



Original software publication

ARGAEL: ARGument Annotation and Evaluation tool

Andrés Segura-Tinoco*, Iván Cantador

Escuela Politécnica Superior, Universidad Autónoma de Madrid, Madrid, Spain



ARTICLE INFO

Article history:

Received 23 November 2022

Received in revised form 21 April 2023

Accepted 10 May 2023

Keywords:

Text annotation

Annotation software

Annotation evaluation

Argument extraction

Argument mining

ABSTRACT

Argument mining aims to automatically extract structured argumentative information existing in natural language text, and it is commonly performed by machine and deep learning models that require accurately and meaningfully labeled corpora. In this paper, we present ARGAE, an open-source desktop tool designed to provide flexibility, effectiveness and efficiency on the manual annotation of related arguments in text documents. ARGAE supports the use of rich, configurable argument models, the labeling of argument components and relations, and the assessment of argument annotations from multiple people, being suitable for large-scale, collaborative argument annotation processes.

© 2023 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

Code metadata

Current code version

Permanent link to code/repository used for this code version

Permanent link to Reproducible Capsule

Legal Code License

Code versioning system used

Software code languages, tools, and services used

Compilation requirements, operating environments and dependencies

If available, link to developer documentation/manual

Support email for questions

v1.0

<https://github.com/ElsevierSoftwareX/SOFTX-D-22-00381><https://github.com/argrecsys/argael/tree/main/exec>

Apache License 2.0

git

Java

JDK 17

<https://argrecsys.github.io/argael/docs/>andres.segurat@uam.es

1. Motivation and significance

Emerged from the intersection of computational linguistics and natural language processing research fields in the late 2000s, argument mining (AM) is a research area whose ultimate goal is the automatic, computer-aided extraction of argumentative structures from natural language text [1].

In the area, much progress has been done on the formulation of argument models, the definition of target tasks – such as identifying argumentative fragments, classifying argument components (e.g., *claims* and *premises*) and recognizing argument relations (e.g., *support* and *attack*) –, and the creation of annotated corpora for different domains and languages [2].

With respect to algorithmic solutions to AM, two mainstream approaches have been consolidated. The first approach consists of machine learning models that are built from linguistic features

extracted from input sentences [1,3], and the second approach extends to deep learning models that are based on word embeddings of the textual content [4,5]. Since both approaches require large corpora previously labeled, there is an increasing need for easy-to-use tools that assist in the complex and costly processes of manual annotation and evaluation of argumentative information in a given text [2].

There are popular general-purpose annotation tools, such as GATE [6] and BRAT [7], which can be used to generate labeled argumentative corpora [2]. Unfortunately, these tools tend to present limitations or require adaptations to deal with the underlying argument model and argumentative annotation methodology. For this reason, argument mining-oriented tools, such as Araucaria [8] and Carneades [9], have been developed. Most of these tools, however, make use of fixed, simple argument models, and do not support the collaborative evaluation of annotations of large texts.

Addressing these limitations, which can potentially impact on the argument annotation quality and on the annotation process

* Corresponding author.

E-mail addresses: andres.segurat@uam.es (Andrés Segura-Tinoco), ivan.cantador@uam.es (Iván Cantador).

effectiveness and efficiency, in this paper, we present ARGAE¹ (which stands for ARGument Annotation and Evaluation tool), an open-source Java desktop application designed to provide flexibility, effectiveness and efficiency on the manual labeling and assessing of argumentative information at scale.

ARGAEL presents a number of novelties and advantages. It allows using configurable, user-defined argument models; it supports both the generation and evaluation of annotations by multiple users; and it provides a visual interface designed to overcome particular needs and difficulties of argumentative annotation processes, such as the exploration of the (textual) context in which an argument is given.

The remainder of the paper is structured as follows. In Section 2, we survey previous work on general-purpose and argument mining-oriented annotation tools. Next, in Section 3, we describe the architecture and functionalities of ARGAE¹. As a proof of concept of the tool, in Section 4, we present an illustrative example in which ARGAE¹ is used to define a novel argument model, to create a new argumentation corpus according to such model, and to conduct a preliminary user study on the perceived usability of the tool. Finally, in Section 5, we conclude by summarizing benefits and potential future extensions of ARGAE¹.

2. Related work

The manual annotation of text is a crucial activity for the majority of natural language processing tasks and applications [10] – such as part of speech tagging, dependency parsing, named entity recognition, and opinion mining –, since they are addressed by models that are built from (very) large labeled corpora.

In general, the generation of a labeled corpus is complex and costly, and requires the use of software tools that assist the user not only in delimiting and categorizing fragments of texts, but also in validating and assessing already generated annotations. In this sense, the creation and evaluation of text annotations are processes that may have a strong collaborative component, which makes some tools inappropriate to certain cases. Argument mining is one of them.

