Daniel Gärber, BSc

Causal Relationship Extraction from Historical Texts using BERT

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Computer Science

submitted to

Graz University of Technology

Supervisor

Ass.Prof. Dipl.-Ing. Dr.techn. Roman Kern

Institute for Interactive Systems and Data Science
Head: Univ.-Prof. Dipl.-Inf. Dr. Stefanie Lindstaedt

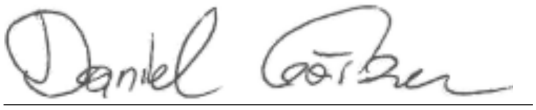
Graz, August 2022

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

30.08.2022

Date



Signature

Acknowledgements

I would like to express my deepest gratitude to my advisor Ass.Prof. Dipl.-Ing. Dr.techn. Roman Kern for his continuous support and guidance.

I also want to thank Mag. Dr.phil. Wolfgang Göderle and Bernhard Ortbauer for their help concerning historical questions and their immense efforts in finding, preparing and annotating the documents.

Additional thank goes to my family and friends for their steady encouragement and moral support.

Abstract

Causality in historic documents is an important source of information for historians. Manually finding relevant causal relations from the immense number of documents is a time-intensive process. To support historians in their work, we created a novel approach for causal relationship extraction and introduced a dataset of historical documents annotated for causal relations in German.

Our proposed model for causality extraction was based on BERT. We extended traditional sequence labeling approaches to allow the model to detect multiple overlapping relations. The model created distinct context embeddings per causal relation, from which associated causal arguments, such as cause and effect, were detected. Additionally, we assigned a causal type and degree to each relation. Our model outperformed a pattern-based approach in all tasks.

We evaluated various BERT models, pre-processing steps, and transfer learning approaches. German BERT models generally performed better than multilingual models, and pre-training on contemporary texts performed similarly well to pre-training on historical texts. Transfer learning on related tasks could overall improve the model. Pre-processing the text to correct historic spelling variations or including additional information about coreferences did not increase the performance. We also found evidence that BERT learns about causal relationships during self-supervised pre-training, indicating that causality is integral for encoding information in natural text.

The promising results of our model demonstrate the potential to support historians in their work by recommending relevant passages containing causal relations or by creating knowledge bases from cause and effect relationships.

Contents

List of Figures

List of Tables

1. Introduction

Historical text documents are of great importance for the work of historians. Various forms of information are embedded in the texts, a prominent form is causal relationships. Causality defines the relationship between two events, where one event (cause) results in another event (effect). The vast amount of available historical documents makes manual approaches for extracting causal relationships time and labor-intensive. The automatic detection of causal relationships has the potential to revolutionize how historians gather information from historic sources. It could help to detect previously unknown causal chains and to establish a more fine-grained understanding of interrelated historical events. This would lead to more holistic knowledge bases and improved information retrieval methods. In this thesis, we use natural language processing (NLP) and deep learning (DL) to extract information in the form of causal relationships from historical texts.

Causality is fundamental for making rational decisions. It can answer the question of why something happened, and helps to predict future outcomes. We frequently use causality to make conclusions or infer information, and require little effort to correctly interpret the following sentences in a causal context.

- *Stress or dehydration cause headaches.*
- *New technologies lead to better medical treatments, which increases the life expectancy of people.*
- *The road leads to the city.*

This seemingly easy task is challenging for machines. Causal relations can occur in various forms, and a strong semantic understanding is required for a correct interpretation. For example, to understand that the sentence *The road leads to the city* is not causal, the machine would need to understand what a road and a city are, and how they interact with each other. Fortunately, recent advances in machine learning and NLP have improved the performance of tasks concerning complex semantics.

(1) **Stress or dehydration**$_{\text{Cause}}$ **cause**$_{\text{Trigger}}$ **headaches**$_{\text{Effect}}$.
Type: *Consequence*, Degree: *Facilitate*

There is no universally agreed definition of causality in natural text, and many different approaches exist to annotate causal relations. In our annotation scheme, a causal relation consists of causal arguments, type and degree, as presented in Example (1). The type and degree are helpful to discern between different classes of causality.

Our proposed model to detect causal relations is based on pre-trained Bidirectional Encoder Representations from Transformers (BERT) [2]. We use a sequence labeling approach, with the extension that we also consider multiple causal relations occurring within one sentence. To ensure that the model focuses on arguments related to a certain causal relation, we generate a context embedding for each relation. This context is an aggregated form of the causal triggers of the relation, for example *because* or *leads to*. To evaluate our approach, we annotate two historic German documents from the late 19th century, both on the topic of managing forests in the Austro-Hungarian Empire. We investigate how additional pre-training on related, contemporary datasets about causal relationships can affect the performance, and experiment with pre-processing by correcting for historical spelling variations and annotating coreferences. As a baseline, we create a rule-based model that learns simple patterns and heuristics found in the annotated corpus.

In addition, we explore BERT's capabilities to understand causality on a fundamental level. BERT has been shown to learn linguistic and semantic relationships from self-supervised pre-training on vast amounts of natural text. However, it is unclear if this includes causal relationships. To answer this question, we investigate BERT's self-attention heads. These heads are trained to connect related tokens and enhance their representations, with heads emerging that correspond to linguistic or semantic relationships. We hypothesize that heads exist that learn to focus on causality.

1.1. Contribution to Research

This thesis contributes to research in several ways. Firstly, we present a general approach for detecting causal relationships in natural text. Secondly, we introduce a

novel corpus of historic documents annotated for causal relationships, and provide insights into how to adjust the historic text for further processing. Lastly, we provide evidence that suggests that BERT already learns causal relationships to some degree during pre-training.

Specifically, we want to answer the following research questions:

1. Can a BERT-based model outperform a pattern-based approach in the extraction and classification of causal structures in historic texts?
2. How does pre-training BERT on contemporary, historic or multilingual texts influence the performance?
3. Does transfer learning, correcting for historical spelling variations or including coreference information improve the performance?
4. Does BERT already learn causal relationships during self-supervised pre-training?

1.2. Thesis Outline

Chapter 2 includes background information about causality in natural text and provides an overview of state-of-the-art solutions. We introduce common patterns of causal relationships and different approaches to group them into semantic classes. Furthermore, we provide information about the algorithms used in this thesis, with a detailed analysis of BERT. Additional necessary technologies are presented, like OCR and a tool for text annotations. Finally, we give an overview of state-of-the-art solutions and datasets concerning causal relations.

In Chapter 3, we describe the methodology of this thesis. First, we introduce the two annotated historic documents that we use for evaluation. Next, we describe in detail the motivation and architecture of the used model, including the pre-processing of the data and the training procedure. Furthermore, we explain the measures we use for evaluation. Lastly, the performed experiments are presented.

In Chapter 4, the results for the experiments concerning the model and the causal attention heads are presented and discussed. Additionally, we present and analyze exemplary predictions of the model.

Chapter 5 summarized our results and proposes possible future steps.

2. Related Work

This chapter gives the necessary prerequisites for the topics discussed in this thesis. Section 2.1 provides an overview of causality in text and the algorithms used later on. Section 2.2 presents relevant research and existing approaches.

2.1. Background

This section is divided into several parts. First, different forms of causality in text are discussed, to get a broader understanding of causal language. Then the concept of word embeddings and their attributes is introduced. Next, we present the highly influential Transformer model and the BERT model, which will form the basis of our proposed method. Afterward, we discuss the architecture of OCR systems and how to correct for errors in the process. Lastly, a short introduction to the text annotation process is given.

2.1.1. Causality in Text

Causality at its core describes a relationship of two events where one event (cause) influences the other event (effect). Causality is a broad subject, connecting various fields such as psychology, philosophy and linguistics. Language plays a central role in constructing causal relationships, and according to Mackie [3] causal language acts as a guideline to what we consider causally related. Causal language can be expressed in various forms, which are not easily separated.

In this section we introduce the concepts of implicit and explicit causality, and present different classification approaches of causal language based on linguistic and semantic attributes.

Implicit and Explicit Causality

A general classification is between implicit and explicit causality [4].

Implicit causalities occur without a linguistic indication of causality. The causal meaning is inferred using knowledge about the word. This is the case in Example (2), where our causal understanding comes from the temporal relationship in combination with general knowledge.

(**2**) Mary went to the concert last night. She is tired today.

In contrast, for explicit causality the causal relationship is explicitly indicated using a causal trigger. Example (2) could be rewritten to form an explicit causality, as can be seen in Example (3). The word *because* acts as a causal trigger.

(**3**) Mary is tired **because**$_{\text{Causal Trigger}}$ she went to the bar last night.

Causal trigger words can however also appear in a non-causal meaning, which makes the trigger word ambiguous [5]. For example, the word *since* can have a causal or temporal meaning, as seen in Examples (4) and (5). An unambiguous trigger is always causal, such as the word *because*.

(**4**) **Since**$_{\text{Causal}}$ it was raining, she took an umbrella.
(**5**) He has lived in this city **since**$_{\text{Non-causal}}$ last year.

Linguistic Classification

From a linguistic standpoint, Khoo et al. [6] distinguished five types of explicit causality: causal links, causative verbs, resultative constructions, conditionals, and causative adverbs and adjectives.

- **Causal Links** Causal links are words or phrases that specify a causal relationship between two events. Examples for this are *so*, *because*, *since* and *the result was*. Example (3) shows a causal link.

- **Causative Verbs** These are verbs that additionally carry a causal meaning. Khoo et al. [6] introduced an extensive classification for causative verbs by the type of their result, with semantic and syntactic sub-categories, which is here shortly summarized with example verbs.

  - Verbs that mean to cause something
    cause, lead to, kill, deactivate, convince, improve, harm
  - Verbs that mean to be caused by something
    proceed from, result from, stem from
  - Verbs that mean to prevent something from happening
    prevent, silence, persuade
  - Verbs that mean to affect something without specifying in what way
    affect, impact

  A special case are verbs where the effect is encoded within the verb, e.g. in Example (6) the effect that *John* is dead is missing, but can be inferred from the verb *killed*. Examples of such verbs are *break* or *melt*.

  (6) He **killed**$_{\text{Causative Verb}}$ John.

- **Resultative Constructions** Here the verb expresses some act that leads to a change in an object, which is specified in an additional phrase. In Example (7), the tearing leads to *the paper* becoming *shreds*.

  (7) She tore the paper to shreds.

- **Conditionals** Conditionals are if-then relations where the if-segment is the cause and the then-segment the effect. An example can be seen in Example (8).

  (8) **If** you eat too much **then** you will become fat.

- **Causative Adverbs and Adjectives** This denotes adverbs and adjectives that incorporate a causal meaning. In Example (9), the adverb *mortally* carries the effect that the man died.

  (9) The man was **mortally**$_{\text{Causative Adverb}}$ wounded.

Semantic Classification

Causal structures can also be classified by their causal meaning. Dunietz et al. [7] identified four types of causality: Consequence, Motivation, Purpose and Inference. The types are separated by fine semantic differences and will be explained in more detail. The example sentences are taken from [7] and [8].

- **Consequence** Consequence is the most common causal type. The cause triggers an effect in a natural manner, without any thinking or acting agents. An example can be seen in Example (10).

  (**10**) The new regulations will prevent future crises.

- **Motivation** This type is similar to Consequence, however, it includes a feeling or thinking agent who anticipates the effect of a cause and acts accordingly, as seen in Example (11).

  (**11**) We don't have much time, so let's move quickly.

- **Purpose** Purpose is for instances where an action is performed to accomplish a certain goal, often including phrases like *in order to*. As the action occurs because of the goal, the goal is the cause and the action the effect. In Example (12), the cause is *strengthen our company*, as this leads to the effect *set clearer policies*. The reverse interpretation is also a valid causality, as *clearer policies* can cause *strengthen our company*. Dunietz et al. [7] however choose to solve this case by always using the non-reversed interpretation.

  (**12**) In order to strengthen our company, we must set clearer policies.

- **Inference** Inference describes instances where a cause directly infers some effect. This type is similar to Consequence, but the cause acts as evidence for the effect. An example is given in Example (13). This type is also called epistemic causation, and the authors argue that this type is not truly causal, however, it corresponds to the use of causal language.

  (**13**) This car was driven recently, because the hood is still hot.

Another classification is given by Wolff et al. [9], who define causal language in terms of three underlying semantic categories of causal meaning: Cause, Enable and Prevent. The classification is based on the relation between two entities, the affector and the patient, with the affector exerting force on the patient. Additionally, the type depends on whether the affector and patient are in agreement and if a result happens. We will now examine the three types in more detail with example sentences from [9].

- **Cause** This type occurs in relations where the patient is against the result and the affector for the result. Against the desire of the patient, the affector

causes the result to occur. In Example (14) the affector *strong winds* results in the patient *the bridge* collapsing. The tendency of *the bridge* is to stay intact, so the affector and the patient are not in agreement.

(**14**) **Strong winds**$_{\text{Affector}}$ caused **the bridge**$_{\text{Patient}}$ to collapse.
Type: Cause

- **Enable** Enable describes relations where the patient and the affector both are in the direction of the result occurring. In Example (15) the affector *Vitamin B* supports the patient *the body* in achieving the result *to digest food.*

  (**15**) **Vitamin B**$_{\text{Affector}}$ enables **the body**$_{\text{Patient}}$ to digest food.
  Type: Enable

- **Prevent** This is the opposite of Cause, as now the patient wants the result to happen, with the affector preventing it. In Example (16) the patient *butter* has a tendency to *burning*, however, the affector *corn oil* hinders the patient, with no result happening.

  (**16**) **Corn oil**$_{\text{Affector}}$ prevents **butter**$_{\text{Patient}}$ from burning.
  Type: Prevent

In their work, Wolff et al. [9] also investigated causal language of other languages than English, including German, and found evidence that their semantic system is present across languages.

2.1.2. Word Embeddings

Word embeddings are a way of representing words in vector space. The goal of word embeddings is to encode the meaning of a word as a vector. Traditional approaches, such as one-hot encoding, result in high-dimensional and sparse vector representations. In contrast, word embeddings are low-dimensional and dense, making them suitable to work with neural networks. They play a central role in modern NLP and are used in many state-of-the-art architectures. There exist two main classes of word embeddings: Non-contextual and contextual word embeddings.

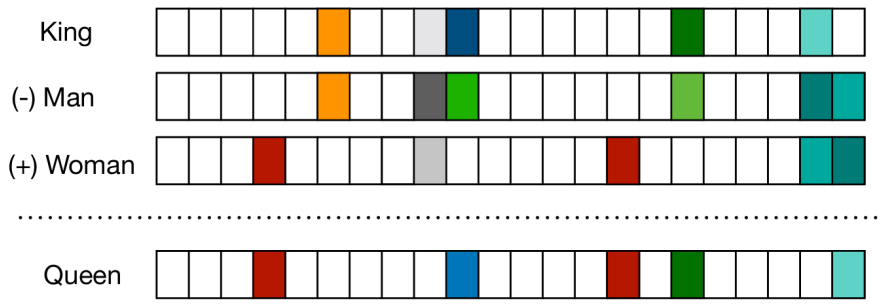


Figure 2.1.: Conceptual example of word embedding vectors. Word Embeddings can be added and subtracted, which adds or removes certain concepts from the vector representation. Image is taken from [10].

Non-Contextual Word Embeddings

Non-contextual word embeddings aim to represent the abstract meaning of a word in vector space, where the embeddings of words with similar meanings tend to be close, while the embeddings of words with dissimilar meanings are far apart. An example of word embeddings is given in Figure 2.1. In the example the colored values of the word embeddings describe certain abstract concepts about the word's meaning. The vectors *King* and *Man* share many similarities, but *King* and *Woman* share few similarities. By subtracting the vector for *Man* and adding the vector for *Woman*, the parts of the vector that correspond to gender are changing. The resulting vector still encodes an abstract concept of royalty, but now in female form, which is closest to the vector for *Queen*.

Non-contextual word embeddings of a word can be learned using its context. The context consists of the words surrounding a word. For each word, a single word embedding is generated. This simplification has the downside that a word with more than one meaning only has a single corresponding word embedding vector, which is approximately the weighted mean of all senses of the word [12]. For example, the meaning of the word *duck* in Examples (17) and (18) can refer to the animal or the verb, but it has the same non-contextual word embedding. Additionally, this has the effect that embeddings of semantically different topics are closer in vector space than they should be [13]. In another example, the word *bat* is both related to the words *ball* and *animal*, which leads to both word embeddings being pulled together due to their common neighbor.

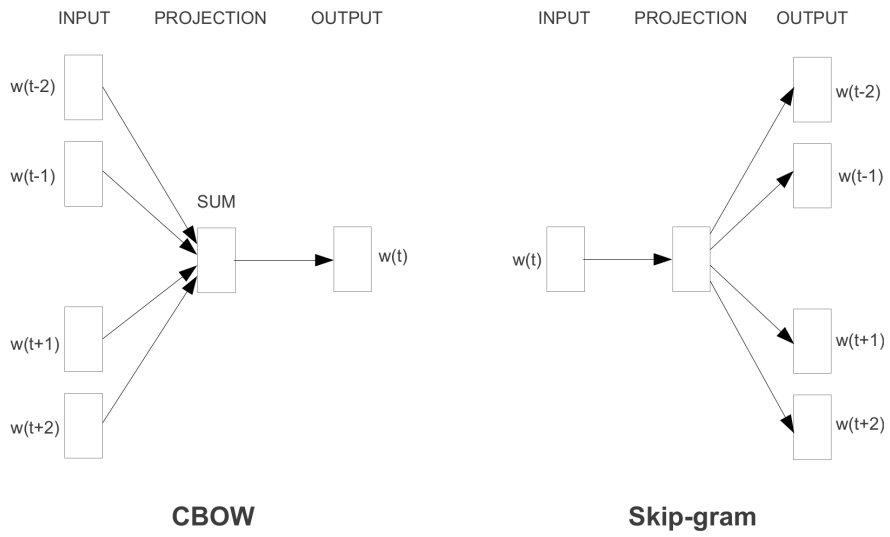


Figure 2.2.: Model architecture for the CBOW and Skip-gram model. In CBOW the input is the context and the output is the target word, and for the Skip-gram model it is the opposite. Image is taken from [11].

(**17**) The **duck**$_{\text{Animal}}$ swam on the lake.
(**18**) I **duck**$_{\text{Verb}}$ my head.

Mikolov et al. [11] have developed two neural network models for learning non-contextual word embeddings. The architecture for both models can be seen in Figure 2.2. The continuous bag-of-words (CBOW) model has the objective to predict a target word from its context. In contrast, the skip-gram model has the objective to predict the context from the word.

The training is done in a self-supervised fashion. This means that the supervisory signals the models are trained on are created directly from the text, which are the target words for the CBOW model and the contexts for the skip-gram model. Self-supervised learning allows models to train on vast amounts of text with no annotation and minimal preprocessing required. As a training corpus, Mikolov et al. [11] used a large Google News corpus, with a vocabulary of the 1 million most frequent words.

Contextual Word Embeddings

Contextual word embeddings solve the lack of adaption to context that non-contextual word embeddings suffer from. The contextual embedding is dependent on the context of the word, which also means that the word embedding is not unique for a word anymore and changes depending on the context. The contextual word embedding of the word *duck* will be different in Examples (17) and (18).

To generate contextual word embeddings, models have to process not only the target word but the whole sequence. A typical approach is using recurrent neural networks (RNN), which are well suited for sequential data. One such model is Embedding from Language Model (ELMo) [14]. ELMo uses a bi-directional RNN to process a sequence of tokens two times, once forward and once backward, and then combines both outputs for a token into the final word embedding. This approach has the advantage that the words are processed in combination with a representation of the context. There are however downsides as well. First, as ELMo looks at the left and right context separately, it is unable to use the information from the whole text at once [2]. Secondly, the RNN architecture has problems with capturing dependencies between words that are far apart. A more sophisticated approach is using the attention mechanism, as it is used in the Transformer.

2.1.3. Transformers

The Transformer is a deep learning model introduced by Vaswani et al. [15] in 2017 and has since been a highly influential building block for NLP and image processing models. It has achieved several state-of-the-art results in machine translation tasks while also needing significantly less training time than models based on recurrent or convolutional neural networks.

Self-Attention

First, we need to understand the concept of self-attention, which is used extensively in the Transformer. At a high level, self-attention is a process to identify and incorporate valuable information from other parts of the sequence into the position of interest. In Example (19) it is clear to see that the pronoun *it* is a reference to the

noun *dog*, and not the noun *street*. Self-attention first determines that the word *dog* is important for the meaning of the word *it*, and then uses the representation of the word *dog* to improve the representation of the word *it*. For algorithms based on RNNs it is difficult to utilize this information efficiently, especially if the relationship in the text is far apart.

(**19**) The **dog** crossed the street because **it**$_{\text{Coreference}}$ saw a bird.

Implementation of Self-Attention

We will now look more closely at how self-attention is implemented. Self-attention works on the word embeddings of the words. Given a sequence $\mathbf{x} = (x_1, x_2, ..., x_n)$ of word embeddings, for each embedding x_i three vectors are generated: the query vector q_i, the key vector k_i and the value vector v_i. These vectors are computed by projecting the word embedding with three learned matrices $\mathbf{W}_Q$, $\mathbf{W}_K$ and $\mathbf{W}_V$ as

$$q_i = x_i\mathbf{W}_Q \qquad k_i = x_i\mathbf{W}_K \qquad v_i = x_i\mathbf{W}_V. \tag{2.1}$$

To calculate the self-attention for a target embedding x_t, first the scores $\mathbf{s_t} = (s_{t,1}, s_{t,2}..., s_{t,n})$ for all embeddings $x_i \in \mathbf{x}$ are calculated using Equation (2.2), with d_k being the dimension of the key vector.

$$s_{t,i} = \frac{q_t k_i^T}{\sqrt{d_k}} \tag{2.2}$$

A softmax operation on the scores $\mathbf{s_t}$ is applied to get the weighting terms $\mathbf{w_t} = (w_{t,1}, w_{t,2}..., w_{t,n})$. Finally the new word embedding x'_t is calculated in Equation (2.3) as a weighted average of the value vectors. Optimally the weights from the softmax operation are high for relevant words, while irrelevant words have a weight close to 0.

$$x'_t = \sum_i v_i w_{t,i} \tag{2.3}$$

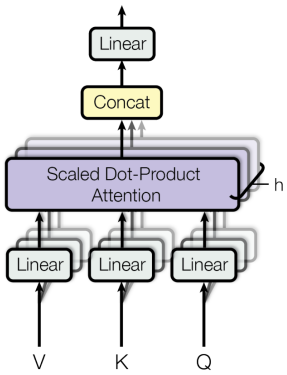


Figure 2.3.: Multi-Head Attention. Each of the h heads calculates self-attention individually, using separate weight matrices $\mathbf{W}_Q$, $\mathbf{W}_K$ and $\mathbf{W}_V$. The individual results are then concatenated and sent into a linear layer. In the original architecture the number of parallel computations $h = 8$. Image is taken from [15].

The above steps can be performed conveniently on the whole sequence in matrix format by stacking the key, query and value vectors into the matrices $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$. The embeddings $\mathbf{x}'$ after self-attention are then computed as

$$\mathbf{x}' = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}. \tag{2.4}$$

Multi-Head Attention

The computation in Equation (2.4) is extended by computing self-attention functions multiple times in separate heads, which is called multi-head attention and can be seen in Figure 2.3. Each head has a separate set of matrices $\mathbf{W}_Q$, $\mathbf{W}_K$ and $\mathbf{W}_V$, aiming for different heads to attend to different aspects of the sequence. The outputs of the heads are then concatenated and processed by a linear layer to compute the final output embeddings. Vaswani et al. [15] proposed to use 8 parallel attention heads.

In Figure 2.4 an example of multi-head attention for the word *it* from Example (19) is given. Each of the 8 heads is associated with a specific color, and the brightness of the color next to a token indicates the attention weight assigned by a head. Some relationships align with linguistic reasoning, for example, the gray head focuses on the referenced word *dog*, or the pink head focuses on the verb *saw*.

