negative factor is a small training data size - due to time constraint and limited computational resources we take only 2,000 training samples from each dataset.

Finally, our current method of corrupting the gold annotations may also be sub-optimal. Further experiments are required to decide how many and which clusters are better to mix, merge or split, how many duplicates to insert, how long they should be and so on. Also, different generation errors may be typical for different datasets, languages and script systems.

5 Conclusion

In this paper we introduce a simple and purely generative end-to-end approach to multilingual coreference resolution. We show that it is capable of the task, but suffers from certain limitations, like a small size of the pre-trained model and a lack of training data, that prevent it from achieving good scores. We believe that replacing mT5-base with a LLM of much larger size can help reach better results and avoid complicated post-processing. We leave such experiments along with a proper ablation study for future work.

Acknowledgments

The research in this paper has been funded by the Horizon Europe project Fluently (grant ID 101058680).

References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. 2022. Large language models are few-shot clinical information extractors. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1998–2022, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2024. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Haitao Zheng, and Maosong Sun. 2022. OpenPrompt: An open-source framework for prompt-learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 105–113, Dublin, Ireland. Association for Computational Linguistics.

Zihao Fu, Wai Lam, Anthony Man-Cho So, and Bei Shi. 2021. A theoretical analysis of the repetition problem in text generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12848–12856.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Nghia T. Le, Fan Bai, and Alan Ritter. 2022. Few-shot anaphora resolution in scientific protocols via mixtures of in-context experts. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*.

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.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The Winograd schema challenge. In *Proceedings of the Thirteenth international conference on the principles of knowledge representation and reasoning*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona Diab, Veselin Stoyanov, and Xian Li. 2022. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052, Abu Dhabi, United Arab Emirates. 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.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022. MetaICL: Learning to learn in context. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2791–2809, Seattle, United States. Association for Computational Linguistics.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

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

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. *Advances in neural information processing systems*, 34:11054–11070.

Ondřej Pražák, Miloslav Konopík, and Jakub Sido. 2021. Multilingual coreference resolution with harmonized annotations. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1119–1123.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Altaf Rahman and Vincent Ng. 2012. Resolving complex cases of definite pronouns: The Winograd schema challenge. In *Proceedings of the 2012 Joint*

Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pages 777–789, Jeju Island, Korea. Association for Computational Linguistics.

Milan Straka. 2023. ÚFAL CorPipe at CRAC 2023: Larger context improves multilingual coreference resolution. In *Proceedings of the CRAC 2023 Shared Task on Multilingual Coreference Resolution*, pages 41–51, Singapore. Association for Computational Linguistics.

Gorka Urbizu, Ander Soraluze, and Olatz Arregi. 2020. Sequence to sequence coreference resolution. In *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 39–46, Barcelona, Spain (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.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Xiaohan Yang, Eduardo Peynetti, Vasco Meerman, and Chris Tanner. 2022. What GPT knows about who is who. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*, pages 75–81, Dublin, Ireland. Association for Computational Linguistics.

Zdeněk Žabokrtský, Miloslav Konopik, Anna Nedoluzhko, Michal Novák, Maciej Ogrodniczuk, Martin Popel, Ondrej Prazak, Jakub Sido, and Daniel Zeman. 2023. Findings of the second shared task on multilingual coreference resolution. In *Proceedings of the CRAC 2023 Shared Task on Multilingual Coreference Resolution*, pages 1–18, Singapore. Association for Computational Linguistics.

Wenzheng Zhang, Sam Wiseman, and Karl Stratos. 2023. Seq2seq is all you need for coreference resolution. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 11493–11504, Singapore. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

A Appendix

A.1 Results

Table 2 presents official F1 scores on 21 test sets in comparison with the scores achieved by the baseline and the winning *straka-twostage* [3] model.