General-purpose annotation tools for natural language processing could be used to label argumentative text fragments. GATE [6,11] is one of the most consolidated and popular of such tools. Originally developed at the University of Sheffield beginning in 1995, it is now used worldwide by a very large community in both academia and industry. It includes a desktop client for developers, and a collaborative annotation web platform called GATE Teamware [12]. Although it allows creating general annotations for the task at hand, it is focused on information extraction. AGTK [13] and BRAT [7], on the other hand, are highly customized tools that put their focus on the annotation of entities, relations and events, which are closer to argumentative structures. Finally, recent research has been conducted to develop semantic annotation tools that enable integrating knowledge bases for concept and fact linking (e.g., INCEPTION [14] and TextAnnotator [15]), performing pre-annotation NLP tasks (e.g., TextAnnotator [15]) and hierarchical, complex annotation tasks (e.g., HUMAN [16]), and querying annotation repositories (e.g., ET [17,18]). In this context, commercial tools have also been developed. Doccano,² Prodigy,³ UBIAI,⁴ and LightTag⁵ are representative examples. In all cases, general-purpose annotation tools require to be configured or even adapted to deal with the argument model and

(collaborative) argumentative annotation methodology at hand, tending to present limitations in this respect.

In addition to general-purpose annotation tools, there exist *argument mining-oriented tools*. Araucaria [8] is one of the first tools aimed to facilitate the creation of fine-grained argumentative structures. It provides a desktop interface that allows the user to make (relatively small) argument diagrams with graph structures. Similarly, OVA+ [19] is an online tool that allows annotating a text with an argument graph via a graphical user interface. It handles the Argument Interchange Format, AIF [20], and has been recently extended with the Argument Scheme Key, ASK, a dichotomous identification method that allows the user to specify an argument type through a series of disjunctive choices [21]. These two tools, however, were not designed to evaluate the quality of annotations made by different users. Carneades [9], on the other hand, is a system focused on the automatic and formal validation of argumentative information and structures, but it is not oriented to support large-scale and collaborative evaluation of annotations. Much more recently developed, VIANA [22] is a web-based tool for interactive, visual annotation of argumentation that augments the manual annotation process by automatically suggesting the text fragments to annotate next. As Araucaria and OVA+, VIANA was not designed for the evaluation of already generated annotations. Last, but not least, TARGER [23] is a web application designed for non-expert users, which automatically extracts potential argumentative units from free text given in real time, using previously trained neural models. This system could be seen as an argument retrieval tool, rather than an argument annotation tool. The above tools work with fixed, non-customizable argument models, and do not support the collaborative evaluation of annotations. Besides, in general, the tools that use a graph-based representation of the argumentation annotations may make difficult the visualization of large argumentative structures.

Targeting high-level argumentation scenarios, there are *tools aimed to annotate discussions and debates*. ART [24] allows users to easily copy and store text fragments, and relate them through formal argument structures. Similarly, ArgScheme [25] provides a user-friendly interface to label support-attack relations between text fragments with predefined argument schemas. Finally, TIARA [26,27] is a web-based tool aimed to deal with the visual complexity of the annotation process. Its versatile visualization, however, is limited to relatively small-scale projects. Some other tools are domain- or application-specific. For instance, in [28], the authors present an annotation tool focused on (online) discussion threads, allowing the creation of both inner- and inter-post relations. MARDY [29] is an environment for relational annotation of political debates. It allows annotating demands raised by politicians and other actors, with claim and actor spans, relations and polarities. Finally, LIDA [30] is an annotation tool designed specifically for conversation (dialogue) data. It supports the integration of machine learning models as annotation recommenders, and includes a dedicated interface to resolve inter-annotator disagreements. All these tools are focused on discourse-level annotations, offering several advantages in this respect, but are not designed to facilitate the generation and evaluation of fine-grained argument annotations.

Motivated by the need to jointly address the collaborative generation and evaluation of annotations of argumentative information in large text documents, as well as to enable the use of personalized, fine-grained argument models and the visualization of the conversation/debate context in which arguments are given, we have developed ARGAE¹, a flexible, open-source tool that we describe in the following sections.

¹ ARGAE¹ tool, <https://github.com/argrecsys/argael>.