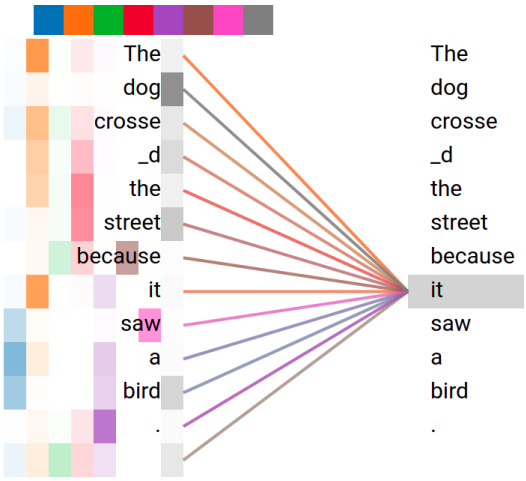


Figure 2.4.: Multi-head attention visualization for the word *it* in Example (19) with the Tensor2Tensor library [16]. Each attention head is associated with a different color. For each token the brightness indicates the assigned attention weight of a head.

Other heads like the orange or green head are not so easily explained by linguistic phenomena. Interestingly, experiments have shown that multiple attention-heads encode similar information and could be pruned without a significant effect on the performance [17]. Additionally, heads with a large effect on the performance can often be associated with specific interpretable roles [18]. There is however debate whether attention weights are suitable to explain predictions of a model [19].

Encoder-Decoder Architecture

The architecture of the Transformer is given in Figure 2.5. It follows a basic encoder-decoder structure. The input to the encoder is a sequence that is mapped to an intermediate representation. The decoder then maps the intermediate representation to the output sequence.

First, the input sequence gets converted into learned embeddings. Then a positional encoding is added to the input for the model to utilize its order. The input with the positional encoding is now processed by the encoder. In the original architecture, the encoder contains 6 identical layers, which can be seen in Figure 2.5 on the left side. For each layer there exist two sub-layers. The first sub-layer is the multi-head

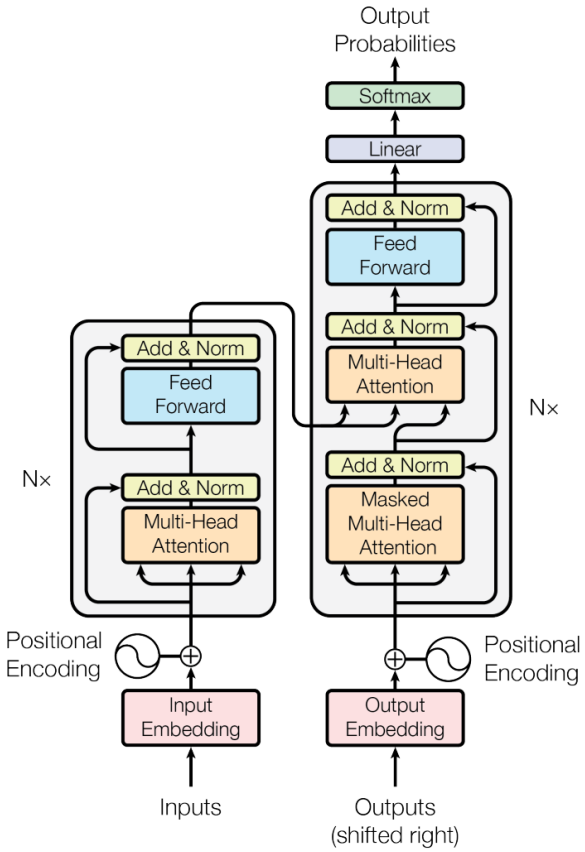


Figure 2.5.: The Transformer architecture. The left side is the encoder, the right is the decoder. Image is taken from [15].

attention and the second sub-layer is a neural network. The unmodified input to each sub-layer is added to the output embedding using residual connections, followed by a normalization step.

The decoder is similar in structure to the encoder. It also contains 6 identical layers, and additionally to the two sub-layers of the encoder, the decoder layer has a third sub-layer where multi-head attention using the encoder's output is performed. This means that in this sub-layer the $\mathbf{K}$ and $\mathbf{V}$ matrices are computed from encoder embeddings, and the $\mathbf{Q}$ matrix is computed from the decoder embedding of the previous self-attention layer.

The decoder re-uses its previous outputs as input. The first output of the decoder only depends on the encoder embeddings, while further outputs also depend on previous outputs. Additionally, the self-attention sub-layer is modified so that the attention can only be applied to previous outputs, which is called masked multi-head attention.

On top of the decoder output is a linear layer and a softmax operation. This yields probability values for all tokens and the token with the highest probability is the final output.

2.1.4. BERT

The Bidirectional Encoder Representations from Transformers (BERT) model is a deep language representation model based on the Transformer architecture. BERT was introduced by Devlin et al. [2] in 2019 and has achieved state-of-the-art results in several NLP tasks, including Question Answering (QA) and Named Entity Recognition (NER). Since then several variations to BERT have been developed.

Since BERT forms the main building block for our later model, we now give a detailed description.

Architecture

The BERT architecture consists only of the encoder stack of the Transformer, the decoder stack is not used. Devlin et al. [2] evaluated two different model sizes: The $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ models. A comparison of the model parameters between

	Transformer	$\text{BERT}_{\text{BASE}}$	$\text{BERT}_{\text{LARGE}}$
Number of Layers	12	12	24
Hidden Size	512	768	1024
Self-attention heads	8	12	16
Total parameters	65M	110M	340M

Table 2.1.: Model comparison between Transformer, $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$. Note that for the Transformer both the encoder and decoder stack are considered. The number of layers for both the encoder and decoder is 6.

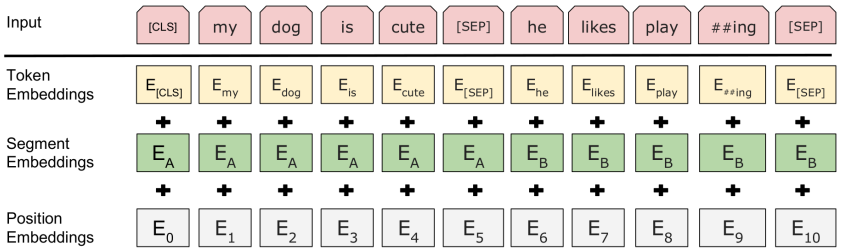


Figure 2.6.: Components of the BERT inputs. The sum of the three embeddings results in the input embedding for the first encoder layer. Image is taken from [2].

the original Transformer and the BERT models are given in Table 2.1. The $\text{BERT}_{\text{BASE}}$ model has a larger hidden size and more self-attention heads than the Transformer, resulting in nearly 70% more parameters. Further, increasing the model size from $\text{BERT}_{\text{BASE}}$ to $\text{BERT}_{\text{LARGE}}$ consistently improved the performance in all evaluated tasks.

Input

Similar to the Transformer input BERT uses pre-trained word embeddings for the individual tokens in a text sequence. In the original architecture, the word embeddings are WordPiece embeddings [20] comprised of over 30.000 different tokens for the English language. The tokens do not always represent full words, there also exist sub-word units to cope with rare or combined words. An example sequence where splitting words is needed is given in Table 2.1. Each sub-word unit corresponds to an input token.

(**20**) I burnish the brass fixtures until they reflect the lamplight.

Original	WordPiece
I	I
burnish	burn ##ish
the	the
brass	brass
fixtures	fixtures
until	until
they	they
reflect	reflect
the	the
lamplight	lamp ##light
.	.

Table 2.2.: Comparison between the original tokens and the BERT-tokenized version of Example (20). The words *burnish* and *lamplight* are split into sub-words. For this example, the BERT tokenizer from the library HuggingFace[1] was used.

There also exist special tokens that do not correspond to any word in the sequence. At the beginning is the [CLS] token and between sentences the [SEP] token. The [CLS] token in the output represents an embedding for the whole input sequence and can be used for classification tasks. The [SEP] token is used to help the model distinguish between two sentences.

Similar to the Transformer the input to the first encoder layer consists of the sum of the token embedding, the positional embedding, and in addition to the input of the Transformer, a segment embedding. The segment embedding is used to differentiate between sub-sequences of tokens separated by the [SEP] token. An example of the embeddings is given in Figure 2.6.

[1] https://huggingface.co/ (Accessed on: 2021-12-15)

Pre-Training

One of the main reasons for BERTs state-of-the-art performance is extensive pre-training. Pre-training on large corpora of text helps BERT to gain a deeper language understanding, which can be built upon further by fine-tuning on specific tasks. The pre-training of BERT consists of two self-supervised tasks performed on a large corpus of unannotated text: Masked Language Model (MLM) and Next Sentence Prediction (NSP).

- **Masked Language Model** In this task, part of the input tokens are chosen as target tokens to be predicted by BERT, with the target tokens being either masked, changed to a random token or kept the same. To train on this task the output of the target token embedding is fed through a neural network with the number of output neurons corresponding to all words of the vocabulary. Then a softmax operation is applied to get the token probabilities, which are used for computing a cross-entropy loss. A token is selected to be a target with a probability of 15%. A target token has an 80% chance of being masked, a 10% chance of being changed to a random token and a 10% chance of staying the same. When a token is masked the token is changed to the special [MASK] token. The reason for not always masking the target token is to cope with the downside that the [MASK] token only appears during the pre-training and not in downstream tasks.
- **Next Sentence Prediction** Here the model is given two sentences from the training corpus and the task is to predict whether the second sentence actually follows the first. With a probability of 50% the second sentence is the actual subsequent sentence, otherwise the second sentence is randomly chosen from the corpus. The actual classification is done using the special [CLS] token. Similar to MLM the output embedding of the [CLS] token is fed through a neural network, this time however with a single output neuron to compute the probability that the second sentence follows the first. The goal of this task is to learn sentence relationships, which are essential for tasks like QA.

Pre-Training Corpus

Devlin et al. [2] used a vast corpus comprised of books and the English Wikipedia, with a combined size of approximately 3,300 million words. They focused on including documents instead of single sentences to be able to work with continual sequences, which is needed for the NSP task.

Fine-Tuning

After pre-training, the model parameters can be further fine-tuned on a variety of downstream tasks. This can be done by changing the inputs and appending an additional layer to the output that suits the task at hand. Similar to MLM or NSP, the output embeddings of the special [CLS] token or the sequence tokens can be used in the additional layer. During fine-tuning, all parameters of BERT and of the task-specific layer are trained in combination. The fine-tuning has a much lower computational cost than the pre-training and can typically be done on a single GPU in reasonable time.

Adaptations of BERT

Since the original BERT model was published, different BERT-based models with increased performance or reduced computational requirements have been created.

There are several ways to increase the performance. Improvements have been made by using different training procedures, more data or changes in the architecture. One such model is RoBERTa, created by Liu et al. [21], which stands for ”Robustly optimized BERT approach”. This model is trained with a different pre-training approach. Training is done on an adapted version of the MLM task without NSP, with larger batch size and on more data. These changes increase the overall performance compared to BERT.

Another modification on the training regime was done by Clark et al. [22] in their approach Efficiently Learning an Encoder that Classifies Token Replacements Accurately (ELECTRA). Instead of the MLM task, they trained their model on replaced token detection, where some tokens are replaced by synthetic but plausible tokens that are generated by an additional network. The model tries to predict

for all tokens whether it was replaced or not. The main advantage to MLM is that this task allows the model to train on all the tokens and not just a small subset of masked tokens. This makes the pre-training more efficient and results in ELECTRA outperforming both BERT and RoBERTa models when trained with similar computational resources.

Even though BERT performs well on a variety of domains, it can be beneficial to create domain-specific models. BioBERT [23] is a BERT model pre-trained on biomedical texts and outperforms BERT on tasks performed in the biomedical domain. A similar model is SciBERT [24], which is pre-trained on scientific texts and additionally uses a domain-specific WordPiece vocabulary.

Performance is not the only important metric. The base model of BERT consists of 110 million parameters, which can be a drawback when executing it with limited computational resources. Sanh et al. [25] have created DistillBERT, which consists of only 66 million parameters and is 60% faster with only a small drop in performance compared to BERT. The model is trained using knowledge distillation, which is a compression method where a small student model is trained to approximate the output of a large teacher model. The teacher model for DistillBERT is the BERT base model.

BERT for Other Languages

BERT can be trained for any language if enough data is available. There exist pre-trained BERT models for a variety of languages. The pre-training for language-specific models is done similarly to the original BERT model, with a large unannotated corpus like Wikipedia or news articles of the specific language. Another approach is to train a single BERT model for multiple languages together, as is the case for Multilingual BERT (mBERT) [2], which is trained with Wikipedia data from 104 languages. Multilingual models like mBERT show an overall good performance, however, they are usually outperformed by language-specific models for most tasks [26].

(**21**) Ärzte empfehlen eine Stunde Sonnenlicht pro Tag.

Original	BERT	mBERT	German BERT
Ärzte	Ä ##rz ##te	Ä ##rz ##te	Ärzte
empfehlen	em ##pf ##eh ##len	em ##pf ##eh ##len	empfehlen
eine	e ##ine	eine	eine
Stunde	St ##und ##e	Stunde	Stunde
Sonnenlicht	Son ##nen ##lich ##t	Sonne ##nl ##icht	Sonnen ##licht
pro	pro	pro	pro
Tag	Tag	Tag	Tag
.	.	.	.

Table 2.3.: Comparison between the original words and the tokenized versions of the original BERT model, the multilingual BERT model and the German BERT model for Example (21). For this example, the respective tokenizers from the library HuggingFace were used.

German BERT

Chan et al. [27] have released a pre-trained BERT version for German that outperform mBERT in language-specific downstream tasks. For pre-training, they used a diverse set of sources, including Wikipedia, books and legal documents. Instead of the MLM task they used Whole Word Masking (WWM), which is similar to MLM, but with the extension that masking one sub-word token results in automatically masking all other sub-word tokens of the original word. Additionally they created a custom WordPiece vocabulary specifically for the German language[2]. A comparison between the effect of different vocabularies can be seen in Table 2.3. The tokenizer of the original BERT model uses a vocabulary for English, therefore it has to split most German words into small sub-words for this example sentence. The mBERT tokenizer is an improvement, but it still needs to split common German words like *empfehlen*. The tokenizer of the German BERT model only needs to splits the compound word *Sonnenlicht*.

[2] https://www.deepset.ai/german-bert (Accessed on: 2022-02-07)

Limitations

BERT and Transformers have revolutionized NLP, however, there are still many open questions to why BERT works and what limitations exist. As with many deep learning models, the complex architecture and data-driven training make it difficult to accurately describe what is happening inside the model.

The following is a non-exhaustive list of BERTs limitations [28].

- **Negation** BERT does not handle negations well. BERT is unable to differentiate between a sentence and its negation, even though the meaning of the sentence has changed.
- **Number Representation** Numbers in BERT are treated the same as character tokens, which leads to insufficient representations. In addition and number decoding tasks, BERT has difficulties generalizing beyond the training data.
- **Reasoning** BERT learns knowledge about the world, however, it cannot reason well. For example, BERT is aware that people can walk into houses, but it does not know that people are smaller than houses.
- **Bias** Since BERT is pre-trained on a vast text corpus, it also inherits the same biases inherited in the data, e.g. race or gender stereotypes. For example, it has been shown that BERT will infer that people with an Italian name are Italian.

2.1.5. Optical Character Recognition

In this section we give a short overview of Optical Character Recognition (OCR) and what challenges need to be solved during the process.

OCR software can extract machine-readable text from an image or scanned document. There has been much progress in OCR technology in the past years, however, it can be error-prone if certain conditions are not met. OCR is a critical step, as OCR errors can propagate to downstream tasks, which can result in a strong negative effect on the performance [29].

OCR Pipeline

Sharma et al. [30] identified the common steps of the traditional OCR pipeline. First, a digital copy of the physical document is created using a camera. Next pre-processing is done to make the image better suited for further processing. Various techniques can be applied, for example skew-detection to correct for misalignments during scanning, or binarization to convert the image from color to greyscale. Pre-processing has a large impact on the performance of OCR systems. The next step is segmentation. Here the document is segmented to identify the individual paragraphs, lines, words and characters. Then features are extracted from the segmented parts. Many possible features are available, including features of the segmented characters, for example structural information like curvature, dots or size. The last step is classification, where the characters and words are predicted based on the extracted features. This can be done with models like support vector machines or k-nearest neighbor classifiers. Additionally, a post-processing step for error correction can be added.

Deep learning techniques however introduce some changes in the traditional OCR pipeline. In modern OCR systems, CNN architectures are used, which work directly on the pixels of the input images. Many OCR systems like Tesseract[3] also include LSTM models for a line-based processing instead of only single characters [31].

Error Correction

OCR software has a very low error rate if certain assumptions are met, e.g. a good quality of the document, simple layouts and an easily recognizable font. Real-world data is however very heterogeneous, especially historic texts.

Nguyen et al. [32] have summarized a topology of prominent spelling errors occurring in OCR texts and ways to correct them. An overview is given in Table 2.4. Most errors in OCR occur as single-error typos, which means a predicted word has a single incorrect character compared to the ground truth word.

An important distinction is between non-word and real-word errors. Non-word errors happen when a predicted word is not part of a dictionary. If the predicted word is found in a dictionary, but still differs from the ground truth, it is called a

[3] https://github.com/tesseract-ocr/tesseract (Accessed on: 2022-04-09)

Error Type	Ground Truth	OCR Prediction
Single-error	school	schopl
Multi-error	school	schopi
Non-word error	peace	piace
Real-word error	peace	piece
Incorrect split error	depend	de pend
Run-on error	is said	issaid
Non-standard mapping	main	rnain

Table 2.4.: Overview of common OCR errors.

real-word error. An additional difficulty arises when the ground truth word is also not part of the dictionary. These words are called out-of-vocabulary words.

Another type of error is given by incorrect word boundaries. An incorrect split error happens due to extra white space, in contrast to the run-on error, where white space is missing. Incorrect split errors are more frequent than run-on errors. A ground truth character that is represented as multiple characters or is left out in the prediction is called a non-standard mapping. Non-standard mappings often occur in a systematic way depending on the OCR engine and the used dataset. An often occurring error, especially in old documents with lower physical quality, are additional punctuation marks like commas and dots.

There are two main ways to correct for errors: dictionary-based and context-based. Dictionary-based methods use dictionaries or character n-gram models to fix incorrect words. As most incorrect words only differ from the ground truth in a few characters, the edit distance is a useful method to find possible solutions. Dictionary-based methods have however the downside that they only work on non-word errors. Context-based methods include the context of the incorrect word in the correction process, which means they can also correct real-word errors.

Two common metrics[4] to evaluate the generated text of OCR systems are the character error rate (CER) and the word error rate (WER), which represent the percentage of incorrectly generated characters and words. Equation (2.5) shows

[4] https://sites.google.com/site/textdigitisation/qualitymeasures/computingerrorrates (Accessed on: 2022-04-01)

the basic definitions of the metrics. The CER is computed using the insertions i_c, substitutions s_c and deletions d_c, that are needed to convert the original text to the OCR text on a character basis, as found in the Levenshtein distance[5]. The sum of these values is divided by the total number of characters n_c in the original text. The WER is computed similarly to the CER, but instead of characters, whole words are used to calculate i_w, s_w, d_w and n_w.

$$\begin{aligned} CER &= \frac{i_c + s_c + d_c}{n_c} \\ WER &= \frac{i_w + s_w + d_w}{n_w} \end{aligned} \tag{2.5}$$

It must be noted that the CER and WER can become larger than 100% if computed using Equation (2.5). For example, the CER of the original text *cause* and the OCR prediction *cause1234567* is 140%, as there are 7 insertions needed to convert the original text to the prediction, while the original text consists of only 5 characters. To avoid error rates over 100%, normalized forms of the CER and WER have been proposed. An example of a normalized CER and WER is given in Equation (2.6), where c_c and c_w are the numbers of correctly predicted characters and words.

$$\begin{aligned} CER_{norm} &= \frac{i_c + s_c + d_c}{i_c + s_c + d_c + c_c} \\ WER_{norm} &= \frac{i_w + s_w + d_w}{i_w + s_w + d_w + c_w} \end{aligned} \tag{2.6}$$

2.1.6. Text Annotation

Text annotation is the process of adding specific information to the text. Many applications in NLP depend on annotated text, especially in the area of machine learning.

To annotate the texts for this work we use the tool BRAT [33]. BRAT is a browser-based text annotation tool with a customizable annotation scheme that supports

[5] https://en.wikipedia.org/wiki/Levenshteindistance (Accessed on: 2022-04-02)

[6] https://github.com/nlplab/brat/tree/master/example-data/tutorials (Accessed on: 2022-08-14)

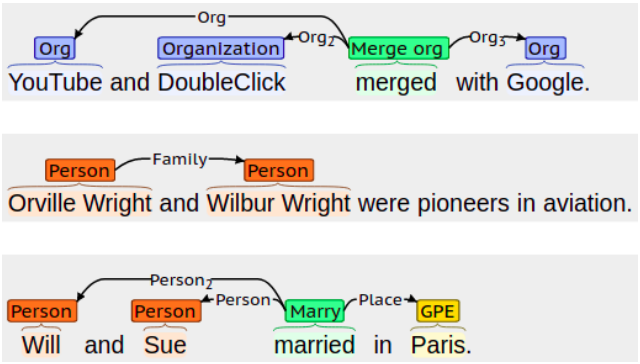


Figure 2.7.: BRAT example annotation of sentences. Text annotated in blue, red and yellow denotes entities and text in green events. Black arrows are relations between annotations. The examples are adapted from the official tutorial[6].

various tasks. The main parts of an annotation scheme in BRAT are Entities, Relations, Events and Attributes.

- **Entities** Entities denote the possible types a text span can be labeled, e.g. *Person* and *Location*.
- **Relations** Relations are the available relationships between two annotations, e.g. two annotations of type *Person* could be connected by the relation *Family*. Which relations are possible between which Entities must be defined.
- **Events** Events are used to annotate specific occurrences between annotations, e.g. *Marriage* between two *Person* entities.
- **Attributes** Attributes define flags to additionally mark annotations, e.g. to note the confidence of an annotation with *Certain* and *Uncertain*.

Examples of sentences annotated with BRAT using a minimal example configuration can be seen in Figure 2.7.

2.2. State of the Art

This section discusses approaches for causal relationship extraction from historical texts from different angles. First, we introduce different types of methods and existing solutions for causal relationship extraction. Afterward, we examine used annotation schemes and their components. Lastly, we present relevant approaches for information extraction from historical text.

2.2.1. Causal Relationship Extraction

Detecting causal structures in natural text is a difficult task, as causal relations can occur in very diverse and ambiguous forms. The topic has a long research history and various approaches have been developed.

Xu et al. [34] have identified three main methods used for causality extraction: text classification, relation extraction and sequence labeling.

- **Text Classification** With text classification a model classifies whether text includes a causality, but does not extract the causal events from the text.
- **Relation Extraction** For relation extraction the events in the text are known and the task is to classify if events are causally related.
- **Sequence Labeling** In sequence labeling, a model assigns a tag to each token of the text. The tagging scheme is pre-defined and includes tags for causal arguments, such as cause and effect.

Existing approaches to the tasks can be broadly categorized into non-statistical and statistical methods [35]. Traditional methods predominantly relied on non-statistical approaches based on pattern matching, often derived from domain-specific knowledge. Statistical methods, specifically machine learning and in recent years deep learning, make use of implicit patterns in the text and outperform non-statistical methods for most tasks.

Li et al. [36] used a deep learning approach based on pre-trained Flair embeddings [37] for causality extraction. They view the problem as a sequence labeling task and annotate cause-effect relationships using three tags: cause, effect and

embedded causality. The tag embedded causality is for entities that are part of multiple causal relations, for example in causal chains. Example (22) shows an annotated sentence from their corpus[7], where the embedded causality *the chronic inflammation* acts as a cause for *an increased acid production* and as an effect from *Helicobacter pylori infection*. They also developed an algorithm called "tag2triplet" to differentiate between multiple cause-effect relationships within one sentence, where multiple relations have a shared embedded causality. To cope with long-range dependencies in the sentences, they added a multi-head self-attention layer to their model.

(22) The current view is that **the chronic inflammation**$_{\text{Embedded}}$ [...] caused by **Helicobacter pylori infection**$_{\text{Cause}}$ results in **an increased acid production**$_{\text{Effect}}$ [...].

Similarly, Rehbein and Ruppenhofer [1] used a sequence labeling model based on BERT to predict the causal arguments and also the type of causality in German texts. The prediction of the causal arguments works reasonably well, and for the prediction of the type of causality, their F1 score is higher than their baseline for all but one causal type. Their baseline for this task is the most frequent causal type of the trigger word in their database, which gives already quite good results.

