Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 6912–6920
Marseille, 11–16 May 2020

TIARA: A Tool for Annotating Discourse Relations and Sentence Reordering

Jan Wira Gotama Putra[†], **Simone Teufel**[‡†], **Kana Matsumura**[§], **Takenobu Tokunaga**[†]
[†]Tokyo Institute of Technology, Japan [‡]University of Cambridge, United Kingdom [§]Waseda University, Japan
gotama.w.aa@m.titech.ac.jp, simone.teufel@cl.cam.ac.uk, kmatsumura@suou.waseda.jp, take@c.titech.ac.jp

Abstract

This paper introduces TIARA, a new publicly available web-based annotation tool for discourse relations and sentence reordering. Annotation tasks such as these, which are based on relations between large textual objects, are inherently hard to visualise without either cluttering the display and/or confusing the annotators. TIARA deals with the visual complexity during the annotation process by systematically simplifying the layout, and by offering interactive visualisation, including coloured links, indentation, and dual-view. TIARA's text view allows annotators to focus on the analysis of logical sequencing between sentences. A separate tree view allows them to review their analysis in terms of the overall discourse structure. The dual-view gives it an edge over other discourse annotation tools and makes it particularly attractive as an educational tool (e.g., for teaching students how to argue more effectively). As it is based on standard web technologies and can be easily customised to other annotation schemes, it can be easily used by anybody. Apart from the project it was originally designed for, in which hundreds of texts were annotated by three annotators, TIARA has already been adopted by a second discourse annotation study, which uses it in the teaching of argumentation.

Keywords: annotation tool, discourse annotation, discourse relations, sentence reordering, teaching of argumentation

1. Introduction

There are many important aspects in writing such as grammar, mechanics, writing style and coherence (Grosz and Sidner, 1986; Lee and Webster, 2012). Out of these aspects, textual coherence, as an aspect of discourse structure, is extremely important. It concerns how sentences or other discourse units form a flow of meaning (Grosz and Sidner, 1986), and has been extensively analysed in the past (Jacobs et al., 1981; Grosz et al., 1995; Mann and Thompson, 1988; Wolf and Gibson, 2005; Garing, 2014). Our long-term goal is to automatically reorder sentences in argumentative essays so that their coherence is improved, as well as providing an explanation for the changes made. Therefore, the analysis of the order of sentence in original texts (which we call *first drafts*) and in their improved versions is necessary. A tool is needed to support these annotations, and we believe that it can be designed in such a way that it is useful for language and argument education in general.

Existing theory of text coherence stipulate that the order of sentences mirrors the intentional structure of discourse (Grosz and Sidner, 1986). In our situation, however, where we are dealing with often badly-structured first drafts, the intentional and surface structure are often not well-aligned. There are many ways how the isomorphism could be restored, leading to an improved draft – some of these ways are computationally and cognitively harder than others. Our idea, also voiced in (Putra et al., 2019), is that given the state of today's natural language processing (NLP) technologies, the fastest and cleanest way is by reordering sentences so that they fit the discourse structure properly, rather than some other, much harder method of repair. For example, should we find a *major claim*[1] in the middle of an essay, it should be moved to the beginning or the end of the essay, while respecting all the other aspects of how it is connected to the argument. Discourse structure provides not only a means to reorder sentences automatically, but it also explains the ways which the improved texts are better than the original ones. Therefore, the annotation of discourse structure is a prerequisite for a sentence order analysis. In its final application, an analysis of both the discourse structure and sentence order is also helpful for downstream tasks such as essay assessment (Al Khatib et al., 2017; Šnajder et al., 2019) and education (Iida and Tokunaga, 2014; Cullen et al., 2018; Matsumura and Sakamoto, 2019)

Since the texts we wish to annotate are argumentative, we employ the approach from the *argument mining* field. In the NLP community, *argument mining* is an emerging area[2] aimed to analyse argumentative texts from a multi-disciplinary perspective, including logic, rhetoric and language (Lippi and Torroni, 2016). It aims to provide structured data for computational models of argument and reasoning engines (Lippi and Torroni, 2016). Traditionally, the annotation of argumentative discourse structure consists of two main steps. The first of these is argument component detection. This step determines the boundaries of discourse units (*segmentation*) and differentiates them into argumentative or non-argumentative components (Stab and Gurevych, 2014; Lippi and Torroni, 2016). Argumentative components (ACs) can be further classified according to their rhetorical function in the discourse, e.g., into *major claim*, *claim* and *premise* (Stab and Gurevych, 2014).

The second step is argumentative discourse structure prediction, which links ACs and labels the links in order to form the structured representation of the text. All ACs must be connected to the structure, while non-ACs remain unconnected. Links can be directed (Stab and Gurevych, 2014) or undirected (Kirschner et al., 2015).

A new discourse annotation study often has specific, so far unserved needs, and we are no exception. Taking an empirical approach to the task of sentence reordering, we

[1] The concept behind a *major claim* has also variously been referred to as *main stance* or *conclusion* in the literature.