² <https://doccano.herokuapp.com>

³ <https://prodigy.ai>

⁴ <https://ubiai.tools>

⁵ <https://www.lighttag.io>

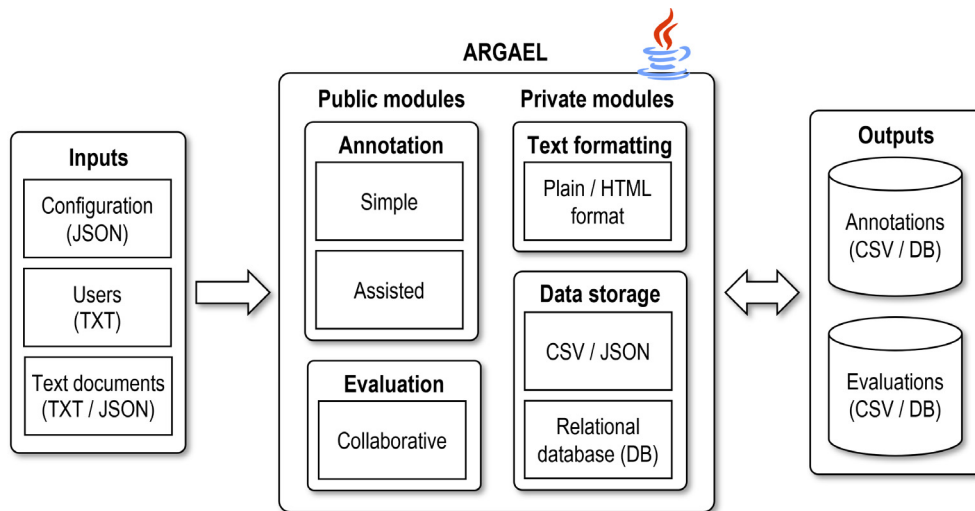


Fig. 1. Overview of ARGAE architecture.

3. Software description

3.1. Software architecture

In this section, we provide an overview of ARGAE architecture, which is depicted in Fig. 1. The tool is composed of several internal modules (explained in Section 3.1.1), receives a number of files as inputs (Section 3.1.2), and allows the user to generate annotations and their evaluations as outputs (Section 3.1.3). The description of these components will facilitate the understanding of the functionalities, explained afterward.

3.1.1. Internal modules

As shown in the middle block of Fig. 1, from a software modeling point of view, ARGAE has two public modules for argument annotation generation and evaluation, and two private modules for text formatting and data storage.

The *annotation (generation) module* consists of two submodules, which offer a variety of manual annotation functionalities, in two modes: simple (or independent) – where the user is not provided with previous annotations from other people – and assisted – which shows and relies on others' annotations. The *(annotation) evaluation module*, on the other hand, provides support to a collaborative assessment process of argumentative component and relation annotations made by several users.

The private modules support the public modules in the annotation generation and evaluation tasks. The *text formatting module* allows presenting input documents in a more readable and context-oriented form, as well as highlighting argumentative components in different colors, to facilitate the user's visual identification of arguments during the generation and evaluation of annotations. The *data storage module* is responsible for writing and reading (to/from CSV or JSON files or a relational database) the annotations and their evaluations made by the users.

3.1.2. Input files

ARGAE is a highly configurable tool, fed by three different types of input files (shown in the left block of Fig. 1), which contain configuration settings, usernames of registered annotation authors and evaluators, and text documents subject to argument annotation tasks.

The JSON configuration file (Fig. 2) allows configuring, in a simple and readable way, the argument model to be used during the annotation (`annotation_model` key), the metrics to be used

during the evaluation (`evaluation_model` key), as well as some characteristics of the text documents to be used (`data` key). The argument model is composed of three fields, which are the components of an argument (e.g., *claim* and *premise*), the intentions of the relations between arguments (commonly, *support* and *attack*), and additional categories and subcategories of such relations (e.g., *reason*, *condition*, *goal*, and *exemplification*). These fields allow using different argumentative structures proposed in the literature. The evaluation model, on the other hand, allows defining the values of the qualitative metrics that will be used to evaluate the annotations (e.g., *incorrect*, *not relevant*, and *relevant* for an 'argument quality' metric). Finally, for the input and output data, it is possible to configure the file type of the text documents subject to annotation, the language of such documents – e.g., English (en) or Spanish (es) –, the source folder path where the target text documents are located, and the result folder path where the evaluations created through the tool are saved. These two latter parameters (i.e., the *source* and *result* folders) are especially important, since they allow the configuration of collaborative work environments, in which, e.g., documents and annotations are stored on a (remote) shared device.

The text documents represent the set of argumentative fragments susceptible to be annotated by users through the tool. So far, two document formats are allowed: TXT (continuous text, without any logical separation other than paragraphs) and JSON (an array of text-valued dictionaries, separated by some logic related to the dataset, such as question-answers or proposal-comments hierarchies). As an illustrative example, Fig. 3 shows part of an input document in JSON format with a logical proposal-comments hierarchy.

