University of Potsdam

Department of Linguistics

Bachelor Thesis

Computational Linguistics B.Sc.

Visualization Approaches for Coherence Relations

Author:

Olha Zolotarenko

Supervisors:

Dr. Dipl.-Ing. Julián Moreno Schneider

Peter Bourgonje

28. Oktober 2020

Contents

List of Figures

List of Tables

List of Listings

Zusammenfassung

Die hier vorliegende Arbeit stellt einen Versuch dar, den Visualisierungsansätzen in dem Feld der annotierten Diskursrelationen nahezukommen und durch Vergleich verschiedener Programmierwerkzeuge eine anforderungsnahe Lösung zu finden. Als Gegenstand der Forschung wurden Kohärenzrelationen ausgewählt, welche eine Reihe an Eigenschaften aufweisen, die für viele Visualisierungsmethoden herausfordernd sein können. Die Arbeit stellt fünf verschiedene Visualisierungsmöglichkeiten sowohl von der Anwendungs- als auch von der Entwicklungsperspektive vor. Die zunächst getesteten einfachen HTML-Ansätze sowie das Softwarepaket displaCy zeigen das unzureichende Niveau für die Visualisierungszwecke dieser Arbeit. Die alternative Implementierung mit D3 würde die Voraussetzungen zwar optimal erfüllen, sprengt aber deutlich den Rahmen des Projektes. Die gewählte Hauptmethode wurde als Single-Web-Anwendung konzipiert und verwendet das Annotationstool brat, welches die meisten definierten Voraussetzungen für die Repräsentation der Kohärenzrelationen erfüllt. Die Anwendung stellt die im Text annotierten Kohärenzrelationen graphisch dar und bietet eine Filterfunktion für verschiedene Relationstypen an.

1 Introduction

Visualization is a powerful tool for exploring data and analyzing different kinds of problems. It helps to model algorithms, generalize results and provide new perspectives. There are many different visualization approaches, depending on the type of data and purpose of visualization, but they all have one point in common: they are created primarily for *human visual perception*.

The nature of human visual perception is a complex topic and has been of particular interest in relation to data visualization and graphical user interfaces (GUIs). There have been many scientific studies on the superiority of the human visual system in comparison to other perceptual systems. Some have indicated that pictures are more likely to be remembered than words (Grady, McIntosh, Rajah, and Craik, 1998; Standing, Conezio, and Haber, 1970). This phenomenon, known as the *"picture superiority effect"* (Paivio, 1971), has been shown in numerous experiments (e.g., Curran and Doyle, 2011) and is prominent in visualization tasks. However, the exact explanation for the effect remains debated.

In addition to the potential evidence of the cognitive advantage of visualization, there is a famous example of how the presentation of non-visualized numerical data might be unclear at first sight: *Anscombe's quartet* is a dataset of nearly identical mean, variance and correlation values which have surprisingly different distributions when presented in a visual form (Anscombe, 1973). This case shows how visualized data could be beneficial for quick analysis and recognizing the main trends in a dataset. However, researchers sometimes use this advantage to mislead their audience by distorting a dataset's measurement axes (Stewart, 2019).

In this thesis, I attempt to explore possible uses of visualization in the fields of discourse analysis and linguistic annotation. More specifically, I attempt to find an approach to present *coherence relations* and to define the main criteria for visualizing this type of data.

2 Coherence Relations

This section defines the terms *coherence* and *coherence relations*. Bußmann (Stede, 2018, p. 24) defines coherence as "a semantic-cognitive meaningful context of a text which can be represented in the form of semantic networks of concepts and relations". Coherence could be considered complementary to *cohesion*, but while cohesion refers to unity within a text or sentence on the surface level, coherence refers to a deep internal connection between text parts (Stede, 2018, p. 132).

In discourse analysis, a *relation* is defined as a certain semantic connection between two sentences. There are two types of coherence relations based on the distance between the units being measured: *local* and *global* coherence (Stede, 2018, p. 24). Local coherence appears between clauses in a single sentence and between neighboring sentences, whereas global coherence composes the text topic. In my research, I focused on the local coherence relations of different senses and types according to the theoretical framework of the Penn Discourse TreeBank (PDTB) presented in Table 1 (Webber, Prasad, Lee, and Joshi, 2019, p.17). In addition to the PDTB framework, there are several other theories concerning discourse relations and, in particular, coherence. The most prominent are the Rhetorical Structure Theory (RST) (Mann and Thompson, 1988), the Cognitive approach to Coherence Relations (CCR) (Sanders, Spooren, and Noordman, 1992) and the Segmented Discourse Representation Theory (SDRT) (Asher, Asher, and Lascarides, 2003).

Level-1	**Level-2**	**Level-3**
TEMPORAL	SYNCHRONOUS	–
	ASYNCHRONOUS	PRECEDENCE
		SUCCESSION
CONTINGENCY	CAUSE	REASON
		RESULT
		NEGRESULT
	CAUSE+BELIEF	REASON+BELIEF
		RESULT+BELIEF
	CAUSE+SPEECHACT	REASON+SPEECHACT
		RESULT+SPEECHACT
	CONDITION	ARG1-AS-COND
		ARG2-AS-COND
	CONDITION+SPEECHACT	–
	NEGATIVE-CONDITION	ARG1-AS-NEGCOND
		ARG2-AS-NEGCOND
	NEGATIVE-CONDITION+SPEECHACT	–
	PURPOSE	ARG1-AS-GOAL
		ARG2-AS-GOAL
COMPARISON	CONCESSION	ARG1-AS-DENIER
		ARG2-AS-DENIER
	CONCESSION+SPEECHACT	ARG2-AS-DENIER+SPEECHACT
	CONTRAST	–
	SIMILARITY	–
EXPANSION	CONJUNCTION	–
	DISJUNCTION	–
	EQUIVALENCE	–
	EXCEPTION	ARG1-AS-EXCPT
		ARG2-AS-EXCPT
	INSTANTIATION	ARG1-AS-INSTANCE
		ARG2-AS-INSTANCE
	LEVEL-OF-DETAIL	ARG1-AS-DETAIL
		ARG2-AS-DETAIL
	MANNER	ARG1-AS-MANNER
		ARG2-AS-MANNER
	SUBSTITUTION	ARG1-AS-SUBST
		ARG2-AS-SUBST

Table 1: *The PDTB-3 Sense Hierarchy*

Because the data from the parser (see Section 5.4.1) used for this project is based on the PDTB 2.0 relation types (Prasad et al., 2008), I define these data according to this version of the PDTB. The framework specifies the following relation types: explicit, implicit, alternative lexicalization (AltLex), entity relation (EntRel) and no relation (NoRel). The latter three labels are used for cases in which an implicit connective could not be provided.

The classification of coherence relations is sometimes defined based on the surface evidence of coherence. If a connective appears in a text, it is recognized as a signal of a coherence relation between the units that it joins. The majority of cases, though, are coherence relations with no apparent connective (Stede, 2018, p. 133). Examples 2.1 and 2.2 illustrate implicit and explicit relations, the most important types of coherence relations:

(2.1) Because I got all wet in the rain, I had to go back home to change my clothes.

(2.2) I got all wet in the rain. I had to go back home to change my clothes.

The first sentence has an explicit marker of a coherence relation: "because", a connective of the sense "reason". The sentence can be rephrased to omit the word "because", as shown in the second sentence. The other relation types - alternative lexicalization, entity relation and no relation - are discarded as they are not returned by the parser used as input for the visualization.

3 Annotation Procedures and Associated Issues

It is crucial to determine the procedure from which coherence and coherence relations result and the type of data in which they appear. We say that the data - text - is *annotated* or *parsed* (Carstensen et al., 2010, p.276) and that the resulting data structure (hierarchical or shallow) can be stored and used for different purposes.