Data	Ours	Bsl.	Best
avg. (place)	33.38 (6)	53.16 (5)	73.90 (1)
ca_ancora	34.77	68.32	82.22
cs_pcedt	32.89	64.06	74.85
cs_pdt	30.88	63.83	77.18
cu_proiel	22.52	24.51	61.58
de_parcorfull	23.07	47.21	69.53
de_potsdamcc	45.85	55.65	71.79
en_gum	35.49	63.19	75.66
en_litbank	46.59	63.54	79.60
en_parcorfull	32.69	33.08	68.89
es_ancora	37.76	69.58	82.46
fr_democrat	36.34	53.62	68.16
grc_proiel	25.87	28.76	71.34
hbo_ptnk	37.96	24.60	72.02
hu_korkor	23.53	35.14	63.17
hu_szegedkoref	33.85	54.51	69.97
lt_lcc	42.73	62.00	75.79
no_bokmaalnarc	37.92	64.96	79.81
no_nynorsknarc	35.69	63.70	78.01
pl_pcc	27.19	66.24	78.50
ru_rucor	47.79	65.83	83.22
tr_itcc	9.65	44.05	68.18

Table 2: F1 scores on the test data.

A.2 Input length impact

As our approach struggles with cluster assignment in longer text sequences, we limit the input length by five sentences up to 512 sub-tokens in total. This leads to the following problems. First, long distance coreference cannot be recovered. Second, certain clusters get split into two or more clusters. Third, the number of singletons grows. To see how many clusters get lost due to such document splitting, we perform an experiment, where we first split the gold data into pieces keeping all the annotations, and then merge them back trying to restore the clusters. Table 3 shows the results for eight development datasets out of 21 official ones. The numbers clearly indicate that even the perfect system will be able to achieve only 84.58 F1 score on average, if its input is limited by five sentences.

[3] It is an updated version of the model presented in Straka (2023)

Data	w sngl.	w/o sngl.
avg.	84.58	82.89
ca_ancora	89.29	90.97
en_gum	89.34	81.67
hbo_ptnk	91.77	82.72
hu_korkor	86.07	86.86
lt_lcc	83.63	86.68
pl_pcc	92.77	85.86
ru_rucor	73.38	76.73
tr_itcc	70.40	71.63

Table 3: F1 scores on the gold development data with and without singleton clusters.

One of the obvious solutions to the problem would be to use a larger pre-trained model that is capable of processing longer inputs. Also, it is important to set the number of sub-tokens as the main constraint, and not the number of sentences, as sentences can be very short in some datasets.

A.3 Data augmentation

The examples below illustrate how we modify the gold coreference annotations in order to create our training data. The gold annotation examples are taken from the *en_gum* corpus.

Gold annotations: *Thus , [0 the time [1 it 1] takes 0] and [2 the ways of visually exploring [3 an artwork 3] 2] can inform about [4 [3 its 3] relevance 4] , [5 interestingness 5] , and even [6 [3 its 3] aesthetic appeal 6] . [7 This paper 7] describes [8 a collaborative pilot project 8] focusing on [9 a unique collection of [10 [11 17th Century 11] [12 Zurbarán 12] paintings 10] 9] . [9 The [13 Jacob 13] cycle at [14 [15 Auckland 15] Castle 14] 9] is [9 the only [16 UK 16] example of [17 a continental collection preserved in situ in [18 purpose - built surroundings 18] 17] 9] .*

Example A.1. Discarding clusters
Thus , the time [0 it 0] takes and the ways of visually exploring [1 an artwork 1] can inform about [2 [1 its 1] relevance 2] , [3 interestingness 3] , and even [1 its 1] aesthetic appeal .

Example A.2. Merging clusters
Thus , [0 the time [1 it 1] takes 0] and [0 the ways of visually exploring [3 an artwork 3] 0] can inform about [1 [3 its 3] relevance 1] , [5 interestingness 5] , and even [3 [3 its 3] aesthetic appeal 3] .

Example A.3. Splitting clusters
Thus , [0 the time [1 it 1] takes 0] and [2 the ways of visually exploring [3 an artwork 3] 2] can inform

about [4 [7 its 7] relevance 4] , [5 interestingness 5] , and even [6 [7 its 7] aesthetic appeal 6] .

Example A.4. Mixing clusters
Thus , [0 the time [1 it 1] takes 0] and [2 the ways of visually exploring [9 an artwork 9] 2] can inform about [4 [9 its 9] relevance 4] , [5 interestingness 5] , and even [6 [9 its 9] aesthetic appeal 6] . [7 This paper 7] describes [8 a collaborative pilot project 8] focusing on [3 a unique collection of [10 [11 17th Century 11] [12 Zurbarán 12] paintings 10] 3] . [3 The [13 Jacob 13] cycle at [14 [15 Auckland 15] Castle 14] 3] is [3 the only [16 UK 16] example of [17 a continental collection preserved in situ in [18 purpose - built surroundings 18] 17] 3] .