3.1.3. Output data

As shown in the right block of Fig. 1, ARGAE is capable of storing both argument annotations and their evaluations into comma-separated-values (CSV) files and local/remote relational databases. More specifically, for each input text document, a user's annotations are saved in two CSV files (or database tables). The first file contains the annotation information associated to argument components (ACs), whereas the second file contains the annotation information corresponding to argument relations (ARs). The information of these elements is structured in fields, which are described in Tables 1 and 2, respectively.

Similarly to annotations, for each input text document, ARGAE handles two files storing annotation evaluations made by a user: one for ACs (Table 3) and another for ARs (Table 4). In

```

{
  "annotation_model": {
    "components": ["major claim", "claim", "premise"],
    "relation_intents": ["support", "attack"]
    "relation_categories": ["cause:condition", "consequence:goal", ...,
                          "elaboration:precision"]
  },
  "evaluation_model": {
    "quality": ["incorrect", "not relevant", "relevant", "very relevant"]
  },
  "data": {
    "file_extension": "json",
    "language": "en",
    "source": "C:/argael/data/proposals",
    "result": "C:/argael/data/results"
  }
}

```

Fig. 2. Example of the JSON configuration file in ARGAEI.

```

[
  {"text": "New public policies should be made for empowering ethnic
        minorities...",
    "type": "proposal", "id": "10675"},
  {"text": "They have been placed in a situation of discrimination
        and institutional racism.",
    "type": "comment", "id": "75704"},
  .
  .
  .
  {"text": "It is not about giving privileges to anyone,
        it is about not taking away anyone's rights.",
    "type": "comment", "id": "10675"},
]

```

Fig. 3. Example of an input JSON document in ARGAEI.

Table 1

Structure of the CSV files that store annotations of argument components (ACs).

Field	Description
<i>ac_id</i>	Identifier of the annotated AC.
<i>ac_text</i>	AC text.
<i>ac_type</i>	AC type, e.g., <i>major claim</i> , <i>claim</i> , or <i>premise</i> .
<i>annotator</i>	Username of the annotator.
<i>timestamp</i>	Timestamp of the annotation.

Table 2

Structure of the CSV files that store annotations of argument relations (ARs).

Field	Description
<i>ar_id</i>	Identifier of the annotated AR.
<i>ac_source_id</i>	Identifier of the source AC of the relation.
<i>ac_target_id</i>	Identifier of the target AC of the relation.
<i>ar_type</i>	Relation (sub)category, e.g., <i>reason</i> , <i>addition</i> , <i>explanation</i> , etc.
<i>annotator</i>	Username of the annotator.
<i>timestamp</i>	Timestamp of the annotation.

Table 3

Structure of the CSV files that store evaluations of argument components (ACs).

Field	Description
<i>ac_id</i>	Identifier of the evaluated AC.
<i>ac_quality</i>	Quality score assigned to the AC.
<i>evaluator</i>	Username of the evaluator.
<i>timestamp</i>	Timestamp of the evaluation.

Table 4

Structure of the CSV files that store evaluations of argument relations (ARs).

Field	Description
<i>ar_id</i>	Identifier of the evaluated AR.
<i>ar_quality</i>	Quality score assigned to the AR.
<i>evaluator</i>	Username of the evaluator.
<i>timestamp</i>	Timestamp of the evaluation.

addition to the evaluators' usernames and the annotations timestamps, the files store the scores assigned for the used evaluation

metrics. In the tables, a quality metric is considered, but others could be added.

For all the elements – argument component/relation annotations and their evaluations –, the CSV format of the output

files offers portability and flexibility, being easily usable to create training datasets for machine and deep learning argument mining models, or to calculate (offline) inter-annotator agreement metrics.

3.2. Software functionalities

In this section, we describe ARGAEEL functionalities, available in its graphical user interface, which has been designed to facilitate the main tasks of the tool: *annotation generation* (Section 3.2.1) and *annotation evaluation* (Section 3.2.2).

3.2.1. Annotation generation

The ARGAEEL interface offers two views for the annotation generation task, namely the *simple view* (Fig. 4) and the *assisted view* (Fig. 5). Both of them allow the user to: (i) identify and manually annotate ACs on a given text, (ii) annotate ARs between pairs of identified ACs, and (iii) interactively validate the created annotations. Each of these functionalities is detailed next.

- *Annotation of ACs.* For the two views, and from a set of text documents loaded via a main menu in the tool (and listed in a left panel of the interface), we can select a particular document and browse its content in a scrollable panel. Then, for the selected document, we can annotate argument components by selecting (highlighting with the mouse) their text fragments, choosing their categories, – e.g., *major claim*, *claim* or *premise* –, according to the configured argument model, and clicking on an “Add” button.
- *Annotation of ARs.* In the annotation interfaces, once we have annotated ACs, we can establish both intra- and inter-argument relations in four easy steps, namely selecting (by clicking it in a table of AC annotations) a source AC, selecting a target AC, choosing the intent and type of the relation, and clicking on an “Add” button.
- *Validation of annotations.* At any time, the annotation interfaces allow the user to modify the annotated arguments components and relations if necessary, by deleting them and creating new ones.