3.1 What is annotation?

In linguistics and computational linguistics, empirical databases known as *corpora* play a central role in programming, testing and research. They are collections of speech or writing consisting of linguistic data, annotations and metadata (Carstensen et al., 2010, p. 482).

In general, *linguistic annotation* involves the association of descriptive or analytic notations with linguistic data (Ide and Pustejovsky, 2017). In corpora, annotations are seen as analytical layers of the data consisting of segments and linguistic, as well as extra-linguistic, information. Among morphologic, syntactic, semantic and other types of annotations, there are also *discourse-related annotations*, which are of particular interest for this research. These can contain information about coreference phenomena, information structure or discourse relations. Among these we can find coherence relations, the focus of this research.

One of the typical formats used for annotations is XML (Extensible Markup Language). An example of such format from the Potsdam Commentary Corpus (PCC) is given in Listing 1 (Stede and Neumann, 2014).

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?relations relSet="Martin1992" lexURL="jar:file:/..."?>
<discourse>
  <unit type="ext" id="3">Trommeln gehört halt zum Geschäft .</unit
   >
  <unit type="ext" id="5">
   <unit type="int" id="3">
     <connective id="3" relation="contrast">Doch</connective>
     unterm Strich stehen Brandenburgs Schulen ganz gut da .
   </unit>
  </unit>
</discourse>
```

Listing 1: *Fragment of an annotated text from PCC (maz-0002.xml)*

Discourse relations can be annotated according to different styles and theories, from RST trees to PDTB-style annotations, which will be discussed in the following sections. When attempting to model similar kinds of relation types, these approaches can be hard to compare. Some attempts of combining both styles and enriching PDTB-styled annotations look promising, while also revealing the complexity of the subject (Bourgonje and Zolotarenko, 2019).

3.2 The goals of annotation visualizations

The approach to data visualization described here is motivated more by science than artistic beauty. Hence, in this project, I pursue the following goals for the visualization:

- **Presentation and communication of information**
  A graphic representation of data is created primarily to communicate relations behind the data. In the case of this thesis, I want to present parsed text data and reveal coherence relations to human viewers.

- **Convenient analysis of data**
  As already discussed, raw data can be misleading at first sight. A good visualization of data could therefore facilitate and speed up data analysis. Annotated relations presented graphically could also be beneficial for educational purposes.

- **Easier debugging of the tools behind represented data**
  During development of any kind of programms, large amounts of raw textual and numerical data might be difficult to interpret, and it may be even more difficult to foresee potential errors in the data. Visual analysis could help in interpreting these data and quickly finding these potential errors, thereby opening new avenues for further research.

3.3 General requirements of visualization tools

There are some general requirements for visualized data (coherence relations in the case of this thesis) besides stylish presentation of the research results. Butz (2009) defines several points as challenging:

- **Layout and positioning**
  The layout of presented data, as well as relative positioning of the elements, play an important role in the easy comprehension and interpretation of data. Thus, when presenting textual data and their annotations, the layout needs to be as simple as possible in order to prevent unnecessary distractions. The visualization has to be effective in terms of giving *convenient access* to the data and *having a clear structure*. In the case of annotated relations, the visualization is expected to present these in such a way that each relation and its components can be easily distinguished and analyzed.

- **Scale**
  Performance of the implementation is as important as the layout. It could be determined by executing software performance tests giving a measure for scalability, reliability and resource usage of a particular program (Beneken, Ernst, and Schmidt, 2016, p.753).

- **Navigation and interaction**
  Another requirement is enabling a convenient user experience and achieving the purpose of use. Applying other tests, such as usability or accessibility tests, could provide insights into the user experience with the program. Visualizations (static or dynamic) are, for example, expected to load quickly, be responsive and act according to the user's requests, such as filtering and storing results, scaling, navigating or applying manual changes.

Some other specific requirements which apply to this project are:

- **Software model**
  The desired tool has to be open-source.

- **Programming effort**
  Solving object-specific representation problems with this tool must be possible within a reasonable amount of time and programming effort.

4 Possible Structures of Text Representation

Modelling discourse relations of a particular text accurately could require complex representation structures that we are currently unable to think of. However, there are several possible structures which would be suitable for the visualization purposes in this thesis.

4.1 Possible structures for coherence relations

The first possible structure is a simple, *flat representation* of a text. "Flat" refers to an approach which works with the text in a given dimension without adding any additional layers like other methods described in this section. These flat representation structures use color, highlighting, subscripts or any other visual tools to mark borders of chunks and relations between them. Examples of such approaches are very easy to implement, though they lack robustness with certain complexities of relations.

The second possible structure, which is postulated by many approaches, is a *tree-like* representation of a text (see RST [Mann and Thompson, 1988]). "Tree" is a term from graph theory, defined as a connected, acyclic, undirected graph. A tree-like representation of a text would be shown as a set of child vertices with segments connected to a certain relation types or senses (see Figure 1 visualized by ANNIS [Krause and Zeldes, 2016]).

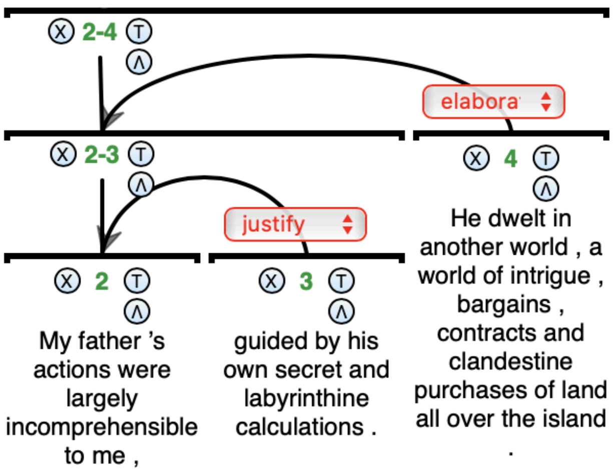

2-4
elabora
2-3
justify
4
2
3
My father 's actions were largely incomprehensible to me ,
guided by his own secret and labyrinthine calculations .
He dwelt in another world , a world of intrigue , bargains , contracts and clandestine purchases of land all over the island .

Figure 1: *An example of visualized RST tree by ANNIS*

However, the tree approach could be complicated by relations between non-adjacent segments (Stede, 2018, p. 149) or other factors. This will be discussed in the following section.

Finally, there are *hybrid representation approaches*, which combine flat and tree-like visualizations. These approaches have many advantages in clarity of presenting coherence relations, allowing the user to reach some degree of complexity and to avoid certain problems, which will be discussed later in this thesis.

4.2 Issues in representing coherence relations

There are many complexity levels which are crucial to consider when developing a good visualization of coherence relations. The most important of them - *discontinuous segments, overlapping relations* and *n-arity of relations* - are explained in this section.

- **Discontinuous segments**
  The simplest relations, at least in terms of visualization, consist of adjacent segments joined by a simple connective. However, relations sometimes occur between non-adjacent segments, or the arguments or connectives are simply not continuous. In the lexicon of German discourse connectives (Stede, 2002), there are continuous vs. discontinuous connectives and single vs. phrasal connectives. The connective "aber" (German for "but") is an example of a simple, continuous connective, whereas "dadurch, dass" (German for "as a result of that") is an example of a discontinuous connective and has to be marked as such in an appropriate way.

- **Overlapping relations**
  Sometimes, annotation reveals segments that are involved in several relations.

  (4.1) Trommeln gehört halt zum Geschäft. Doch unterm Strich stehen Brandenburgs Schulen ganz gut da. Zum einen wurden nach der Wende fast alle Schulbuchbestände ausgetauscht, zum anderen müssen sich märkische Eltern am Buchkauf beteiligen.

  In Example 4.1, there are two relations:

  - ***Explicit.Comparison.Concession.Arg2-as-denier:***
    1. Segment: Trommeln gehört halt zum Geschäft.
    2. Segment: Doch unterm Strich stehen Brandenburgs Schulen ganz gut da.
  - ***Explicit.Expansion.Conjunction:***
    1. Segment: Doch unterm Strich stehen Brandenburgs Schulen ganz gut

da.
2. Segment: Zum einen wurden nach der Wende fast alle Schulbuchbestände ausgetauscht, zum anderen müssen sich märkische Eltern am Buchkauf beteiligen.

These two relations have the segment "Doch unterm Strich stehen Brandenburgs Schulen ganz gut da" in common, which we call *"overlapping"*.

- **N-arity of relations**
  Normally, discourse relations are binary or ternary. Binary relations include two segments and ternary, three. In the PDTB, the explicit relations consist of three segments - two arguments and a connective. The implicit relations are binary, having only two arguments and no connective. Potentially, relations may involve more segments or, in some cases, fewer than two relations, due to particular parser settings, which should be considered when choosing an appropriate visualization method.

5 Towards Visualization Approaches for Coherence Relations

It is important to find a proper way to efficiently show coherence relations while dealing with the issues described above. The main tool used in this project - *brat* - best meets the requirements for showing coherence relations and will be discussed in more detail in Section 5.4. However, other tools are discussed first in order to compare them to brat. This will provide a better understanding of the problems related to the visualization of coherence relations while justifying the choice of the main approach.

Section 5.1 describes simple *HTML tools* available for visualization purposes, Section 5.2 presents the visualization package *displaCy* and Section 5.3 provides a detailed overview for the implementation of the visualization library *D3*.

5.1 Simple HTML tools

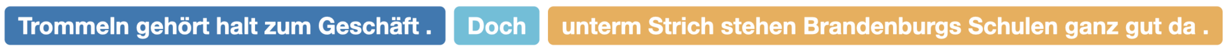
5.1.1 Coloring and highlighting spans

The first simple method that could be used for visualization is highlighting or coloring spans with *bootstrap labels*. The following contextual label classes which could be used within a <span> element of HTML code:

- `.label-default`
- `.label-primary`

- `.label-success`
- `.label-info`
- `.label-warning`
- `.label-danger`

Figure 2 shows the bootstrap labels style for the visualization of a single relation. The code for it is provided in Listing 2. The spans are easy to identify by color even though there are no labels marking them as the first argument, the second argument or the connective.



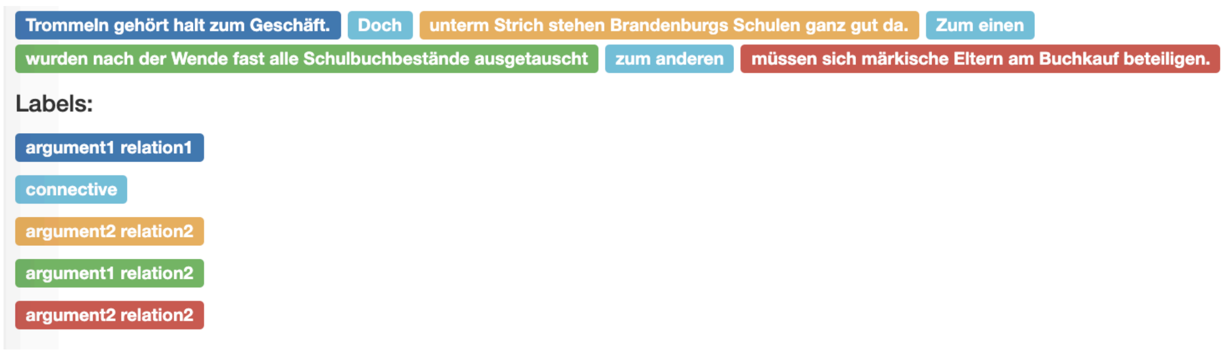
Trommeln gehört halt zum Geschäft . Doch unterm Strich stehen Brandenburgs Schulen ganz gut da .

Figure 2: *Bootstrap labels: an example of one relation*

