# Fine-grained quotation detection and attribution in German news articles

**Fynn Petersen-Frey and Chris Biemann**
Hub of Computing and Data Science & Language Technology Group
Universität Hamburg
{fynn.petersen-frey, chris.biemann}@uni-hamburg.de

## Abstract

The task of quotation detection and attribution deals with identifying quotation spans together with their associated role spans such as the speaker. We describe an approach to solve the task of fine-grained quotation detection and attribution using a sequence-to-sequence transformer model with constrained decoding. Our model improves vastly upon the existing baselines on the German news articles quotation dataset, thereby making it feasible for a first time to automatically extract attributed quotations from German news articles. We provide an extensive description of our method, discuss alternative approaches, performed experiments using multiple foundation language models and method variants, and analyzed our model's prediction errors. Our source code and trained models are available.[1]

## 1 Introduction

Identifying who says what to whom is the central piece in analyzing written human communication. It enables scientists or journalists to analyze how a discourse changes over time, which people participate in a discourse, what their points of view are and much more. With today's ever-increasing amounts of data such as online news articles, it is typically not feasible to manually process the wealth of data that is of interest for a specific research question.

### 1.1 Task description

The task of quotation detection and attribution deals with identifying quotation spans (*Quote*) in a document together with their associated role spans such as the *Speaker* or *Addressee*. While early, simple variants of the task only considered *Direct* quotations spanning at most a single sentence, the task's complexity has increased over time (see Section 2.1). We base our work on our recent dataset (Petersen-Frey and Biemann, 2024) that also features a significantly more complex task that requires a fine-grained quotation detection of five quotation types and attribution of up to four roles per *Quote*.

Example 1.1 shows a human-friendly representation of an annotated text passage. It contains two quotations (*Direct* and *Indirect*) uttered by the same *Speaker*. The *Indirect* quotation is further invoked by a *Cue* word "said" within a *Frame* in the same sentence. A quotation with all its associated roles is called a *Quote* group. The integers behind each span type indicate which group that span belongs to. In case of the *Speaker* "someone", it belongs to both the first and second group.

**Example 1.1 (Annotated text passage)**

*Most sentences do not contain a quotation.*
*This is an indirect quote, said someone.*
*Indirect 1* — *Cue 1* *Speaker 1,2* — *Frame 1*
*"Followed by a direct quote in a new sentence."*
*Direct 2*

In the fine-grained task variant and dataset (Petersen-Frey and Biemann, 2024), a single quotation can span multiple sentences, each quotation can be associated with zero to four roles, *Quote* groups can be nested inside each other, the same annotation span can belong to multiple groups and annotation spans can be discontinuous. We solve the full task with all its challenges and predict *Direct*, *Indirect*, *Reported*, *Free Indirect* and *Indirect/Free Indirect* quotations together with the *Speaker*, *Cue*, *Addressee* and *Frame* roles.

### 1.2 Example use cases

Reliably extracting *Quote* groups from a large document collection allows researchers or journalists to quickly analyze the quotations contained in their data of interest. The type of quotation and existence of certain roles can be used to filter and/or

[1] Source code and model weights: https://github.com/uhh-lt/seq2seq-quotation-attribution



*Proceedings of the 20th Conference on Natural Language Processing (KONVENS 2024)*, pages 196–208
September 10-13, 2024 

aggregate quotations enabling both a quantitative view and finding individual occurrences to analyze in detail. To compare different news outlets, time frames or topics, the fine-grained annotations enable corpus comparisons with statistics such as the distribution of quotation types, fraction of quotations with or without a *Speaker* or *Addressee*.

In this paper, we describe our approach to automatically identify who said what to whom in German news articles. Our model improves vastly upon the existing baselines and makes it possible for a first time to automatically extract fine-grained, attributed quotations with high precision and recall from plain text.

## 2 Related work

We first review previous work on quotation detection and attribution for English and German. Then, we review approaches to structured generation as an essential component to our chosen approach.

### 2.1 Quotation detection and attribution

The NewsExplorer system (Pouliquen et al., 2007) is the first system to tackle the task of *Direct* quotation detection and attribution to a *Speaker*. Krestel et al. (2008) also detected *Indirect* quotations using rule-based reported verb and speech finder. O'Keefe et al. (2012) created a new labeled dataset and reformulated *Direct* quotation detection and attribution as a sequence labeling task. Pareti et al. (2013) focus on the more challenging *Indirect* quotations by training classifiers for the *Cue*, *Speaker* and *Quote*. Almeida et al. (2014) introduced a model that jointly solves the problems of quotation attribution and coreference resolution. Newell et al. (2018) created the Citron software that implements an improved variant of the approach by Pareti et al. (2013) using trainable content resolver and source resolver. Zhang and Liu (2022) focus only on *Direct* quotations and test multiple sequence labeling methods including three neural models.

While quotation detection and attribution in English news has been addressed by many works, less have dealt with German news. Bögel and Gertz (2015) created a rule-based system using dependency trees to extract and attribute *Direct* and *Indirect* quotations from German news articles. Papay and Padó (2019) created a corpus-agnostic, neural quotation detection model that can detect *Direct* and *Indirect* quotations while omitting *Cue* and *Speaker*. Brunner et al. (2020b) trained simple sequence taggers on the Redewiedergabe corpus (Brunner et al., 2020a) to individually detect *Direct*, *Indirect*, *Reported* and *Free Indirect* quotations without any roles or attribution.

### 2.2 Structured generation

Conditional language modeling has become a useful technique to tackle structured prediction tasks using pre-trained language models. The target structure is flattened into a sequence and a conditional language model is trained to predict it. Paolini et al. (2021) solve structured prediction language tasks (e.g. joint entity/ relation extraction and SRL) by performing translation between augmented natural languages. Liu et al. (2022) argue that flattening structured information leads to inferior performance. They model structures as action sequences and achieve a new state-of-the-art on named-entity recognition, relation extraction and coreference resolution. Geng et al. (2023) suggest grammar-constrained decoding without fine-tuning can solve many structured NLP tasks and show this for information extraction, entity disambiguation and constituency parsing. Zhang et al. (2023) show that task-specific models are not necessary for state-of-the-art coreference resolution and train a model to translate the input to a sequence encoding the coreference information.