ARGAEEL offers a document-driven visualization of the texts that are being annotated, providing the entire context of every argument, and making inter-argument relations possible. Next, we describe particular graphical controls and functionalities provided by each of the annotation views.

The *simple annotation view* (Fig. 4), accessible through the *Independent Annotation* tab of the interface, allows the user to identify and annotate argumentative components and their respective relations independently, i.e., without taking into account previous annotations from other people on the same document.

Next, we describe the graphical controls available in this view, following their numeration in Fig. 4:

1. *Main annotation panel.* It displays the selected document text along with the argument components annotated by the current user. It supports particular text highlighting of each component type.
2. *Annotator toolbar.* It provides graphical controls for creating and deleting argument components and relations. It offers as options the elements configured in the argument model (cf. Section 3.1.2).
3. *Argument components (ACs) table.* It displays the argument components annotated by the current user, by means of three fields: AC id, AC text, and AC type.
4. *Argument relations (ARs) table.* It displays the argument relations annotated by the current user, by means of five fields: AR id, AC id 1, AC id 2, AR type, and AR intent.

5. *Argument text area.* It displays the selected argument in a user-friendly way. Its text is shown when selecting an argument relation in the ARs table.

The *assisted annotation view* (Fig. 5), accessible through the *Assisted Annotation* tab of the interface, allows the user to identify and annotate argumentative components and relations in an assisted way, i.e., based on previous annotations of other users.

Next, we describe the graphical controls available in this view, following their numeration in Fig. 5:

1. *Drop-down list of target (reference) annotators.* It contains the set of other annotators registered into the system. By choosing one of these annotators, the current user specifies that she is interested in accessing such annotator's work on a selected document.
2. *Assisted annotation panel.* It displays the annotations made by the chosen target annotator on a selected document. These annotations serve as a guide for the current user to annotated the document by herself.

The graphical controls 3, 4, 5 and 6 have the same behavior as 1, 2, 3 and 4 of the simple annotation view, respectively.

3.2.2. Annotation evaluation

The annotation evaluation view (Fig. 6), accessible through the *Evaluation* tab of the interface, allows the user to manual assess annotations provided by other people, in terms of the evaluation metrics defined in the configuration file.

In ARGAEEL, the default metric for assessing annotations corresponds to an argument quality metric that takes the following values: *incorrect*, *not relevant*, *relevant*, and *very relevant*; allowing the simultaneous evaluation of the correctness and relevance of the annotations.

Next, we describe the graphical controls available in this view, following their numeration in Fig. 6:

1. *Drop-down list of target (reference) annotators.* It contains the set of other annotators registered into the system. By choosing one of these annotators, the current user specifies that she is interested in accessing such annotator's work on a selected document.
2. *Annotation panel.* It displays the annotations made by the chosen target annotator on a selected document.
3. *Argument components (ACs) table.* Its rows show the argument components annotated by the target user. For each row, the control allows the assessment of such components by selecting one of the values of a drop-down list in the *evaluation* column of the table.
4. *Argument relations (ARs) table.* Its rows show the argument relations annotated by the target user. For each row, the control allows the assessment of such relations by selecting one of the values of a drop-down list in the *evaluation* column of the table.
5. *Argument text area.* It displays the selected argument in a user-friendly way. The text is shown when selecting an argument relation in the ARs table.

4. Illustrative example and preliminary evaluation

As a proof of concept, in this section, we test the correct functioning of ARGAEEL by considering a novel argument model (Section 4.1), creating a new argumentative corpus (Section 4.2), and conducting a user study on the perceived usability of the tool (Section 4.3).