```html
<div class="container">
  <span class="label label-primary">Trommeln gehört halt zum
   Geschäft.</span>
  <span class="label label-info">Doch</span>
  <span class="label label-warning">unterm Strich stehen
   Brandenburgs Schulen ganz gut da.</span>
</div>
```

Listing 2: *Code example for bootstrap labels*

Figure 3 shows a similar example as shown above, but with two relations. As the relations do not overlap, they are still easy to distinguish but the problem with labelling remains. This could be solved by using some kind of label information given below the annotation itself. This solution might become inconvenient if the number of relations increases (even with different colors, the boundaries of the relations become unclear). Furthermore, the information on the relation type and sense is not presented, but could be included by some descriptive means similar to label names.



Trommeln gehört halt zum Geschäft. Doch unterm Strich stehen Brandenburgs Schulen ganz gut da. Zum einen wurden nach der Wende fast alle Schulbuchbestände ausgetauscht zum anderen müssen sich märkische Eltern am Buchkauf beteiligen.

Labels:

argument1 relation1

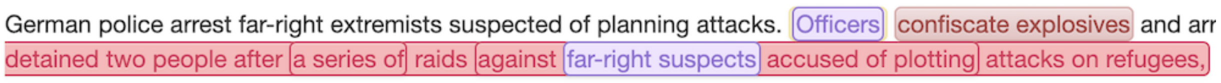
connective

argument2 relation2

argument1 relation2

argument2 relation2

Figure 3: *Bootstrap labels: an example of two relations*



German police arrest far-right extremists suspected of planning attacks. Officers confiscate explosives and arr
detained two people after a series of raids against far-right suspects accused of plotting attacks on refugees,

Figure 4: *Bootstrap labels: an example of multiple overlapping annotation spans*

A more complex example with overlapping spans is shown in Figure 4. Although the solution for the labelling problem could be similar to the example in Figure 3, the overlapping of spans or complete relations makes the annotation unclear and difficult to interpret.

These examples have shown the following drawbacks to using bootstrap labels to highlight and color spans:

- **Interpretation**
  The annotations become hard to interpret as the number of spans and/or relation grows.

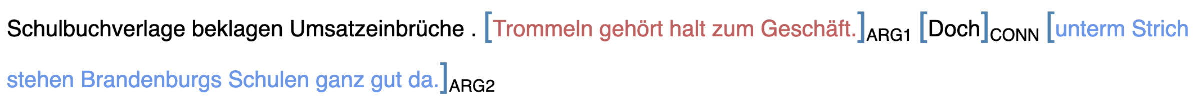
- **Relations boundaries**
  There is no convenient method of displaying boundaries of the annotated relations.

- **Annotation labels**
  Information on spans and relations is separated from the annotation.

- **Discontinuous units**
  There is no possibility to show discontinuous units.

5.1.2 Brackets

The second method in this section refers to ANNIS [1], a "web browser-based search and visualization architecture for complex multilayer linguistic corpora with diverse types of annotation" (Krause and Zeldes, 2016). It has an information structure visualization whose style could be convenient to use for visualization of coherence relations. One of the examples implemented in the research for this thesis is shown in Figure 5. The brackets are defined by spans before and after using internal CSS style (see Listing 3).



Schulbuchverlage beklagen Umsatzeinbrüche . [Trommeln gehört halt zum Geschäft.]$_{ARG1}$ [Doch]$_{CONN}$ [unterm Strich stehen Brandenburgs Schulen ganz gut da.]$_{ARG2}$

Figure 5: *ANNIS-style square brackets visualization example*

[1] https://corpus-tools.org/annis/visualizations.html