## 3 Method

We frame the task of detecting and attributing quotations as a special clustering task with labeled text spans. Each cluster contains a single *Quote* that may consist of multiple spans. Further, a cluster may contain multiple spans of the different roles, e.g. *Cue*, *Frame*, *Speaker*, *Addressee*. In contrast to typical clustering, the same role span may belong to different clusters. We indicate this by applying multiple IDs to the same span.

On a high level, our method to solve the task of detecting and attributing quotations with a sequence-to-sequence model works as follows:

1. Training data pre-processing: Transform the original text and annotated cluster information to a suitable linearized token representation as a training target.
2. Training: Train the sequence-to-sequence model to predict the linearized token representation from the original input text.
3. Inference: During generation, constrain the model to produce a valid linearized output

from original input text.

4. Post-processing: Transform the linearized representation back to the cluster information with token offsets in the original text.

### 3.1 Linearization strategies for clustered quotation annotations

We transform the clustering information into a sequential representation to employ a standard sequence-to-sequence model to solve the task. Our method is inspired by Zhang et al. (2023), who show that it is feasible to perform state-of-the-art coreference resolution using standard sequence-to-sequence models with the appropriate textual representations. They experimented with a number of approaches to transform coreference mention spans and their clusters into linear text. With full linearization, the full input text is reproduced in the output – either as tokens or using a special copy action that forces the model to predict the token from the original input sequence to prevent deviations between the input and target sequences.

With partial linearization, only the tokens being part of a mention span are reproduced in the output. This mode is incompatible with the copy action, potentially has alignment issues when matching the predicted output back to the original text and achieves slightly worse coreference resolution scores. Its advantage is a significantly shorter output length. As generating long token sequences is a slow and computationally expensive process with non-linear runtime, partial linearization provides a way to mitigate this issue.

As such, we focus on the partial linearization outputting tokens with sentence markers for quotation detection and attribution to create a model useable with fewer computational resources. We call this method **partial token linearization**. Since full linearization with the copy action had the best results for coreference resolution, we use this as our second method for quotation detection and attribution named **full copy linearization**.

Zhang et al. (2023) also experimented with multiple techniques to include the cluster ID for each span. For both our methods, we re-use their best-working policy and include the cluster ID with a separation token | and a plain text integer number right before the closing tag of every span.

### 3.2 Forward transformation

Our transformation uses two special tokens per span type for both linearization methods (in addition to the separation | and sentence marker <sent>, </sent> tokens). We insert specific tokens that indicate the start resp. end of a span. Table 1 shows all special tokens used for our method. In total, we add 20 special tokens for full copy linearization, 21 special tokens for partial token linearization. For both, we also add 500 integer numbers for the cluster IDs as special tokens. To explain the forward transformation, we continue with Example 1.1. We apply both linearization methods to produce a sequence out of this cluster information.

| Special token | start | end |
|---|---|---|
| *Cue* | <cue> | </cue> |
| *Addressee* | <addr> | </addr> |
| *Speaker* | <speaker> | </speaker> |
| *Frame* | <frame> | </frame> |
| *Direct* | <direct> | </direct> |
| *Indirect* | <indirect> | </indirect> |
| *Reported* | <reported> | </reported> |
| *Free Indirect* | <frin> | </frin> |
| *Indirect/Free Indirect* | <infrin> | </infrin> |
| Copy from input | <cp> | |
| Cluster separation | \| | |
| Sentence marker | <sent> | </sent> |
| Cluster IDs | 0 | 499 |

Table 1: Special tokens used

Example 3.1 shows the partial token linearization sequence. All tokens from the original text that are not part of any span are removed. For each sentence, a sentence marker is inserted at the beginning and end of it. Likewise, each span is marked using a special token directly before the text span begins. Spans are ended with a three-token sequence: Cluster ID separation token |, cluster ID, span closing token.

**Example 3.1 (Partial token linearization)**

<sent> </sent> <sent> <indirect> *This is an indirect quote* | 1 </indirect> <frame> , <cue> *said* 1 | </cue> <speaker> <speaker> *someone* | 1 </speaker> | 2 </speaker> | 1 </frame> </sent> <sent> <direct> *"Followed by a direct quote in a new sentence."* | 2 </direct> </sent>

Example 3.2 shows the full copy linearization sequence. Annotated spans are handled identical to the partial token linearization. In contrast, it does not contain sentence markers and all originals tokens are replaced with the special <cp> copy token. The copy token enforces that the token is copied directly from the input.

**Example 3.2 (Full copy linearization)**

<cp> <cp> <cp> <cp> <cp> <cp> <cp> <cp> <indirect> <cp> <cp> <cp> <cp> <cp> | 1 </indirect> <cp> <frame> <cp> <cue> <cp> 1 | </cue> <speaker> <speaker> <cp> | 1 </speaker> | 2 </speaker> | 1 </frame> <cp> <direct> <cp> <cp> <cp> <cp> <cp> <cp> <cp> <cp> <cp> <cp> <cp> | 2 </direct>

For both linearization methods, nested spans are handled by first opening the span that ends later. As seen in Example 1.1, the *Frame* and *Cue* start at the same token. This results in a linearized sequence such as <frame> <cue> . . . </cue> . . . </frame>. It enables a precise reconstruction of the original clustering information as long as opening and closing brackets match. To ensure this, we use constrained decoding.

### 3.3 Constrained Decoding Strategy

Sequences-to-sequence models can generate arbitrary output that might not be possible to be converted back to a cluster representation. To prevent this and enforce valid outputs, we employ constrained decoding by masking the log-probabilities of vocabulary entries (see Daza and Frank (2018); De Cao et al. (2021)). We use beam search to improve generation quality. While it complicates the constrained decoding implementation, beam search has no effect on the rules we enforce.

**Partial token linearization** Most of the time, we allow the model to generate any normal vocabulary subword token, any special start-span token (so spans can be nested) or the | token used to separate span text from the cluster ID. However, all these are forbidden after the | token has been generated before a special end-span token is issued. During this short time, only the cluster ID and the correct end-span token can be generated by the model. The correct end-span token is the closing "bracket" for the most recently started span that is yet unclosed.

**Full copy linearization** We never allow the model to generate a normal token. Instead, we only allow the copy token to copy from the input or use the other special tokens to create spans resp. clusters. Similar to the first setting, we usually allow the model to generate the copy token, any start-span or cluster ID separation token. The copy token is replaced with the actual token from the input when generating the next token.

### 3.4 Reverse transformation

Our system extracts a token offset-based cluster representation from the linearized sequence to identify the predicted spans and clusters in the original plain text.