[2] The interest of the NLP community towards this is proven by Argument Mining Workshop series at ACL conferences. Readers may refer to Lippi and Torroni (2016) and Lytos et al. (2019) as an overview of this field.

need two kinds of annotation: (1) discourse structure and (2) sentence reordering. This will allow us to analyse and correlate the discourse characteristics of the first drafts and their improved versions. While there are publicly available argumentative essay corpora in which some aspect of discourse structure has been annotated (Peldszus and Stede, 2016; Stab and Gurevych, 2017), sentence reordering annotation is the problem: no ready-made corpora with sentence reordering annotation exists, at least not for our target domain of student essays, and no existing annotation tool supports sentence reordering, as we will show in Section 4. There are however some studies that explain how to order sentences to generate coherent texts in NLP applications, (Barzilay et al., 2002; Okazaki et al., 2004; Li and Jurafsky, 2017; Xu et al., 2019). However, they operate on different, non-argumentative texts such as news, and because they are statistical, they lack the explanatory power we need for educational purposes.

Although general-purpose annotation tools exist (Kaplan et al., 2010; Stenetorp et al., 2012), the modification of an existing annotation tool is still often not realistic due to many real-life constraints (e.g. the time involved in modification rather than fresh implementation, the availability of documentation and the entire redesign necessary when annotation needs diverge too much). Task-specific tools are often simply better for the annotation process and can lead to a better inter-annotator agreement (Sonntag and Stede, 2014).

This paper presents TIARA,[3] a new client-side tool for annotating discourse structure and sentence reordering to support our goal. We outline our annotation needs (i.e., target domain, annotation scheme) in Section 2, and describe how these requirements translate to design considerations and features of the tool in Section 3. Section 4 shows how it sits among other annotation tools. Section 5 outlines how TIARA benefits other domains such as education, i.e., in the teaching of argumentative writing. Finally, Section 6 concludes this paper and describes what can be improved in the future.

2. Annotation Needs

2.1. Target Domain

Our target texts are ICNALE[4] essays, short argumentative essays written in English by Asian college students studying English as a foreign language (EFL), of 200-300 words in length (Ishikawa, 2013). These essays are an ideal target for our study since EFL writings often require improvement in terms of organisation and coherence, although they are very different from the texts normally used for discourse annotation, where all parts of the text can be assumed to be coherently connected (Mann and Thompson, 1988). On top of having benefits for the NLP community (in the form of corpus), this paper also describes how discourse analysis of argumentative essays will be useful in the education domain (Section 5).

2.2. Discourse Structure Annotation Scheme

Discourse annotation aims to create a structured representation out of text, which explains how discourse units (e.g., sentences, clauses or clause-like segments) relate to each other and which role they play in the overall discourse (Mann and Thompson, 1988; Wolf and Gibson, 2005). A text is usually represented as a tree (Mann and Thompson, 1988; Stab and Gurevych, 2014) or graph (Wolf and Gibson, 2005; Sonntag and Stede, 2014; Kirschner et al., 2015). The discourse units are represented as nodes, and discourse relations hold between nodes (Wolf and Gibson, 2005; Kirschner et al., 2015; Stab and Gurevych, 2014) or between nodes and edges (for example, when challenging the acceptability of the inference between nodes–Peldszus and Stede (2016)).

In this study, we aim to annotate student essays at the sentence-granularity level and represent the (argumentative) discourse structure as a tree. This step is performed before the sentence reordering step. Following the workflow of *argument mining* as mentioned in Section 1, we first separate argumentative components (ACs) and non-ACs. Our approach differs from the previous *argument mining* work on student essays by Stab and Gurevych (2014) that a further classification of ACs into *main claim*, *claim* and *premise* is not necessary or appropriate for our analysis. Instead, our scheme allows these rhetorical functions to be inferred from the structure. It treats the *major claim* as the *root* of the structure. Furthermore, a sentence can act as both *claim* and *premise* at the same time, i.e., a *claim* also serve as *premise* for another *claim* (as has been noted by Stab and Gurevych (2014) themselves). Thus, differentiating both labels is rather difficult. The second step is to link ACs, in which we use both directed and undirected links in our scheme. In computation, we represent undirected links as directed, provided which nodes act as *source* and *target*. It means, our definition of "undirected" link is just a matter of visualisation (i.e., the presence of arrow) and the interpretation of the link label in question, but not of computation. This strategy is adopted to eliminate circular links which is not allowed in our scheme. Readers may refer to Putra et al. (2019) for a more complete explanation of our discourse structure annotation scheme.

2.3. Reordering and Text Repair

Our long-term goal, that motivates the development of this tool, is to reorder sentences to improve text coherence. If the sentences in the first draft are not already in the best order they could be, annotators are asked to arrange them into a more logically well-structured text. However, reordering may introduce errors in referring and connective expressions (Iida and Tokunaga, 2014). To revert these negative changes, annotators are allowed to perform some carefully limited, superficial repair of the text, such as replacing a pronoun with its referent noun phrase.

3. Annotation Tool TIARA

3.1. Design

There are several considerations that influence the TIARA's technical and visual design:

[3] TIARA stands for **TI**tech **AR**gument **A**nnotation. The tool and a detailed user manual are publicly available at `https://github.com/wiragotama/TIARA-annotationTool`.

[4] `http://language.sakura.ne.jp/icnale/`

(a) **Intuitive interface and visualisation**
We believe an annotation tool should provide an intuitive interface and visualisation. In the context of this study, it means the annotators must be able to read the sentences in linear order while viewing the discourse structure at the same time. The novelty of TIARA lies in this dual-view, which we believe provides an important aspect of global overview to the annotators, who operate by making local changes.