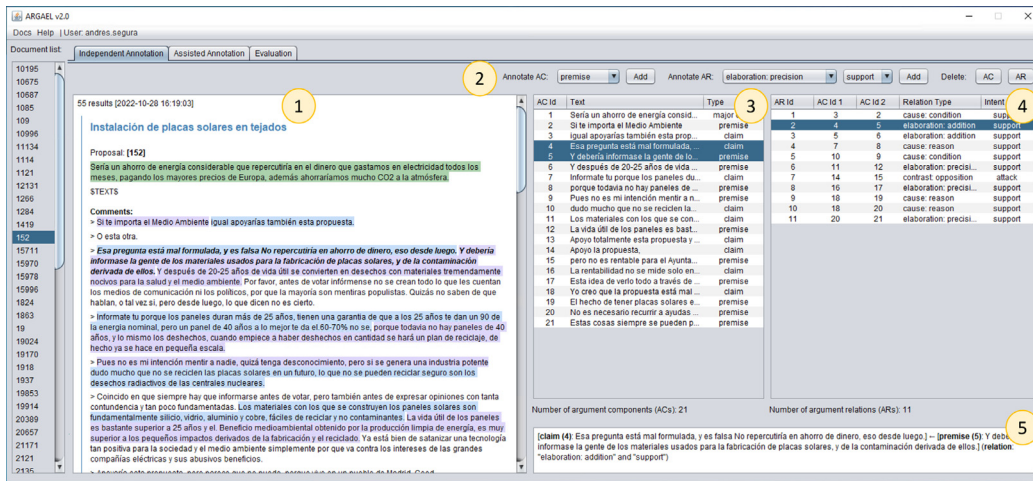


Fig. 4. ARGAEI simple annotation view.

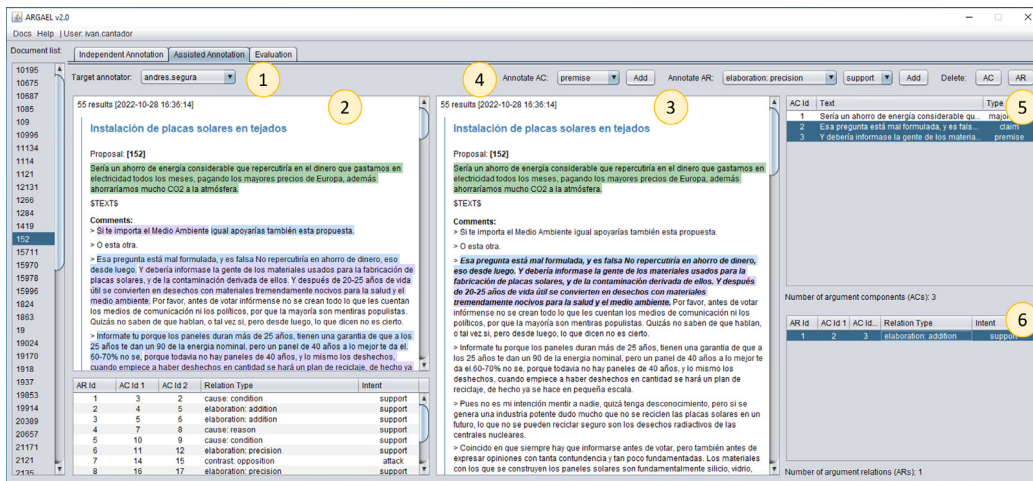


Fig. 5. ARGAEI assisted annotation view.

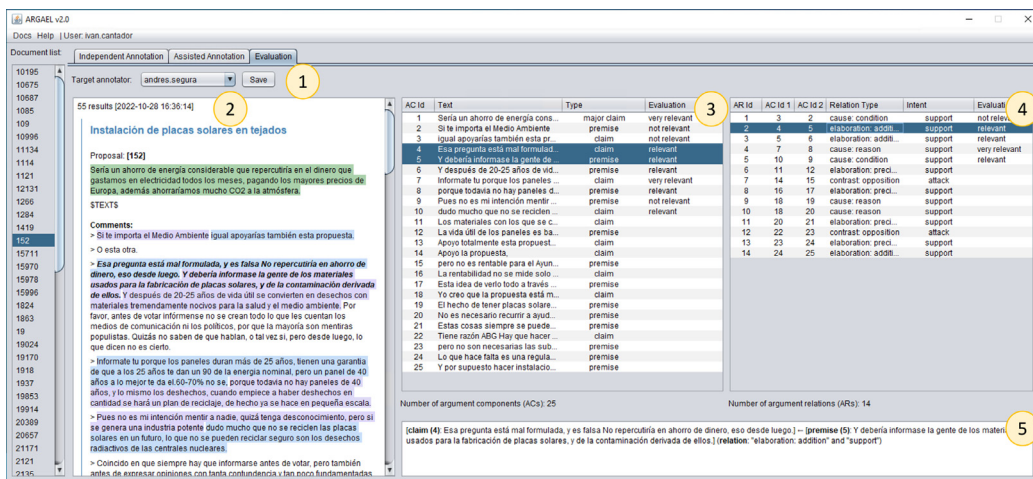


Fig. 6. ARGAEI evaluation view.

4.1. Argument model

Several argument models have been proposed in the literature, among which the traditional *premise-claim* model with *support-attack* relations stands out [4,31,32].