```
<style type="text/css">
.custom {
font-family: sans-serif;
color: black;
font-size:20px;
}

span:before {
   content: '[';
   font-size:30px;
   color:steelblue;
 }

span:after {
   content: ']';
   font-size:30px;
   color:steelblue;
 }
</style>
```

Listing 3: *ANNIS-style square brackets code*

As relation complexity increases, this method tends to show behavior similar to the HTML labels method. Figure 6 presents the annotation with two familiar overlapping relations from Section 4.1. Although the annotation labels are integrated in the visualization, the brackets and the annotated relations are difficult to distinguish.

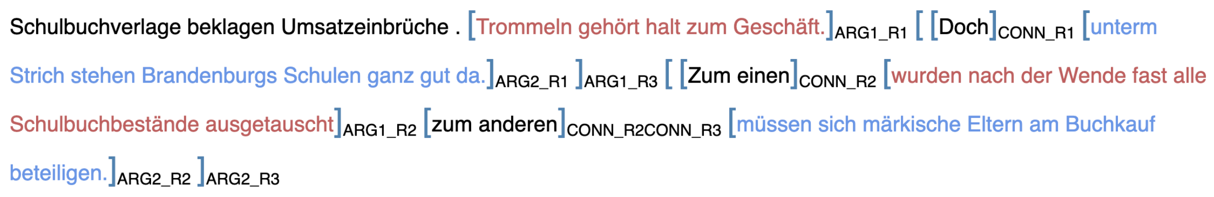

Schulbuchverlage beklagen Umsatzeinbrüche . [Trommeln gehört halt zum Geschäft.]$_{\text{ARG1\_R1}}$ [[Doch]$_{\text{CONN\_R1}}$ [unterm Strich stehen Brandenburgs Schulen ganz gut da.]$_{\text{ARG2\_R1}}$]$_{\text{ARG1\_R3}}$ [[Zum einen]$_{\text{CONN\_R2}}$ [wurden nach der Wende fast alle Schulbuchbestände ausgetauscht]$_{\text{ARG1\_R2}}$ [zum anderen]$_{\text{CONN\_R2CONN\_R3}}$ [müssen sich märkische Eltern am Buchkauf beteiligen.]$_{\text{ARG2\_R2}}$]$_{\text{ARG2\_R3}}$

Figure 6: *ANNIS-style square brackets visualization example*

The brackets method has shown the following disadvantages, which should be avoided when displaying coherence relations:

- **Interpretation**
  The annotations remain hard to interpret as the number of spans/relation grows due to high complexity of the brackets clusters.

- **Relations boundaries**
  Although the relation boundaries are implemented in this method, it is difficult to distinguish, as well as code, the opening and closing brackets in order to interpret the annotation.

- **Annotation labels**
  Information on spans and relations is implemented as subscripts after the closing bracket. Although a convenient solution for a small number of relations that have neither overlapping nor discontinuous spans, they make the visualization more complex, rather than help explain it.

- **Discontinuous units**
  There is no possibility to show discontinuous units.

5.2 displaCy

DisplaCy[2] is a visualizing package of spaCy, a free, open-source library for advanced Natural Language Processing (NLP) written in Python. The package offers two possibilities for visualization: Dependency and Named Entity Visualization. Figures 7 and 8 show visualizations of Example 4.1. Figure 7 presents the Named Entity Visualizer with highlighted relation spans.

Figure 8 shows an attempt at using the Dependency Visualizer, which results in some disadvantages for displaying discourse relations. Selected spans are not marked and the image in the Scalable Vector Graphics (SVG) format must be rendered manually via the options parameter (Figure 9, for comparison, is adjusted manually), which is not always suitable for longer texts.

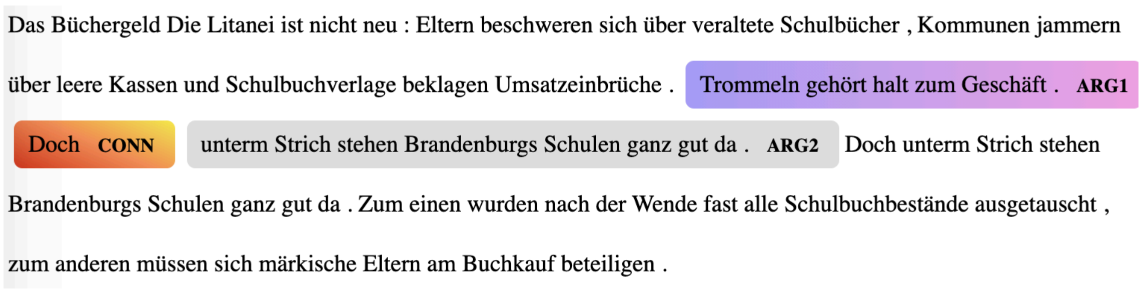

Das Büchergeld Die Litanei ist nicht neu : Eltern beschweren sich über veraltete Schulbücher , Kommunen jammern über leere Kassen und Schulbuchverlage beklagen Umsatzeinbrüche . Trommeln gehört halt zum Geschäft . ARG1 Doch CONN unterm Strich stehen Brandenburgs Schulen ganz gut da . ARG2 Doch unterm Strich stehen Brandenburgs Schulen ganz gut da . Zum einen wurden nach der Wende fast alle Schulbuchbestände ausgetauscht , zum anderen müssen sich märkische Eltern am Buchkauf beteiligen .

Figure 7: *Visualization using displaCy Named Entity Visualizer*

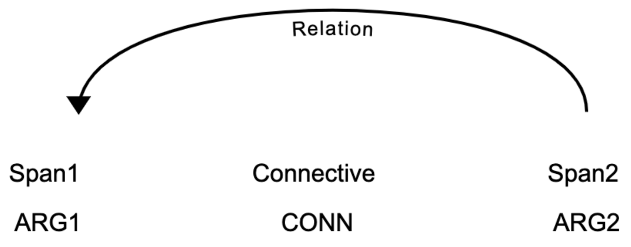


Figure 8: *Visualization using displaCy Dependency Visualizer (default rendering scheme)*

[2] https://spacy.io/usage/visualizers

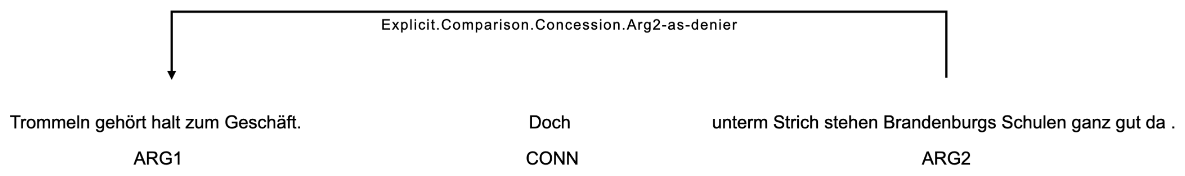

Explicit.Comparison.Concession.Arg2-as-denier
Trommeln gehört halt zum Geschäft.
ARG1
Doch
CONN
unterm Strich stehen Brandenburgs Schulen ganz gut da .
ARG2

Figure 9: *Visualization using displaCy Dependency Visualizer (manual rendering with options)*

The displaCy visualization was set up using Python script `bottle_displacy.py`. As the name suggests, the web framework applied for displaCy is Bottle. Alternatively, Flask can be used as a microframework for starting development server and debugger. Once the Bottle script is up and running, the visualization can be found under the specified localhost and port. The main function for the displaCy Named Entity Visualizer is given in Listing 4.