**Full copy linearization** Since the constrained decoding as described above guarantees a valid output, we can transform the linearized sequence back to a cluster representation. As described in Section 3.2, nested spans are handled in a way so that it is always clear to which span a cluster ID belongs and which span is ended. To obtain the span begin and end offsets, we keep a stack of currently opened spans and close them in reverse order. Offsets are only affected by the number of copy tokens as only they correspond to sub-words in the original text; all other tokens are disregarded for this purpose.

**Partial token linearization** To find the offsets in the original text, the partial linearization requires an alignment by finding the sequence of predicted tokens of a span in the input text. We follow Zhang et al. (2023) who used Gotoh's algorithm (Gotoh, 1982) together with sentence markers to efficiently find an optimal alignment. The sentence markers constrain the alignment to pairs of sentences.

### 3.5 Alternative methods considered

To solve the task of quotation detection and attribution, we evaluated multiple approaches before we developed the system using a task-agnostic sequence-to-sequence model. In this section, we briefly describe what alternative approaches we considered and why we decided against implementing them.

**Semantic role labeling (SRL)** SRL deals with discovering the predicate-argument structure of a sentence. This task is related to quotation detection and attribution as e.g. *Cue*, *Speaker*, *Addressee* and *Quote* can potentially be seen as predicate-argument groups. An apparent solution for the task is to re-use an existing SRL system. However, they are strictly designed around the SRL task where roles must be identified for a single predicate (*Cue*) within a single sentence. This setup is incompatible with the task of general quotation detection and attribution where *Quotes* can occur without any *Cue* and often span multiple sentences.

**Coreference resolution with tagging** Coreference resolution is the task of resolving mentions (text spans) that refer to the same entity by identifying and clustering the mentions. When considering neural models, the task is very similar to quotation detection and attribution: Identifying and clustering spans of texts across an entire document. The primary difference is coreference resolution uses unlabeled spans, while the quotation task has spans with different labels. Thus, another potential solution is to re-use a task-specific coreference model and combine it with a span tagging model. The coreference part can already handle nested spans and the clustering. However, the labeling part would need to be deeply integrated in the coreference resolution model. Consequently, the system would no longer predict unlabeled mentions with arbitrary antecedent relations to form clusters. Instead, it would need to predict labeled spans with a specific set of allowed relations. Coreference resolution models include many special cases to cope with the computational complexity that make them ill fit for quotation detection and attribution. For example, word-level coreference (Kirstain et al., 2021; Dobrovolskii, 2021) would be incompatible with the quotation detection task as the same token would need to be used for multiple spans. Most architectures typically set a maximum span length to achieve practical computational complexity. This would also be an issue for the quotation attribution as the quotation spans vary greatly in length and can be much longer than a mention.

**Creating a new task-specific model architecture** We also decided against creating a new task-specific model architecture for two reasons. First, this approach makes the system dataset-specific to a large degree; thereby making the re-use of our model for similar datasets much more challenging. Second, the approach of creating task-specific model architectures for every NLP task is cumbersome, time-consuming and dated compared to a modern, more generic solution.

## 4 Experiments

### 4.1 Data

We use the dataset with attributed quotations in German news articles described in Petersen-Frey and Biemann (2024) for our experiments. It consists of 998 news articles split into 700 for training, 150 for development and 148 for test. We present an overview in Table 2. All 4,182 annotated quotation groups contain a *Quote*. While all roles are optional, most groups also contain a *Speaker*. For more details refer to Petersen-Frey and Biemann (2024). The data is available as JSON with the tokenized text and grouped annotations specified with token offsets for the span start and end. We transform this data into different sequential representation for the training routine depending on the linearization method and the foundation language model's tokenizer.

| | count | avg. len. |
|---|---|---|
| Documents | 998 | 249.0 |
| Sentences | 13 186 | 18.8 |
| Tokens | 248 480 | |
| *Quote* | 4182 | 16.7 |
| *Direct* | 873 | 17.5 |
| *Indirect* | 2 250 | 14.7 |
| *Reported* | 454 | 18.0 |
| *Free Indirect* | 171 | 20.4 |
| *Indirect/Free Indirect* | 434 | 22.3 |
| Roles | 10 212 | |
| *Speaker* | 3 908 | 3.5 |
| *Cue* | 2 929 | 1.6 |
| *Frame* | 3 038 | 9.0 |
| *Addressee* | 337 | 2.7 |

Table 2: Dataset overview

### 4.2 Variants

We perform extensive experiments on a number of combinations of foundation language model, model size and linearization method. For the foundation language models, we test T5 (Raffel et al., 2020), T5 v1.1, Flan-T5 (Chung et al., 2022) as well as the multilingual mT5 (Xue et al., 2021). We test all models in three sizes: base ($\approx$ 250 million parameters), large ($\approx$ 800 million parameters), xl ($\approx$ 3 billion parameters). While there are even larger models available, we do not have the computational resources to train such large models. Further, we test both of our linearization methods: Full copy and partial token linearization. In total, we test $4 * 3 * 2 * 2 = 24$ combinations for our models. We could not obtain results on the two xl variants of the original T5 model because the computation ran into timeout errors on our available hardware. Thus, we only report results on the remaining 22 combinations.

### 4.3 Training and evaluation details

To enable efficient training in batches with low amount of padding and limited memory use, we use a sliding window approach and slice the training documents into chunks of 2048 subword tokens with an overlap of half the length. The evaluation and test documents are not sliced and are fed as whole into the system. Evaluation uses generation with 4 beams to improve prediction results.

We train each model for 100 epochs and use early stopping to prevent overfitting by evaluating on the development set. Weight decay of 0.01 is further used as a regularization method. For the base and large models we use a learning rate of $5 \cdot 10^{-4}$, for the xl models a reduced learning rate of $5 \cdot 10^{-5}$ as the models would not converge otherwise. AdamW (Loshchilov and Hutter, 2019) is used as the optimizer. To help the model adapt to it's drastically changed task objective, we use a learning rate warmup of 0.1.

We train with batch sizes of 8 using a single A100 GPU for base/large models. For the xl models, we use two A100 GPUs with DeepSpeed (Rasley et al., 2020) ZeRO-2 (Rajbhandari et al., 2020) and a batch size of 4 per device, resulting in a total batch of 8. In all cases, we use gradient checkpointing and train the models in bfloat16.