(b) **Annotation consistency checking**
An annotation tool ideally prevents violations of its annotation scheme, such as illogical annotations (e.g., connecting a sentence to itself). Compliance guarantees offered by annotation tools are attractive: annotators can follow their natural workflow without having to worry about doing something wrong or having to perform separate checks. Project owners are served too by this as they do not have to ask the annotators for a post-hoc repair of the annotations. TIARA checks in real-time whether the annotation violates any constraint of the annotation scheme, and warns the annotator when it does.

(c) **Annotation tracking**
Tracking changes and actions performed by the annotators is important since it provides information about annotation behaviour. It is also valuable for troubleshooting annotation schemes, because it can identify the parts which cause most confusion or re-annotation. For example, we know that label `X` and `Y` are potentially confusing when annotators often change the links labelled with `X` to `Y` (and vice versa). TIARA does this by logging the actions performed by annotators in each annotation-file.

(d) **Ease of use, installation and deployment**
Ease of use and installation for annotators is often prioritised for annotation design, but we believe that deployment is equally important. Not every project owner is tech-savvy; for them, an annotation tool that is hard to deploy is practically unusable. In contrast, tools that are usable without deployment and may run at client-side, such as `EasyTree`[5] (Little and Tratz, 2016), are able to reach and help as many potential users as possible, including those who have no knowledge in programming. Therefore, TIARA shares the same principle. Users only need a web browser and the TIARA package. This tool is written in `javascript`, `html` and `css`. We use `JsPlumb`[6] and `Treant-js`[7] as the visualisation libraries.

We understand that the necessity of deployment (server-side) is often coupled with file and/or annotation management features (Yimam et al., 2013), and this is important in a large annotation project. While the current version of TIARA does not actively support such annotation management yet, we plan to do so in future TIARA versions.

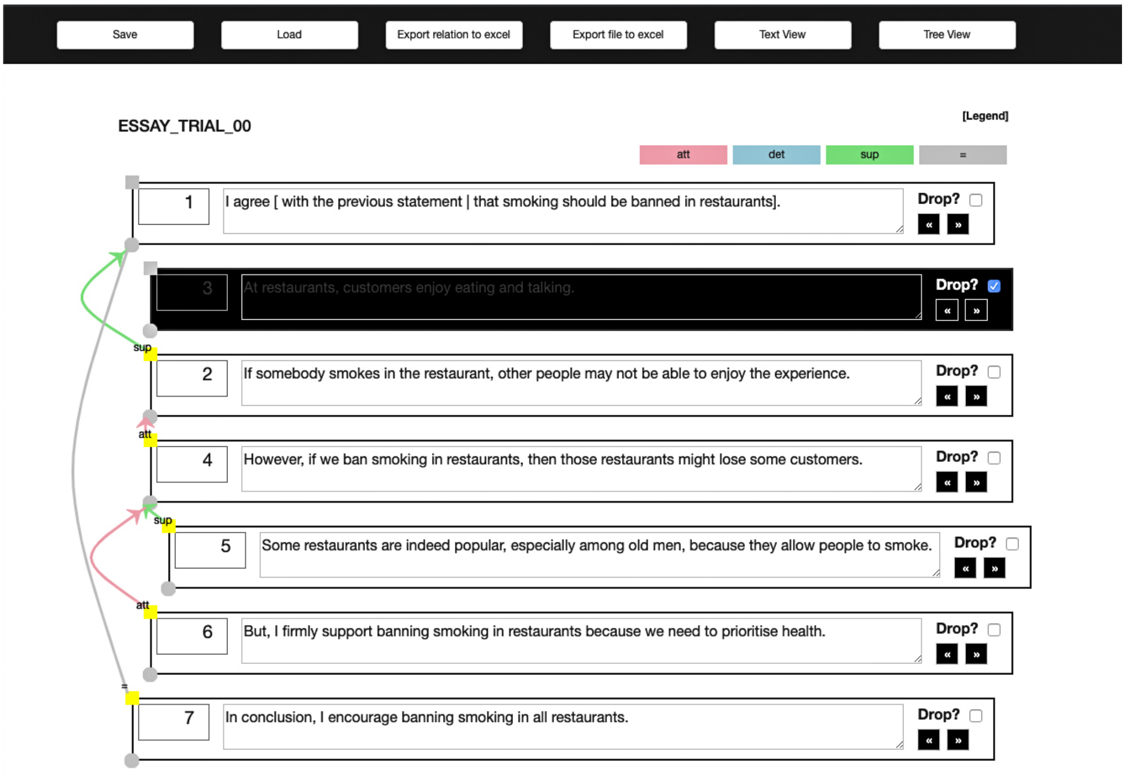
(e) **Customisability**
An annotation tool must be flexible in order to accommodate a wide variety of annotation tasks (Kaplan et al., 2010). This is especially important in the early stage of an annotation study when the project goal and annotation scheme might frequently change. We adhere to the principle that users should never have to touch the main code at all; they should be able to customise the annotation tool easily in some other way. Similar to `BRAT` (Stenetorp et al., 2012), the annotation scheme of TIARA can be changed by editing a configuration file, and thus not part of the main code. At the beginning of an annotation project, the project owners should define this configuration script. We chose this approach over the alternative, a user interface provided by the tool, e.g., as in `RSTTool` (Mann and Thompson, 1988), since `javascript` should not modify local files (for security reason).