```python
def displacy_visual_ents():
    colors = {"ARG1": "linear-gradient(90deg, #aa9cfc, #fc9ce7)",
              "INT": "linear-gradient(203deg, #e0eed4, #f6ec00, #4e7a27)",
              "CONN": "linear-gradient(203deg, #f6ec00, #ff8647, #e32400)"}
    options = {"ents": ["ARG1", "ARG2", "CONN"], "colors": colors}
    ent_input = [{"text": "...Some text to visualize here...",
            "ents": [{"start": 182, "end": 217, "label": "ARG1"},
                     {"start": 218, "end": 222, "label": "CONN"},
                     {"start": 223, "end": 278, "label": "ARG2"}],
            "title": "Sentence 1"}]
    html = displacy.render(ent_input, style="ent", manual=True, options=options)

    # Example of exporting the image
    doc = nlp("Trommeln gehört halt zum Geschäft . Doch unterm Strich stehen Brandenburgs Schulen ganz gut da .")
    svg = displacy.render(doc, style="dep")
    output_path = Path("./images/sentence.svg")
    output_path.open("w", encoding="utf-8").write(svg)

    return html
```

Listing 4: *Python function for displaCy Named Entity Visualizer*

The function for the Dependency Visualizer named `displacy_visual_deps`, which generates the images shown in Figures 7 and 8, can also be found in the mentioned Bottle script.

Using displaCy in the field of coherence relations leads to the following conclusions:

- **Spans and default rendering**
  The displaCy package has good visualization possibilities: entities and dependencies. Dependencies are modelled for short spans, though (words, POS tags, etc.), and default rendering is not properly displayed for large spans.

- **Overlapping and discontinuous entities**
  DisplaCy cannot show overlapping relations and discontinuous entities, which is very important for the objective of this project.

- **Lacking of structure for discourse relations**
  Manual tweaking of displaCy is convenient and well-documented. However, it is not designed to deal with discourse relations and it makes it hard to meet all the needs of visualizing them. A combination of both the Named Entity and Dependency Visualizer would be a perfect strategy to visualize coherence relations using spaCy tools.

SpaCy has other tools with powerful features for natural language processing and annotation. For instance, *Prodigy*[3], which is powered by active learning, seems to offer good features to annotate dependencies and relations, but must remain a topic for later research as it is not open source.

5.3 D3

Another possible visualization method which offers high levels of customization is the purely javascript *d3-annotation* by Susie Lu[4]. This method uses the D3.js library and gives annotations a new perspective with built-in annotation types. Figure 10 shows a possible visualization of an explicit comparison relation for a portion of Example 4.1.

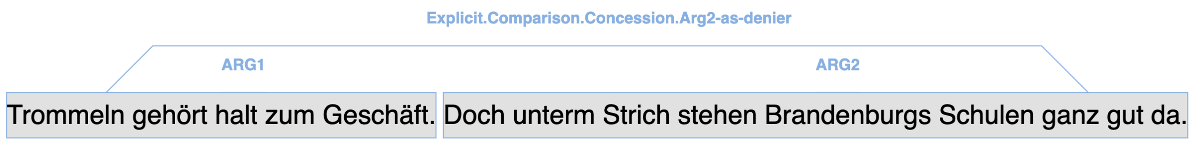

Explicit.Comparison.Concession.Arg2-as-denier
ARG1
ARG2
Trommeln gehört halt zum Geschäft.
Doch unterm Strich stehen Brandenburgs Schulen ganz gut da.

Figure 10: *Visualization using d3-annotation*

[3] https://prodi.gy/features/dependencies-relations
[4] https://d3-annotation.susielu.com

The suggested implementation uses the D3 module to position annotation elements over SVG with the text. This is done according to the following algorithm:

- **SVG Container**
  An SVG container is created in the script part.
- **SVG Text Element**
  SVG text element is added to the SVG container.
- **SVG Container <– SVG Text Element Attributes**
  Text element and their attributes are added to SVG container:
  - coordinates x and y
  - text
  - font family
  - font size
  - positioning of the text
  - text color
- **Text coordinates**
  To calculate the coordinates of a text or subtext the following features need to be estimated:
  - coordinates x and y
  - width and height of the text or subtext
  - top, right, bottom, left coordinates of the text or subtext

  It is critical that the font properties are correct, as the location of the text or subtext depend on the font family and size.
- **Annotation object**
  Annotation object is created (constant variable annotations)
- **Annotation features**
  Annotation features are defined. The following features were used in this implementation to mark the spans and construct the relation:
  - `d3.annotationCalloutRect` and
  - `d3.annotationCalloutElbow`
- **Features of annotation types**
  For each annotation type there are following features specified:
  - x and y are absolute coordinates of an object, which give the position of the data point to which the annotation refers.

    – dx and dy are relative coordinates, which indicate a shift along the x- or y-axis on the position of an element or its content.

- **Building annotation**
  `d3.annotation()` is called to build the annotation

Apart from relatively high scripting effort comparing to the tools which only need to be embedded into the project, d3-annotation has some potentially important properties:

- **Layout and aesthetics**
  It has a high degree of freedom of styles, forms and complexity of constructed annotations.

- **Pure javascript**
  The entire annotation visualization can be scripted using one language, which is a potentially important feature.

Since choosing d3-annotation would mean implementing a whole annotation tool from scratch and it goes beyond the required programming effort, it would be best suitable to explore it in another project.

5.4 brat

The main approach used in this thesis is based on the *brat* rapid annotation tool[5], a web-based tool for text annotation (Stenetorp et al., 2011). This tool allows for convenient embedding of the visualized annotations in other projects.

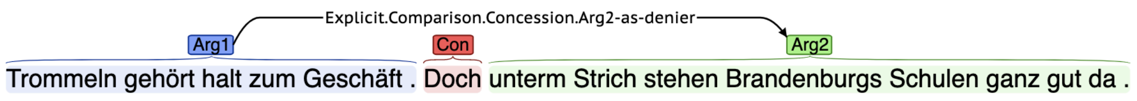

Explicit.Comparison.Concession.Arg2-as-denier
Arg1
Con
Arg2
Trommeln gehört halt zum Geschäft . Doch unterm Strich stehen Brandenburgs Schulen ganz gut da .

Figure 11: *An example of a relation annotated with brat*

5.4.1 Data used and their format

The output of the German Shallow Discourse Parser[6] of Peter Bourgonje was used as a base for exploring the visualization strategies. The parser is the result of a PhD dissertation (to appear), individual components of which are published in the papers „Explicit Discourse Argument Extraction for German“ (Bourgonje and Stede, 2019) and „Identifying Explicit Discourse Connectives in German“ (Bourgonje and Stede, 2019).

[5] http://brat.nlplab.org
[6] https://github.com/PeterBourgonje/GermanShallowDiscourseParser

The visualization implementation with brat uses JSON format for both input data of the parser and its converted version, in order to suit brat internal format. For proper embedding brat needs two objects to be defined: the *collection data object* with defined properties of entity types; and the *document object* with text, entities and relations to be visualized. Listings 5 and 6 show these objects for a minimal example with one explicit relation.

```
const docData = {
"text" : "Das Büchergeld  Die Litanei ist nicht neu : " +
     "Eltern beschweren sich über veraltete Schulbücher , " +
     "Kommunen jammern über leere Kassen und " +
     "Schulbuchverlage beklagen Umsatzeinbrüche . " +
     "Trommeln gehört halt zum Geschäft . " +
     "Doch unterm Strich stehen Brandenburgs Schulen ganz gut da .
   ",
"entities" : [["E1", "Arg1", [[182, 217]]],
       ["E2", "Arg2", [[223, 278]]],
       ["E3", "Connective", [[218, 222]]]
       ],
"relations": [["R1",
        "Explicit.Comparison.Concession.Arg2-as-denier",
        [["1", "E1"],
        ["2", "E2"]]]
       ]
};
```

Listing 5: *The document data object of brat*