### 4.4 Evaluation Metrics

We use the evaluation method as described in Petersen-Frey and Biemann (2024). The method is based on the precision, recall and F1-metrics on individual tokens of corresponding spans in matched clusters. Consequently, a predicted role span can only be matched to the gold span if they belong to a matched cluster. While unmatched predicted spans increase the false positives for this type, unmatched gold spans increase the corresponding false negatives. Clusters are matched via linear sum assignment of the fraction of token overlap on the *Quote* span. Correctly matched clusters produce true positives for all correct roles and quote spans according to the set intersection of tokens and false negatives resp. false positives for two set differences. Precision, recall and F1 are provided both independently for the quotations and roles as well as a joint metric.

### 4.5 Results

We present our main results summarized in Table 3. It shows the result of our best combinations per model size as compared to baselines for the development and test set across all metrics. We include both baseline systems from Petersen-Frey and Biemann (2024):

1. A rule-based system (RBS) build on top of neural components for dependency parsing, part-of-speech tagging, named-entity recognition etc.
2. Citron (Newell et al., 2018) consists of individually trained classifiers for *Cue*, *Speaker*, *Quote* and resolvers to group the predicted spans together.

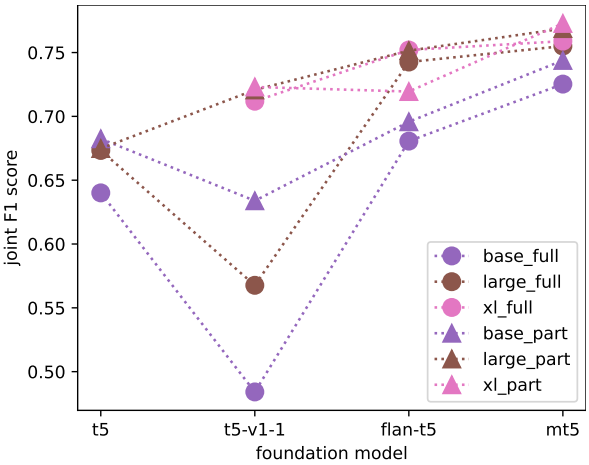


Figure 1: joint F1 score per foundation model

From our results in Table 3 it is immediately obvious that our trained models far exceed the two baselines. Our best models deliver almost a two times higher F1 score than the baselines. While the rule-base system (RBS) and Citron obtain a joint F1 score of 39.1 resp. 46.8 on the test set, our best model scores 80.8. The difference in performance mainly originates from the vast increase in recall; going from 29.0 resp. 33.0 to 77.3 for our model. While we achieve a substantial improvement on the precision compared to RBS (60.7 versus 84.6), we manage a slight improvement over Citron (82.4 versus 84.6) – albeit at an entirely different recall level. Regarding the prediction of the quotation type, our model again deliver almost twice the F1 score (44 vs 85). To put an F1 score of 80 into perspective: It corresponds to a nicely useable prediction from a manual evaluation of our model's outputs, although there are small errors from time to time. We provide some example outputs in Section 4.8.

### 4.6 Ablation study

In this section, we discuss the effect of the foundation model type, model size and linearization method.

| | | | quotation | | | roles | | | joint | | | type | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| model | sz | lin. | prec. | rec. | F1 | prec. | rec. | F1 | prec. | rec. | F1 | prec. | rec. | F1 |
| *development set* | | | | | | | | | | | | | | |
| RBS* | | | 75.1 | 36.1 | 48.8 | 55.0 | 25.5 | 34.9 | 60.7 | 28.7 | 38.9 | 57.8 | 29.6 | 39.1 |
| Citron* | | | **91.5** | 27.6 | 42.4 | 79.3 | 31.5 | 45.1 | 82.4 | 30.3 | 44.3 | 87.0 | 26.6 | 40.8 |
| mt5 | b | part | 85.5 | 75.2 | 80.0 | 83.1 | 71.4 | 76.8 | 83.8 | 72.5 | 77.8 | 84.9 | 74.3 | 79.2 |
| mt5 | l | part | 89.7 | 78.4 | **83.7** | 83.2 | 74.5 | 78.6 | 85.1 | 75.6 | **80.1** | 89.2 | 78.6 | **83.6** |
| mt5 | xl | part | 86.4 | **79.5** | 82.8 | 82.5 | 75.8 | 79.0 | 83.7 | 76.9 | **80.1** | 85.3 | 79.0 | 82.0 |
| *test set* | | | | | | | | | | | | | | |
| RBS* | | | 70.8 | 36.2 | 47.9 | 55.6 | 26.1 | 35.5 | 59.9 | 29.0 | 39.1 | 63.5 | 33.6 | 43.9 |
| Citron* | | | 88.2 | 30.1 | 44.9 | 77.9 | 34.2 | 47.6 | 80.5 | 33.0 | 46.8 | 86.5 | 29.6 | 44.1 |
| mt5 | b | part | 85.9 | 77.2 | 81.3 | 81.8 | 73.2 | 77.3 | 83.0 | 74.4 | 78.4 | 87.5 | 78.1 | 82.5 |
| mt5 | l | part | 88.8 | **81.1** | **84.8** | 83.0 | 75.2 | 78.9 | 84.7 | 76.9 | 80.6 | 89.5 | **81.0** | **85.0** |
| mt5 | xl | part | 88.2 | 80.0 | 83.9 | 83.2 | **76.2** | **79.5** | 84.6 | **77.3** | **80.8** | 88.5 | 80.2 | 84.2 |

*Baseline results taken from Petersen-Frey and Biemann (2024)

Table 3: Selected evaluation results (baselines and our best model combination per model size). The column *sz* is short for size, with *b* for base and *l* for large. Complete results are shown in Table 5 and 6 in the appendix.