Khetan et al. [38] developed a BERT-based architecture for extracting cause-effect relations between events. In this task, the events in the sentence are already provided, and the model classifies whether both events form a cause-effect relationship. Their model combines BERT's [CLS] token of a sentence with the individual embeddings of the events. The event embeddings increase the performance compared to an approach with only the [CLS] token. Their model achieved state-of-the-art performance for three different datasets.

It should be noted that tasks related to causality are sometimes improperly labeled as such. One example is textual entailment, which is often included in NLP benchmarks. In this task, given a textual statement that is assumed to be correct, it must be inferred whether a hypothesis text is plausible. However, the task does not assume that the statement and the hypothesis are causally related.

[7] https://github.com/Das-Boot/scite (Accessed on: 2022-3-24)

Datasets

The number of datasets for causality extraction is rather small, with the additional drawback that most datasets use a custom annotation scheme and are of limited size. The schemes range from simply annotating cause and effect pairs to annotating intricate linguistic and semantic causal components. Extending a scheme has the advantage of including more information about causal relationships, which comes with an increase in complexity.

The International Workshop on Semantic Evaluation (SemEval) has introduced various tasks and annotated datasets in the context of causality. Two notable examples are the SemEval-2007 Task 4 [39] and the SemEval-2010 Task 8 [40]. SemEval-2007 Task 4 is about semantic relation classification of noun phrases, including cause-effect relations. The dataset[8] includes 114 causal relations, however not all found causal relationships are annotated. This task was continued in Semeval-2010 Task 8 with a corpus[9] of over a thousand cause-effect relations. The sentences for both tasks were collected using a pattern-based web search.

The SemEval-2010 Task 8 data was also used by Li et al. [36], who created a dataset with a more detailed annotation scheme. The original sentences contained only a single annotated causal relation, while in their extended dataset they annotated all available relations.

Dunietz et al. [8] created the BECAUSE[10] corpus using a more elaborate annotation scheme. They annotated and classified causal relationships based on pre-defined causal patterns. The scheme also allows for annotations to overlap, which happens if two different causal relations include the same part of the text. The corpus contains over five thousand sentences, with nearly two thousand including causal language. The sentences are taken from various sources, including newspaper articles and congress hearings.

Based on the annotation scheme in [8], Rehbein and Ruppenhofer [1] created their

[8] http://nlp.cs.swarthmore.edu/semeval/tasks/task04/data.php (Accessed on: 2022-3-29)

[9] https://huggingface.co/datasets/semeval2010task8 (Accessed on: 2022-3-29)

[10] https://github.com/duncanka/BECAUSE (Accessed on: 2022-2-09)

dataset[11] of German texts from the TiGer [41] and Europarl [42] corpus. They used a fine-grained annotation scheme, with the restriction of annotating at most one causal relation per sentence. The dataset contains over four thousand sentences, of which nearly three thousand are causal.

Caselli and Vossen [43] created the Event StoryLine Corpus[12] as a benchmark corpus for storyline detection. Their goal is to monitor and connect information about a specific topic over events occurring in different documents. The dataset consists of 258 documents about disaster events from the Event Coreference Bank+ [44]. The documents are annotated for events and various relation types. Of the over two thousand annotated relations, only 117 are explicit causal relations. A special characteristic of the corpus is that relations are not restricted to being within the same sentence.

Also related to causality, great efforts have been made to collect and structure commonsense knowledge. One such project is ATOMIC [45], an atlas about inferential knowledge. The atlas is a collection of "if-then" relations between events and commonsense inferential knowledge, categorized into nine dimensions. A relation for an event from the corpus is given in Example (23). ATOMIC was created using crowdsourcing and comprises over 877 thousand relations.

(23) If **Person X pays Person Y a compliment**$_{\text{Event}}$, then **Person Y will smile**$_{\text{Inference}}$.

Similar to ATOMIC is GLUCOSE [46], a dataset for causal knowledge in stories. GLUCOSE is built around short children's stories, where for each sentence in the story commonsense explanations for ten different dimensions are created in the form of semi-structured inference rules. The dimensions cover different causal aspects and state changes. In contrast to ATOMIC, GLUCOSE is contextual and bases its causal knowledge on specific situations, which is similar to how humans do causal reasoning. GLUCOSE includes more than 670 thousand annotations, gathered using crowdsourcing.

[11] https://github.com/josefkr/causalannotationsDE (Accessed on: 2022-2-09)

[12] https://github.com/tommasoc80/EventStoryLine (Accessed on: 2022-3-24)

2.2.2. Annotation of Causality

In this section, we take a detailed look into two annotation schemes for sequence labeling of causal language, namely the works of Dunietz et al. [8] and Rehbein and Ruppenhofer [1].

The scheme used by Dunietz et al. [8] annotates explicit causal relations based on causal connectives. Causal connectives are linguistic patterns based on the concept of construction grammar [47] that signal causality in a sequence. Examples are <*effect*> *arises from* <*cause*> and <*cause*> *leads* <*effect*> *to* <*effect*>, with <*cause*> and <*effect*> acting as placeholders. Dunietz et al. collected a catalog of occurring causal connective patterns. Examples (24) and (25) from their annotated corpus BECAUSE[13] show the causal connectives in sentences.

(24) **A slight unpleasantness**$_{\text{Effect}}$ **arose from**$_{\text{Causal Connective}}$ **this discussion**$_{\text{Cause}}$.
Type: *Consequence*, Degree: *Facilitate*

(25) **Recent events**$_{\text{Cause}}$ have **led**$_{\text{Causal Connective}}$ **the Basel Committee on Banking Supervision**$_{\text{Effect}}$ **to**$_{\text{Causal Connective}}$ **consider higher capital charges for such items**$_{\text{Effect}}$.
Type: *Motivation*, Degree: *Facilitate*

A causal connective must be present for an annotation to occur. In addition, the annotation includes the causal type and semantic category, as introduced in 2.1.1, but with some modifications. Firstly the type Inference for evidentiary causation is not included in the scheme. Secondly, to make the three semantic categories Cause, Enable and Prevent of [9] more practicable, Dunietz et al. [8] combined them into the positive and negative degrees Facilitate and Inhibit. Facilitate corresponds to Cause and Enable, and Inhibit is used instead of Prevent.

Rehbein and Ruppenhofer [1] adapted the scheme from Dunietz et al. [8] to annotate causal language on a German corpus. They kept the annotation of type and degree, but instead of causal connectives, they used a trigger-based approach. They defined a set of causal triggers, which similar to the causal connectives, can introduce a causal relation. Causal trigger words are for example *wegen* (because of), *auslösen* (trigger) or *Grund* (reason). As a simplification, for each annotated sentence they restricted themselves to only annotate a single trigger with corresponding arguments, even if more possible trigger words are present. The causal triggers are similar to the

[13] https://github.com/duncanka/BECAUSE (Accessed on: 2022-2-09)

causal links and verbs identified by Khoo et al. [6] in Section 2.1.1, but Rehbein and Ruppenhofer also included nouns that introduce causality as possible triggers.

Given the presence of a trigger word in an instance, first it is evaluated if the trigger occurs in a causal context, as some trigger words can be ambiguous (see Examples 4 and 5). After a causality is found, the other causal arguments are determined. If a trigger word is found in a non-causal context, the trigger word is annotated as non-causal.

Rehbein and Ruppenhofer [1] increased the number of causal arguments compared to Dunietz et al. [8]. Their scheme consists of four participant roles, two auxiliary roles and two types of trigger annotations. The different arguments are presented in Table 2.5. The additional participant and auxiliary roles are added to introduce a more fine-grained annotation and are based on the annotation scheme used in FrameNet [48]. We will explain the roles in more detail, with examples from their annotated corpus[14].

Participant Roles	**Auxiliary Roles**	**Trigger Annotation**
• Cause	• Controller	• Trigger
• Effect	• Support	• None
• Actor		
• Affected		

Table 2.5.: Causal arguments in [1].

Participant Roles

Participant roles consist of the basic causal arguments cause and effect, and the additional roles actor and affected. Actor and affected are included to capture more information about the causal structure. Cause and effect are for inanimate actions or events, while actor and affected are for entities. For actor the entity is typically sentient.

The actor role denotes entities responsible for initiating the effect, while affected entities are influenced by the consequences of the cause. Actor and affected can occur in combination with cause and effect (Example 24), but also alone (Example 25).

[14] https://github.com/josefkr/causalannotationsDE (Accessed on: 2022-2-09)

(24) Vielleicht kann **die Kommission**$_{\text{Actor}}$ da auch **ihren Teil**$_{\text{Cause}}$ **dazu**$_{\text{Effect}}$ **beitragen**$_{\text{Trigger}}$.
(Perhaps the Commission can do its part there as well.)
Type: *Consequence*, Degree: *Facilitate*

(25) **Die Kommission**$_{\text{Actor}}$ will **einen Teil dieser Vereinbarung**$_{\text{Affected}}$ **kippen**$_{\text{Trigger}}$.
(The Commission wants to overturn part of this agreement.)
Type: *Motivation*, Degree: *Inhibit*

Auxiliary Roles

The roles support and controller connect the trigger to sections not reached by its syntactic projection. This is typically needed only for noun triggers. Support marks verbs and prepositions like *ist* (is) which link a subject to its complement (Example 26). Controller is used in cases where an event is presented, for example with the verb *nennen* (to name) in combination with the preposition *als* (as) (Example 27).

(26) Der **Grund**$_{\text{Trigger}}$ **ist**$_{\text{Support}}$ **vor allem Enttäuschung über die traditionellen Parteien**$_{\text{Cause}}$.
(The reason is mainly disappointment with the traditional parties.)
Type: *Motivation*, Degree: *Facilitate*

(27) **Als**$_{\text{Controller}}$ **Grund**$_{\text{Trigger}}$ wird **die gute Auftragslage**$_{\text{Cause}}$ **genannt**$_{\text{Controller}}$.
(The good order situation is cited as the reason.)
Type: *Motivation*, Degree: *Facilitate*

Trigger Annotations

Here the scheme differentiates between trigger and none. Trigger is used when a potential trigger word occurs in a causal meaning, whereas none is used if it occurs in a non-causal meaning.

As an example, the trigger *durch* (through/by) can have a causal and non-causal meaning. Example (28) has a causal meaning, but Example (29) has a non-causal meaning.

(28) Ich begrüße **die einhellige Unterstützung des Fahrplans für die Überwachung der Finanzmärkte**$_{\text{Effect}}$ **durch**$_{\text{Trigger}}$ **den Europäischen Rat**$_{\text{Actor}}$.

(I welcome the unanimous support for the roadmap for financial market supervision by the European Council.)
Type: *Motivation*, Degree: *Facilitate*

(29) Dann gehen wir **durch**$_{\text{None}}$ die Galerie.
(Then we go through the gallery.)
Type: -, Degree: -

2.2.3. Information Extraction from Historical Texts

There are several obstacles to overcome when using modern tools and frameworks with historical texts. Since the documents are initially in printed form, they need to be converted to text. There has been much progress in OCR technology in the past years, however, errors during the process frequently occur. This can have a strong negative effect on the performance of pre-trained language models in downstream tasks [49].

A notable example of how to deal with a faulty vocabulary in OCR text is presented by the approach Boros et al. [50] submitted to the HIPE evaluation campaign of CLEF 2020 [51]. In their solution to the NER task, they included misspelled and out-of-vocabulary words into the vocabulary of their BERT model. By relearning the new words the influence of OCR errors was reduced.

Another problem when working with historical texts is the change of language over time. BERT and similar deep learning models are pre-trained on a vast corpus using masked language modeling and the next sentence prediction task [2]. This however makes BERT dependent on the training data distribution, which typically consists of modern texts like Wikipedia articles. Fine-tuning BERT on the specific dataset can increase the performance, especially when the dataset is different from the data BERT was trained on. Brunner et al. [52] compared deep language models, including BERT, on a corpus of historical German documents. Their tasks consisted of detecting types of speech, thought and writing representation (STWR) in fiction and non-fiction texts. Fine-tuning BERT on the corpus increased its performance in all of their evaluated tasks, compared to the standard BERT model.

For most tasks concerning historical documents, an important aspect to consider is that the spelling of words might have changed over time. To convert historical spelling variations to their modern equivalent, text normalization can be applied.

Historical text normalization has a long research history, and there exist many viable approaches [53]. A simple but effective normalization approach is to use substitution lists of historical spelling variants. More advanced methods include rule-based systems and machine learning methods, however many approaches depend on manually labeled training data.

Exploring historical documents in the context of event extraction (EE) is also an active research domain. An event is commonly defined as a specific incident at a particular time and place with some actors involved [54]. Lai et al. [55] applied state-of-the-art EE models based on BERT on African-American newspapers of the 19th century. The task included identifying event trigger words, classifying the event and finding arguments that specify the event. The models applied to the historic documents yielded a inferior performance compared to modern EE datasets. Fine-tuning BERT to historic text proved to be helpful, but a large performance difference remained.

Cybulska and Vossen [56] created a more traditional pipeline for historic EE on a corpus of historical Dutch documents. First, they annotated event actions, participants, locations and time markers using semantic and syntactic information from part-of-speech (PoS) tagging, dependency parsing, word sense disambiguation and the Dutch WordNet. The labeled text was then combined into events, while also filtering out non-historical events. Their model achieved a reasonably good performance on their corpus, indicating it to be a viable approach for historic EE.

Similar to EE, NER is also a popular task performed on historical documents. Riedl and Padó [57] evaluated NER models based on Conditional Random Fields (CRF) and Bidirectional LSTMs (BiLSTM) on historical and contemporary German texts. They showed that with the help of transfer learning on another contemporary NER corpus, the BiLSTM model outperforms the CRF model in both settings. The performance for both models was however lower for the historical texts, mainly due to errors and formating inconsistencies introduced by OCR.

Schweter and Baiter [58] continued the experiments of [57] and evaluated a pre-trained character-based language model on the same historical datasets. Their model achieved better results than [57], without the need for additional transfer learning on NER. Recently, various BERT models created by Schweter [59] outperformed the models of [57] and [58]. These BERT models were pre-trained on a large historic newspapers corpus and used a custom vocabulary.

As language is subject to change over time, so is the use of causality. This includes linguistic changes as well as how frequently causal language is used. Iliev and Axelrod [60] have identified an increase of causal language in English in the 20th century. They hypothesize that this might be due to better education and the importance of causality in scientific discoveries and technology.

In the context of semantic shift, Hara [61] investigated the change in the meaning of the word *wenn* (if) in Early New High German and New High German. Initially, its sense was temporal and it was used for sequential events. The meaning changed from a temporal relation between events to one event being caused by another event. Later on, other causal words emerged, such as *weil* (because) and *denn* (so), and *wenn* developed the conditional sense it currently has.

Datasets

The number and quality of accessible digitized historical documents are steadily increasing. Still, creating a curated dataset requires time and effort. Even though available text is abundant, it is often not trivial to find relevant documents for a certain task or topic. Historical documents also often include OCR errors or spelling variations, which might need to be corrected. Additionally, many tasks require meta-information about the documents to be available.

For the CLEF HIPE 2020 task, Ehrmann et al. [62] prepared a large corpus[15] of historical newspapers annotated for NER. The documents are in French, German and English, and are sampled diachronically from 1798 to 2018. They removed task-irrelevant sections like tabular data or weather forecasts, however, they did not normalize the OCR generated text to simulate a varying text quality which is typically seen when working with historic documents. The corpus consists of 563 documents with nearly half a million tokens.

Brunner et al. [63] collected the REDEWIEDERGABE[16] corpus, consisting of over 800 German documents, published between 1840 and 1919, with nearly 500 thousand tokens. The corpus contains texts from various sources, with an equal share of

[15] https://github.com/impresso/CLEF-HIPE-2020/tree/master/data (Accessed on: 2022-3-29)

[16] https://github.com/redewiedergabe/corpus (Accessed on: 2022-3-29)

non-fictional and fictional text, as well as text from different decades. In their preprocessing, they corrected OCR errors and exchanged archaic characters with their contemporary representations. Old spelling variations were however not corrected. The corpus is annotated for STWR, and additionally includes metadata for each text, such as author information, the text type and dialect information.

To create a benchmark dataset for historical event extraction, Lai et al. [55] introduced the BRAD corpus. Their goal was to build a dataset of high quality with a focus on a single topic, in this case, rebellions of the African diaspora. The texts are extracted from an African American newspaper, ranging from 1827 to 1909. Their corpus contains 151 documents with over 150 thousand tokens and is annotated for entities and events.

The NER corpus created by Riedl and Padó [57] is a refined subset of the corpus in [64] and contains historical German texts from the Europeana collection[17]. The corpus is split into two datasets: The first dataset consists of Tyrolean newspapers from 1926 with around 87 thousand tokens and the second of Austrian newspapers published between 1710 and 1873 with around 35 thousand tokens. As the corpus is sampled from various periods, the language is not consistent over time.

In the context of cultural heritage, a large effort has been made in recent years to digitize historic documents [65]. Cultural heritage includes all historic or cultural assets of society, with a significant portion being written text. The main goals of this movement are to keep the information of documents safe and increase its accessibility for the public and historians. An example of this is AustriaN Newspapers Online (ANNO)[18], which is a project of the Austrian National Library to provide access to historic Austrian newspapers and journals. Since the project started in 2003, over 24 million pages have been scanned and made freely available over their website. Both the original scans and the computer-readable texts generated by OCR are available.

[17] https://www.europeana.eu/de (Accessed on: 2022-4-6)
[18] https://anno.onb.ac.at/ (Accessed on: 2022-01-06)

3. Materials and Methods

This chapter describes our method for causal relationship extraction using an extended sequence labeling approach. The goal of this thesis is not exclusively a study of causality in text, but it should also be of practical use for information extraction. Therefore, we aim to capture a wide array of naturally appearing causal structures with as few linguistic and semantic limitations as possible.

The problem is defined as follows: Given a sequence of tokens $S = (t_1, t_2, \ldots, t_l)$, with l being the sequence length, the task is to extract all occurring causal relations $R = \{r_1, \ldots, r_n\}$. A causal relation is defined as a sequence of tags in the form of $r_i = (l_1, l_2, \ldots, l_l)$, with l_i being a tag from the set of causal arguments, or the O tag for non-causal tokens. In addition, for each r_i, a causal type and degree is determined.

In Section 3.1 we introduce our annotation scheme and the datasets used in this work. Section 3.2 presents the model to solve the task. Section 3.3 explains the different metrics we apply to evaluate the performance of the model. Lastly, Section 3.4 outlines experiments to comprehend the possibilities and limitations of our approach.

3.1. Data and Annotation

This section introduces the annotation scheme and the two newly created datasets for this work. Additionally, we compare our datasets to the datasets of Dunietz et al. [8] and Rehbein and Ruppenhofer [1].

3.1.1. Annotation Scheme

Our annotation scheme is similar to the scheme of Rehbein and Ruppenhofer [1], as explained in Section 2.2.2, but with a few adaptations. Rehbein and Ruppenhofer annotated only one causal relation per sentence, even if other potential relations existed. In our scheme, however, we annotate all available relations. A token can be tagged as a causal argument in multiple relations, and if two relations share one or more causal tokens, we call them overlapping. Examples (30.1) and (30.2) show two overlapping relations in the same sentence, where the relation in (30.1) overlaps with the cause argument in (30.2). Within a relation, a token must be of a single argument type, e.g. a token cannot be trigger and cause in the same relation.

(30.1) Indessen war auch **eine nicht unbedeutende Fläche an Wald- und Weideland**$_{\text{Affected}}$ **im Wege**$_{\text{Trigger}}$ **der Servitutenablösung**$_{\text{Cause}}$ **an die Berechtigten zur Abtretung gelangt**$_{\text{Effect}}$, so dass [...] 45 Procent der Landesfläche im Staatsbesitze und in Verwaltung des Staates verblieben.
Type: *Consequence*, Degree: *Facilitate*

(30.2) **Indessen war auch eine nicht unbedeutende Fläche an Wald- und Weideland im Wege der Servitutenablösung an die Berechtigten zur Abtretung gelangt**$_{\text{Cause}}$**, so dass**$_{\text{Trigger}}$ [...] **45 Procent der Landesfläche im Staatsbesitze und in Verwaltung des Staates**$_{\text{Effect}}$ **verblieben**$_{\text{Trigger}}$.
Type: *Consequence*, Degree: *Facilitate*
(Meanwhile, a not insignificant area of forest and pasture land had been ceded to the entitled people by way of servitude redemption, such that [...] 45 percent of the land area remained in state ownership and administration.)

Rehbein and Ruppenhofer [1] annotate sentences with triggers from a pre-defined list and use the none argument for triggers occurring in a non-causal context. We discontinue this, as our scheme aims to capture all observable causal structures without restriction to specific triggers. Alternatively to the None argument, we depict non-causality of a sentence as implied by the absence of an annotated trigger.

Another difference is that we interpret the arguments controller and support in a causal instead of a syntactic context. In our scheme, controller denotes circumstances that influence the causal mechanism and specify the context in which causality occurs. In Example (31), the phrase *in den Alpen* (in the Alps) adds contextual information because the causal mechanism might differ for another location.

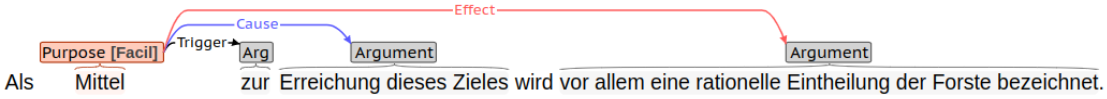


Figure 3.1.: BRAT annotation with triggers of different priorities. The trigger *Mittel* (means) has a higher priority than *zur* (to). Therefore, *Mittel* is annotated as the causation event, while *zur* is annotated as an additional trigger entity. The example is taken from our newly annotated corpus.

Support is used for arguments that increase the impact or highlight the causal relation by further specification. The phrase *neben anderen Factoren* (among other factors) in Example (32) reinforces the causal meaning of the relation.

(31) Während der beherzte Weidmann im Osten die Spur des Bären [...] verfolgt, **lockt**$_{\text{Trigger}}$ **ihn**$_{\text{Affected}}$ **in den Alpen**$_{\text{Controller}}$ **die flüchtige Gemse**$_{\text{Cause}}$ **in unwirtliches Gefelse**$_{\text{Effect}}$ [...].
Type: *Motivation*, Degree: *Facilitate*
(While the courageous hunter pursues the bear's trail in the east [...], the elusive chamois lures him in the Alps into inhospitable terrain.)

(32) Für einzelne bedeutendere Forstcomplexe lassen sich [...] Gesteinsarten anführen, **deren Auftreten**$_{\text{Cause}}$ **neben anderen Factoren**$_{\text{Support}}$ **das vorwiegende Vorkommen einer gewissen Baum- und Strauchflora**$_{\text{Effect}}$ **mitbedingt**$_{\text{Trigger}}$.
Type: *Consequence*, Degree: *Facilitate*
(For some more important forest complexes, [...] rock types can be mentioned whose occurrence, along with other factors, contributes to the predominant occurrence of a certain tree and shrub flora.)

As a sentence might include more than one triggers, we constructed syntactic priority rules to ensure consistent annotations. Predicate triggers have the highest priority, then nouns, and lastly prepositions. In BRAT, trigger entities of the highest available priority are marked as the causation event with causal type and degree, while triggers of lower priority are annotated as separate entities. An example of this can be seen in Figure 3.1. The noun trigger *Mittel* (means) is annotated as the causation event, and the preposition *zur* (to), which has a lower priority, is annotated as an additional trigger entity. However, in this work we simplify the annotations and do not distinguish between the priorities of triggers, therefore both triggers *Mittel* and *to* are treated equally.

The exact argument boundaries are often a question of interpretation. Xu et al. [34] suggest that a model can better learn short argument spans. On the other hand, Rehbein and Ruppenhofer [1] used rather generous spans. We choose to follow the approach of Rehbein and Ruppenhofer, as it makes the arguments more coherent.

An ambiguous case occurs with negations, for example, triggers like *not increase*. From a strict causal interpretation, *not* is not a trigger, as it does not introduce causality but only modifies it. Therefore we choose to split such cases into a strictly causal trigger, in this case, *increase*, and an additional controller argument *not*. This change also inverts the causal degree of the example from Inhibit to Facilitate.

Similar to [8], we annotate coreferences if they involve a causal argument. A sentence including a coreference is presented in Example (33). The pronoun *diese* (it/this) is a reference to the noun phrase *die Besoldung der Beamten* (the renumeration of the civil servants). Note that only *diese* is annotated as effect, as *die Besoldung der Beamten* is not directly part of the causal relation.