```
const collData = {
  entity_types: [ {
  type  : "Arg1",
  labels : ["Arg1", "Arg1"],
  bgColor: "#7fa2ff",
  borderColor: "darken"
  },
  {
  type  : "Arg2",
  labels : ["Arg2", "Arg2"],
  bgColor: "#9FF781",
  borderColor: "darken"
  },
  {
  type  : "Connective",
  labels : ["Con", "Con"],
  bgColor: "#FA5858",
  borderColor: "darken"
  }]
};
```

Listing 6: *The collection data object of brat for defining entity types*

5.4.2 brat features

Brat is a powerful annotation tool with a wide range of useful features. The full list of them can be found on their web page. Some of the most important features for this project are listed below:

- **High-quality visualization at any scale**
  The annotation elements of brat maintain high quality at any scale which is crucial for annotated relations. An example of a zoomed relation is shown in Figure 12.

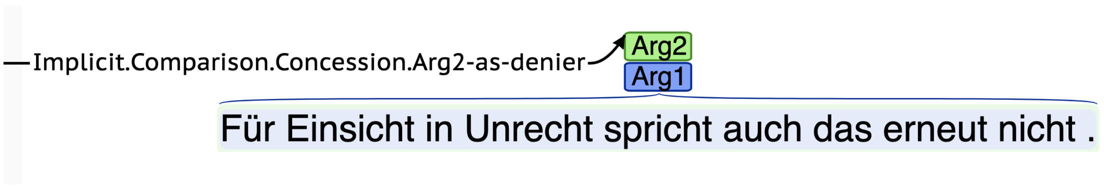

Implicit.Comparison.Concession.Arg2-as-denier
Arg2
Arg1
Für Einsicht in Unrecht spricht auch das erneut nicht .

Figure 12: *Zoomed relation in brat*

- **Rich set of annotation primitives**
  Brat has a good variety of annotation primitives, which are especially useful when dealing with discourse relations. As described on the brat website, the primitives provide marking for text spans (e.g., entity annotation), binary relations, equivalence classes, n-ary associations (e.g., event annotation) and attributes. They can be applied together in any combination to define specific annotation tasks. Figure 13 shows how brat deals with the problem of discontinuous segments; the dotted line and red highlighted spans mark the discontinuous connective "entweder...oder" (German for "either...or").

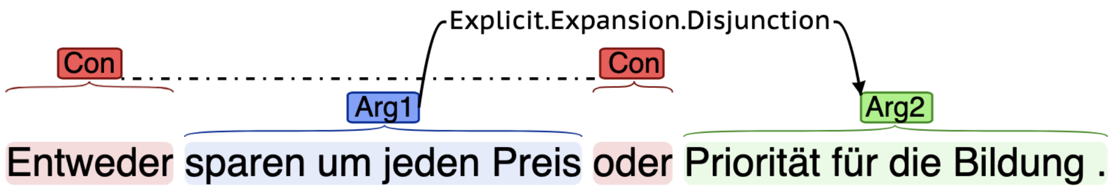

Explicit.Expansion.Disjunction
Con
Con
Arg1
Arg2
Entweder sparen um jeden Preis oder Priorität für die Bildung .

Figure 13: *Discontinuous connective shown with brat*

- **Wide browser support**
  The brat tool also has an important technical feature for web apps: wide browser support (see also brat browser compatibility in Table 2).

5.4.3 The architecture of brat visualization

The visualization of coherence relations with brat is implemented with a web app in the form of a single-page application (SPA). It is designed to be used offline without a constant connection to the central server. The input content is dynamically updated in response to user actions.

To describe the architecture of the implementation, the topics directory infrastructure, backend, frontend and data conversion are discussed in the following sections.

5.4.3.1 Directory infrastructure

The project structure consists of main directories for javascript, CSS and for the data converter. The project is also available as a GitHub repository[7].

```
brat
├─ converter
├─ js
│   └─ brat.js
├─ css
│   └─ style_brat.css
└─ index_brat.xhtml
```

5.4.3.2 Backend

The backend of the web app includes getting the output data and converting it, the brat embedding and the javascript file for website functionality which defines the filtering of the displayed relations.

Parsed input data

As mentioned in Section 5.4.1, the input data for this application is the output of the German Shallow Discourse Parser. It is provided as a JSON file with the structure presented in Listing 7. The structure is almost identical to the file format of the CoNLL-2015 Shared Task on Shallow Discourse Parsing (Xue et al., 2015).

Each relation is stored under a unique id and contains attributes like `docID`, `sense`, `type`, `arg1`, `arg2` and `connective` of the parsed relation. The attributes `arg1`, `arg2` and `connective` also have the subattributes `character span list`, `raw text` and `token list`. The most important attributes which must be converted are `sense`, `type` and `character span list` of each argument and connective.

[7] https://github.com/zolotarenko/VisualizationOfCoherenceRelations

```
[{"ID": 1,
  "DocID": "09-01-2020_13:03:49",
  "Sense": "Comparison.Concession.Arg2-as-denier",
  "Type": "Explicit",
  "Arg1":
    {"CharacterSpanList": [[182, 217]],
     "RawText": "Trommeln gehört halt zum Geschäft .",
     "TokenList": [[182, 190, 27, 1, 0],
             [191, 197, 28, 1, 1],
              [198, 202, 29, 1, 2],
              [203, 206, 30, 1, 3],
              [207, 215, 31, 1, 4],
              [216, 217, 32, 1, 5]]},
  "Arg2":
     {"CharacterSpanList": [[223, 278]],
     "RawText": "unterm Strich stehen Brandenburgs Schulen ganz gut
    da .",        "TokenList": [[223, 229, 34, 2, 1],
              [230, 236, 35, 2, 2],
              [237, 243, 36, 2, 3],
              [244, 256, 37, 2, 4],
              [257, 264, 38, 2, 5],
              [265, 269, 39, 2, 6],
              [270, 273, 40, 2, 7],
              [274, 276, 41, 2, 8],
              [277, 278, 42, 2, 9]]},
  "Connective":
     {"CharacterSpanList": [[218, 222]],
      "RawText": "Doch",
      "TokenList": [[218, 222, 33, 2, 0]]}},
```

Listing 7: *Fragment of the parser output*

Brat embedding

One of the big advantages of brat is how simply it is embedded in other projects. Some main technical features will be listed in this section.

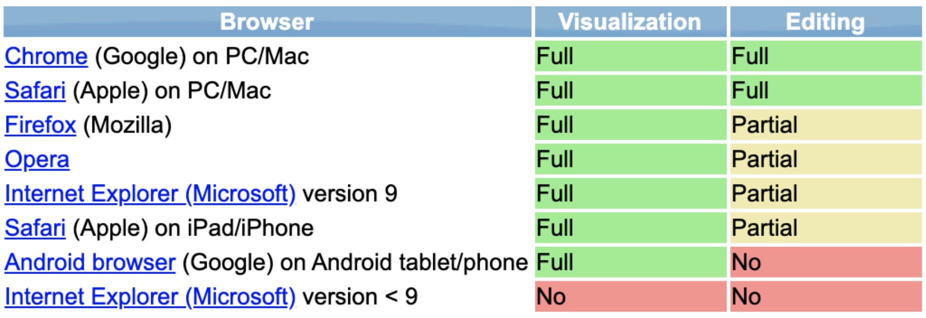
- **Installation**
  In this project, brat is used as an embedded stand-alone serverless visualization method. Once downloaded, brat is fully configured to work using just two lines in the script box as shown in Listing 8 (the dependencies need to match the systems paths and brat location).