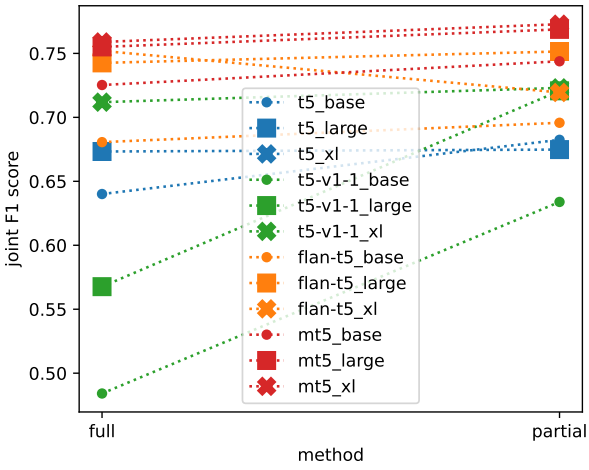


Figure 2: joint F1 score per linearization method

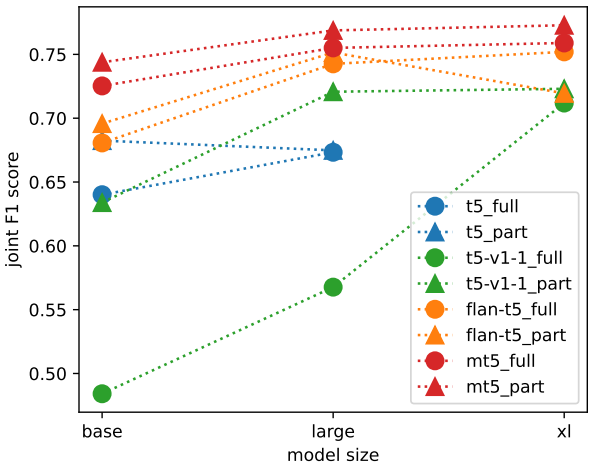


Figure 3: joint F1 score per model size

Figure 1 makes it easy to compare the performance across different foundation models. We can see a clear trend: mT5 outperforms Flan-T5 over all combinations of size and linearization method. Flan-T5 in turn outperforms T5 and T5 version 1.1 over all combinations. T5 version 1.1 behaves oddly in comparison to the other three foundation models: It exhibits a rather large difference of over 20 F1 points between its best and worst combinations. We attribute this to the fact this specific foundation model was only pre-trained using language modeling excluding any supervised training on other tasks.

Figure 2 compares the performance across our two tested linearization methods: Full copy linearization and partial token linearization. In all but one cases, we see an increase in performance using the partial token linearization. The two smaller T5 version 1.1 models profit substantially from using the partial linearization. For most models, it provides a low, consistent performance boost – only for Flan-T5 XL the full copy linearization performs slightly better. Albert not directly comparable, this is in contrast to the results of Zhang et al. (2023), who reported a slight performance decrease in coreference resolution by using their partial linearization method. While the task of coreference resolution seems to be affected by the alignment issue inherent to any partial linearization, our task of

| size | lin. | joint F1 | time | time batched |
|---|---|---|---|---|
| base | full | 78.3 | 44m | 24m |
| base | part | 78.4 | 23m | 13m |
| large | full | 80.3 | 82m | 60m |
| large | part | 80.6 | 40m | 31m |
| xl | full | 78.7 | 90m | - |
| xl | part | 80.8 | 49m | - |

Table 4: Inference runtime

quotation detection and attribution is less affected.

Figure 3 shows the effect of increased model size. While the large model variants noticeably outperform the base size equivalents for all but one combination, using the xl model instead of the large model only provides significant gains for the combination of T5 version 1.1 with full copy linearization. The Flan-T5 using the partial token linearization even shows a drop in performance. Consequently, we do not see a reason to use a xl model size as it requires significantly more computational resources than a base or large model.

### 4.7 Inference runtime efficiency

As we intend to use our system on large document collections, inference runtime and compute resource requirements play an important role. Consequently, we evaluated the runtime efficiency across the three model sizes and both linearization methods. Tests were performed using the mT5 models with batch sizes of 1 and 8 (only base and large). We ran the inference measurements on a single A100 GPU (80 GB) and predicted the development and test set together (298 documents). Table 4 shows the total time usage when predicting all documents including post-processing. Across all model and batch sizes, the partial token linearization is roughly twice as fast as the full copy linearization resp. needs only half of the computational resources. For the base model, batching is highly recommended as long as enough memory is available because it also reduces the inference time in half compared to no batching. The large model is 25% faster with batching. The base model is roughly twice as fast as the large model. Using a smaller model has the additional advantage of a lower memory usage allowing higher batch sizes with potentially even faster inference. The xl model is only 25% slower than the large model without batching because the xl model utilizes the GPU better.

When considering the quality of the predictions together with the required computational resources, only two options should be considered: Partial token linearization with either a base or large model depending on the preference for performance or quality. Across all scenarios, the full linearization performs slightly worse than partial linearization and takes roughly twice as long to compute.

### 4.8 Error Analysis

We performed a manual verification of our best model's output on both the development and test set. Beside the comparison with the gold annotations, we also checked whether false positives might be annotation errors. We found that the system sometimes has difficulties with the *Addressee* role: It often predicts the span as a *Speaker* instead. This is somewhat expected since *Addressee* is tiny minority class in the dataset. Similarly, *Free Indirect* is also a rare class and quotations of this type are either not predicted or used in debatable situations where the gold annotations did annotate the span, e.g. when the grammatical structure matches, but it represents the opinion of the article author.

It also occurred that multiple *Speaker* spans were predicted for a single *Quote* whereas the gold annotations only contain one *Speaker*. Further, the system sometimes predicts very short *Direct* quotes not annotated in the gold data. While some of these could be declared as annotation errors, most of these errors are justified. However, we also encountered multiple documents where the system predicted full quotations with roles that were most likely missed during the annotation.

Example 4.1 shows the predictions on a random document from the dev set. The human annotations for the same document are shown in Example 4.2. Our system predicts two false positive quotations according to the curated annotations: The first is an edge case as someone very likely uttered the predicted quote but the sentence is written as a description of an action. The second quotation was likely missed during the annotation. However, the identified *Speaker* and *Cue* are partly wrong, only *Nachricht* (message) should be the *Speaker* and *Cue*. The third predicted quotation and it's roles are identical to the gold annotation. In the last quotation, the system's prediction is identical except for potentially false positive *Addressee* – another edge case where one could argue the BBC was the *Addressee* during the interview.