3.2. Text and Tree View

TIARA's **text view** (cf. Figure 1) is the view in which annotation is performed. Annotators can read the discourse units sequentially while viewing at the annotated discourse structure at the same time. The interface in the text view is split into two parts, menu navigation on top and work area at the bottom. After loading a text file, the contents are shown in the work area. Each discourse unit appears framed in a box (denoting node), numbered ("ID") according to its original order in the input text. Coloured (defined by user) links depict the annotated relations and their labels. For example in Figure 1, `att` (*attack*), `det` (*detail*) and `sup` (*support*) are directed while "=" (*restatement*) is an undirected link. Text repair is present in sentence (1). Note that sentence (3) is "dropped" (i.e., deemed non-AC and blacked-out), and users cannot establish a link to or from sentence (3). Sentence (2) and (3) are swapped in position (reordering). Sentence (3) to (6) are indented to the right to quickly simulate the hierarchical-nature of the discourse structure, e.g., denoting sub-arguments. The earliest working version of TIARA (only "text view) was deployed to eight students of a graduate NLP course in summer 2018, and they commented its use intuitive.

While the text view can be used to illustrate a hierarchical-like structure of the discourse, we think that it is not enough for the analysis of the whole discourse structure. Another alternative view offered by TIARA is the **tree view**, which illustrates the shape of the discourse structure as a whole. Figure 2 shows the tree view of the annotation in Figure 1. The text view emphasises the text analysis on logical sequencing while the tree view emphasises the analysis of the overall discourse structure. Annotators annotate in the text view and then verify their annotation in the tree view, and they can freely switch between both while annotating. We believe that providing the tree view enhances the annotation experience, and therefore, the annotation quality. Annotators may also fold/un-fold subtrees in this view, which is useful for analysing longer texts as it prevents annotators from feeling overwhelmed by too much content. Users can download the hierarchical visualisation by clicking the "capture image" button, e.g., to be used in presentation.

[5] `https://github.com/alexalittle/easytree`
[6] `https://jsplumbtoolkit.com`
[7] `https://fperucic.github.io/treant-js/`



ESSAY_TRIAL_00

1 I agree [with the previous statement | that smoking should be banned in restaurants]. Drop? □ « »

3 At restaurants, customers enjoy eating and talking. Drop? ☑ « »

2 If somebody smokes in the restaurant, other people may not be able to enjoy the experience. Drop? □ « »

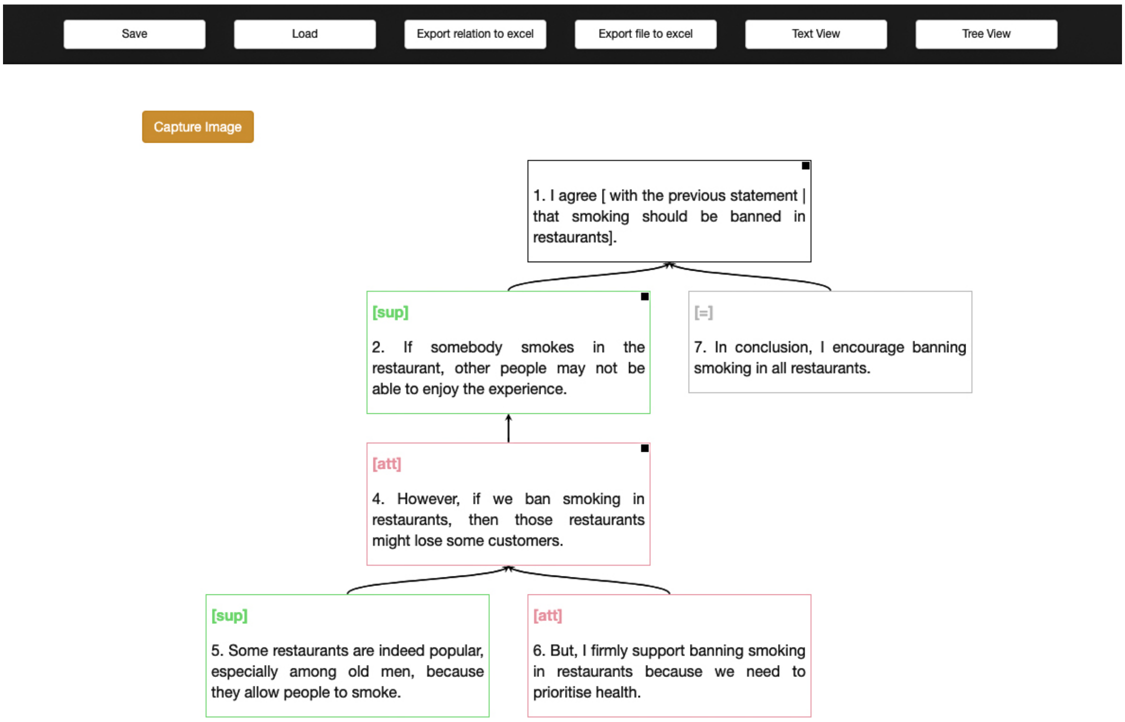
4 However, if we ban smoking in restaurants, then those restaurants might lose some customers. Drop? □ « »

5 Some restaurants are indeed popular, especially among old men, because they allow people to smoke. Drop? □ « »

6 But, I firmly support banning smoking in restaurants because we need to prioritise health. Drop? □ « »

7 In conclusion, I encourage banning smoking in all restaurants. Drop? □ « »

Figure 1: A screenshot illustrating TIARA text view.