```
<link rel="stylesheet" type="text/css" href="./brat-v1.3
   _Crunchy_Frog/static/style-vis.css"/>
```

```
<script type="text/javascript" src="./brat-v1.3_Crunchy_Frog/
    client/lib/head.load.min.js"></script>
```

Listing 8: *Including brat in the project*

- **Data transport**
  brat uses the following techniques for data transport:
  - JSON - both for annotation configurations and for data to be visualized
  - AJAX - a technique for asynchronous data transfer without reloading of the existing page (Beneken et al., 2016, p.733).
- **Client-side scripting**
  The main programming languages of brat are:
  - JavaScript: central scripting language: `head.js` was needed to load its scripts in parallel (can be also adjusted to load in another way)
  - XHTML 1.0: mainly needed for embedding the annotation SVG, otherwise the embedding won't show up properly
  - CSS: used for styling the annotations
- **Browser compatibility**[8]
  The main requirement for the web browser is that the document should be XHTML compliant (SVG image embedding mentioned previously). Compatibility for several popular browser versions supporting visualization and/or editing is displayed in Table 2.



Browser	Visualization	Editing
Chrome (Google) on PC/Mac	Full	Full
Safari (Apple) on PC/Mac	Full	Full
Firefox (Mozilla)	Full	Partial
Opera	Full	Partial
Internet Explorer (Microsoft) version 9	Full	Partial
Safari (Apple) on iPad/iPhone	Full	Partial
Android browser (Google) on Android tablet/phone	Full	No
Internet Explorer (Microsoft) version < 9	No	No

Table 2: *Browser compatibility of brat*

Converter

As mentioned in Section 5.4.1, the input data for the brat web app needs to be converted to match the brat-specific document object format. For this purpose, the Python script `convert_input_brat.py` is used. The script takes two files as input: the text file, which needs to be converted to a one-line string; and the parser output in JSON format, as follows:

[8] http://brat.nlplab.org/embed.html

```
python3 convert_input_brat.py text-file file-parsed
```

The formats of the entities and relations have to be converted as given below.

- **The entities (arguments and connectives)**

  ```
  [${ID}, ${TYPE}, [[${START}, ${END}]]]
  ```

- **The relations**

  ```
  [${ID}, ${TYPE}, [[${ARGNAME}, ${TARGET}],
  [${ARGNAME}, ${TARGET}]]]
  ```

The procedure of conversion is mostly iterative. The script is taking every parsed relation from the input and is extracting entities (arguments and connectives) with their character span lists as well as relations consisting of the corresponding type and sense and entity segments (see Listing 5). The output of the script is also a JSON file which can be used as brat input document object.

JS script

The programming core of the web application is `brat.js`. It implements the filtering functionality of the visualization, renders the relations to be presented on the main page and styles the page with a loader. The filtering types and senses in the given document are implemented with the following algorithm:

- **Checking and unchecking**
  Checking and unnchecking events are being tracked and passed to the filtering function.

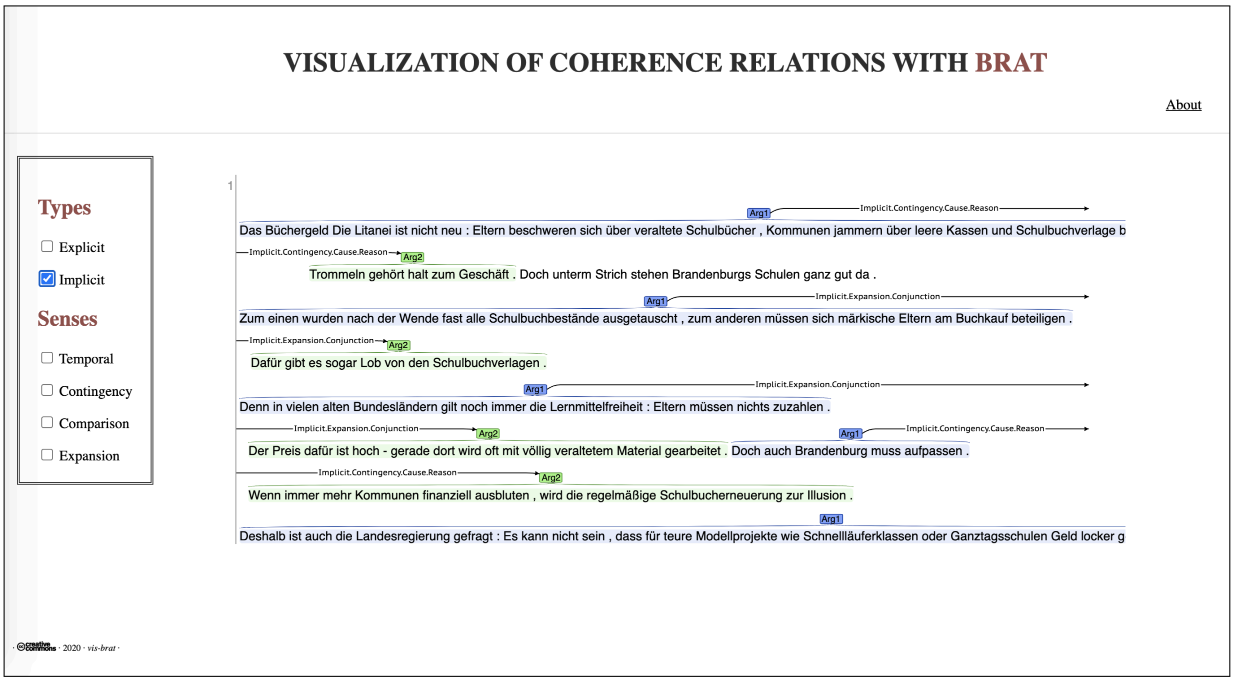
- **Selecting and deleting relations**
  All selected relations and corresponding entities (arguments and connectives in case of explicit relation) need to be found in the input object and then added to the JSON object to be handed over to brat. An unchecking event triggers another search and deletes specified relations to be hidden from the visualization.

- **Filtering**
  In this implementation, filtering occurs when types and senses are mixed in the selection (e.g. the user wants only the explicit contingency relations from all the explicit relations to be displayed).

- **Rendering**
  Once all of the relations are selected and filtered, brat can be called with `Util.embed`, which returns the embedded visualizer's dispatcher object.

5.4.3.3 Frontend

- **GUI - main page**
  The main page is implemented with the functionality to filter relations (their types and senses) and the visualization itself (see `index_brat.xhtml`). Checking the corresponding checkboxes on the left side of the main page filters the types and senses. The annotation visualization is placed in the center and takes up most of the GUI. There is also an "About" popup page that provides some additional information on this project.


VISUALIZATION OF COHERENCE RELATIONS WITH BRAT

About

Types

- [ ] Explicit
- [x] Implicit

Senses

- [ ] Temporal
- [ ] Contingency
- [ ] Comparison
- [ ] Expansion

Arg1 Implicit.Contingency.Cause.Reason
Das Büchergeld Die Litanei ist nicht neu : Eltern beschweren sich über veraltete Schulbücher , Kommunen jammern über leere Kassen und Schulbuchverlage b
Implicit.Contingency.Cause.Reason Arg2
Trommeln gehört halt zum Geschäft . Doch unterm Strich stehen Brandenburgs Schulen ganz gut da .
Arg1 Implicit.Expansion.Conjunction
Zum einen wurden nach der Wende fast alle Schulbuchbestände ausgetauscht , zum anderen müssen sich märkische Eltern am Buchkauf beteiligen .
Implicit.Expansion.Conjunction Arg2
Dafür gibt es sogar Lob von den Schulbuchverlagen .
Arg1 Implicit.Expansion.Conjunction
Denn in vielen alten Bundesländern gilt noch immer die Lernmittelfreiheit : Eltern müssen nichts zuzahlen .
Implicit.Expansion.Conjunction Arg2 Arg1 Implicit.Contingency.Cause.Reason
Der Preis dafür ist hoch - gerade dort wird oft mit völlig veraltetem Material gearbeitet . Doch auch Brandenburg muss aufpassen .
Implicit.Contingency.Cause.Reason Arg2
Wenn immer mehr Kommunen finanziell ausbluten , wird die regelmäßige Schulbucherneuerung zur Illusion .
Arg1
Deshalb ist auch die Landesregierung gefragt : Es kann nicht sein , dass für teure Modellprojekte wie Schnellläuferklassen oder Ganztagsschulen Geld locker g

· creative commons · 2020 · *vis-brat* ·

Figure 14: *Main page of the web app*

- **CSS style for the page**
  The page style of the web app is set using external CSS (see `style_brat.css` file). The main attributes described in the file are:
  - Headings, subheadings and background
  - Boxes for annotation, checkboxes and title (Listing 9 provides an example of the CSS style of the annotation box and shows how to make the box fit the content of the annotation)
  - Loader (styled as an animated spinner)
  - Footer (contains information on the rights of the project and the year of developing).
  - "About" page (styled as popup page to maintain focus on the annotation visualization)