(33) Was *die Besoldung der Beamten* anbelangt, so **richtet sich**$_{\text{Trigger}}$ ***diese***$_{\text{Effect}}$ **nach**$_{\text{Trigger}}$ **den Bezügen der betreffenden allgemeinen Rangsclasse**$_{\text{Cause}}$, [...].
Type: *Consequence*, Degree: *Facilitate*
(As far as the remuneration of civil servants is concerned, it is based on the the remuneration of the relevant general rank classes, [...].)

3.1.2. Data Sources

Four different datasets are used to train and evaluate our model. Two of them are newly annotated texts from the late 19th century, the other two are the already annotated dataset of Rehbein and Ruppenhofer [1] and the BECAUSE dataset of Dunietz et al. [8]. They are however only utilized for pre-training purposes and not for evaluation.

Table 3.1 provides a summarization of all four datasets. The BECAUSE dataset is unfortunately not complete, as about two thirds of the text have not been publicly released. The causal arguments actor, affected, controller and support do not exist in BECAUSE, and the additional argument means, which occurs only rarely, has been disregarded. It must be noted that the number of arguments in Table 3.1 is

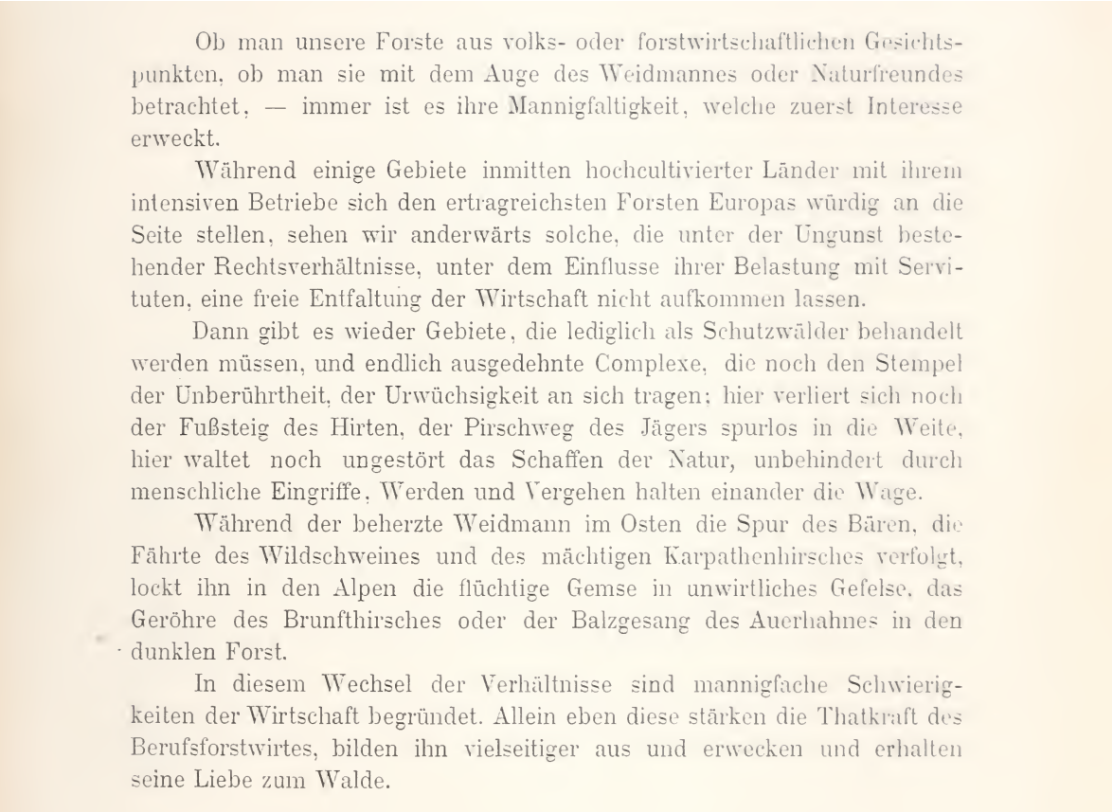
		SF	FV	Rehbein and Ruppenhofer	BECAUSE
	Sentences	399	1299	4,389	1,527
	% causal	51.13	16.86	60.65	24.89
	Tokens	14,725	19,316	126,084	33,569
Type	**Purpose**	68 (21.7%)	109 (28.3%)	292 (11.0%)	78 (14.1%)
	Motivation	82 (26.2%)	83 (21.6%)	997 (37.5%)	149 (26.9%)
	Conseq.	163 (52.1%)	193 (50.1%)	1,373 (51.6%)	327 (59.0%)
Degree	**Facilitate**	289 (92.3%)	370 (96.1%)	2,494 (93.7%)	497 (89.7%)
	Inhibit	24 (7.7%)	15 (3.9%)	168 (6.3%)	57 (10.3%)
Arg.	**Trigger**	657 (42.7%)	705 (39.7%)	2,739 (29.8%)	703 (39.2%)
	Actor	40 (2.6%)	30 (1.7%)	176 (1.9%)	0 (0.0%)
	Affected	71 (4.6%)	50 (2.8%)	321 (3.5%)	0 (0.0%)
	Cause	355 (23.1%)	456 (25.7%)	2,318 (25.2%)	494 (27.6%)
	Controlling	38 (2.5%)	56 (3.2%)	116 (1.3%)	0 (0.0%)
	Effect	365 (23.7%)	468 (26.4%)	3,033 (33.0%)	595 (33.2%)
	Support	12 (0.8%)	9 (0.5%)	489 (5.3%)	0 (0.0%)

Table 3.1.: Overview of the different datasets used in this work. The percentages show the distribution of causal type, degree and arguments in the respective datasets. The available dataset of Dunietz et al. [8] consists of about one-third of their annotated corpus, as the text of the remainder has not been released publicly.

slightly higher than the number of arguments during annotation, as we treat split entities as separate arguments.

Staats- und Fondsforste (SF)

The first newly annotated document is titled ”Die Staats Und Fondsforste Österreichs: Ein Leitfaden Für Den Besuch Der Ausstellung Der Österreichischen Staatsforstverwaltung In Paris 1900” (The state and fund forests of Austria: A guide for visiting the exhibition of the Austrian State Forestry Administration in Paris 1900), published by the K. K. Ackerbauministerium in 1900. We call this document ”Staats- und Fondsforste” (SF) for short. It contains a detailed summarization of the forests administrated by the Ministry of Agriculture of the Austro-Hungarian Empire,



Ob man unsere Forste aus volks- oder forstwirtschaftlichen Gesichtspunkten, ob man sie mit dem Auge des Weidmannes oder Naturfreundes betrachtet, — immer ist es ihre Mannigfaltigkeit, welche zuerst Interesse erweckt.

Während einige Gebiete inmitten hochcultivierter Länder mit ihrem intensiven Betriebe sich den ertragreichsten Forsten Europas würdig an die Seite stellen, sehen wir anderwärts solche, die unter der Ungunst bestehender Rechtsverhältnisse, unter dem Einflusse ihrer Belastung mit Servituten, eine freie Entfaltung der Wirtschaft nicht aufkommen lassen.

Dann gibt es wieder Gebiete, die lediglich als Schutzwälder behandelt werden müssen, und endlich ausgedehnte Complexe, die noch den Stempel der Unberührtheit, der Urwüchsigkeit an sich tragen; hier verliert sich noch der Fußsteig des Hirten, der Pirschweg des Jägers spurlos in die Weite, hier waltet noch ungestört das Schaffen der Natur, unbehindert durch menschliche Eingriffe. Werden und Vergehen halten einander die Wage.

Während der beherzte Weidmann im Osten die Spur des Bären, die Fährte des Wildschweines und des mächtigen Karpathenhirsches verfolgt, lockt ihn in den Alpen die flüchtige Gemse in unwirtliches Gefelse, das Geröhre des Brunfthirsches oder der Balzgesang des Auerhahnes in den dunklen Forst.

In diesem Wechsel der Verhältnisse sind mannigfache Schwierigkeiten der Wirtschaft begründet. Allein eben diese stärken die Thatkraft des Berufsforstwirtes, bilden ihn vielseitiger aus und erwecken und erhalten seine Liebe zum Walde.

Figure 3.2.: Excerpt from SF document (Page 3).

including their magnitude, organization, and cultural and economic impact. The scanned documents can be accessed online[1]. The OCR text was generated using the Tesseract OCR engine[2] and contains only few OCR artifacts. An excerpt from the document can be seen in Figure 3.2, and a part of the corresponding annotation is given in Figure 3.3.

[1] https://archive.org/details/diestaatsundfond00aust (Accessed on: 2022-05-18)

[2] https://github.com/tesseract-ocr/tesseract (Accessed on: 2022-04-23)

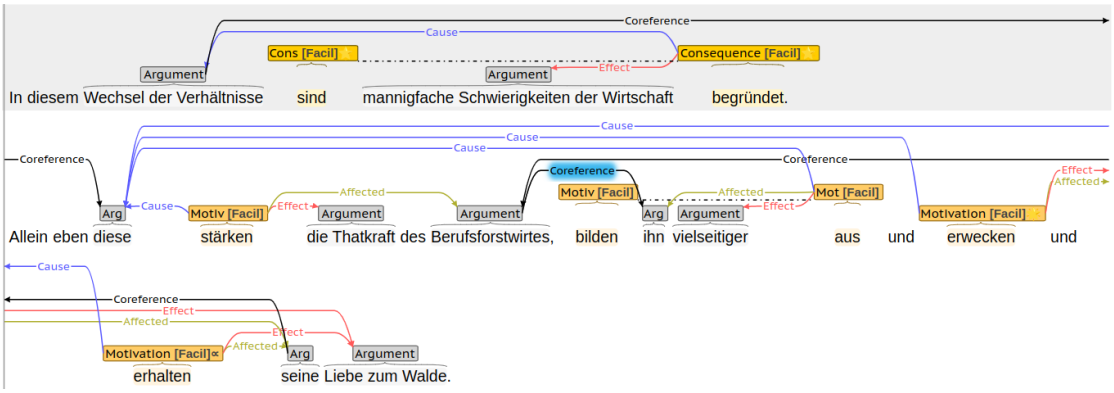


Figure 3.3.: BRAT Annotation example of SF. The second sentence includes four separate causal relations with several overlapping arguments.

Forstvermessung (FV)

The second annotated document is titled ”Instruction für die Begrenzung, Vermarkung, Vermessung und Betriebseinrichtung der Österreichischen Staats- und Fondsforste” (Instruction for the Delimitation, Marking, Surveying and Operating Equipment of the Austrian State and Fund Forests) and was published by the Kaiserlich-Königliche Hof- und Staatsdruckerei in 1878. We call this document ”Forstvermessung” (FV) for short. FV is of the same domain as SF and gives detailed instructions on properly managing the forests of the Austro-Hungarian Empire. This document is longer and noisier than SF, with more frequent OCR errors and heterogeneous formatting. The OCR was generated by the Münchner Digitalisierungs Zentrum and can be accessed online[3]. We only use approximately half of the document, the remaining text is excluded from our experiments. The first paragraph of FV is given in Figure 3.5, with the annotated first sentence in Figure 3.4.

3.1.3. Annotation Process

In this section, we provide information about the annotation process, which occurred in several steps accompanied by joint discussions.

[3] https://digitale-sammlungen.de/de/view/bsb11367093?page=,1 (Accessed on: 2022-05-05)

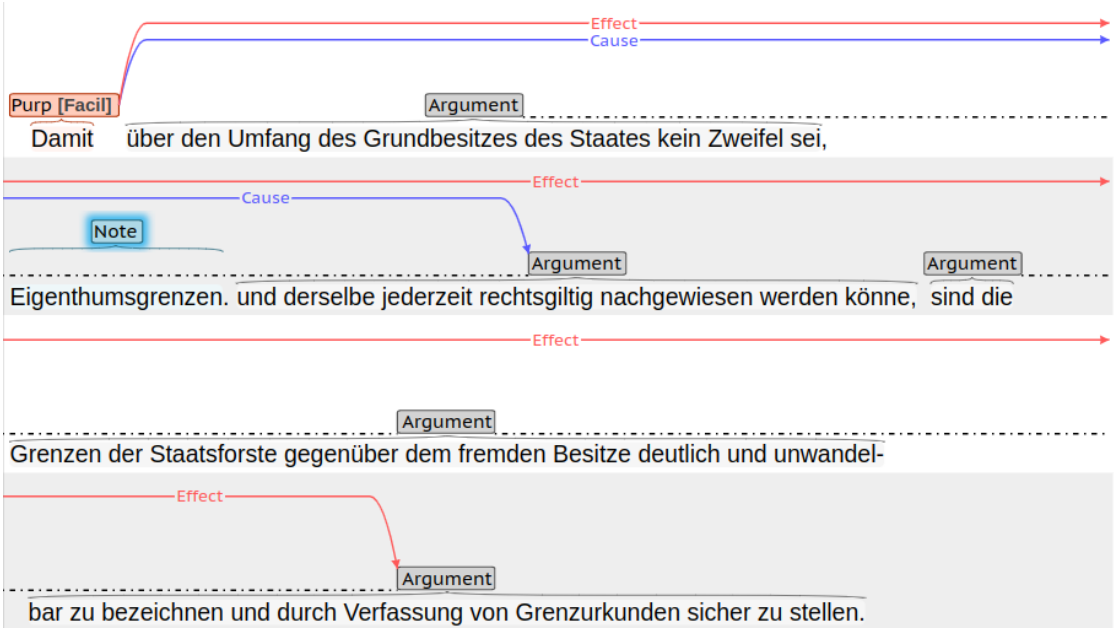
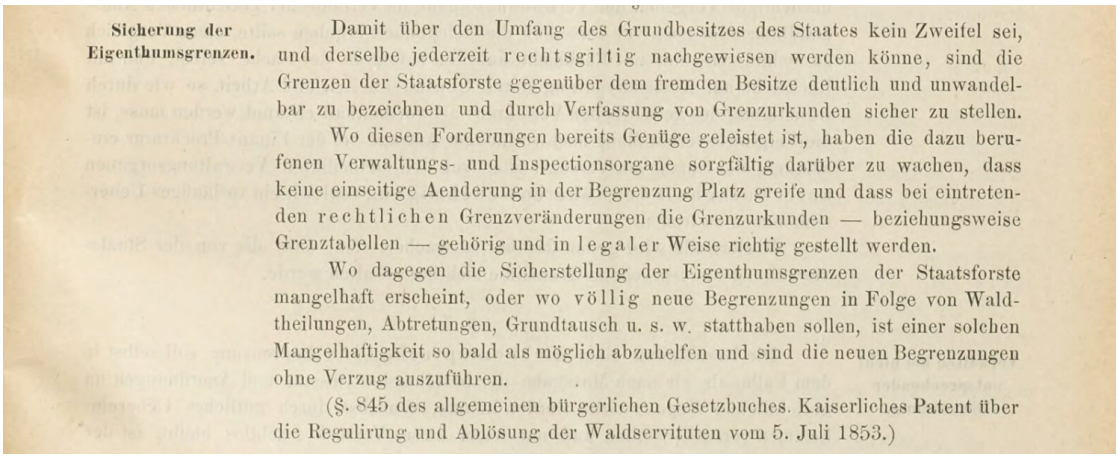


Figure 3.4.: BRAT Annotation example of FV.


Sicherung der Eigenthumsgrenzen.

Damit über den Umfang des Grundbesitzes des Staates kein Zweifel sei, und derselbe jederzeit rechtsgiltig nachgewiesen werden könne, sind die Grenzen der Staatsforste gegenüber dem fremden Besitze deutlich und unwandelbar zu bezeichnen und durch Verfassung von Grenzurkunden sicher zu stellen.

Wo diesen Forderungen bereits Genüge geleistet ist, haben die dazu berufenen Verwaltungs- und Inspectionsorgane sorgfältig darüber zu wachen, dass keine einseitige Aenderung in der Begrenzung Platz greife und dass bei eintretenden rechtlichen Grenzveränderungen die Grenzurkunden — beziehungsweise Grenztabellen — gehörig und in legaler Weise richtig gestellt werden.

Wo dagegen die Sicherstellung der Eigenthumsgrenzen der Staatsforste mangelhaft erscheint, oder wo völlig neue Begrenzungen in Folge von Waldtheilungen, Abtretungen, Grundtausch u. s. w. statthaben sollen, ist einer solchen Mangelhaftigkeit so bald als möglich abzuhelfen und sind die neuen Begrenzungen ohne Verzug auszuführen.

(§. 845 des allgemeinen bürgerlichen Gesetzbuches. Kaiserliches Patent über die Regulirung und Ablösung der Waldservituten vom 5. Juli 1853.)

Figure 3.5.: Excerpt from FV document (Page 9).

Trigger (*F1*)	23.31
Causal Degrees (κ)	65.88
Causal Types (κ)	57.09
Arguments Strict (*F1*)	31.23
Arguments Relaxed (*F1*)	61.71

Table 3.2.: Inter-Annotator Agreement results for the SF document. For triggers, the F1 score is calculated on a token-level. The scores for causal degree, type and arguments are based only on relations where at least 50% of the trigger tokens are overlapping. For the arguments score, the trigger tokens are ignored. The agreement for triggers and strict arguments is low. Degrees, types and relaxed arguments achieve a moderate to substantial agreement.

Preparation

The first step was to find historic documents and an annotation scheme. Requirements for the documents were that they should represent the historical language of the 19th century, frequently include causal relations, and have a reasonably good OCR quality. To find an appropriate annotation scheme, we performed a literary survey on causal relation extraction and decided to focus on the schemes introduced by Rehbein and Ruppenhofer [1] and Dunietz et al. [8]. We then annotated a small part of the SF document. After discussing the results thoroughly, we formalized the initial guidelines of our scheme, which is for the most part similar to the scheme presented in Section 3.1.1.

Phase I

The first annotation phase consisted of annotating the SF document, which was annotated by two annotators, a historian and an advanced student. The inter-annotator agreement can be seen in Table 3.2. The F1 score for triggers, which was calculated on a token-level, is rather low. For the causal types and degrees on the other hand, the Cohen's Kappa reports a moderate to substantial agreement. The causal arguments except trigger have a low F1 score for the strict regime, however, for the relaxed regime the F1 score almost doubles.

In closer inspection, we identify several reasons that could explain the low trigger F1 score. Firstly, our annotation scheme aims to capture all observable causal relations, which leads to the inclusion of semantically bordering relations such as temporal structures. As the causal interpretation of such relations is often debatable, the annotations often diverged between the annotators. Secondly, our texts are written by government officials in an administrative style, which is defined by long and complicated sentences. Despite our efforts to keep the annotation scheme consistent, for many sentences several viable annotations exist, especially in regard to the trigger. Lastly, the language of the historic documents is harder to interpret due to different meanings of words and spelling variations compared to modern texts we are accustomed to.

Consolidation of annotations

Using the insights from annotating the SF document, we adapted the annotation scheme slightly. We decided to annotate overlapping relations, which are introduced in Dunietz et al. [8], only in obvious cases because they are often difficult to detect and do not provide much additional information. Another modification is for the special case of relations with negating words. For triggers like *not increase*, we decided to annotate the negating words as controller and use the opposite causal degree. In this case, the degree changes from Inhibit to Facilitate.

Phase II

In the final phase, the FV document was annotated by the student annotator. This document includes a more complicated layout, which introduces OCR artifacts, where headers are often moved within the paragraph text. These OCR artifacts have been marked with an additional argument.

For training and evaluation of our method, we use the combined data of SF and FV of the student annotator. The annotations of the historian are excluded, as he only annotated the shorter SF document and we want to keep the dataset consistent.

3.2. Model

This section describes the proposed model architecture, the pre-processing, the training procedure, and baselines for performance comparisons.

3.2.1. Motivation and Assumptions

The proposed task is to find causal relations $R = \{r_1, ..., r_n\}$ in a sequence. A causal relation is defined as $r_i = (l_1, l_2, \ldots, l_l)$, with l_i being a tag from the set of causal arguments, or the O tag for non-causal tokens. l is the number of tokens in the sequence. Additionally, for each r_i a causal type and degree is determined. The difficulty of this task arises due to the possibility that there may be more than one causal relation present, and that the causal arguments might be fully or partially overlapping. Simple sequence labeling, where a tag is assigned to each token, does not suffice to detect multiple relations. A more elaborate approach is needed.

The sequence labeling approach could be extended by performing the tagging step multiple times, each time tagging arguments of one causal relation. However, some problems remain: We do not know how many causal relations are present in a sequence, and we need a mechanism for the model to only focus on a single relation in each tagging pass.

Before trying to solve these problems, we simplify matters by introducing a restriction on causal relations: Each relation must correspond to a unique combination of trigger tokens, which we call trigger group. As an example, the sentence depicted in Examples (30.1) and (30.2) contains two causal relations, where the trigger group of the first relation is made up of the tokens *im* (by) and *Wege* (means), and the second trigger group is made up of *so* (so), *dass* (that) and *verblieben* (remained). For most sentences, this restriction naturally holds, however, there exist relations with the same combination of triggers but different causal arguments. This is the case for the relations in the Examples (34.1) and (34.2), where the trigger groups of both relations are equal and made up of the trigger token *erfolgte* (took place). Therefore we treat both relations as a single combined relation, as seen in Example (34.3). Fortunately, these cases are rare and even after combining relations that have the same trigger group, the general causal information remains.

(34.1) **Die Etatsermittlung**$_{\text{Cause}}$ **erfolgte**$_{\text{Trigger}}$ **in den Kahlschlagbetriebsclassen**$_{\text{Controller}}$ **nach der Maßenfachwerksmethode**$_{\text{Effect}}$, beim Plenterbetriebe nach dem Hundeshagen'schen Nutzungsprocente.
Type: *Motivation*, Degree: *Facilitate*

(34.2) **Die Etatsermittlung**$_{\text{Cause}}$ **erfolgte**$_{\text{Trigger}}$ in den Kahlschlagbetriebsclassen nach der Maßenfachwerksmethode, **beim Plenterbetriebe**$_{\text{Controller}}$ **nach dem Hundeshagen'schen Nutzungsprocente**$_{\text{Effect}}$.
Type: *Motivation*, Degree: *Facilitate*

(34.3) **Die Etatsermittlung**$_{\text{Cause}}$ **erfolgte**$_{\text{Trigger}}$ **in den Kahlschlagbetriebsclassen**$_{\text{Controller}}$ **nach der Maßenfachwerksmethode**$_{\text{Effect}}$, **beim Plenterbetriebe**$_{\text{Controller}}$ **nach dem Hundeshagen'schen Nutzungsprocente**$_{\text{Effect}}$.
Type: *Motivation*, Degree: *Facilitate*
(The budget was calculated in the clearcutting operating classes according to the Massenfachwerk method, and Plenterwald operations according to Hundeshagen's utilization percentage.)

Our approach is now as follows: Instead of directly extracting causal relations, the first step of the model is to detect available trigger tokens, regardless of their relation association. Triggers usually occur in similar forms and patterns in a language, which helps to detect them. Next, we cluster the found trigger tokens into unique groups, with the goal to recreate the original trigger groups of the causal relations. For each predicted trigger group, we perform a tagging pass to detect the remaining causal arguments. The information of the respective trigger group is used as context information during the tagging pass. This enables the model to focus on arguments that are causally connected to the trigger group. Lastly, we view the trigger group as a proxy for the causal relation and use it to predict the causal type and degree.

3.2.2. Architecture

The model is designed to perform a series of five connected tasks, which can be seen in Figure 3.6. The model begins with generating BERT embeddings for each token of the sequence, which are utilized as features in subsequent tasks. Next, trigger tokens are detected that introduce causal relations. Found triggers are combined into trigger groups, where each group corresponds to a causal relation. For each trigger group, the remaining causal arguments in the sequence are detected, as well as the causal type and degree.

The input to the model consists of a sequence of tokens S and, during training, corresponding ground truth labels.

The five performed tasks will now be explained in more detail.

Task 1: BERT Embeddings Generation

The first task is to generate contextualized word embeddings $E = (e_1, e_2, \ldots, e_l)$ for the sequence of tokens $S = (t_1, t_2, \ldots, t_l)$ using a BERT model. The embeddings are used as features in the following tasks.

Task 2: Trigger Detection

Next, all occurring triggers in the sequence are identified, without differentiating between causal relations. We view this as a sequence labeling task, with a tag set containing only the causal argument trigger and O. Similar to the proposed architecture in [2], the BERT embeddings E are processed in a single linear layer followed by a softmax activation function, with output neurons for trigger and O.