1. I agree [with the previous statement | that smoking should be banned in restaurants].

[sup]
2. If somebody smokes in the restaurant, other people may not be able to enjoy the experience.

[=]
7. In conclusion, I encourage banning smoking in all restaurants.

[att]
4. However, if we ban smoking in restaurants, then those restaurants might lose some customers.

[sup]
5. Some restaurants are indeed popular, especially among old men, because they allow people to smoke.

[att]
6. But, I firmly support banning smoking in restaurants because we need to prioritise health.

Figure 2: A screenshot illustrating TIARA's tree view for the annotation in Figure 1. Users may fold and un-fold a subtree by clicking the rectangular button on the top-right corner of its root.

We are not the first to offer both tree (structural) and text view in our annotation tool; DiGAT (Kirschner et al., 2015) also did this. But we switch between the two, instead of presenting them simultaneously, and will argue in Section 4 why this is advantageous.

3.3. Functionality

In detail, TIARA offers the following functionality:

(a) **Loading a file**
Annotators can load an unannotated text file, where

the discourse units must have previously already been separated by a newline. TIARA can also load an annotated file in its own internal-format.

(b) **Linking and link labelling**
Annotators link ACs by dragging an arrow from the rectangular endpoint of the source to the circular endpoint of the target unit, and TIARA shows a dialogue box for choosing the link label. Annotators may delete or change the link label by clicking the established link in question.

(c) **Reordering**
Annotators may move the position of discourse unit boxes by drag and drop operation of the box.

(d) **Text revision**
TIARA allows annotators to edit the text inside boxes. To track changes, some notation can be employed. For example, annotators may modify parts of text if needed in "[*original expression* | *revised expression*]" notation. An illustration is shown in sentence (1) of Figure 1.

(e) **Dropping discourse units**
TIARA supports the differentiation between ACs and non-ACs. ACs are connected to form the argumentative discourse structure while non-ACs are not connected to the structure (cf. Section 2). Annotators mark non-ACs by checking the "drop" checkbox located at the right-hand side of each box. When ticked, the box is blacked-out and annotators cannot establish a relation to or from the non-AC units. Annotators may uncheck the checkbox to revert back.

(f) **Indentation**
TIARA supports indentation of discourse units by clicking the indentation buttons at the right-hand side of boxes (under the "drop" checkbox). This is useful to quickly visualise the hierarchical structure of the discourse (De Kuthy et al., 2018).

(g) **Customisation**
Users may customise the relation types, labels and colours by modifying an external configuration script. They can also disable or enable certain functions. Figure 3 shows a configuration script example.

```
var disableDropping = false;
var disableReordering = false;
var allowIntermediarySave = false;
var availableRels = [”support”, ”attack”]
var relColors = [”lightgreen”, ”lightred”]
var relDirections = [true, true]
```

Figure 3: Example of TIARA's configuration script (written in `javascript`).

(h) **Saving annotation**
Users may save the annotation in TIARA's internal format using the "save" menu. Annotators may then load and/or modify the annotation. On top of that, TIARA offers exporting the annotation into excel-friendly (spreadsheet) format as follows.

- "Export relation to excel" menu. TIARA extracts information on relations of all combinations of discourse units which is useful for measuring inter-annotator agreement (Kirschner et al., 2015).
- "Export file to excel" converts the annotation as "`.tsv`" file in which the row contains <essay code, unit ID, text, corresponding target unit ID, relation label, dropping flag> information.

(i) **Logging**
On top of visualisation features previously mentioned, TIARA logs the actions (and their timestamp) performed by annotators in background. The log information is stored at each saved annotation-file (TIARA's internal format). Therefore, users know the history of each file. This feature is also useful for the analysis of annotation behaviour.

4. Related Work

In the NLP community, many annotation tools have been developed. Among them, `BRAT`[8] (Stenetorp et al., 2012) is relatively popular as a general-purpose linguistic annotation tool. It offers annotation visualisation and collaboration features. `BRAT` also has been used for argument mining in the study of Stab and Gurevych (2017). Built in the same spirit as `BRAT`, `WebAnno`[9] (Yimam et al., 2013) offers additional management and monitoring features. These tools are easy to customise, offering the flexibility to accomodate a wide range of annotation tasks. However, `BRAT` and `WebAnno` were originally designed for morphological, syntactic and semantic annotations (i.e., rather local word or phrase-level annotation). While they support link display and could thus theoretically be used for discourse annotation, the visual display of links appears as drawn directly on top of text. This style of display has already been identified by others as a source of confusion for argumentation and discourse annotation projects (Kirschner et al., 2015). `PDTB annotator`[10] (Prasad et al., 2008) also falls into the class of annotation tools designed for local relations. When it comes to the display of larger-scale hierarchical or graphical structure of discourse, this falls entirely outside the purview of these tools.