Example A.5. Inserting lonely cluster IDs
Thus , [0 the [0 0] time [1 it 1] 1] [0 0] takes 0] and [0 0] [2 the ways of visually exploring [3 an artwork [3 3] 3] 2] 2] [3 3] can inform about [4 [3 its 3] [3 3] relevance 4] 4] , [5 [2 2] interestingness 5] , and even [6 [4 [3 its 3] [4 4] aesthetic appeal 6] .

Example A.6. Inserting repetitions
Thus , [0 the time [1 it 1] [1 it 1] [1 it 1] takes 0] and [2 the ways of visually exploring [3 an artwork 3] 2] can inform about [4 [3 its 3] [3 its 3] relevance 4] , [5 interestingness 5] , and even [6 [3 its 3] [3 its 3] [3 its 3] [3 its 3] [3 its 3] aesthetic appeal 6] .

Example A.7. Violating mention boundaries
Thus , [0 [1 the time it 1] takes and [2 0] the ways of [3 visually exploring an artwork 3] 2] can inform about [4 [3 its 3] relevance [5 4] , interestingness 5] , and even [6 [3 its 3] aesthetic appeal 6] . This paper describes a collaborative pilot project focusing on a unique collection of 17th Century Zurbarán paintings .

Additionally, we modify original coreference annotations in short text sequences (containing up to three sentences) inserting empty brackets.
Gold: *[0 Aesthetic Appreciation 0] and [1 Spanish Art 1] : [2 Insights from [3 [4 Eye 4] - Tracking 3] 2]*

Example A.8. Adding empty brackets
[0 Aesthetic Appreciation [0] and [1 Spanish Art 1] : Insights from Eye - Tracking

A.4 Alignment

A generated sequence with cluster identifiers may differ a lot from the original string. It can contain hallucinated tokens, unnecessary repetitions, or be truncated. Such problems are especially frequent when smaller models like *mT5-base* are used. In such cases it is impossible to extract the correct mention indices, even if certain mentions were correctly identified and clustered. To solve this problem, we try to transform the generated text into the gold one, keeping the cluster indices.

We use the *difflib* library for this task. The algorithm splits the generated string into parts and suggests one of the four actions, namely *'insert'*, *'delete'*, *'replace'* and *'equal'*, to be performed for each part. If *'delete'* or *'replace'* actions are chosen, we follow the commands, but keep all the cluster identifiers, no matter wrong or correct. We show how a generated text can be aligned with the gold one below. Example A.9 illustrates the steps of the sequence matcher from *difflib*.
Generated text: *Thus , [0 [1 the time it 1] takes and [2 0] the ways of [3 visually exploring an artwork 3] 2] can inform about [4 [3 its 3] relevance [5 4] relevance [5 4] relevance [5 4] relevance [5 4] relevance [5 4]*
Gold text: *Thus , the time it takes and the ways of visually exploring an artwork can inform about its relevance , interestingness , and even its aesthetic appeal .*

Example A.9. Alignment

equal	*"Thus ," → "Thus ,"*
delete	*"[0 [1" → ""*
equal	*"the time it" → "the time it"*
delete	*"1]" → ""*
equal	*"takes and" → "takes and"*
delete	*"[2 0]" → ""*
equal	*"the ways of" → "the ways of"*
delete	*"[3" → ""*
equal	*"visually exploring an artwork" → "visually exploring an artwork"*
delete	*"3] 2]" → ""*
equal	*"can inform about" → "can inform about"*
delete	*"[4 [3" → ""*
equal	*"its" → "its"*
delete	*"3]" → ""*
equal	*"relevance" → "relevance"*
replace	*"[5 4] relevance [5 4] relevance [5 4] relevance [5 4] relevance [5 4]" → ", interestingness , and even its aesthetic appeal ."*

Aligned result: *Thus , [0 [1 the time it 1] takes and [2 0] the ways of [3 visually exploring an artwork*

3] 2] can inform about [4 [3 its 3] relevance [5 4] [5 4] [5 4] [5 4] [5 4] , interestingness , and even its aesthetic appeal .

Note that the aligned text above contains some excessive cluster identifiers, and certain mention boundaries are wrong. We discard all the opening brackets that cannot be properly closed later during the post-processing.