We define the detected triggers as a set of word embeddings $T = \{e_i, \ldots, e_j\}$. For example, in Figure 3.6, the triggers are $T = \{e_2, e_3, e_7\}$.

Task 3: Trigger Combination

In this task, the detected triggers are grouped to represent the causal relations in the sequence. First it is determined if a pair of triggers is part of the same relation, and then, using methods from graph theory, the pairwise results are clustered to form trigger groups.

Given the set of triggers T from the previous step, all pairs (e_i, e_j) with $e_i, e_j \in T, i < j$ are classified whether they are part of the same relation. The two embeddings are concatenated and processed by a two-layer neural network. The additional hidden layer with a non-linear activation function helps to combine the information of both embeddings better than a single layer as used in Task 1, however, more layers seem to have little effect on the performance. The network has a single output neuron with a sigmoid activation function to compute the

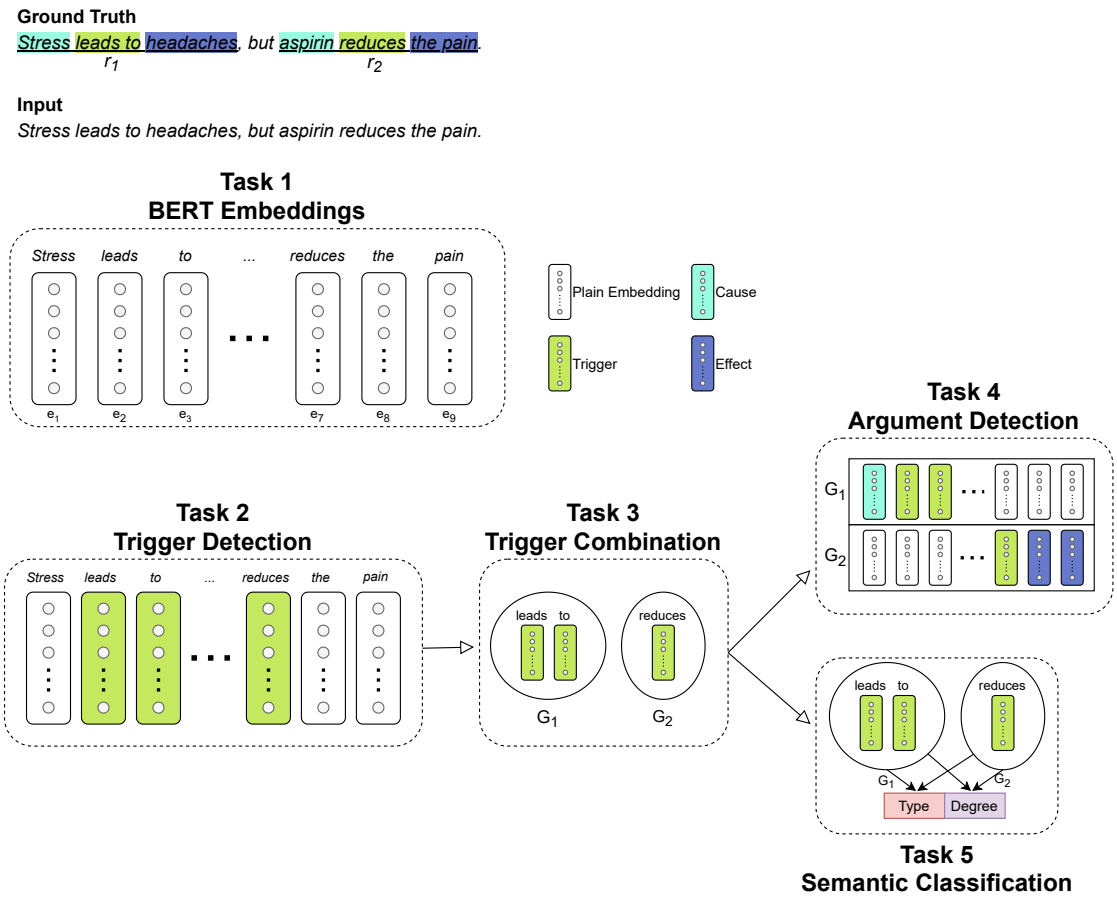


Figure 3.6.: The conceptual structure of the model. The arrows indicate the information flow between the individual tasks. In this example, the ground truth sequence contains two causal relations r_1 and r_2 that are unknown to the model. First, the BERT embeddings for all tokens are created in Task 1, which are used as features in the following tasks. In Task 2, the trigger tokens *leads*, *to* and *reduces* are detected. In Task 3, the triggers are combined into separate groups, with the goal that a group contains triggers that are part of the same causal relation. This results in two trigger groups G_1 and G_2, where the triggers are grouped in the same way as they occur in r_1 and r_2. In Task 4, the remaining causal arguments are detected for each trigger group G_i. A combined embedding of G_i is used as context, so the model focuses on arguments that are causally related to the triggers in G_i. In Task 5, each combined embedding of G_i is used for semantic classification of the causal type and degree.

probability that the pair of triggers is part of the same causal relation. We keep the positional order of the triggers in the pairs and do not evaluate the pair with reversed order (e_j, e_i), as this does not benefit the performance. The number of pairwise comparisons has an order of $\mathcal{O}(n_t^2)$, with n_t being the number of detected triggers in Task 1. This might seem problematic, fortunately, the majority of causal sentences contain fewer than five triggers. Still, to guard against cases where the number of triggers predicted in Task 1 is exceedingly large, we restrict the number of used triggers to 20 using a random selection.

Next, we use the pairwise classification results to identify the trigger groups. As described before, a trigger group is a unique set of triggers that corresponds to exactly one causal relation. Formally, we define a trigger group as the set of embeddings $G = \{e_j, \ldots, e_m\}$, where the embeddings are marked as triggers in the same relation. For example, the trigger groups of Figure 3.6 are $G_1 = \{e_2, e_3\}$ and $G_2 = \{e_7\}$. To find these groups, we interpret the pairwise classification results as an undirected graph where triggers are nodes, and links represent whether two linked triggers are part of the same causal relation. The graph for the example in Figure 3.6 is depicted in Figure 3.7. All triggers within one relation are connected. In this simple case, where each trigger token is part of only one causal relation, the trigger groups are the same as the connected components in the graph.

There exist however some edge cases, due to overlapping relations or incorrect predictions, that must be considered. If triggers are shared between two causal relations, splitting into connected components is not useful anymore. An example sentence for this can be seen in Examples (35.1) and (35.2), where the trigger *durch* (by) is shared. The corresponding graph for this example can be seen in Figure 3.8. We solve this by extracting all maximal cliques in the graph, exploiting the fact that each pair of triggers in a causal relation is connected. This results in the two groups $G_1 = \{e_{10}, e_{19}\}$ and $G_2 = \{e_{10}, e_{28}\}$, both containing the embedding e_{10}.

(35.1) Diesem Grundsatz trägt auch die Betriebseinrichtung Rechnung, indem **sie**$_{\text{Actor}}$ **durch**$_{\text{Trigger}}$ **die Schaffung zahlreicher Anhiebe**$_{\text{Effect}}$ **die Beweglichkeit der Wirtschaft**$_{\text{Cause}}$ **erhöht**$_{\text{Trigger}}$ und die Vertheilung der Jahresschlagfläche auf mehrere Orte ermöglicht.
Type: *Purpose*, Degree: *Facilitate*

(35.2) Diesem Grundsatz trägt auch die Betriebseinrichtung Rechnung, indem **sie**$_{\text{Actor}}$ **durch**$_{\text{Trigger}}$ **die Schaffung zahlreicher Anhiebe**$_{\text{Effect}}$ die Beweglichkeit der Wirtschaft erhöht und **die Vertheilung der Jahresschlagfläche auf**

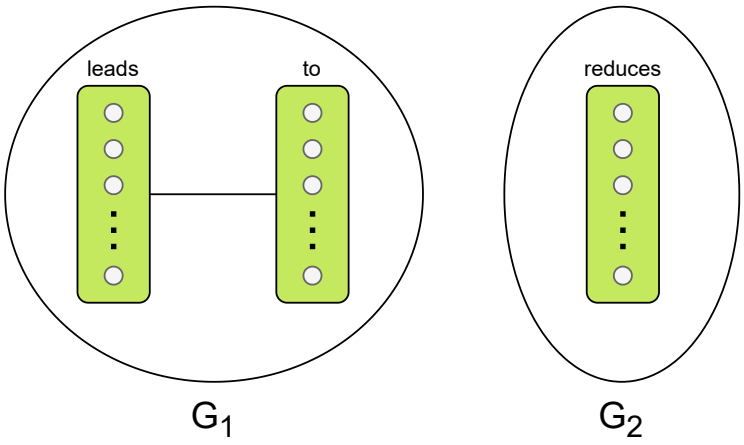


Figure 3.7.: An example of trigger groups in the simple case from Figure 3.6. The trigger groups are disjoint and form two connected components, corresponding to the two causal relations in the sequence.

mehrere Orte$_{\text{Cause}}$ **ermöglicht**$_{\text{Trigger}}$.
Type: *Purpose*, Degree: *Facilitate*
(This basic principle is also taken into account by the company's equipment, as it creates numerous cuts to increase the mobility of the economy and enables the distribution of the annual cut area to several locations.)

Task 4: Argument Detection

The goal of this task is to detect the remaining causal arguments cause, effect, actor, affected, support and controller, to construct the complete causal relations $R = \{r_1, \ldots, r_n\}$. This is similar to ordinary sequence labeling, but with the extension that the assigned tag not only depends on the token itself, but also its relation.

As each predicted group of triggers G_i from the previous step denotes a separate relation, we perform argument detection for all groups. First, a combined embedding e_{G_i} of G_i is created to represent the context of the causal relation. e_{G_i} is calculated in Equation 3.1 as a weighted average of the individual trigger embeddings $e_j \in G_i$.

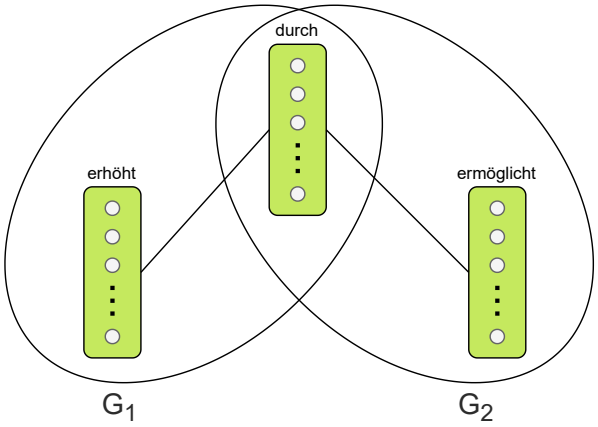


Figure 3.8.: An example of trigger groups in the complex case. The triggers come from the Examples (35.1) and (35.2). Both causal relations in the sequence share the trigger *durch* (by). To resolve this, the trigger groups are given by the maximal cliques of the graph.

$$e_{G_i} = \sum_{e_j \in G_i} e_j w_j \tag{3.1}$$

The weights w_j are generated separately using an attention mechanism based on the approach by Yang et al. [66]. Each trigger embedding e_j is processed with a single layer neural network to generate a query vector q_j. The weights are the result of the dot product between each q_j and a key vector k, followed by a softmax operation, as seen in Equation 3.2. The key vector k is randomly initialized and jointly trained with the network. k should learn to estimate the causal relevance of a trigger embedding e_j given its query vector q_j.

$$w_j = \frac{\exp(q_j^T k)}{\sum_k \exp(q_k^T k)} \tag{3.2}$$

The individual token embeddings of the sequence are concatenated with e_{G_i}, resulting in the extended vector of embeddings $E_{G_i} = (e_1|e_{G_i}, e_2|e_{G_i}, \ldots e_l|e_{G_i})$.

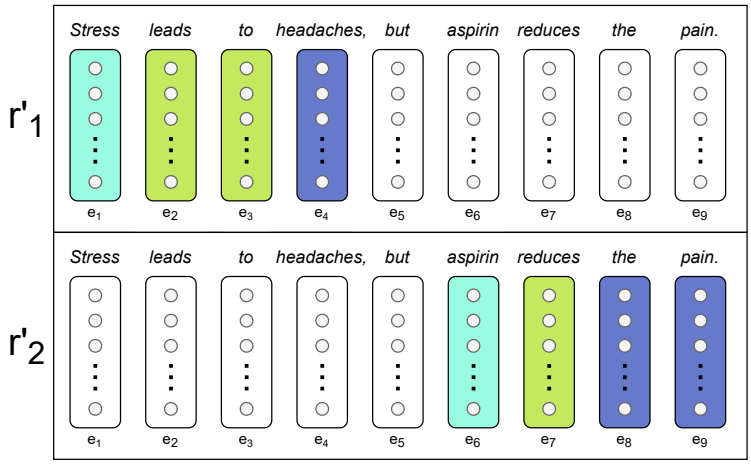


Figure 3.9.: The output for the two causal relations of the example in Figure 3.6. The triggers of each relation are combined with the sequence labeling output of the argument detection task to form the final predicted causal relations r'_1 and r'_2.

These embeddedings are then processed by a neural network with three hidden layers and 13 output neurons with a softmax activation function. Similar to the network in Task 2, the hidden layers with non-linear activation functions seem to support the combination of information from the the concatenated embeddings. The neurons correspond to the BI tags for each argument and the O tag. For each relation, the argument detection output is combined with the tags of the already detected triggers inferred from G_i, resulting in the final predicted causal relation r'_i. The causal relations r'_1 and r'_2 for the example sentence of Figure 3.6 can be seen in Figure 3.9.

Task 5: Semantic Classification

In this task, the model predicts the causal type and degree of a relation. In both cases we utilize the combined trigger group embeddings e_{G_i}, which are processed with a neural network. For the type prediction, we use a single layer network with three output neurons and a softmax activation function, representing the causal types Consequence, Motivation and Purpose. For the degree prediction, we use a similar network, however with only a single neuron and a sigmoid activation function, where 0 corresponds to Facilitate and 1 to Inhibit.

3.2.3. Training and Inference

All tasks of the model are trained in combination, and, except for the BERT embeddings generation, create a cross-entropy loss term for backpropagation. The inputs each task receives during training are independent of the predictions of previous tasks to prevent an accumulation of errors. For example, the argument detection task always receives the correct trigger groups, even though the predictions in the trigger detection or combination tasks might have been wrong.

During inference, however, the output of a task is relayed to subsequent tasks. This means the detected triggers are relayed to the trigger combination task, and the classified trigger groups are relayed to the argument detection and semantic classification tasks.

Training is done in batches. Sequences shorter than the longest sequence of the batch are padded to the length of the longest sequence using the [PAD] token. Padded tokens are excluded from the attention computations in the BERT model, and their created embeddings are not used in further tasks.

For the pre-training on the data of Dunietz et al. [8] and Rehbein and Ruppenhofer [1] we split the data into 90% training and 10% validation data and use the validation data for early stopping to reduce overfitting. The final performance evaluations are exclusively done on the SF and FV datasets. We conduct a 5-fold cross-validation with stratified folds to ensure similar distributions of causal types and degrees.

3.2.4. Pre-processing

To capture the heterogeneous nature of historical documents, we refrain from performing any customized pre-processing based on the structure of the documents. Due to discrepancies between the annotation scheme and our model assumptions, some pre-processing is necessary. Additionally, we try to enhance the input quality by automatically correcting for spelling variations and including coreferences information.

Original	Normalized	Frequency
anderseits	andererseits	15
Procent	Prozent	14
theils	teils	12
Rangsclasse	Rangsklasse	10
Instruction	Instruktion	9
Theile	Teile	9
Directionen	Direktionen	8
Gesammtarea	Gesamtara	6
Theil	Teil	6
Communicationen	Kommunikationen	6

Table 3.3.: The 10 most frequent normalizations in the SF document. Most corrections are reasonable, except *Gesammtarea* to *Gesamtara*, with *Gesamtareal* being a better correction.

To generate training samples of reasonable size, we use spacy[4] to split the text into sentences. We use the BIO scheme to encode the causal arguments and the tagging output of the model. For triggers the I-tag is not used, because the individual tokens are needed for the trigger combination in Task 2.

The original BRAT annotation data is simplified in two ways: First, we remove the distinction between triggers of different priorities, as stated in Section 3.1.1. Secondly, as we defined that a causal relation must correspond to a unique set of triggers in Section 3.2.1, relations with the same set of triggers are combined into a single relation.

Text Normalization

To normalize the text we use the online tool CAB[5] [67]. CAB translates historical spelling variations to their modern equivalents based on rules and statistical properties. For SF, 460 words, and for FV, 2180 words are corrected. In Tables 3.3 and 3.4, the 10 most frequently spelling corrections can be seen for SF and FV respectively.

[4] Version 3.1.4

[5] https://www.deutschestextarchiv.de/demo/cab/file (Accessed on: 2022-04-29)

Original	Normalized	Frequency
Eintheilung	Einteilung	48
mittelst	mittels	36
Abtheilungen	Abteilungen	30
Theil	Teil	26
Eigenthumsgrenzen	Eigentumsgrenzen	24
Coordinaten	Koordinaten	24
fiir	vier	19
Abtheilung	Abteilung	17
Masse	Maße	17
Wirthschaftsstreifen	Wirtschaftsstreifen	15

Table 3.4.: The 10 most frequent normalizations in the FV document. Similarly to SF, most corrections are acceptable. The correction of *fiir* to *vier* is however clearly erroneous, and should have been *für*.

Most corrections of CAB are reasonable, however, there are also some dubious results, e.g. the town name *Bad Aussee* is corrected to *Bad Aussäe*, or the OCR error *fiir* of the actual word *für* (for) results in the correction *vier* (four).

Coreference Information

In our annotations, coreferences occur frequently, but they are marked separately from the causal arguments. Still, coreferences might include valuable information, as they specify causal arguments in more detail. Therefore, we evaluate the effect of extending the annotations of causal arguments to coreferenced entities. For Example (33), this means that we additionally mark the referenced phrase *die Besoldung der Beamten* as an effect and evaluate whether the gained information affects the performance. Since the samples are generated as sentences, coreferences over multiple sentences are not considered.

3.2.5. Baselines

Each task is compared to a simple baseline based on pattern matching and heuristics. The goal of these baselines is to give a rough estimate of the difficulty of the tasks.

For the trigger detection task, the tokens of all unique trigger groups in the training set are matched on the test set, with the restriction that if a trigger group consists of more than one word, all tokens have to occur correctly ordered in the test sample. This is done with the original words and the lemmatized forms.

For the trigger combination baseline, two trigger tokens are predicted as connected if there exists a trigger group in the training set containing both tokens. The tokens are compared in their original and lemmatized form.

The argument detection baseline is of a similar form as the trigger detection baseline: First, we construct a database of all arguments in the training data and try to find occurrences in the test set. To make more precise predictions, we add the restriction that a causal argument must occur in the area around or between triggers. In contrast to the trigger detection baseline, only half of the tokens of a training argument must be matched in the test sample for a prediction to be made. Again we do this in original and lemmatized form.

For the semantic classification task, the trigger groups of the test set are matched to the training trigger groups. If a match occurs, the prediction is the most frequent sense of the training trigger group. If the groups could not be matched, the most frequent class is predicted, which is Consequence for the type and Facilitate for the degree. Again, we use the original and the lemmatized forms for matching.

3.3. Measures

We evaluate our model both on the individual tasks and the final predicted relations.

For the trigger and argument detection tasks we compute the recall, precision and micro F1 score using a strict and relaxed regime. The regimes differ in what constitutes as a correct prediction.

The strict regime only considers predictions with exactly matching boundaries as true positives. Predictions that do not occur in the ground truth are counted as false positives, and ground truth entities that are not predicted are counted as false negatives. For the strict regime we use the seqeval [68] implementation available online[6].

For the relaxed regime we also consider incorrect boundaries of entities. Only a single predicted token has to be correct for the whole prediction of an entity to be counted as correct. In Table 3.5, we can see a sequence of tokens and the ground truth labels, and in Table 3.6 are the evaluation results of the strict and relaxed regime. For Prediction 1, where an additional entity is predicted, and Prediction 2, where nothing is predicted, both regimes give the same results. For Prediction 3, however, the strict regime counts the prediction as incorrect, whereas the relaxed regime counts it as correct. To compute the relaxed metric, we conduct an iterative greedy search to match partially overlapping entities. First, the Jaccard index for all pairs of predicted and ground truth entities of the same type is calculated on a token level, which yields a measure of their overlap. Then we iteratively match the pair with the highest Jaccard index and remove both entities from further matching. This is continued until all pairs are either matched or cannot be matched. A matched pair counts as a true positive, an unmatched ground truth entity as a false negative and an unmatched predicted entitiy as a false positive. Using these values we can calculate the micro F1-score.

The relaxed regime gives better insights into how well the model can capture arguments in general, as it shows whether the model detects the general location of the arguments.

For the trigger combination and semantic classification tasks, we evaluate the accuracy, macro F1 score and the Matthews correlation coefficient (MCC).

To evaluate the final prediction we inspect how well the predicted relations correspond to the ground truth relations. We call this task relation matching. Similar to before, we use a strict approach, where a match only occurs if at least 90% of the labels are correct, and a relaxed approach, where only a single label has to be correct. Matched relations count as true positives, unmatched predicted relations as false positives and unmatched ground truth relations as false negatives. From these values we calculate the overall precision, recall and micro F1-score.

[6] https://github.com/chakki-works/seqeval (Accessed on: 2022-05-05)

Tokens	Ground Truth	Pred 1	Pred 2	Pred 3
The	B-ORG	B-ORG	O	O
company	I-ORG	I-ORG	O	B-ORG
Apple	I-ORG	I-ORG	O	I-ORG
is	O	O	O	O
not	O	O	O	O
an	O	O	O	O
apple	O	B-ORG	O	O

Table 3.5.: A simple sequence labeling example in BIO format with ground truth and three predicted tag sequences.

	strict			relaxed		
	Precision	Recall	F1	Precision	Recall	F1
Pred 1	0.5	1.0	0.67	0.5	1.0	0.67
Pred 2	0.0	0.0	0.00	0.0	0.0	0.00
Pred 3	0.0	0.0	0.00	1.0	1.0	1.00

Table 3.6.: The results of the strict and relaxed regimes for the three predictions in Table 3.5. The relaxed regime for Prediction 3 differs because it does not require the correct boundaries.

3.4. Model Variations

Pre-training is one of the main reasons for the recent success of deep language models like BERT. Many pre-trained models exist, and it is not trivial to find the best model for a specific task. We evaluate a selection of BERT models that have been pre-trained on different datasets. Additionally, we investigate the effect of transfer learning on the performance.

3.4.1. BERT Models

In this experiment, we compare different pre-trained BERT models for generating the embeddings used throughout the model. As our corpus is German, there exists only a limited number of pre-trained models compared to English. We choose to evaluate the following models, accessed from the HuggingFace model portal[7].

- **BERT-German**[8] The German BERT model was trained by the Bavarian State Library on Wikipedia documents, books and news articles with a cased vocabulary.

- **BERT-Europeana**[9] The BERT-Europeana model, also created by the Bavarian State Library, was trained on the Europeana newspapers corpus with historical documents published between the 17th and 20th centuries. The model outperformed models trained on contemporary texts on tasks involving historical datasets with OCR errors [69]. The model has a cased vocabulary.

- **BERT-Multilingual**[10] The BERT-Multilingual model is trained on documents of 104 different languages with a cased vocabulary.

[7] https://huggingface.co/ (Accessed on: 2021-12-15)
[8] https://huggingface.co/dbmdz/bert-base-german-cased (Accessed on: 2022-04-18)
[9] https://huggingface.co/dbmdz/bert-base-german-europeana-cased (Accessed on: 2022-04-18)
[10] https://huggingface.co/bert-base-multilingual-cased (Accessed on: 2022-04-18)

3.4.2. Transfer Learning

Supervised learning tasks have the disadvantage that they often require large amounts of training data for a good performance. As the available annotated data in the SF and FV documents is rather small, we perform transfer learning. Transfer learning describes the technique of training a model on a related task before using the aquired knowledge on the task of interest, similar to how BERT is pre-trained on MLM and then fine-tuned on a specific task.

We evaluate the effect of pre-training our model on the data of Rehbein and Ruppenhofer [1] and the available BECAUSE corpus [8] before fine-tuning on SF and FV. Despite the differences between these datasets and our annotated documents, we hypothesize that a model can benefit from the additional information. For the experiment, the models introduced in Section 3.4.1 are pre-trained for different configurations, either trained only on one of the two datasets, or both combined.