Annotation tools which are specifically aimed at visualising larger-scale and more global discourse structure have also been developed, e.g., `RSTTool`[11] (Mann and Thompson, 1988), `TreeAnno`[12] (De Kuthy et al., 2018), `OVA`[13] (Janier et al., 2014), `DiGAT`[14] (Kirschner et al.,

[8] `https://brat.nlplab.org/`

[9] `https://webanno.github.io/webanno/`

[10] `https://www.cis.upenn.edu/~pdtb/pdtb3-tools.shtml`

[11] `http://www.wagsoft.com/RSTTool/`

[12] `https://github.com/nilsreiter/treeanno`

[13] `http://www.arg-tech.org/index.php/ova/`

[14] `https://github.com/UKPLab/argmin2015-DiGAT`

	Feature	GraPat	DiGat	OVA	TreeAnno	RSTTool	TIARA
1.	Discourse structure	Graph	Graph	Graph	Tree	Tree	Tree
2.	Segmentation (arbitrary)			✓		✓	
3.	Detecting AC and non-AC	✓	✓	✓			✓
4.	AC labelling	✓					
5.	Linking	✓	✓	✓	✓	✓	✓
6.	Link labelling/polarity	✓	✓	✓		✓	✓
7.	Structure visualisation	✓	✓	✓	✓	✓	✓
8.	Ease of scheme customisation			✓		✓	✓
9.	Discourse unit reordering						✓
10.	Text editing			✓		✓	✓

Table 1: Comparison of features in TIARA and discourse annotation tools in terms of *argument mining* tasks (1–7) and our additional needs (8–10).

2015), and GraPat[15] (Sonntag and Stede, 2014), but neither fulfilled our desiderata. Table 1 shows in detail how TIARA is situated in terms of its visual features amongst other annotation tools, in particular with respect to its support of argument mining tasks (1–7) and our additional needs (8–10), which of course it is designed to fulfill.

RSTTool is a strong competitor of TIARA in terms of features implemented and visual elegance. However, it only allows RST-style annotation, i.e., only two adjacent units can be attached, in contrast to our scheme that allows relation between arbitrary units. It also excludes annotation of any text that contains non-AC material, as all units have to be connected in the RST scheme. Meanwhile, our scheme allows us to keep and isolate non-ACs.

Similar to TIARA, TreeAnno and RSTTool allow a general tree structure of discourse. TreeAnno is easy to use, but falls short in number of features implemented. While the visualisation of hierarchy via node indentation in TreeAnno illustrates the discourse structure (to some extent), it does not show the links between discourse units.

GraPat, DiGAT and OVA offer features that support discourse annotation tasks, and they assume a graph structure of texts. However, GraPAT and DiGAT require considerable effort to customise the annotation scheme, and all three suffer from an additional problem. While any tree structure is by definition also a graph (with some constraints), these tools cannot ensure annotation compliance to the specific tree structure we assumed in our annotation (cf. Section 3).

While it is not suited to our specific task, GraPat is the only tool among the surveyed tools that supports AC labelling (as *proponent* or *opponent*); it thus is at an advantage over other tools for (argumentative) discourse annotation. It shows the text for the corresponding node (box); thus, annotators can view the text and the discourse structure at the same time. However, Kirschner et al. (2015) argued that the visual in GraPat might be confusing for texts with multiple long sentences. Their solution to the problem, DiGAT, splits the display into a text and structure view, a design that both OVA and TIARA have followed. DiGAT and OVA present both views simultaneously, but in DiGAT's structure view, the text corresponding to a node is not shown, and text and nodes are instead associated by IDs. We think it is essential to see both text and structure in the same visual (as both OVA and TIARA offer), because it is cognitively expensive to have to synthesise two visuals in one's mind by switching between left and right side of the screen, as is necessary in DiGAT. Our dual-view is also advantageous because it allows annotators to focus on one type of analysis, i.e., either logical sequencing and the overall discourse structure, at one time (cf. Section 3).

Among the annotation tools that assume graph structure, we consider OVA our strongest competitor. It offers almost all features needed for our research, with the exception of the discourse unit reordering feature, since it was not designed for the analysis of logical sequencing.

Overall, there is no one-for-all discourse annotation tool, but TIARA with its middle-ground visual solution is efficient for annotation and is a strong general tool for relation-focused discourse annotation. In particular, because it provides versatile visualisation for representing structure (the dual-view, indentation, collapsible features), annotators can choose the method that works best for them.

Despite its advantages, TIARA does not yet offer all the features we would like it to have. Its lack of segmentation and AC labelling features may be an obstacle to its use in some more complex discourse and argumentation annotation studies. If the annotation happens at the sentence-level (as here), segmentation features may not be critical: excellent tools for breaking text into sentences are available, e.g., nltk.[16] When it comes to enabling the full *argument mining* pipeline for exhaustive discourse annotation, the ability to label arbitrary AC becomes more important and needs to be solved before TIARA can be applied to such projects.

TIARA was originally designed for the annotation of relations between discourse units and sentence reordering in monologue text, as motivated in Section 1. Four link labels

[15] https://github.com/discourse-lab/GraPat

[16] https://www.nltk.org/

were used (three directed and one undirected). Three annotators used an early version of TIARA to annotate a total of around 450 short (200-300 words) argumentative essays from the ICNALE corpus. Putra et al. (2019) showed that the annotation time compared to using a general-purpose tool was reduced from 40 to 25 minutes. Since then, TIARA has found another use case, which is described in the next section.