We extend this model with ARGAEI by defining an argument with respect to argumentative components (ACs) and relations between pairs of ACs. The ACs are the core of an argument, and usually are *major claims* (*mc*), *claims* (*c*) and *premises* (*p*). The relations between pairs of components – a.k.a., argumentative

relations (ARs) – give the form and meaning to the arguments and, in our model, are characterized by two attributes: its *intent* (e.g., support or attack) and its *category* (and *subcategory*), based on the two-level taxonomy we introduced in [33].

The main elements of this taxonomy (which is also available online⁶) are the following:

- *Cause*. It states the *reason* or *condition* for an argument. Examples of sentences that include this relation are: “[The pollution levels in the city center are very high]_c because [most people use the car to get around]_p”, “[If the government wants to favor tourism]_p, [it must offer free tourist information]_c”.
- *Clarification*. It introduces a *conclusion*, *exemplification*, *re-statement*, or *summary* of an argument. Examples of sentences: “As a conclusion, [we suggest the government to authorize this initiative]_c”, “In short, [we have to wait for the results of the elections so that they can start to do something]_c”.
- *Consequence*. It evidences an *explanation*, *goal*, or *result* of an argument. Examples of sentences: “[The use of public transport should be facilitated]_p to [avoid pollution in the downtown area]_c”, “[I have not seen garbage trucks for a week]_p, hence [the bins are full, and people have to throw the garbage in the streets]_c”.
- *Contrast*. It conflicts with an argument by giving *alternatives*, doing *comparisons*, making *concessions*, or providing *oppositions*. Examples of sentences: “On the other hand, [we must think about the costs that this work will cause due to its maintenance]_c”, “[Restricting the access of private vehicles to the downtown area helps in mitigating noise]_c, but [it is still insufficient due to the presence of buses, taxis, etc.]_c”.
- *Elaboration*. It introduces an argument that provides details about another one, entailing *addition*, *precision*, or *similarity* issues. Examples of sentences: “[The asphalt of the streets is in very bad conditions]_c, moreover, [the sidewalks have holes]_c”, “[The youth unemployment rate has increased compared to last year]_c, specifically, [it has gone from 23% to 28%]_c”.

4.2. Argumentation corpus

To assess the effectiveness of ARGAE, we used it to create a corpus with public citizen-generated content from the *Decide Madrid*⁷ e-platform. Since September 2015, the platform is an ad-hoc website used by the City Council of Madrid (Spain) as part of its participatory budgeting initiative. Through *Decide Madrid*, residents of Madrid can post proposals to address issues and problems in the city, and comment and vote others' proposals.

The *Decide Madrid* platform is built upon CONSUL⁸, an open-source framework supported by the City Council of Madrid, and is used in tens of cities in Spain, Italy, France and South America. *Decide Madrid* follows the standard structure of online forums and social networks, which is based on trees of hierarchical, nested comments. In particular, each citizen proposal has associated a tree whose root contains the proposal's title and description, and whose nodes are comments with positive or negative textual opinions and arguments about the proposal or a parent node.

More specifically, we built the corpus upon the *Decide Madrid* open dataset presented in [34]. The dataset contains information about 21,744 citizen proposals – automatically classified into 30 categories and 325 topics, and annotated with controversy scores.

Table 5

Some statistics about the generated corpus.

	Total	Per proposal	Per comment
Proposals	40		
Comments	1,355	33.9	
Words	66,989	1,674.7	49.44
Arg. components	1,460	36.50	1.08
Arg. relations	538	13.45	0.40

To narrow the scope of the case study, we limited the first use of ARGAE to a subset of 80 proposals and their associated 5633 comments.

We next report some statistics about the current version of the corpus, which is composed of the 40 most controversial proposals (selected as established in [34]), together with their 1355 associated comments. Each of these proposals deals with a certain topic, e.g., street cleaning services, recycling habits, and wealth balance. Using ARGAE, we annotated 1460 argument components (922 *claims* and 538 *premises*) and 538 argument relations (distributed by category as: 77 relations belonging to *Cause*, 64 to *Clarification*, 76 to *Consequence*, 120 to *Contrast*, and 201 to *Elaboration*). Table 5 and Fig. 7 show additional details about these statistics.

Thanks to ARGAE, we were able to annotate the 40 proposals from scratch, with no execution errors and in approximately 84 h, which represents a ~42% of efficiency improvement with respect to the time we spent in a previous annotation process using a commercial annotation tool. Besides, we believe that ARGAE allowed us to be more effective in identifying argumentative relationships since we always had access to the argumentation contexts (i.e., proposals plus comments). On the other hand, through the ARGAE evaluation module, it was easier for us to correct some annotations based on the feedback given in a collaborative validation stage.

4.3. User study

We conducted a preliminary user study aimed to assess the users' perceived usability of ARGAE for the creation and evaluation of argument annotations. Specifically, we used the System Usability Scale (SUS) questionnaire [35,36] as the instrument to measure and assess the usability of the tool for the above two tasks.