3.5. Causal Attention Heads

In this section, we describe an experiment to investigate whether attention heads in BERT correspond to causal relations. We hypothesize that after unsupervised pre-training, attention heads emerge that focus on relationships that are related to causality. Our BERT model consists of 12 layers with 12 heads each, resulting in 144 attention heads in total for our experiment.

Our experiment is similar to the approach of Clark et al. [70]. First, the sequence of tokens is processed by a BERT model to generate the attention weights of the heads. The experiment is done on a word-level, therefore the attention weights of subwords are combined into a single word. Given a word that is split into subwords, the attention weights of attended subwords are summed up, and then the mean of all attending subwords is computed. This has the advantage that the attention weights of an attending token still sum up to 1. An example of this can be seen in Figure 3.10. In the summation step, the two green and blue weights to the subwords *Ra* and *##diation* are respectively added up, resulting in the complete attended token *Radiation*. In the following averaging step, the green and blue weights from two subwords are averaged to the complete attending token *Radiation*.

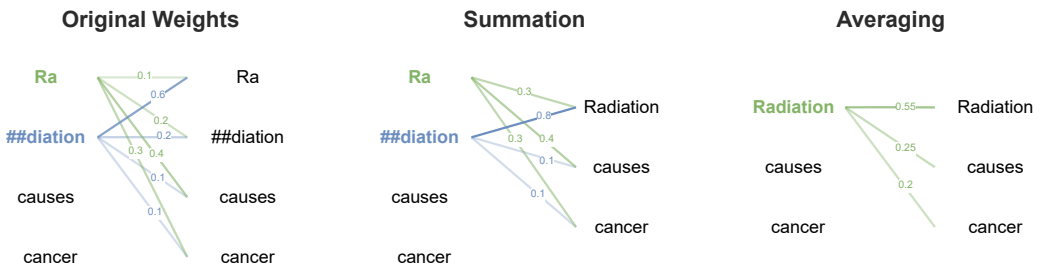


Figure 3.10.: Transformation of the raw attention weights to word-level weights for the example sentence *Radiation causes cancer*. In each step, the tokens on the left are the attending tokens, and on the right are the attended tokens. The word *Radiation* is split into subwords, and for more clarity, only the weights of the subwords *Ra* and *##diation* are included. In the summation step, the weights of all attended subwords are summed up, and in the averaging step, the mean of the subword weights is computed.

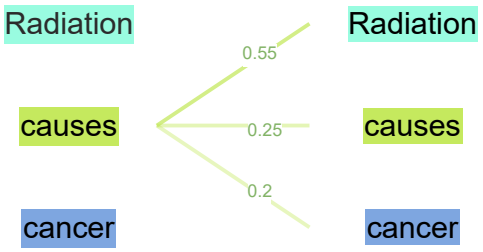


Figure 3.11.: Example attention weights for the trigger *causes* in a sentence with cause *Radiation* and effect *cancer*. The highest attention weight of the trigger is to the cause argument. For cause, this relation would be a hit, while for effect, it would be a miss.

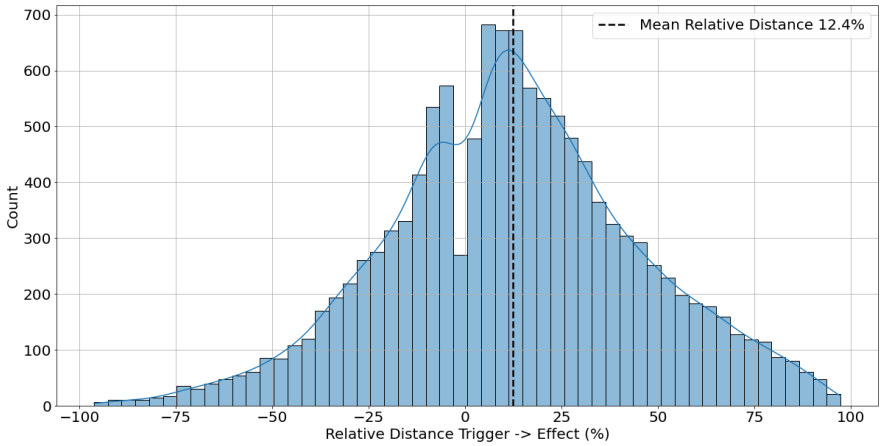


Figure 3.12.: Histogram of the relative distances between trigger and effect in the combined data of SF and FV. The relative distance is given as the percentage of tokens of a sentence between trigger and effect. The dashed black line shows the mean relative distance.

To find heads that focus on causal arguments, we evaluate for each head whether one of the trigger tokens assigns the highest attention weight to a token of a causal argument. An example of this is given in Figure 3.11, where the attention weight between trigger *causes* and the cause *Radiation* is the highest. This step is repeated for all relations in the dataset for each causal argument except trigger. For the argument cause, the example in Figure 3.11 would be a hit, but for effect it would be a miss. Finally, an accuracy score for each head and causal argument is computed. We are interested in the head with the highest accuracy score for an argument. It must be noted that we only evaluate attention weights to tokens corresponding to a word and disregard punctuation or special tokens.

To create a baseline, we use the mean relative distance between the trigger and the causal argument. The relative distance is the number of tokens between a trigger and causal argument divided by the total sentence length, which can be interpreted as a percentage. To illustrate this, Figure 3.12 shows the histogram of relative distances between trigger and effect token for all relations in the combined dataset of SF and FV. An effect token occurs on average after a trigger token, indicated by the dashed black line, with a distance of 12.4% of the total sentence length. In this case, the baseline prediction is the token that has a distance of 12.4% of the

	German (SF & FV)	English (BECAUSE)
Cause	671	488
Effect	684	538
Actor	68	-
Affected	112	-
Support	13	-
Controller	82	-

Table 3.7.: The number of relations that include one of the causal arguments for the German and English data. In the English data, consisting of the BECAUSE [8] corpus, only the causal arguments cause and effect are used.

sequence length to the trigger.

The mean relative distance differs from the approach of Clark et al. [70], who used the absolute number of tokens as distance. This could lead to problems, as our sentences heavily vary in size, and causal arguments are often far apart. For example, the mean absolute number of tokens might be larger than the sentence length for shorter sentences.

We perform this experiment in German and English separately, using two language-specific BERT models[11,12]. The German evaluation data consists of the combination of SF and FV, and for English we use the BECAUSE corpus of Dunietz et al. [8]. The number of relations per causal argument for each dataset can be seen in Table 3.7.

Additionally, we conduct the same experiment after fine-tuning on the causal relationship extraction task to find out whether heads learn to attend to causal relations better after training on a related task. To make use of the full datasets, we perform 5-fold cross-validation.

[11] https://huggingface.co/dbmdz/bert-base-german-cased (Accessed on: 2022-04-18)

[12] https://huggingface.co/bert-base-cased (Accessed on: 2022-04-18)

4. Evaluation

In this chapter, we present the results of our experiments on the model and the BERT attention heads. Section 4.1 contains the results for different backbone BERT models in combination with transfer learning, as well as the influence of pre-processing the input. Section 4.2 shows the results of the causal attention head experiment.

In the following tables, the bolded entry denotes the highest score for a measure, and the number in parentheses is the standard deviation over the 5 cross-validation folds.

4.1. Model Performance

This section reports and discusses the cross-validation results for different BERT models and pre-processing configurations, in combination with transfer learning. The evaluation dataset is a combination of both historic documents SF and FV. The reported measures are introduced in Section 3.3.

4.1.1. Results

We now present the results for Tasks 2-5 of the model, plus the additional task of matching the final predictions to the ground truth relations. For these experiments, we use the unmodified text and labels.

Transfer Learning	BERT Model	Trigger		
		Precision	Recall	F1
—	**German**	**50.1 (3)**	56.7 (2)	**53.1 (2)**
	Europeana	49.3 (2)	53.2 (2)	51.1 (2)
	Multilingual	45.1 (4)	51.9 (2)	48.2 (2)
BECAUSE	**German**	48.5 (4)	54.7 (1)	51.4 (3)
	Europeana	49 (3)	53.1 (4)	50.8 (2)
	Multilingual	45.5 (4)	50.9 (2)	48 (3)
Rehbein and Ruppenhofer	**German**	42.6 (2)	56.9 (2)	48.6 (1)
	Europeana	44.5 (1)	**57.3 (2)**	50 (1)
	Multilingual	41.7 (3)	52.2 (4)	46.3 (3)
Rehbein and Ruppenhofer, BECAUSE	**German**	45.1 (3)	54.3 (2)	49.2 (2)
	Europeana	47.1 (3)	54.3 (3)	50.4 (3)
	Multilingual	43.2 (3)	52 (3)	47.1 (3)
Baseline		26.9 (3)	45.7 (4)	33.8 (3)

Table 4.1.: F1 scores for trigger detection. The evaluated dataset is the combination of the SF and FV documents. We use 5-fold cross-valdation, with the number in parentheses denoting the standard deviation over the 5 folds. The scores are derived on a token-level, therefore we do not distinguish between the strict and relaxed regime, as each span is of length one.

Transfer Learning	BERT Model	MCC	Accuracy	Connected Trigger F1	Separate Trigger F1
—	**Ger.**	75.1 (5)	87.4 (2)	86.3 (2)	88.3 (3)
	Europ.	74.3 (5)	87 (3)	85.8 (3)	87.9 (3)
	Multi.	75 (7)	87.7 (3)	85.9 (4)	88.9 (3)
BEC	**Ger.**	77.6 (4)	88.8 (2)	87.4 (2)	**89.9 (2)**
	Europ.	76.1 (4)	87.7 (2)	86.8 (2)	88.5 (2)
	Multi.	77.1 (4)	88.7 (2)	87.1 (2)	89.8 (2)
R&R	**Ger.**	73.4 (3)	86.5 (2)	85.3 (2)	87.5 (1)
	Europ.	**78.1 (2)**	88.8 (1)	**87.9 (2)**	89.6 (1)
	Multi.	73.6 (6)	86.7 (3)	85.5 (3)	87.7 (3)
R&R, BEC	**Ger.**	77.9 (4)	**88.9 (2)**	87.7 (2)	89.8 (2)
	Europ.	76.2 (5)	87.7 (3)	86.8 (3)	88.4 (3)
	Multi.	74.7 (6)	87 (3)	86.1 (3)	87.8 (3)
Baseline		18.2 (5)	61 (2)	37 (4)	71.7 (3)

Table 4.2.: Results for the trigger combination task. All pairs of trigger tokens are evaluated whether they are within the same causal relation or not.

Trigger Detection

In Table 4.1 we present the F1 score for trigger detection. The model with the best F1 and precision score is BERT-German without transfer learning, the best recall score is achieved by BERT-Europeana with transfer learning on the Rehbein and Ruppenhofer [1] data. Additional transfer learning data slightly increases the recall but decreases the precision and F1 score. In all transfer learning configurations, the BERT-Europeana and the BERT-German scores are close, while the BERT-Multilingual scores are about 3-4 percentage points lower. All models outperform the baseline.

Trigger Combination

Table 4.2 shows the results for trigger combination. The transfer learning configuration and BERT models are abbreviated, but the same as in Table 4.1. For

Transfer Learning	BERT Model	Cause	Effect	Actor	Aff.	Sup.	Contr.
—	**Ger.**	25.6 (3)	24.7 (3)	32.7 (12)	20 (8)	0 (0)	8 (5)
	Europ.	23.5 (5)	23.5 (5)	**34.3 (16)**	16.1 (7)	0 (0)	3.5 (5)
	Multi.	23.9 (3)	22.5 (4)	27.8 (14)	9.9 (6)	0 (0)	5.6 (8)
BEC	**Ger.**	27.3 (4)	25.8 (5)	29.6 (10)	20.2 (8)	0 (0)	**9 (3)**
	Europ.	22 (4)	22.9 (2)	29.5 (9)	17.8 (5)	0 (0)	5.4 (5)
	Multi.	26.8 (5)	25.2 (4)	31.4 (16)	13.9 (4)	0 (0)	8.3 (5)
R&R	**Ger.**	26.4 (3)	29.8 (3)	31.3 (8)	20.6 (3)	0 (0)	8.7 (4)
	Europ.	27.9 (4)	29.4 (3)	30.6 (12)	**25.2 (6)**	0 (0)	8.8 (6)
	Multi.	26.4 (4)	26.8 (4)	27.5 (8)	19.4 (8)	0 (0)	7.9 (5)
R&R, BEC	**Ger.**	28.7 (2)	**31 (5)**	33 (13)	20.5 (6)	0 (0)	5.8 (5)
	Europ.	**30.3 (5)**	27.2 (4)	33.4 (11)	18.7 (5)	0 (0)	8.8 (9)
	Multi.	27.4 (4)	27.7 (4)	34 (9)	21.3 (10)	0 (0)	6.5 (7)
Baseline		0.7 (0)	0.7 (0)	5 (10)	0 (0)	0 (0)	0 (0)

Table 4.3.: F1 scores for each causal argument in the task of argument detection using the strict regime. All words of a argument must be predicted correctly for a prediction to be a true positive. Ground truth arguments without a predicted counterpart are treated as false negatives, and predicted arguments without corresponding ground truth are counted as false positives.

this task, the BERT-Europeana model with transfer learning on the Rehbein and Ruppenhofer [1] data has the highest MCC and F1 score for connected triggers. The highest accuracy is given by the BERT-German model with transfer learning on both datasets, and the highest F1 for separate triggers by the BERT-German model with transfer learning on BECAUSE [8]. For all models, the MCC indicates a strong correlation between the predictions and the ground truth. The BERT models outperform the baseline, which is slightly better than a random prediction.

Argument Detection

In the argument detection task, we distinguish between strict and relaxed detection. For strict, all predicted tokens in the argument must match exactly to the ground truth, whereas for relaxed only a single token must match. Table 4.3 presents

Transfer Learning	BERT Model	Cause	Effect	Actor	Aff.	Sup.	Contr.
—	**Ger.**	49.6 (2)	52.7 (3)	44.5 (17)	23.5 (8)	0 (0)	24.8 (10)
	Europ.	47.6 (3)	51.8 (4)	**47.7 (15)**	22.4 (8)	0 (0)	15.9 (6)
	Multi.	46.3 (2)	52.5 (3)	29.7 (16)	16.6 (9)	0 (0)	8.7 (6)
BEC	**Ger.**	51.5 (4)	54.8 (2)	42.6 (13)	27.2 (13)	0 (0)	**29.5 (6)**
	Europ.	47.1 (2)	50.9 (5)	39.3 (13)	23.3 (8)	0 (0)	18.7 (7)
	Multi.	50.4 (3)	54.5 (2)	38 (15)	18.1 (7)	0 (0)	13 (7)
R&R	**Ger.**	51.5 (4)	58.2 (4)	36.7 (13)	27 (7)	0 (0)	19.9 (2)
	Europ.	**53.2 (2)**	56.9 (3)	39.1 (15)	**28.8 (7)**	0 (0)	25.2 (10)
	Multi.	52 (4)	55 (5)	34.7 (9)	26.2 (8)	0 (0)	22.4 (6)
R&R, BEC	**Ger.**	52.7 (2)	**59.4 (3)**	36 (13)	28.7 (8)	0 (0)	18.7 (3)
	Europ.	**53.2 (2)**	56.1 (3)	44.2 (17)	26.5 (8)	0 (0)	18.7 (7)
	Multi.	52.3 (4)	56.5 (7)	42.1 (16)	25.1 (8)	0 (0)	15.1 (9)
Baseline		19.3 (3)	24.1 (2)	19.8 (9)	1.9 (4)	0 (0)	10.3 (8)

Table 4.4.: F1 scores for each argument in the task of argument detection using the relaxed regime. The relaxed regime differs from the strict regime as now only a single word has to be predicted correctly for a prediction to count as a true positive.

the F1 scores of the arguments for the strict setting. No model performs best on more than one causal argument. For cause and effect, which are the two most frequent arguments, the highest F1 scores are achieved with transfer learning on both datasets. BERT-Europeana has the highest score for cause, and BERT-German for effect. The best scores for the other arguments are scattered for different transfer learning configurations, with a BERT-Europeana model performing best for actor and affected arguments, and BERT-German for controller arguments. The F1 score of support is zero for all models. The baseline for this task is very low, with some correct predictions for cause, effect and actor.

The F1 scores for the relaxed setting are presented in Table 4.4. The best scores are achieved by the same models as in the strict regime. Again, the support scores are zero for all models. The baseline scores are higher than in the strict setting, but most models are clearly above it.

To inspect the combined performance over all arguments, Figure 4.1 shows the macro F1 scores of the causal arguments for the BERT models in respect to the transfer learning configurations. No model is above the others for all configurations, for the relaxed regime the BERT-German model achieves the highest score when transfer learning on BECAUSE [8], and for the strict regime the BERT-Europeana model with transfer learning on the Rehbein and Ruppenhofer [1] is the best.

Causal Type Prediction

Table 4.5 shows the results for the prediction of the causal type. The BERT-German model with transfer learning on both datasets performs best overall, with the highest MCC, accuracy and Consequence F1 scores. The best F1 scores for Motivation and Purpose are achieved by the BERT-German and BERT-Europeana models with transfer learning on the Rehbein and Ruppenhofer [1] data. The baseline MCC for this task is low and only slightly better than random. Figure 4.2 shows the MCC for the different models with respect to the transfer learning data. BERT-German and BERT-Europeana perform similarly, but the BERT-Multilingual model has the lowest MCC in all configurations. The predictions of the best model compared to the ground truth labels can be seen in the confusion matrix in Table 4.6. Motivation is frequently confused with the other types.

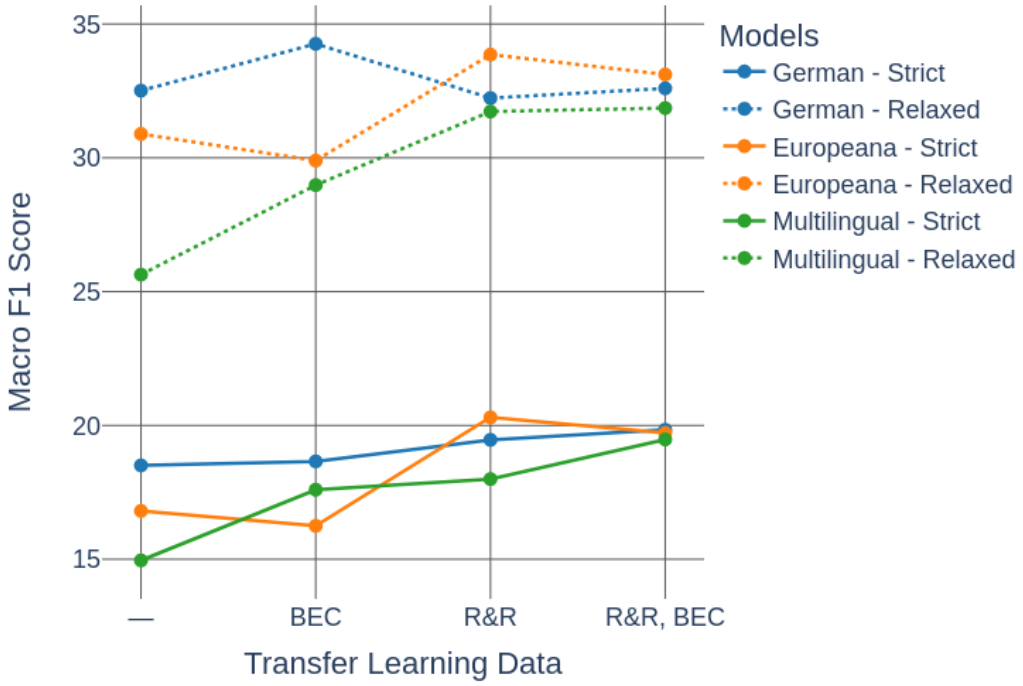


Figure 4.1.: Comparison of the macro F1 of the causal arguments for different models in respect to the transfer learning configuration. The solid lines show the scores of the strict regime, and the dotted lines of the relaxed regime.

Transfer Learning	BERT Model	MCC	Accuracy	Consequence F1	Motivation F1	Purpose F1
—	**Ger.**	58.1 (9)	73.8 (5)	82.3 (6)	55.2 (8)	74.2 (6)
	Europ.	56.2 (6)	72.8 (4)	81.6 (4)	49.8 (5)	76 (4)
	Multi.	55.3 (8)	72.3 (5)	81 (4)	52.3 (6)	73.8 (6)
BEC	**Ger.**	57.1 (4)	73.6 (2)	82 (3)	52.8 (3)	74.7 (6)
	Europ.	56.9 (5)	73.5 (3)	81.9 (3)	50.7 (5)	75.7 (6)
	Multi.	50.8 (4)	69.7 (3)	78.4 (3)	46.3 (2)	73.8 (3)
R&R	**Ger.**	59.6 (6)	74.7 (4)	82.3 (4)	**57.3 (6)**	76.3 (5)
	Europ.	59.2 (5)	74.6 (3)	82.2 (3)	54.4 (6)	**78.1 (5)**
	Multi.	51 (3)	69.1 (2)	78 (3)	48.3 (5)	72.9 (3)
R&R, BEC	**Ger.**	**59.9 (7)**	**75.3 (4)**	**82.9 (4)**	56.2 (6)	77.5 (6)
	Europ.	59.3 (2)	74.8 (1)	82.3 (2)	57 (5)	76.2 (5)
	Multi.	50.7 (4)	68.1 (3)	76.5 (5)	51.5 (2)	71 (3)
Baseline		17 (4)	53.2 (2)	66.8 (1)	16.4 (7)	39.5 (5)

Table 4.5.: Results for the prediction of the causal type.

		Ground Truth		
		Purpose	**Motivation**	**Consequence**
Predicted	**Purpose**	132	21	9
	Motivation	28	94	48
	Consequence	17	50	299

Table 4.6.: Confusion matrix of causal type prediction for BERT-German using all data for transfer learning.

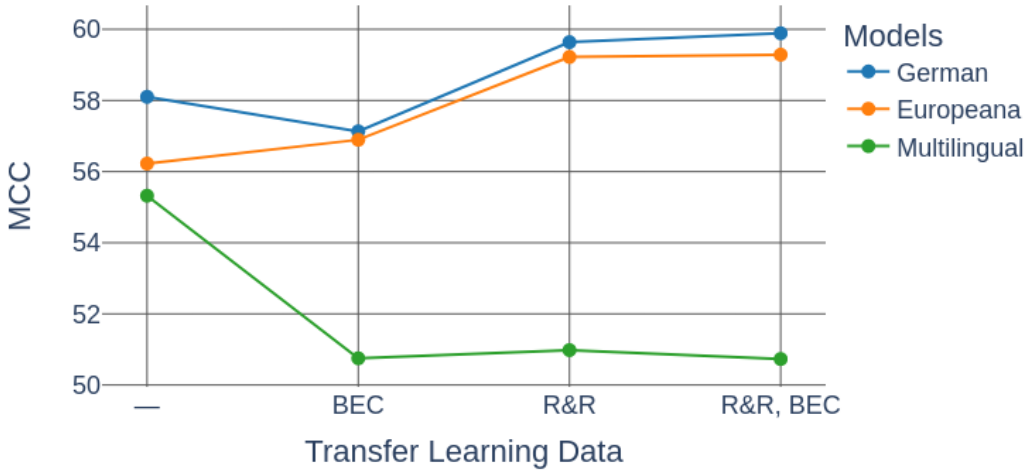


Figure 4.2.: Comparison of the MCC for causal type prediction for different models in respect to the transfer learning configuration. For this graph we use the MCC, in contrast to the macro-F1 score presented in Figure 4.1. The reason we do not use the MCC for argument detection, as well as relation matching, is that for these two tasks no true negative samples exist (see Section 3.3). As this creates unusual MCC scores, we choose to use the F1 score for these tasks.