5. Possible Application of TIARA in Teaching Environment

Teaching students to argue well is difficult, because so many constraints need to be satisfied for an argument to be convincing, i.e., the text needs to contain the desired argumentative elements. By means, the ideas should be clearly stated, connected to each other, and supported by reasons. They should also be logically developed in a particular sequencing such as time or importance, and accompanied by appropriate discourse markers. Only then can the writing ultimately communicate the desired ideas as a whole (Toulmin, 2003; Jacobs et al., 1981; Reed and Wells, 2007; Peldszus and Stede, 2013; Matsumura and Sakamoto, 2019). Despite its benefits, the analysis of the implicit logical structure of argumentative texts is rarely taught explicitly in universities (Cullen et al., 2018). Teaching how to argue can be supported by the construction of mental models of the structure, which require checks for completeness (*are all the parts there?*) and for coherence (*do relations among parts make sense?*). Teaching this with purely symbolic means (i.e., the use of words) is less efficient than using visual explanations, e.g., in the form of diagrams. Visual information can also act as an intuitive platform for inference (Bobek and Tversky, 2016). For these reasons, visual information has been widely used to promote effective communication.

Cullen et al. (2018) assessed the effect that argument visualisation has on analytical reasoning and argument understanding. They performed a controlled study where one group of students were taught how to use automatic argument visualisation (`MindMup`[17]), whereas the control group was taught traditionally. The targeted texts were contemporary academic texts. When measuring the improvement of both groups in logical reasoning test before and after the teaching sessions, they found a larger increase in the visually-taught group than in the control group, suggesting that learning how to visualise arguments led to improvements in students' analytical-reasoning skills.

Beyond the benefits of checking for completeness, coherence and inference, argument visualisation also helps in conveying what students understand about the texts. The graphical visualisations of an argument can be shared with instructors, allowing students to easily discuss their interpretations with the instructors, and allowing instructors to quickly identify gaps in students' understanding of the reading material, and hence suggest ways for improving future texts by the students. This feedback should enable students to write more accurate and effectively structured essays (Cullen et al., 2018).

Matsumura and Sakamoto (2019) studied in detail how the analysis of argument visualisation is helpful for instructors' diagnostic assessment. They used TIARA to investigate organisation problems in texts written by Japanese EFL learners. They defined six types of directed link: five inspired by Toulmin's argument model (Toulmin, 2003), and added a special link label "?" for feedback to students. Two annotators annotated 15 short (∼140 words) argumentative essays written by 5 Japanese EFL learners in a classroom setting. Their analysis aims to assess whether textual segments written by their ESL students are logically connected and relevant to each other, and whether material presented as if it was supporting some claim was indeed relevant to the target claim. The annotation was then used to provide evidence-based feedback to the students using the visualisation strategies available in TIARA. The diagnostic assessment should enable instructors to make inferences about learners' strengths and weaknesses in the skills being taught (Jang and Wagner, 2013); in this case, Matsumura and Sakamoto (2019) found a remarkable difference between the discourse structures in the high-scored essays (where they typically form a balanced tree) to those in low-scored ones (where the the overall structure tends to be flat and linear, and isolated elements occur). Visual explanation enables instructors to better explain students' mistakes. In addition, it enables students to comprehend and accept why certain ways of writing are considered logically weak, and thus, attract poor scores. In the long run, observations like the ones gained by this study can be used to formulate hypotheses about better assessment and teaching of argumentative writing.

Beyond this, TIARA and other discourse annotation tools may play an even bigger role in the education domain, especially in the teaching of argument. Apart from learning-to-read (understanding the argumentative discourse structure), we believe that TIARA can be useful for learning-to-write as well. In a classroom setting, students could write argumentative essays and simultaneously draw the intended structures on TIARA in parallel, allowing instructors to interactively and quickly point out and address those student mistakes that are visible in TIARA's visualisation. Instructors can then suggest improvements in the overall discourse flow (i.e., by reordering sentences) and in the textual realisation (e.g., by editing discourse markers directly in TIARA). Thereby, TIARA should enhance the process of student-instructor communication and feedback.

6. Conclusion

This paper presents TIARA, a new web-based annotation tool which is specifically aimed at annotating discourse relations and sentence reordering. It offers versatile visualisation to enhance discourse structure annotation. The tool is easily customisable via a configuration script, and is a strong competitor among other discourse annotation tools. It has been used to annotate hundreds of texts in discourse annotation studies and has also proved its usefulness for education, i.e., the analysis and construction of arguments.

Future versions of TIARA will provide segmentation and argumentative-component labelling features, and will im-

[17] `https://www.mindmup.com/tutorials/argument-visualization.html`

prove the visualisation by easy comparisons of original and edited text. In addition, we plan to allow relations between nodes and edges.

On the purely technical side, the current version of TIARA is appropriate for relatively small-scale projects, while for bigger and more complex projects, additional features would improve the experience substantially. We therefore consider the provision of two parallel versions of TIARA: a light-weight TIARA versus one with more extensive management, collaboration and monitoring features.

7. Acknowledgements

This work was supported by Tokyo Tech World Research Hub Initiative (WRHI) and Support Centre for Advanced Telecommunication Technology Research. We also would like to thank anonymous reviewers for their useful comments.

8. References

Al Khatib, K., Wachsmuth, H., Hagen, M., and Stein, B. (2017). Patterns of argumentation strategies across topics. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1351–1357.

Barzilay, R., Elhadad, N., and Mckeown, K. R. (2002). Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:35–55.