```css
/* Annotation */
.box2 {
    background-color: transparent;
    overflow: auto;
    padding: 40px;
    font-family: 'Work Sans', sans-serif;
    height:fit-content;
}
```

Listing 9: *An example of the CSS styled boxes for the annotation*

6 Outlook

The purpose of this section is to offer possible improvements for the implemented visualization web app. These suggestions can be categorized into two groups: improvements of the existing brat visualization and improvements of the general further developmental stages.

- **Interactive input**
  Getting rid of hard-coded data would be the first important step to improve customization flexibility and accessibility of the implemented app. Due to limited availability of the parser, the input data is stored in a JSON variable that has to be manually replaced. The suggested solution is to take input data from the parser or other source and perform an automatic conversion, thereby avoiding a manual data transfer. This could be completed with a simple API call to the parser and to the converter. Alternatively, the converter could be integrated into the parser in order to provide the format required, without additional steps. In addition, it is important to include an input form (a text field) in the web page script so that users can paste in text they want to visualize with the app.

- **Ternary relation to include connectives**
  In the current version of the implementation, connectives are sometimes hard to assign to a relation. Ternary relations, rather than binary relations with only argument spans, would improve the readability of annotations.

- **Additional information by moving the mouse over annotation elements**
  The parser provides more information on the relations and arguments than displayed in the current implementation. Those attributes, like id, raw text or token list, could be useful for debugging or analyzing data and could be easily accessed when implemented in the GUI (e.g., as brat normalization).

- **Optimization of filtering functionality**
  At the current stage, the filtering function seems to be excessively long and could be optimized in terms of readability and performance.

- **More functionality for debugging module**
  Since the web app could also be used as a debugging tool, it would be useful to have some functions like statistics - number of different relation types or senses, the total number of tokens covered in the relation argument - or warnings for empty spans.

- **Possibility of live customization the brat visualization**
  Another nice feature would be an interactive settings panel for changing the colour of the spans, font size, font family or other elements.

The following ideas may help to develop the project further:

- **More styles**
  Embedding more visualization styles may offer more variety and help to analyze the data from another perspective. This could be implemented as a filter or selection option.

- **More functions apart from visualization**
  Some functions such as exporting the shown annotation to different formats (PDF, JPG, PNG etc.) for further use would be beneficial on later stages of the project.

- **Expanding project to include larger discourse structures**
  As mentioned in Section 2, this project focuses on local coherence. The next step in visualizing larger discourse structures would be including global coherence and showing hierarchical relations between longer text spans.

7 Conclusion

Visualizations improve and elevate human perception of data. This thesis has demonstrated how the graphical presentation of discourse relations is easier to interpret than raw data. The implemented visualization web app is a step towards exploring and using graphical presentations of discourse annotations for different purposes. Implementation of custom visualizations for a single project is a very difficult task, and often goes beyond the scope of the project. Hence, being familiar with the existing tools, their potential facilities and techniques before building a custom visualization could be very beneficial and time-saving.

Developing the web app with brat has illustrated how convenient it can be to embed a tool and customize it for specific needs. Brat meets most of the requirements for visualizing coherence relations: It addresses the overlapping relations and entity spans, manages to display discontinuous units, is an open-source tool and embeds easily into new projects. Several other visualization methods - d3-annotation, displaCy and the more straight-forward HTML tools - were compared to the existing implementation with brat.

Section 6, Outlook, gives some suggestions for further developing as the project has currently only basic functionality and can be extended when needed.

Bibliography

Anscombe, F. J. (1973). Graphs in Statistical Analysis. *The American Statistician*, *27*(1), 17–21. doi:10.1080/00031305.1973.10478966

Asher, N., Asher, N. M., & Lascarides, A. (2003). *Logics of conversation*. Cambridge University Press.

Beneken, G., Ernst, H., & Schmidt, J. (2016). Grundkurs Informatik: Grundlagen und Konzepte für die erfolgreiche IT-Praxis-Eine umfassende, praxisorientierte Einführung (Auflage von 2015).

Bourgonje, P., & Stede, M. (2018). Identifying Explicit Discourse Connectives in German. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue* (pp. 327–331). Melbourne, Australia: Association for Computational Linguistics.

Bourgonje, P., & Stede, M. (2019). Explicit Discourse Argument Extraction for German. In *Proceedings of the 21st International Conference on Text, Speech and Dialogue*, Ljubljana, Slovenia. Retrieved from https://link.springer.com/chapter/10.1007/978-3-030-27947-9_3

Bourgonje, P., & Zolotarenko, O. (2019). Toward cross-theory discourse relation annotation. In *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019* (pp. 7–11).

Carstensen, K., Ebert, C., Ebert, C., Jekat, S., Langer, H., & Klabunde, R. (2010). *Computerlinguistik und Sprachtechnologie: Eine Einführung. 3. überarb. u. erw. Aufl. Heidelberg: Spektrum*. Akademischer Verlag.

Curran, T., & Doyle, J. (2011). Picture superiority doubly dissociates the ERP correlates of recollection and familiarity. *Journal of Cognitive Neuroscience*, *23*(5), 1247–1262.

Grady, C. L., McIntosh, A. R., Rajah, M. N., & Craik, F. I. (1998). Neural correlates of the episodic encoding of pictures and words. *Proceedings of the National Academy of Sciences*, *95*(5), 2703–2708.

Ide, N., & Pustejovsky, J. (2017). *Handbook of linguistic annotation*. Springer.

Krause, T., & Zeldes, A. (2016). ANNIS3: A new architecture for generic corpus query and visualization. *Digital Scholarship in the Humanities*, *31*(1), 118–139.

Mann, W. C., & Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text*, *8*(3), 243–281.

Paivio, A. (1971). Imagery and verbal processes. New York, NY: Holt, Rinheart & Winston. Paivio, A. 1986. Mental representation: A dual-coding approach. New York, NY: Oxford University Press.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A. K., & Webber, B. L. (2008). The Penn Discourse TreeBank 2.0. In *LREC*. Citeseer.

Sanders, T. J., Spooren, W. P., & Noordman, L. G. (1992). Toward a taxonomy of coherence relations. *Discourse processes*, *15*(1), 1–35.

Standing, L., Conezio, J., & Haber, R. N. (1970). Perception and memory for pictures: Single-trial learning of 2500 visual stimuli. *Psychonomic Science*, *19*(2), 73–74.

Stede, M. (2002). DiMLex: A lexical approach to discourse markers.

Stede, M. (2018). *Korpusgestützte Textanalyse: Grundzüge der Ebenen-orientierten Textlinguistik*. Narr Francke Attempto Verlag.

Stede, M., & Neumann, A. (2014). Potsdam Commentary Corpus 2.0: Annotation for Discourse Research. In *LREC* (pp. 925–929).

Stenetorp, P., Topić, G., Pyysalo, S., Ohta, T., Kim, J.-D., & Tsujii, J. (2011). BioNLP Shared Task 2011: Supporting Resources. In *Proceedings of BioNLP Shared Task 2011 Workshop* (pp. 112–120). Portland, Oregon, USA: Association for Computational Linguistics. Retrieved from http://www.aclweb.org/anthology/W11-1816

Stewart, M. (2019). The Power of Visualization in Data Science. Retrieved May 16, 2019, from https://towardsdatascience.com/the-power-of-visualization-in-data-science-1995d56e4208

Webber, B., Prasad, R., Lee, A., & Joshi, A. (2019). The Penn Discourse Treebank 3.0 Annotation Manual. Philadelphia: University of Pennsylvania. Available online: https://catalog ?

Xue, N., Ng, H. T., Pradhan, S., Prasad, R., Bryant, C., & Rutherford, A. (2015). The CoNLL-2015 Shared Task on Shallow Discourse Parsing. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning - Shared Task* (pp. 1–16). doi:10.18653/v1/K15-2001