Transfer Learning	BERT Model	MCC	Accuracy	Facilitate F1	Inhibit F1
—	**Ger.**	52.7 (14)	95.8 (1)	97.8 (1)	50.4 (12)
	Europ.	42.9 (16)	95.5 (1)	97.7 (0)	37.7 (14)
	Multi.	42.1 (17)	94.9 (2)	97.4 (1)	42.7 (16)
BEC	**Ger.**	55.4 (15)	95.8 (2)	97.8 (1)	54.8 (14)
	Europ.	38.7 (12)	95.3 (1)	97.5 (0)	32.5 (13)
	Multi.	52.4 (12)	95.8 (1)	97.8 (1)	51.9 (13)
R&R	**Ger.**	64.5 (18)	96.4 (2)	98.1 (1)	65.7 (17)
	Europ.	58.9 (23)	96.5 (2)	98.2 (1)	57.6 (22)
	Multi.	46.5 (20)	95.1 (2)	97.4 (1)	45.4 (18)
R&R, BEC	**Ger.**	**69.1 (18)**	**97 (2)**	**98.4 (1)**	**69.8 (17)**
	Europ.	62.8 (33)	**97 (2)**	**98.4 (1)**	62.7 (32)
	Multi.	52.5 (11)	95.7 (1)	97.7 (0)	52.2 (11)
Baseline		0.9 (9)	92.3 (2)	96 (1)	4 (8)

Table 4.7.: Results for the prediction of the causal degree.

Causal Degree Prediction

For the causal degree prediction, the results are presented in Table 4.7. The BERT-German model with transfer learning on both datasets achieves the best results in all metrics, with the BERT-Europeana having the same accuracy and F1 score for Facilitate. The MCC and Inhibit F1 scores vary strongly over the cross-validation folds. In Figure 4.3, the MCC for the different models with respect to the transfer learning data is presented. The results have a high variation, however, the BERT-German model performs best in all configurations. Table 4.8 presents the confusion matrix for the predictions of the best model. Facilitate occurs much more frequently than Inhibit, and Inhibit is relatively often confused with Facilitate.

Relation Matching

Here we present the results for relation matching, where the final predicted causal relations are compared to the ground truth relations on a word level. The differ-

		Ground Truth	
		Facilitate	**Inhibit**
Predicted	**Facilitate**	653	15
	Inhibit	6	24

Table 4.8.: Confusion matrix of causal degree prediction for BERT-German using all data for transfer learning.

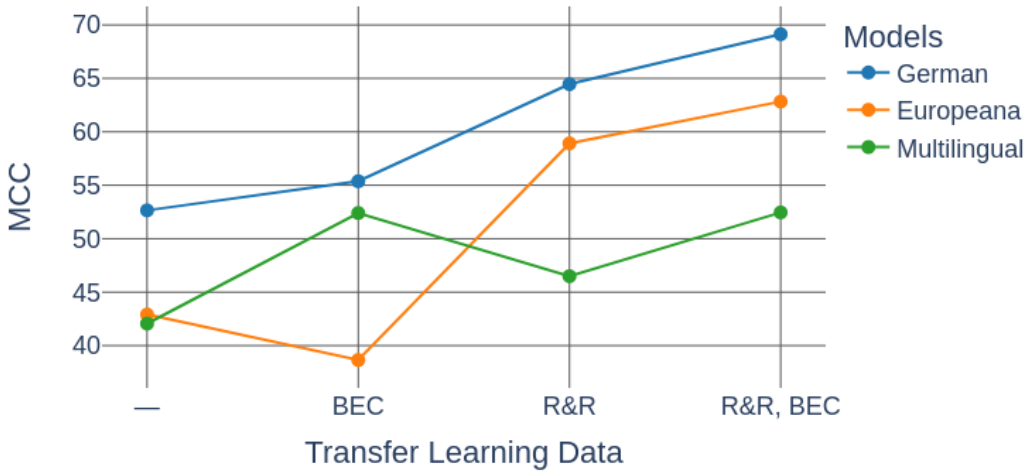
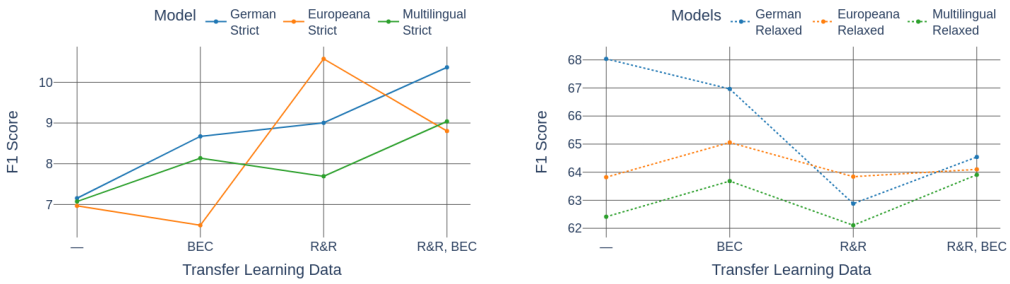


Figure 4.3.: Comparison of the MCC for causal degree prediction for different models in respect to the transfer learning configuration.

Transfer Learning	BERT Model	Strict			Relaxed		
		Precision	Recall	F1	Precision	Recall	F1
—	**Ger.**	6.4 (1)	8.2 (1)	7.2 (1)	**60.5 (4)**	78 (3)	**68 (2)**
	Europ.	6.3 (1)	7.7 (1)	7 (1)	58.2 (2)	70.8 (2)	63.8 (1)
	Multi.	6.2 (2)	8.3 (2)	7.1 (2)	54.3 (4)	73.9 (4)	62.4 (3)
BEC	**Ger.**	7.7 (1)	10 (2)	8.7 (2)	59.1 (3)	77.6 (3)	67 (2)
	Europ.	5.9 (1)	7.2 (1)	6.5 (1)	59.6 (2)	72 (5)	65.1 (1)
	Multi.	7.2 (1)	9.5 (1)	8.1 (1)	56.1 (4)	74 (3)	63.7 (2)
R&R	**Ger.**	7.4 (1)	11.5 (2)	9 (1)	51.9 (4)	80.1 (3)	62.9 (3)
	Europ.	**8.9 (3)**	**13 (4)**	**10.6 (4)**	53.6 (1)	79 (4)	63.8 (2)
	Multi.	6.4 (1)	9.8 (2)	7.7 (2)	51.5 (3)	78.5 (3)	62.1 (3)
R&R, BEC	**Ger.**	8.8 (2)	12.7 (3)	10.4 (2)	54.3 (1)	79.7 (3)	64.5 (0)
	Europ.	7.7 (1)	10.3 (2)	8.8 (2)	56.1 (3)	74.9 (4)	64.1 (4)
	Multi.	7.7 (1)	10.9 (2)	9 (1)	54.7 (3)	77.2 (5)	63.9 (3)
Baseline		0 (0)	0 (0)	0 (0)	32.7 (3)	**83.8 (4)**	46.9 (2)

Table 4.9.: Results for relation matching for the strict and relaxed regime. In this task, the predicted labels are compared to the ground truth labels on a token level, ignoring the B- and I-tags.



(a) Strict (b) Relaxed

Figure 4.4.: Comparison of the F1 score for relation matching for different models in respect to the transfer learning configuration. Figure (a) shows the results for the strict regime, (b) for the relaxed.

entiation between B- and I-labels is ignored. For the strict regime, a match occurs if at least 90% of predicted labels are correct, while for the relaxed regime only a single label has to be correct. In the previous tasks, the input was derived from the ground truth to reduce the influence of accumulated errors. For example, the trigger combination task is evaluated on pairs of ground truth triggers. For this task, the model does not utilize any intermediary results of the ground truth. The output of a task is used as input for the subsequent tasks, which means that for example the trigger combination task receives the predicted triggers of the trigger detection task.

The strict and relaxed scores for recall, precision and F1 are presented in Table 4.9. For the strict regime, the best results are achieved by the BERT-Europeana model with transfer learning from the data of Rehbein and Ruppenhofer [1]. For the relaxed regime, the results differ. The best precision and F1 score is given by the BERT-German model without transfer learning. The baseline for the strict regime is zero for all measures, but for the relaxed regime, the scores are similar to the performances of the BERT-models, with the recall score being the highest overall.

Figure 4.4 shows the F1 scores for the strict and relaxed regime for the different BERT models and transfer learning configurations. For the strict regime, the BERT-Multilingual and the BERT-German models follow a similar trend, with the BERT-German model having higher scores. The BERT-Europeana model has the lowest score for all configurations except when using the data of Rehbein and

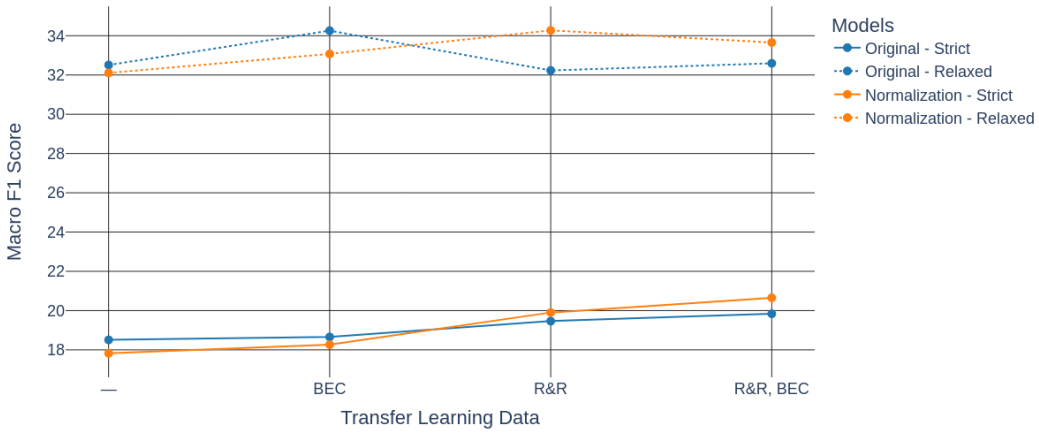
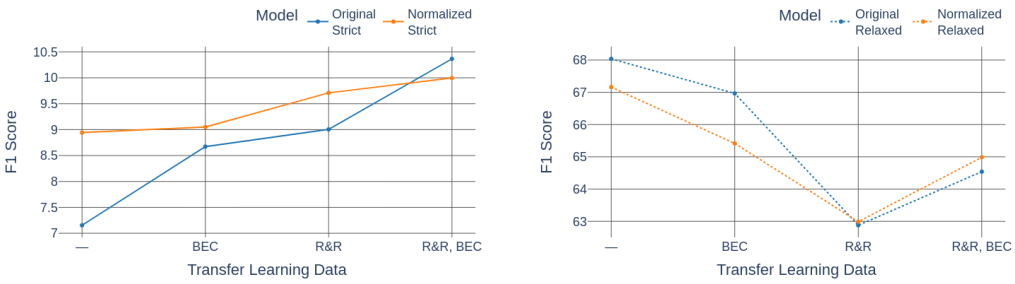


Figure 4.5.: Comparison of the macro F1 scores for strict and relaxed argument detection between the original and normalized text for BERT-German.

Ruppenhofer [1] for transfer learning, where it achieves the overall highest score. For the relaxed regime, the BERT-German model achieves the highest F1 score for most configurations.

4.1.2. Pre-Processing

In this section, we present the results when applying pre-processing to the text or the labels, and compare them to the results achieved without pre-processing. The first evaluated pre-processing step is text normalization, where historic or OCR-induced spelling variations are corrected. The second is the additional annotation of tokens referenced by causal arguments. We do not present results for all tasks, but only for argument detection and relation matching, as we regard them as the most relevant. To further simplify the evaluation, we only use the BERT-German model.



(a) Strict (b) Relaxed

Figure 4.6.: Comparison of the F1 score for relation matching between the original and normalized data with respect to the transfer learning configuration for BERT-German. Figure (a) shows the results for the strict regime and (b) for the relaxed regime.

	Actor	Affected	Cause	Effect	Overall
Coreference MCC	-2.6	39.9	-3.93	15.4	9.03

Table 4.10.: MCC for predictions of coreference tokens evaluated on a token-basis. Only the causal arguments involved in coreferences are shown.

Text Normalization

In Figure 4.5 we can see the argument detection results for the original text in comparison with the results when using prior text normalization. Both results seem to be very similar across different transfer learning configurations. Figure 4.6 shows the results for relation matching. For the strict regime, the model with normalized input is better for most transfer learning configurations, however, the best score is achieved by the original model using transfer learning on both datasets. For the relaxed regime, the original model overall performs slightly better.

Coreference Information

Figure 4.7 and Figure 4.8 show the comparison between the original model and when including coreference labels for argument detection and for relation matching.

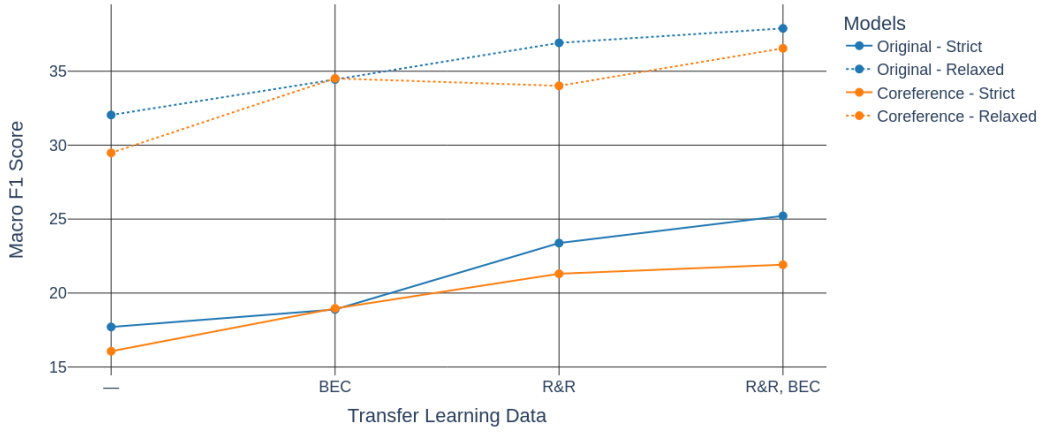
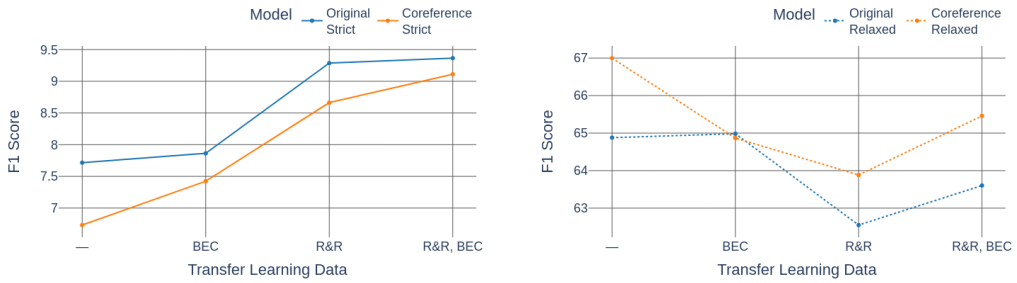


Figure 4.7.: Comparison of the macro F1 scores for strict and relaxed argument detection between the original labels and the labels including coreferences.



(a) Strict

(b) Relaxed

Figure 4.8.: Comparison of the F1 score for relation matching between the original and data including coreferences in respect to the transfer learning configuration for BERT-German. Figure (a) shows the results for the strict regime and (b) for the relaxed regime.

Both models perform similarly well, with the original model being slightly better except for the relaxed relation matching. To evaluate how well the additional coreferences are detected, Table 4.10 presents the MCC scores on a token-basis for tokens labeled as coreference in the ground truth annotations. The scores are low for most arguments, with affected being the highest.

4.1.3. Discussion

In this section we discuss the evaluation results and answer the first three research questions.

Can a BERT-based model outperform a pattern-based approach in the extraction and classification of causal structures in historic texts? The BERT-based models outperform the pattern-based baseline for nearly all tasks. Most causal arguments are predicted reasonably well, especially trigger, cause, effect and actor. The models cannot detect support arguments, which is probably due to the small number of annotated samples. Trigger combination works surprisingly well, both for the combination and separation of triggers. For the classification of causal types, Consequence and Purpose have high F1 scores, while the score for Motivation is much lower. It seems that the distinction between Consequence-Motivation and Motivation-Purpose is difficult. The confusion matrix shows only few wrong predictions for Purpose-Consequence, which indicates that they are semantically more different. Regarding the causal degree, the scores for Inhibit are lower than for Facilitate, and they are subject to a high variation in the cross-validation folds. This might be explained by the imbalanced class distribution and BERT's inability to comprehend negations. The relation matching task gives the best approximation to the general performance of the model, as it combines the results of the trigger detection, trigger combination, and argument detection tasks. The scores for the strict regime are low, which is expected, as this requires the correct prediction of multiple causal arguments within a sentence. The relaxed regime represents more accurately the overall ability of the model to find causal relations, and the scores are considerably higher than for the strict regime. Overall, the recall scores for both regimes are higher than the precision scores, which is desirable for our application of extracting causal relations from historic documents. Interestingly, the baseline

achieves the highest recall, but it must be kept in mind that this is due to a high number of predicted relations of poor quality.

How does pre-training BERT on contemporary, historic or multilingual texts influence the performance? For most tasks, either the BERT-Europeana or BERT-German model performs best, both outperforming the multilingual model. Which of BERT-Europeana or BERT-German is to be preferred seems to depend on the task. The standard deviations between the cross-validation folds are quite high for some of the tasks, making a comparison difficult. Both models seem to be viable choices for causal relation extraction on historic texts.

Does transfer learning, correcting for historical spelling variations or including coreference information improve the performance? For trigger combination, argument detection, and strict relation matching, transfer learning benefits the performance. However, for trigger detection and relaxed relation matching, the Rehbein and Ruppenhofer [1] dataset has a negative effect. For semantic classification of type and degree, the effect of transfer learning seems to be slightly positive, however with a high amount of noise. Interestingly, transfer learning on the BECAUSE [8] data yields an overall similar performance increase in the German models as in the multilingual model. This is surprising, as we expected the multilingual model to profit more from the English data than the other models. The negative influence of the Rehbein and Ruppenhofer [1] data in the trigger detection task could be due to their annotation approach. They annotated only a single relation per sentence, even if more were present. Therefore, during pre-training on the Rehbein and Ruppenhofer data, the model learns to detect only a subset of triggers. In argument detection, however, this is not a problem, as the predictions are conditioned on a trigger group. The results of relation matching show opposite trends between the strict and relaxed evaluation regimes regarding transfer learning. The strict regime seems to profit from more data during transfer learning, while the scores for the relaxed regime decrease. Generally, it appears that the data of Rehbein and Ruppenhofer [1] does help improve the argument spans and create more precise relations, with the downside of finding fewer overall relations, probably due to missing triggers. Normalizing the text does not seem to improve the performance. This result is counter-intuitive, as normalization should lead to a more coherent input representation for BERT, with fewer words split into subwords.

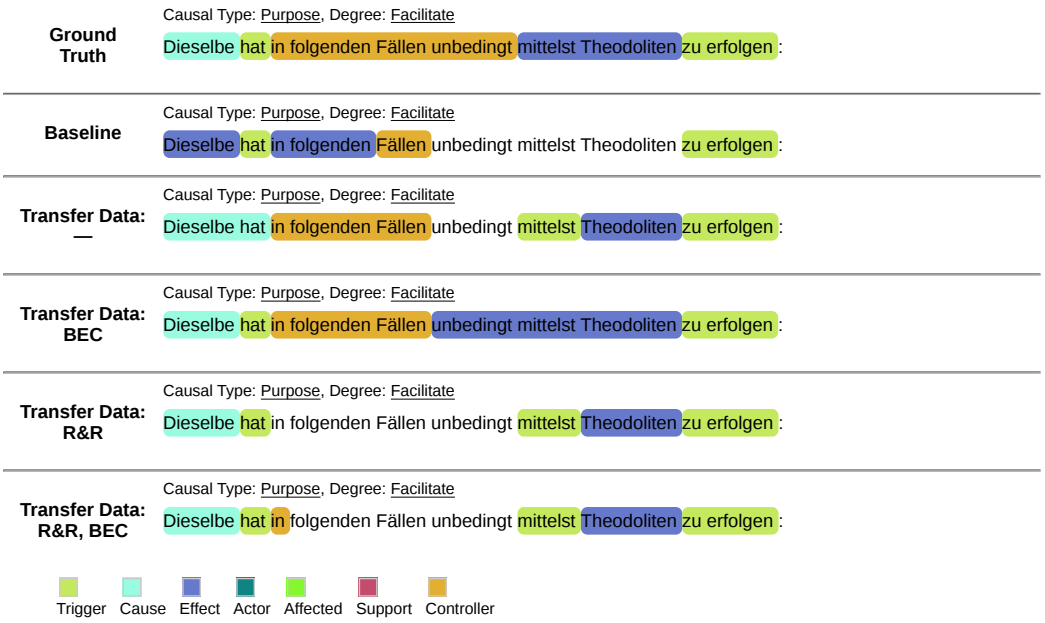


Figure 4.9.: Sentence 1052 predicted using different transfer learning configurations and BERT-German.

Additionally, as the data used during MLM pre-training of BERT-German seldomly includes spelling mistakes, normalization should help make the historical text more like the pre-training data. Several reasons might explain this behavior: First, it might be that BERT naturally copes well with historical spelling variations, and that additional spelling correction is unnecessary. Another possibility is that the positive and negative effects of valid and erroneous corrections cancel out. Adding coreference information also does not increase the model performance. For relaxed relation matching, it performs slightly better, but it must be noted that the number of arguments is also higher due to the coreferences. This increases the chance of matching a predicted relation to a ground truth relation in the relaxed regime, which might explain the performance difference. Furthermore, the predictions for coreference tokens appear to be close to random predictions, indicating that the model cannot usefully capture coreference relationships.

To confirm the validity of the predictions, Figure 4.9 and Figure 4.10 present two sample causal relations from the dataset with ground truth, baseline and model

Figure 4.10.: Sentence 1649 predicted using different transfer learning configurations and BERT-German.

predictions. The used model is BERT-German with different transfer learning configurations. Figure 4.9 shows high variability between the different transfer learning configurations. Still, the models create meaningful causal relations, even though some arguments or spans differ from the ground truth. A common mistake is arguments where only the first token is annotated, as can be seen for the controller argument for the model using both transfer learning datasets. The baseline prediction can successfully find the triggers, however, has problems with the remaining arguments. The second example in Figure 4.10 is predicted similarly by all models. Interestingly, the affected entity is only found once, and only the model with both transfer learning datasets could correctly determine the correct causal type Motivation.

Considering all results, both the BERT-Europeana and BERT-German models seem to be viable choices for extracting causal relations from historical documents. If the quality of the predicted arguments is of interest, it is advisable to utilize transfer learning. If the focus is on detecting the general locations of causal relations, transfer learning seems to be detrimental. There is also no need for text normalization and additional coreference information.

4.2. Causal Attention Heads

This section presents the results of the experiment of detecting causal attention heads in BERT, including a discussion and interpretation of the findings.

4.2.1. Results

Table 4.11 shows the results for the causal attention head experiment for the German data with 5-fold cross-validation. In this experiment we investigate whether BERT contains attention heads that correspond to certain causal relationships. For each head we determine if a trigger token of a causal relation assigns the highest attention weight to a token labeled as a causal argument, where we consider all arguments except trigger. This step is repeated for all sentences in our German and English dataset, where the German dataset consists of the SF and FV documents, and the English dataset of the available BECAUSE [8] corpus. This makes it possible to calculate an accuracy score for each head in respect to each causal argument except

	Cause	Effect	Actor	Aff.	Sup.	Contr.
Baseline	26.06 (3)	38.69 (6)	29.33 (19)	26.46 (8)	20.0 (27)	28.15 (15)
Original BERT	**60.55** (3)	**64.44** (5)	**70.1** (7)	49.64 (13)	**93.33** (15)	47.41 (8)
Fine-tuned BERT	59.87 (2)	61.26 (7)	67.6 (6)	**52.22** (14)	90.0 (22)	**56.33** (6)

Table 4.11.: The accuracy results of the best performing attention heads for the combined German datasets SF and FV. The fine-tuned BERT model is first trained on causal relation extraction as part of the model presented in Section 3.2.2. Using 5-fold cross-validation, the model is fine-tuned on the training data, and the evaluation data is used for the causal attention head experiment. The number in parentheses indicates the standard deviation over the cross-validation folds.

	Cause	Effect
Baseline	59.59 (5)	37.79 (6)
Original BERT	**64.95 (2)**	60.13 (3)
Fine-tuned BERT	64.66 (3)	**67.69 (8)**

Table 4.12.: The accuracy results of the best performing attention heads for the English BECAUSE corpus.

trigger. We report the accuracy of the best-perfoming attention head. The number in parentheses denotes the standard deviation between the folds. Except for affected and controller, the best results of a head are achieved using the original BERT model without fine-tuning. Both models are clearly above the baseline for all arguments. The results of some arguments vary highly between folds, for example for affected or support.