Bobek, E. and Tversky, B. (2016). Creating visual explanations improve learning. *Cognitive Research: Principles and Implications*, 1(1).

Cullen, S., Fan, J., van der Brugge, E., and Elga, A. (2018). Improving analytical reasoning and argument understanding: a quasi-experimental field study of argument visualization. *NPJ Science of Learning*, 3(21).

De Kuthy, K., Reiter, N., and Riester, A. (2018). QUD-based annotation of discourse structure and information structure: Tool and evaluation. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 1932–1938.

Garing, A. G. (2014). Coherence in argumentative essays of first year college of liberal arts students at de la salle university. *DLSU Research Congress*.

Grosz, B. J. and Sidner, C. L. (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204.

Grosz, B. J., Weinstein, S., and Joshi, A. K. (1995). Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

Iida, R. and Tokunaga, T. (2014). Building a corpus of manually revised texts from discourse perspective. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 936–941.

Ishikawa, S. (2013). The icnale and sophisticated contrastive interlanguage analysis of asian learners of english. *In S. Ishikawa (Ed.), Learner corpus studies in Asia and the world*, pages 91–118.

Jacobs, H. L., Zinkgraf, S. A., Wormuth, D. R., Harfiel, V. F., and Hughey, J. B. (1981). *Testing ESL composition: a practical approach*. Newbury House, Rowley, Massachusetts.

Jang, E. E. and Wagner, M. (2013). Diagnostic feedback in the classroom. In Antony Kunnan, editor, *Companion to language assessment*, volume 2, chapter 6. Wiley-Blackwell, Oxford.

Janier, M., Lawrence, J., and Reed, C. (2014). Ova+: an argument analysis interface. In *International conference on computational models of argument (COMMA)*, pages 463–464.

Kaplan, D., Iida, R., and Tokunaga, T. (2010). Annotation process management revisited. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 3654 – 3661.

Kirschner, C., Eckle-Kohler, J., and Gurevych, I. (2015). Linking the thoughts: Analysis of argumentation structures in scientific publications. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 1–11.

Lee, J. and Webster, J. (2012). A corpus of textual revisions in second language writing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 248–252.

Li, J. and Jurafsky, D. (2017). Neural net models of open-domain discourse coherence. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 198–209.

Lippi, M. and Torroni, P. (2016). Argumentation mining: State of the art and emerging trends. *ACM Transaction of Internet Technology*, 16(2):10:1–10:25.

Little, A. and Tratz, S. (2016). EasyTree: A graphical tool for dependency tree annotation. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 2343–2347.

Lytos, A., Langkas, T., Sarigiannidas, P., and Bontcheva, K. (2019). The evolution of argument bining: From models to social media and emerging tools. *Information Processing and Management*, 56.

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

Matsumura, K. and Sakamoto, K. (2019). The relevance of reasoning in l2 argumentative writings of japanese efl students: A mixed methods approach. In *Proceedings of the Mixed Methods International Research Association Asia Regional Conference (MMIRA)*.

Okazaki, N., Matsuo, Y., and Ishizuka, M. (2004). Improving chronological sentence ordering by precedence relation. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 750 – 756.

Peldszus, A. and Stede, M. (2013). From argument diagrams to argumentation mining in texts: A survey. *International Journal of Cognitive Informatics and Natural Intelligence*, 7(1):1–31.

Peldszus, A. and Stede, M. (2016). An annotated corpus of argumentative microtexts. In *Argumentation and Reasoned Action - Proceedings of the 1st European Conference on Argumentation, Lisbon, 2015.*

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., and Webber, B. (2008). The Penn discourse TreeBank 2.0. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

Putra, J. W. G., Teufel, S., and Tokunaga, T. (2019). An argument annotation scheme for the repair of student essays by sentence reordering. In *Proceedings of Annual Meeting of the Association for Natural Language Processing Japan*, pages 546–549.

Reed, C. and Wells, S. (2007). Dialogical argument as an interface to complex debates. *IEEE Intelligent Systems*, 22(6):60–65.

Šnajder, J., Sladoljev-Agejev, T., and Kolić Vehovec, S. (2019). Analysing rhetorical structure as a key feature of summary coherence. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 46–51.

Sonntag, J. and Stede, M. (2014). GraPAT: a tool for graph annotations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 4147 – 4151.

Stab, C. and Gurevych, I. (2014). Annotating argument components and relations in persuasive essays. In *Proceedings of International Conference on Computational Linguistics (COLING)*, pages 1501–1510.

Stab, C. and Gurevych, I. (2017). Parsing argumentation structures in persuasive essays. *Computational Linguistics*, 43(3):619–659.

Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S., and Tsujii, J. (2012). brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 102–107.

Toulmin, S. E. (2003). *The uses of argument*. Cambridge University Press, 2 edition.

Wolf, F. and Gibson, E. (2005). Representing discourse coherence: A corpus-based study. *Computational Linguistics*, 31(2):249–288.

Xu, P., Saghir, H., Kang, J. S., Long, T., Bose, A. J., Cao, Y., and Cheung, J. C. K. (2019). A cross-domain transferable neural coherence model. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 678–687, Florence, Italy.

Yimam, S. M., Gurevych, I., Eckart de Castilho, R., and Biemann, C. (2013). WebAnno: A flexible, web-based and visually supported system for distributed annotations. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL): System Demonstrations*, pages 1–6.