**Example 4.1 (system prediction)**

. . .
*Jetzt bereiten sich die lokalen Autoritäten darauf*
*Cue 0* *Frame 0* *Speaker 0* *Cue 0*
*vor den Tourismusansturm in geordnete Bahnen*
*Indirect 0*
*zu lenken, um die Einnahmen zum Schutze der Umgebung des Wasserfalls zu verwenden.*
. . .
*Auch er zeigte sich überrascht und stolz von der*
*Speaker 1* *Cue 1* *Frame 1* *Cue 1*
*Nachricht, dass nur wenige Kilometer von*
*Indirect 1*
*seinem Haus einer der höchsten Wasserfälle der Welt liegt.*
. . .
*Er hofft, dass für sein Dorf etwas von den zu*
*Speaker 2* *Cue 2* *Indirect 2*
*Frame 2*
*erwartenden Einnahmen abfällt, denn bisher gibt es nicht einmal ein Telefon, um mit seinen weit entfernt wohnenden Kindern zu kommunizieren.*
. . .
*In einem Interview mit der BBC erklärte er:*
*Frame 3* *Addressee 3* *Cue 3* *Speaker 3*
*„Der Anblick dieses Wasserfalls ist einfach spektakulär.“* *Direct 3*

**Example 4.2 (gold annotation)**

. . .
*Er hofft, dass für sein Dorf etwas von den zu*
*Speaker 0* *Cue 0* *Indirect 0*
*Frame 0*
*erwartenden Einnahmen abfällt, denn bisher gibt es nicht einmal ein Telefon, um mit seinen weit entfernt wohnenden Kindern zu kommunizieren.*
. . .
*In einem Interview mit der BBC erklärte er:*
*Frame 1* *Cue 1* *Speaker 1*
*„Der Anblick dieses Wasserfalls ist einfach spektakulär.“* *Direct 1*

Another class of errors is that a role span is sometimes not used for multiple quotations, although the system correctly predicted all quotations – but without linking a *Speaker* etc. On a positive note, the system is perfectly capable of handling interrupted quotations and linking roles in a different sentence. Most nested quotations are also handled correctly, so the nesting itself does not appear to be a problem.

In general, it can be said that although the predictions are not perfect, but they are reasonable and very much useable both for an in-depth analysis of extracted groups and a quantitative analysis.

## 5 Conclusion

We have presented the first model for fine-grained, high-quality quotation detection and attribution in German news articles. The system allows to automatically identify who said what to whom. Our model can predict five different types of quotations together with the four different roles and connect these together. The model is based on a sequence-to-sequence transformer architecture generating structured, linearized output from plain text using constrained decoding. We described our method in detail, evaluated our two linearization methods across multiple foundation models and model sizes, performed an ablation study showing the importance of choosing the right foundation model, and performed a manual error analysis. Our models deliver a very strong performance on both a manual verification of the outputs and the evaluation metrics, almost doubling the scores of the available baselines.

Our system can be used for a range of use cases in the digital humanities, computational social sciences and journalism. Especially when combined with additional NLP tasks such as coreference resolution and entity linking, identified quotations can be easily grouped and analyzed, thereby providing researchers and journalists new means to work with quotations in large document collections.

In the future, we look forward to even multilingual foundation models that will likely further improve the quality of models using our approach.

## Ethical Considerations

Relying on automation to solve a task always introduced room for issues. The models may have (unknown) biases due to its training data. The foundation models have been pre-trained on a huge amount of diverse web texts (see Raffel et al. (2020); Chung et al. (2022); Xue et al. (2021) for details). Our task-specific fine-tuning is performed using a dataset was created from a random sample of data of an open and freely available source that has been manually annotated and curated. During manual evaluation of our model's prediction, we did not encounter apparent biases such as a preference of gender for the speaker. Another potential issue with automation of an information extraction task is low recall. While precision is easy to check manually (by checking whether the system's prediction is valid), a low recall is problematic for certain use cases as there is so way to efficiently

verify that the model detected most quotations in a large article collection. In our previous work (Petersen-Frey and Biemann, 2024), we reported a low recall for the baseline systems. We evaluated our models on a held-out test set and they achieve a high recall across quotations and roles.

Automation using machine learning can often open the door for misuse. In our case, we do not see a direct issue as detecting quotations in text is not harmful. It can become an issue if the source articles contain false quotations (e.g. because the texts were generated) and the extracted quotations are blindly believed to be valid. However, it is already possible to simply generate a list of fake quotations using readily available generative models. Thus, we do not see our model making things worse than the current state. In contrast, it could be used in helping to identify fake articles by comparing the found quotes with either a database or more reputable sources.

## Limitations

We see two main limitations of our models. First, the computational resources required during inference are rather high as each generated token requires a pass through the decoder. We mitigate this to some extent by reducing the required output length with the partial token linearization. For documents that only include few quotations (or none at all) in the prediction, this makes the inference very quick. For documents with a high amount of quotations, it is still faster than the full copy linearization although less pronounced. The second issue is the amendable handling of rare events. Sometimes, deeply nested spans or span re-use across groups are not as well captured as they would be in isolation. Rare spans classes such a *Addressee* have a lower recall compared to common classes. More training data would help significantly with this issue.

## References


Mariana S. C. Almeida, Miguel B. Almeida, and André F. T. Martins. 2014. A joint model for quotation attribution and coreference resolution. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 39–48, Gothenburg, Sweden. Association for Computational Linguistics.

Thomas Bögel and Michael Gertz. 2015. Did i really say that? – combining machine learning and dependency relations to extract statements from german news articles. In *Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology*, pages 13–21, Duisburg-Essen, Germany. German Society for Computational Linguistics and Language Technology.

Annelen Brunner, Stefan Engelberg, Fotis Jannidis, Ngoc Duyen Tanja Tu, and Lukas Weimer. 2020a. Corpus REDEWIEDERGABE. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 803–812, Marseille, France. European Language Resources Association.

Annelen Brunner, Ngoc Duyen Tanja Tu, Lukas Weimer, and Fotis Jannidis. 2020b. To BERT or not to BERT - comparing contextual embeddings in a deep learning architecture for the automatic recognition of four types of speech, thought and writing representation. In *Proceedings of the 5th Swiss Text Analytics Conference and the 16th Conference on Natural Language Processing*, volume 2624 of *CEUR Workshop Proceedings*, Zurich, Switzerland. CEUR-WS.org.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *Preprint*, arXiv:2210.11416.

Angel Daza and Anette Frank. 2018. A sequence-to-sequence model for semantic role labeling. In *Proceedings of the Third Workshop on Representation Learning for NLP*, pages 207–216, Melbourne, Australia. Association for Computational Linguistics.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *9th International Conference on Learning Representations (ICLR 2021)*. OpenReview.net.

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.

Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. 2023. Grammar-constrained decoding for structured NLP tasks without finetuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10932–10952, Singapore. Association for Computational Linguistics.

Osamu Gotoh. 1982. An improved algorithm for matching biological sequences. *Journal of Molecular Biology*, 162(3):705–708.

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.