The accuracy results for the English data are presented in Table 4.12. For cause, the original BERT model and the fine-tuned model achieve similar accuracies, with the baseline result being slightly worse. For effect, both BERT models outperform the baseline, and the fine-tuned BERT model has the best accuracy.

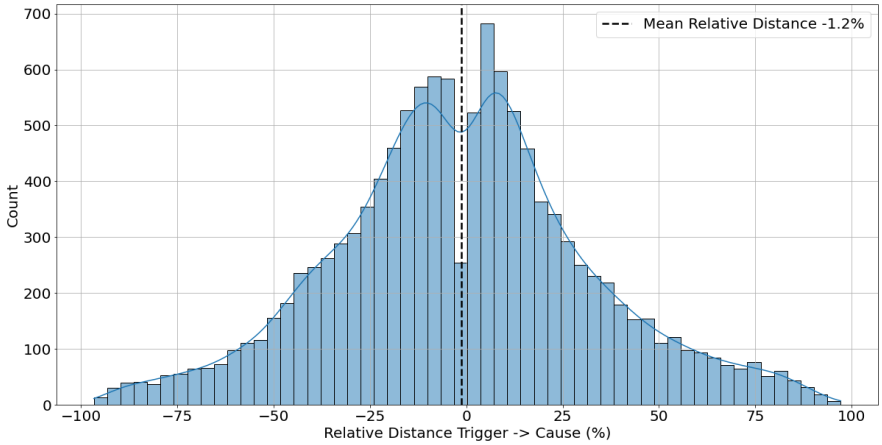
4.2.2. Discussion

This section answers the last research questions and discusses possible flaws of the experiment.

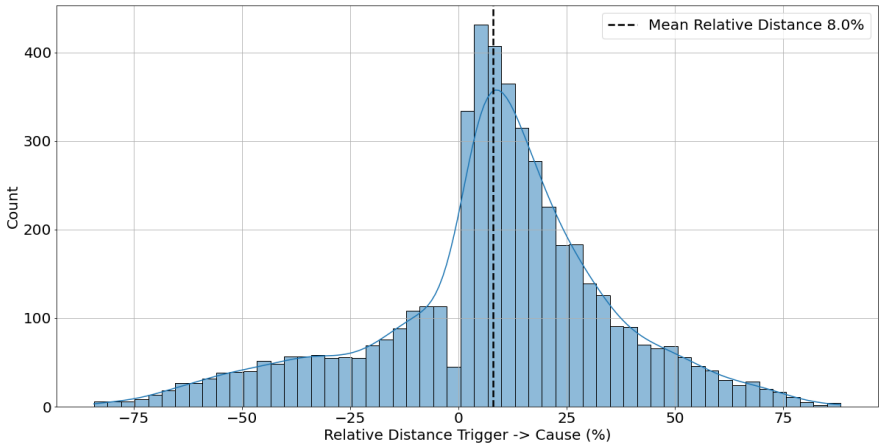
Does BERT already learn causal relationships during self-supervised pre-training? The German BERT model outperforms the baseline, suggesting that some heads already focus on causal relationships. Additional fine-tuning seems to have a minimal effect on the accuracy. For English, the results for effect also perform better than the baseline, however for cause the difference is much smaller. One explanation for the strong baseline can be found in the data. Figure 4.11 shows the relative distances between trigger and cause arguments for the German and English datasets. For the German case, the cause argument seems to be equally distributed around the trigger tokens, whereas in the English case, cause occurs frequently directly after the trigger. As cause arguments in the English data are located mostly in one position relative to the trigger, the baseline performance increases.

We also want to address some of the shortcomings of this experiment. Both BERT models we use for evaluation each contain 144 heads, from which we choose the one with the best accuracy. The comparison to a single baseline might therefore be biased, as it is likely that out of 144 heads one performs better than the baseline by chance. Also, the number of relations for an argument can influence the result, e.g. for support there exist only 13 relations in the whole German dataset. This increases the chance to find a head that correlates with support but does not express the underlying relationship. Another problematic possibility is the existence of heads that focus on semantic or syntactic relationships that mimic causal relationships. For example, the trigger is often a verb, while effect is often an object, therefore heads that capture verb-object relationships might also score high for trigger-effect relationships.

Still, the performance differences between the best heads and the baselines in the German case supports the claim that some heads indeed learn causal relationships during self-supervised pre-training. For the English case, more experimentation is required.



(a) German Corpus



(b) English Corpus

Figure 4.11.: Histograms of the relative distance between trigger and cause arguments for the German and English dataset. For German, the cause arguments are distributed similarly before and after the trigger, while for English, most cause arguments occur shortly after the trigger.

5. Conclusions

In this thesis, we introduced a new annotated dataset for causal relationship extraction on historic documents and used it to evaluate a novel method to solve this task. Additionally, we found evidence that causal relationships might be learned by deep language models during self-supervised pre-training.

The annotated dataset contains two German documents from the late 19th century. We constructed our annotation scheme to capture most of the naturally occurring causal structures. It consists of a fine-grained set of causal arguments and additionally assignes a causal type and degree to each relation. The dataset was subject to a notable amount of noise, which probably resulted from the historical nature of the documents and the overall difficulty of the task. For many sentences there existed more than one way of representing the contained causal relation. Bordering semantic relationships made a distinction between what could be interpreted as causal challenging.

Our approach for causal relationship extraction is based on BERT and extends traditional sequence labeling to also detect multiple separate or overlapping relations within a sequence of text.

We will now provide conclusive answers to our research questions.

1. **Can a BERT-based model outperform a pattern-based approach in the extraction and classification of causal structures in historic texts?**
   The approach outperforms a pattern-based model, for the detection of causal arguments as well as for the semantic classification of the relations.
2. **How does pre-training BERT on contemporary, historic or multilingual texts influence the performance?**
   BERT models pre-trained on contemporary and historical texts both worked similarly well on our dataset, and BERT models pre-trained on a single language performed stronger than multilingual models.

3. **Does transfer learning, correcting for historical spelling variations or including coreference information improve the performance?**
   Transfer learning on related datasets improved the model, however, the results depended on the annotation scheme and language. Correcting historic spelling variations and adding additional information in the form of coreferences did not increase the performance.
4. **Does BERT already learn causal relationships during self-supervised pre-training?**
   We found evidence that causality as an inherent property of natural language is already learned by the BERT model during self-supervised pre-training. We detected attention heads in BERT that seem to focus on causal relationships, without fine-tuning the model on datasets concerning causal relations. In the German BERT model, these heads outperformed a location-based baseline. For English, the results were not conclusive.

Our approach for causal relation extraction captured most relations in the annotated dataset and yields great potential to support historians in their work.

5.1. Future Work

There exist many different future directions to focus on. Two areas are of particular interest: Increasing the model performance and efficiently utilizing the causal relations found in historical documents to aid historians in their work.

To improve the performance, several modifications are possible. In this work, we used the standard BERT model [2] for generating word embeddings. A larger version of BERT, or improved models such as RoBERTa [21] or ELECTRA [22] could potentially create more useful embeddings. This could also be extended to an ensemble architecture, where results from models with different backbones are combined.

Concerning the input to BERT, we might improve the embeddings by using longer sequences of text. Currently, we feed the model separate sentences to keep the computational overhead feasible. By processing larger text sequences together, individual sentences could profit from the information in surrounding sentences.

Models like Longformer [71] are well suited to process long sequences with reasonable computational cost. However, a major impediment to our task is that many promising BERT models are not yet available in German.

Another direction is to annotate more diverse historic documents, regarding the time period, topic, or text type. In this work, we focused on documents regarding forests created by government officials, however, other sources like newspapers or journals could be of interest. Additionally, our model is currently trained on texts from a single time period, therefore training on an annotated sample from the 18th or 17th century might increase the generalization ability of the model.

For the predictions of a model to be useful to historians, practical problems arise that need to be considered. For example, scaling the model to process large amounts of documents with time and resource constraints could be solved by training smaller models with decreased quality but faster execution. It is also essential to convert the predicted relations into a usable and informative format, especially for large documents. The predictions could be converted into representations that are easier to infer knowledge from, such as a knowledge graph.

Bibliography

[1] I. Rehbein and J. Ruppenhofer, "A new resource for German causal language," in *Proceedings of the 12th Language Resources and Evaluation Conference*, (Marseille, France), pp. 5968–5977, European Language Resources Association, May 2020.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "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)*, (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[3] J. L. Mackie, *The Cement of the Universe: A Study of Causation*. Oxford, England: Oxford, Clarendon Press, 1974.

[4] T. Solstad and O. Bott, "Causality and causal reasoning in natural language," *The Oxford handbook of causal reasoning*, pp. 619–644, 2017.

[5] E. Blanco, N. Castell, and D. Moldovan, "Causal relation extraction," in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, (Marrakech, Morocco), European Language Resources Association (ELRA), May 2008.

[6] C. Khoo, J. Kornfilt, R. Oddy, and S.-H. Myaeng, "Automatic extraction of cause-effect information from newspaper text without knowledge-based inferencing," *Literary and Linguistic Computing*, vol. 13, pp. 177–186, 12 1998.

[7] J. Dunietz, L. Levin, and J. Carbonell, "Annotating causal language using corpus lexicography of constructions," in *Proceedings of The 9th Linguistic Annotation Workshop*, (Denver, Colorado, USA), pp. 188–196, Association for Computational Linguistics, June 2015.

[8] J. Dunietz, L. Levin, and J. Carbonell, "The BECauSE corpus 2.0: Annotating causality and overlapping relations," in *Proceedings of the 11th Linguistic Annotation Workshop*, (Valencia, Spain), pp. 95–104, Association for Computational Linguistics, Apr. 2017.

[9] P. Wolff, B. Klettke, T. Ventura, and G. Song, "Expressing causation in english and other languages.," 2005.

[10] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," 2018.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[12] A. Neelakantan, J. Shankar, A. Passos, and A. McCallum, "Efficient non-parametric estimation of multiple embeddings per word in vector space," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1059–1069, Association for Computational Linguistics, Oct. 2014.

[13] J. Camacho-Collados and M. T. Pilehvar, "From word to sense embeddings: A survey on vector representations of meaning," *J. Artif. Int. Res.*, vol. 63, p. 743–788, sep 2018.

[14] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[16] A. Vaswani, S. Bengio, E. Brevdo, F. Chollet, A. N. Gomez, S. Gouws, L. Jones, L. Kaiser, N. Kalchbrenner, N. Parmar, R. Sepassi, N. Shazeer, and J. Uszkoreit, "Tensor2tensor for neural machine translation," *CoRR*, vol. abs/1803.07416, 2018.

[17] P. Michel, O. Levy, and G. Neubig, "Are sixteen heads really better than one?," 2019.

[18] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 5797–5808, Association for Computational Linguistics, July 2019.

[19] S. Jain and B. C. Wallace, "Attention is not Explanation," 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)*, (Minneapolis, Minnesota), pp. 3543–3556, Association for Computational Linguistics, June 2019.

[20] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. R. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. S. Corrado, M. Hughes, and J. Dean, "Google's neural machine translation system: Bridging the gap between human and machine translation," *ArXiv*, vol. abs/1609.08144, 2016.

[21] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[22] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators," 2020.

[23] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "BioBERT: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, sep 2019.

[24] I. Beltagy, K. Lo, and A. Cohan, "Scibert: A pretrained language model for scientific text," in *EMNLP*, Association for Computational Linguistics, 2019.

[25] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," 2020.

[26] D. Nozza, F. Bianchi, and D. Hovy, "What the [mask]? making sense of language-specific bert models," 2020.

[27] B. Chan, S. Schweter, and T. Möller, "German's next language model," 2020.

[28] A. Rogers, O. Kovaleva, and A. Rumshisky, "A primer in bertology: What we know about how bert works," 2020.

[29] T. T. H. Nguyen, A. Jatowt, M. Coustaty, and A. Doucet, "Survey of post-ocr processing approaches," *ACM Comput. Surv.*, vol. 54, jul 2021.

[30] R. Sharma, B. Kaushik, and N. Gondhi, "Character recognition using machine learning and deep learning - a survey," in *2020 International Conference on Emerging Smart Computing and Informatics (ESCI)*, pp. 341–345, 2020.

[31] C. Reul, U. Springmann, C. Wick, and F. Puppe, "State of the art optical character recognition of 19th century fraktur scripts using open source engines," 2018.

[32] T.-T.-H. Nguyen, A. Jatowt, M. Coustaty, N.-V. Nguyen, and A. Doucet, "Deep statistical analysis of ocr errors for effective post-ocr processing," in *2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pp. 29–38, 2019.

[33] P. Stenetorp, S. Pyysalo, G. Topić, T. Ohta, S. Ananiadou, and J. Tsujii, "brat: a web-based tool for NLP-assisted text annotation," in *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, (Avignon, France), pp. 102–107, Association for Computational Linguistics, Apr. 2012.

[34] J. Xu, W. Zuo, S. Liang, and X. Zuo, "A review of dataset and labeling methods for causality extraction," in *Proceedings of the 28th International Conference on Computational Linguistics*, (Barcelona, Spain (Online)), pp. 1519–1531, International Committee on Computational Linguistics, Dec. 2020.

[35] N. Asghar, "Automatic extraction of causal relations from natural language texts: A comprehensive survey," *ArXiv*, vol. abs/1605.07895, 2016.

[36] Z. Li, Q. Li, X. Zou, and J. Ren, "Causality extraction based on self-attentive bilstm-crf with transferred embeddings," *Neurocomputing*, vol. 423, p. 207–219, Jan 2021.

[37] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual string embeddings for sequence labeling," in *Proceedings of the 27th International Conference on Computational Linguistics*, (Santa Fe, New Mexico, USA), pp. 1638–1649, Association for Computational Linguistics, Aug. 2018.

[38] V. Khetan, R. Ramnani, M. Anand, S. Sengupta, and A. E. Fano, "Causal bert : Language models for causality detection between events expressed in text," 2021.

[39] R. Girju, P. Nakov, V. Nastase, S. Szpakowicz, P. Turney, and D. Yuret, "SemEval-2007 task 04: Classification of semantic relations between nominals," in *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, (Prague, Czech Republic), pp. 13–18, Association for Computational Linguistics, June 2007.

[40] I. Hendrickx, S. N. Kim, Z. Kozareva, P. Nakov, D. Ó Séaghdha, S. Padó, M. Pennacchiotti, L. Romano, and S. Szpakowicz, "SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals," in *Proceedings of the 5th International Workshop on Semantic Evaluation*, (Uppsala, Sweden), pp. 33–38, Association for Computational Linguistics, July 2010.

[41] S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith, "The TIGER treebank," in *Proceedings of the workshop on treebanks and linguistic theories*, pp. 24–41, 2002.

[42] P. Koehn, "Europarl: A parallel corpus for statistical machine translation," in *Proceedings of machine translation summit x: papers*, pp. 79–86, 2005.

[43] T. Caselli and P. Vossen, "The event StoryLine corpus: A new benchmark for causal and temporal relation extraction," in *Proceedings of the Events and Stories in the News Workshop*, (Vancouver, Canada), pp. 77–86, Association for Computational Linguistics, Aug. 2017.

[44] A. Cybulska and P. Vossen, "Using a sledgehammer to crack a nut? lexical diversity and event coreference resolution," in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, (Reykjavik, Iceland), pp. 4545–4552, European Language Resources Association (ELRA), May 2014.

[45] M. Sap, R. L. Bras, E. Allaway, C. Bhagavatula, N. Lourie, H. Rashkin, B. Roof, N. A. Smith, and Y. Choi, "ATOMIC: an atlas of machine commonsense for if-then reasoning," in *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in*

Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019, pp. 3027–3035, AAAI Press, 2019.

[46] N. Mostafazadeh, A. Kalyanpur, L. Moon, D. Buchanan, L. Berkowitz, O. Biran, and J. Chu-Carroll, "GLUCOSE: GeneraLized and COntextualized story explanations," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Online), pp. 4569–4586, Association for Computational Linguistics, Nov. 2020.

[47] C. Fillmore, P. Kay, and M. O'Connor, "Regularity and idiomaticity in grammatical constructions: the case of let alone," *Language*, vol. 64, pp. 501–538, 09 1988.

[48] J. Ruppenhofer, M. Ellsworth, M. R. L. Petruck, C. R. Johnson, and J. Scheffczyk, "Framenet ii: Extended theory and practice," 2006.

[49] D. A. van Strien, K. Beelen, M. C. Ardanuy, K. Hosseini, B. McGillivray, and G. Colavizza, "Assessing the impact of ocr quality on downstream nlp tasks," in *ICAART*, 2020.

[50] E. Boros, E. Linhares Pontes, L. A. Cabrera-Diego, A. Hamdi, J. G. Moreno, N. Sidère, and A. Doucet, "Robust Named Entity Recognition and Linking on Historical Multilingual Documents," in *Conference and Labs of the Evaluation Forum (CLEF 2020)*, vol. 2696 of *Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum*, (Thessaloniki, Greece), pp. 1–17, CEUR-WS Working Notes, Sept. 2020.

[51] M. Ehrmann, M. Romanello, A. Flückiger, and S. Clematide, "Extended overview of clef hipe 2020: Named entity processing on historical newspapers," in *CLEF*, 2020.

[52] A. Brunner, N. D. T. Tu, L. Weimer, and F. Jannidis, "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 *SwissText/KONVENS*, 2020.

[53] M. Bollmann, "A large-scale comparison of historical text normalization systems," 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)*, (Minneapolis, Minnesota), pp. 3885–3898, Association for Computational Linguistics, June 2019.

[54] *ACE (Automatic Content Extraction) English Annotation Guidelines for Events*, 5.4.3 2005.07.01 ed., 2005.

[55] V. Lai, M. V. Nguyen, H. Kaufman, and T. H. Nguyen, "Event extraction from historical texts: A new dataset for black rebellions," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, (Online), pp. 2390–2400, Association for Computational Linguistics, Aug. 2021.

[56] A. K. Cybulska and P. Vossen, "Historical event extraction from text," in *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, (Portland, OR, USA), pp. 39–43, Association for Computational Linguistics, June 2011.

[57] M. Riedl and S. Padó, "A named entity recognition shootout for German," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Melbourne, Australia), pp. 120–125, Association for Computational Linguistics, July 2018.

[58] S. Schweter and J. Baiter, "Towards robust named entity recognition for historic German," in *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, (Florence, Italy), pp. 96–103, Association for Computational Linguistics, Aug. 2019.

[59] S. Schweter, "Europeana bert and electra models," Nov. 2020.

[60] R. Iliev and R. Axelrod, "Does causality matter more now? increase in the proportion of causal language in english texts," *Psychological Science*, vol. 27, no. 5, pp. 635–643, 2016. PMID: 26993741.

[61] Y. Hara, "Semantic shift from conjunction/causal to conditional," 2021.

[62] M. Ehrmann, M. Romanello, A. Flückiger, and S. Clematide, "Extended Overview of CLEF HIPE 2020: Named Entity Processing on Historical Newspapers," in *CLEF 2020 Working Notes. Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum* (L. Cappellato, C. Eickhoff, N. Ferro, and A. Névéol, eds.), vol. 2696, (Thessaloniki, Greece), p. 38, CEUR-WS, 2020.

[63] A. Brunner, S. Engelberg, F. Jannidis, N. D. T. Tu, and L. Weimer, "Corpus REDEWIEDERGABE," in *Proceedings of the 12th Language Resources and Evaluation Conference*, (Marseille, France), pp. 803–812, European Language Resources Association, May 2020.

[64] C. Neudecker, "An open corpus for named entity recognition in historic newspapers," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, (Portorož, Slovenia), pp. 4348–4352, European Language Resources Association (ELRA), May 2016.

[65] C. Sporleder, "Natural language processing for cultural heritage domains," *Language and Linguistics Compass*, vol. 4, pp. 750–768, 09 2010.

[66] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (San Diego, California), pp. 1480–1489, Association for Computational Linguistics, June 2016.

[67] B. Jurish, *Finite-State Canonicalization Techniques for Historical German*. PhD thesis, Universität Potsdam, January 2012. (completed 2011, published 2012).

[68] H. Nakayama, "seqeval: A python framework for sequence labeling evaluation," 2018. Software available from https://github.com/chakki-works/seqeval.

[69] S. Schweter and L. März, "Triple e - effective ensembling of embeddings and language models for ner of historical german," in *CLEF*, 2020.

[70] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does BERT look at? an analysis of BERT's attention," in *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, (Florence, Italy), pp. 276–286, Association for Computational Linguistics, Aug. 2019.

[71] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," 2020.

[72] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017.

Appendix A.

Hyperparameters

In this section we present the hyperparameters of our model in more detail.

The model is built in PyTorch[1]. We use an AdamW [72] optimizer and a linear learning rate scheduler with 1000 warmup steps. The initial learning rate is 2.5×10^{-5}. Before backpropagation, the gradients are clipped to 1 to prevent exploding gradients, and to increase the speed we train with mixed precision. We use a batch size of 2, larger batch sizes seem to decreases the performance. The model is trained on a Tesla P100 GPU.

[1] Version 1.11.0

Appendix B.

Brat Configuration

The configuration for annotating with BRAT is an adapted version of the configuration used by Dunietz et al. [8], which can be found online[1]. We added the arguments actor, affected, support, controller and trigger, and removed the argument means.

Listing B.1: BRAT annotation configuration

```
# Simple text-based definitions of hierarchial
   ontologies of
# (physical) entity types, relation types, event
   types, and
# attributes.

[entities]

# Definition of entities.
# Format is a simple list with one type per line.

Argument
Note

[relations]

# Definition of (binary) relations.
```

[1] https://github.com/duncanka/BECAUSE/blob/master/annotation.conf (Accessed on: 2022-07-19)

```
# Format in brief: one relation per line, with
    first space-separated
# field giving the relation type and the rest of
    the line the
# comma-separated arguments in ROLE:TYPE format.
    The roles are
# typically "Arg1" and "Arg2".

Coref          Arg1:Argument,    Arg2:Argument
<OVERLAP>  Arg1:<ENTITY>, Arg2:<ENTITY>, <OVL-
    TYPE>:<ANY>

[events]
# Definition of events.

# Format in brief: one event per line, with first
    space-separated
# field giving the event type and the rest of the
    line the
# comma-separated arguments in ROLE:TYPE format.
    Arguments may be
# specified as either optional (by appending "?" to
     role) or repeated
# (by appending either "*" for "0 or more" or "+"
    for "1 or more").

!Causation
    Consequence       Cause*:Argument, Effect*:
       Argument, Actor*:Argument, Affected*:
       Argument, Support*:Argument, Controlling*:
       Argument, Trigger?:Argument
    Motivation        Cause*:Argument, Effect*:
       Argument, Actor*:Argument, Affected*:
       Argument, Support*:Argument, Controlling*:
       Argument, Trigger?:Argument
    Purpose           Cause*:Argument, Effect*:
       Argument, Actor*:Argument, Affected*:
```

```
        Argument, Support*:Argument, Controlling*:
        Argument, Trigger?:Argument
#   Inference          Cause*:Argument, Effect*:
    Argument, Actor*:Argument, Affected*:Argument,
    Support*:Argument, Controlling*:Argument,
    Trigger?:Argument
NonCausal          Arg0?:Argument, Arg1?:Argument

[attributes]

# Definition of entity and event attributes.

# Format in brief: first tab-separated field is
    attribute name, second
# a set of key-value pairs. The latter must define
    "Arg:" which
# specifies what the attribute can attach to (
    typically "<EVENT>").
# If no other keys are defined, the attribute is
    binary (present or
# absent). If "Value:" with multiple alternatives
    is defined, the
# attribute can have one of the given values.

Degree          Arg:Consequence|Motivation|Inference|
    Purpose, Value:Facilitate|Inhibit
Temporal Arg:Consequence|Motivation|Inference|
    Purpose|NonCausal
Correlation Arg:Consequence|Motivation|Inference|
    Purpose|NonCausal
Hypothetical Arg:Consequence|Motivation|Inference|
    Purpose|NonCausal
Obligation-permission Arg:Consequence|Motivation|
    Inference|Purpose|NonCausal
Creation-termination Arg:Consequence|Motivation|
    Inference|Purpose|NonCausal
Extremity-sufficiency Arg:Consequence|Motivation|
```

```
        Inference|Purpose|NonCausal
    Context  Arg:Consequence|Motivation|Inference|
        Purpose|NonCausal
```