Ralf Krestel, Sabine Bergler, and René Witte. 2008. Minding the source: Automatic tagging of reported speech in newspaper articles. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

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.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations (ICLR 2019)*, New Orleans, LA, USA. OpenReview.net.

Chris Newell, Tim Cowlishaw, and David Man. 2018. Quote extraction and analysis for news. In *Proceedings of the Workshop on Data Science, Journalism and Media, KDD*, pages 1–6, London, UK.

Timothy O'Keefe, Silvia Pareti, James R. Curran, Irena Koprinska, and Matthew Honnibal. 2012. A sequence labelling approach to quote attribution. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 790–799, Jeju Island, Korea. 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. In *9th International Conference on Learning Representations (ICLR 2021)*. OpenReview.net.

Sean Papay and Sebastian Padó. 2019. Quotation detection and classification with a corpus-agnostic model. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 888–894, Varna, Bulgaria. INCOMA Ltd.

Silvia Pareti, Tim O'Keefe, Ioannis Konstas, James R. Curran, and Irena Koprinska. 2013. Automatically detecting and attributing indirect quotations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 989–999, Seattle, Washington, USA. Association for Computational Linguistics.

Fynn Petersen-Frey and Chris Biemann. 2024. Dataset of quotation attribution in German news articles. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4412–4422, Torino, Italia. European Language Resources Association (ELRA) and International Committee on Computational Linguistics (ICCL).

Bruno Pouliquen, Ralf Steinberger, and Clive Best. 2007. Automatic detection of quotations in multilingual news. In *The International Conference on Recent Advances in Natural Language Processing*, RANLP 2007, pages 487–492.

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. *J. Mach. Learn. Res.*, 21(1).

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. ZeRO: memory optimizations toward training trillion parameter models. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '20, pages 1–16. Institute of Electrical and Electronics Engineers (IEEE).

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. DeepSpeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

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.

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.

Yuanchi Zhang and Yang Liu. 2022. DirectQuote: A dataset for direct quotation extraction and attribution in news articles. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6959–6966, Marseille, France. European Language Resources Association.

## Appendix

| | | | quotation | | | roles | | | joint | | | type | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| model | sz | lin. | prec. | rec. | F1 | prec. | rec. | F1 | prec. | rec. | F1 | prec. | rec. | F1 |
| RBS* | | | 75.1 | 36.1 | 48.8 | 55.0 | 25.5 | 34.9 | 60.7 | 28.7 | 38.9 | 57.8 | 29.6 | 39.1 |
| Citron* | | | **91.5** | 27.6 | 42.4 | 79.3 | 31.5 | 45.1 | 82.4 | 30.3 | 44.3 | 87.0 | 26.6 | 40.8 |
| t5 | b | full | 82.4 | 68.3 | 74.7 | 76.2 | 61.9 | 68.3 | 78.0 | 63.8 | 70.2 | 81.6 | 68.0 | 74.2 |
| t5 | b | part | 83.0 | 74.5 | 78.5 | 79.0 | 68.6 | 73.4 | 80.2 | 70.3 | 74.9 | 83.2 | 74.3 | 78.5 |
| t5 | l | full | 86.7 | 70.0 | 77.5 | 80.1 | 63.2 | 70.7 | 82.1 | 65.2 | 72.7 | 87.6 | 70.4 | 78.1 |
| t5 | l | part | 86.2 | 69.3 | 76.8 | 80.7 | 63.8 | 71.3 | 82.3 | 65.4 | 72.9 | 86.6 | 69.4 | 77.0 |
| t5-v1-1 | b | full | 80.4 | 50.8 | 62.3 | 62.3 | 44.5 | 51.9 | 67.3 | 46.3 | 54.9 | 79.8 | 50.8 | 62.1 |
| t5-v1-1 | b | part | 81.6 | 65.9 | 72.9 | 77.8 | 60.4 | 68.0 | 79.0 | 62.0 | 69.5 | 83.5 | 66.5 | 74.0 |
| t5-v1-1 | l | full | 71.8 | 63.7 | 67.5 | 62.6 | 52.0 | 56.8 | 65.4 | 55.5 | 60.0 | 69.6 | 62.5 | 65.9 |
| t5-v1-1 | l | part | 81.8 | 76.5 | 79.1 | 80.3 | 71.1 | 75.4 | 80.8 | 72.7 | 76.5 | 82.2 | 77.0 | 79.5 |
| t5-v1-1 | xl | full | 85.9 | 74.0 | 79.5 | 80.6 | 71.3 | 75.6 | 82.1 | 72.1 | 76.8 | 86.5 | 75.3 | 80.5 |
| t5-v1-1 | xl | part | 86.9 | 76.5 | 81.4 | 82.2 | 69.6 | 75.4 | 83.6 | 71.6 | 77.2 | 86.7 | 76.0 | 81.0 |
| flan-t5 | b | full | 81.8 | 73.4 | 77.4 | 76.1 | 67.2 | 71.3 | 77.8 | 69.0 | 73.1 | 82.7 | 74.1 | 78.2 |
| flan-t5 | b | part | 81.7 | 72.7 | 77.0 | 76.9 | 66.4 | 71.3 | 78.4 | 68.2 | 73.0 | 82.2 | 73.7 | 77.7 |
| flan-t5 | l | full | 86.1 | 73.5 | 79.3 | 80.5 | 70.3 | 75.0 | 82.1 | 71.2 | 76.3 | 86.2 | 74.0 | 79.6 |
| flan-t5 | l | part | 87.3 | 74.2 | 80.2 | 82.8 | 69.9 | 75.8 | 84.1 | 71.2 | 77.1 | 88.1 | 74.0 | 80.4 |
| flan-t5 | xl | full | 89.0 | 77.3 | 82.7 | **83.9** | 73.3 | 78.2 | **85.4** | 74.5 | 79.6 | **89.6** | 77.6 | 83.2 |
| flan-t5 | xl | part | 84.9 | 73.0 | 78.5 | 80.5 | 69.0 | 74.3 | 81.9 | 70.1 | 75.5 | 86.1 | 74.1 | 79.7 |
| mt5 | b | full | 89.9 | 72.6 | 80.3 | 81.3 | 70.9 | 75.7 | 83.7 | 71.4 | 77.1 | 89.2 | 73.5 | 80.6 |
| mt5 | b | part | 85.5 | 75.2 | 80.0 | 83.1 | 71.4 | 76.8 | 83.8 | 72.5 | 77.8 | 84.9 | 74.3 | 79.2 |
| mt5 | l | full | 88.4 | 75.5 | 81.4 | 83.4 | 73.2 | 78.0 | 84.9 | 73.9 | 79.0 | 89.1 | 76.3 | 82.2 |
| mt5 | l | part | 89.7 | 78.4 | **83.7** | 83.2 | 74.5 | 78.6 | 85.1 | 75.6 | **80.1** | 89.2 | 78.6 | **83.6** |
| mt5 | xl | full | 84.8 | 78.8 | 81.7 | 82.2 | **76.5** | **79.3** | 83.0 | **77.2** | 80.0 | 84.5 | **80.2** | 82.3 |
| mt5 | xl | part | 86.4 | **79.5** | 82.8 | 82.5 | 75.8 | 79.0 | 83.7 | 76.9 | **80.1** | 85.3 | 79.0 | 82.0 |

*Baseline results taken from Petersen-Frey and Biemann (2024)

Table 5: Complete evaluation results on the development set. Highest score per metric marked in bold.

| | | | quotation | | | roles | | | joint | | | type | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| model | sz | lin. | prec. | rec. | F1 | prec. | rec. | F1 | prec. | rec. | F1 | prec. | rec. | F1 |
| RBS* | | | 70.8 | 36.2 | 47.9 | 55.6 | 26.1 | 35.5 | 59.9 | 29.0 | 39.1 | 63.5 | 33.6 | 43.9 |
| Citron* | | | 88.2 | 30.1 | 44.9 | 77.9 | 34.2 | 47.6 | 80.5 | 33.0 | 46.8 | 86.5 | 29.6 | 44.1 |
| t5 | b | full | 81.3 | 68.1 | 74.2 | 75.2 | 62.3 | 68.2 | 77.0 | 64.0 | 69.9 | 82.6 | 69.0 | 75.2 |
| t5 | b | part | 84.7 | 72.9 | 78.4 | 79.6 | 66.4 | 72.4 | 81.1 | 68.2 | 74.1 | 84.5 | 72.7 | 78.2 |
| t5 | l | full | 89.4 | 71.1 | 79.2 | 82.5 | 65.8 | 73.2 | 84.5 | 67.3 | 74.9 | 90.1 | 72.4 | 80.3 |
| t5 | l | part | 86.2 | 72.2 | 78.6 | 80.5 | 65.6 | 72.3 | 82.2 | 67.5 | 74.1 | 86.3 | 72.4 | 78.7 |
| t5-v1-1 | b | full | 77.8 | 53.7 | 63.5 | 61.6 | 46.3 | 52.9 | 66.0 | 48.4 | 55.9 | 80.7 | 55.1 | 65.5 |
| t5-v1-1 | b | part | 83.8 | 67.2 | 74.6 | 78.4 | 61.9 | 69.2 | 80.0 | 63.4 | 70.7 | 85.2 | 66.9 | 74.9 |
| t5-v1-1 | l | full | 71.4 | 62.8 | 66.8 | 63.3 | 54.3 | 58.5 | 65.7 | 56.8 | 60.9 | 71.2 | 63.7 | 67.2 |
| t5-v1-1 | l | part | 83.5 | 77.1 | 80.2 | 78.0 | 70.0 | 73.8 | 79.6 | 72.1 | 75.7 | 84.3 | 77.3 | 80.6 |
| t5-v1-1 | xl | full | 88.1 | 74.6 | 80.8 | 80.6 | 69.8 | 74.8 | 82.8 | 71.2 | 76.5 | 88.4 | 75.2 | 81.3 |
| t5-v1-1 | xl | part | 89.3 | 76.8 | 82.6 | 83.2 | 70.5 | 76.3 | 85.0 | 72.3 | 78.1 | 89.6 | 77.0 | 82.8 |
| flan-t5 | b | full | 83.0 | 73.2 | 77.8 | 73.3 | 66.0 | 69.4 | 76.1 | 68.1 | 71.8 | 83.9 | 74.2 | 78.8 |
| flan-t5 | b | part | 81.0 | 75.5 | 78.2 | 74.9 | 67.2 | 70.8 | 76.7 | 69.6 | 73.0 | 82.4 | 76.2 | 79.2 |
| flan-t5 | l | full | 86.6 | 76.8 | 81.4 | 78.9 | 73.2 | 75.9 | 81.1 | 74.3 | 77.5 | 86.8 | 77.5 | 81.8 |
| flan-t5 | l | part | 89.4 | 77.9 | 83.3 | 82.5 | 74.0 | 78.1 | 84.5 | 75.1 | 79.6 | 90.1 | 78.2 | 83.7 |
| flan-t5 | xl | full | 89.5 | 78.2 | 83.5 | 83.5 | 74.0 | 78.5 | 85.3 | 75.2 | 79.9 | 89.2 | 78.2 | 83.3 |
| flan-t5 | xl | part | 89.9 | 76.3 | 82.5 | **84.2** | 70.2 | 76.6 | **85.8** | 71.9 | 78.3 | **90.7** | 76.7 | 83.1 |
| mt5 | b | full | **91.2** | 75.4 | 82.6 | 82.6 | 71.4 | 76.6 | 85.0 | 72.5 | 78.3 | 90.0 | 74.5 | 81.5 |
| mt5 | b | part | 85.9 | 77.2 | 81.3 | 81.8 | 73.2 | 77.3 | 83.0 | 74.4 | 78.4 | 87.5 | 78.1 | 82.5 |
| mt5 | l | full | 89.5 | 78.3 | 83.5 | 84.1 | 74.4 | 78.9 | 85.6 | 75.5 | 80.3 | 90.3 | 78.7 | 84.1 |
| mt5 | l | part | 88.8 | **81.1** | **84.8** | 83.0 | 75.2 | 78.9 | 84.7 | 76.9 | 80.6 | 89.5 | **81.0** | **85.0** |
| mt5 | xl | full | 85.2 | 78.6 | 81.8 | 80.3 | 74.8 | 77.4 | 81.7 | 75.9 | 78.7 | 86.1 | 79.6 | 82.7 |
| mt5 | xl | part | 88.2 | 80.0 | 83.9 | 83.2 | **76.2** | **79.5** | 84.6 | **77.3** | **80.8** | 88.5 | 80.2 | 84.2 |

*Baseline results taken from Petersen-Frey and Biemann (2024)

Table 6: Complete evaluation results on the test set. Highest score per metric marked in bold.