A total of 10 people participated in the study. They were 7 male and 3 female of ages ranging 30–39 years old (7), 40–49 years old (2), and 50–59 years old (1), with different education levels: Bachelor's degree (3), Master's degree (2), and Doctoral degree (5). They had studied Engineering (6), Sciences (1), Social Sciences (2), and Arts and Humanities (1) careers. The participants had relatively medium levels of knowledge/expertise on annotation tools – certain knowledge (6), medium expertise (3), and high expertise (1) – and natural language processing – null knowledge and expertise (2), certain knowledge (3), low expertise (2), medium expertise (2) and high expertise (1) –, and low levels of knowledge/expertise on argument mining – null knowledge and expertise (7), certain knowledge (2), and low expertise (1).

After a training session to learn how to use ARGAE, the participants were requested to freely use the tool for creating and evaluating argument annotations on the corpus described in Section 4.2, for about 20 min. Afterward, they filled the 10-item SUS questionnaire according to their experience using ARGAE. In a scale of 100, the resultant scores of the SUS questionnaire were 83.8 and 86.3 for the annotation creation and evaluation modules of ARGAE, respectively, which mean acceptable levels (>70) of usability.

⁶ Argument two-level taxonomy, <https://github.com/argrecsys/connectors>.

⁷ Decide Madrid, <https://decide.madrid.es>.

⁸ CONSUL, <http://consulproject.org>.

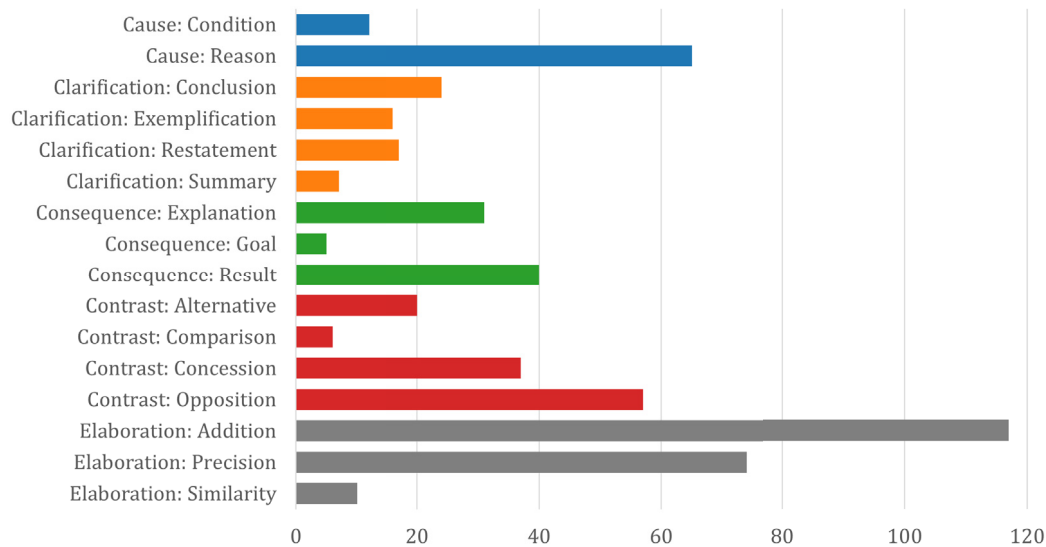


Fig. 7. Number of argument relations by (sub)category in the corpus.

Apart from obtaining these quantitative results, we took advantage of the study to ask participants for their opinions about the tool. In general, they found ARGAEEL easy or very easy to use and considered it very valuable for annotating argumentative texts. By contrast, they had difficulties with the argument annotation itself, whose complexity has been widely recognized in the research literature [2]. The identification of those fragments that are relevant arguments, and the recognition of the existing relations between argument components, are tasks whose outcomes do not necessarily be the same for different human annotators. In this context, some participants suggested us to incorporate into ARGAEEL a dialog window showing an explanation of the underlying argument model; in particular, providing typical connectors and example sentences of its 16 subtypes of argument relations. There were also comments on the possibility of having an interactive, graph-based visualization of the annotated argument structures.

5. Impact and conclusions

In this paper, we have presented ARGAEEL, an open-source Java desktop tool for the annotation and evaluation of argumentative information in text documents. The tool entails several benefits and novelties, such as allowing the definition and use of different argument models, taking advantage of the context of the arguments during the annotation process, and supporting collaborative work by multiple annotators and evaluators.

Through ARGAEEL and a survey on (argument) annotation tools we have presented in this paper, we aim to raise awareness of the need for software applications that allow the joint creation and evaluation of high-quality argument annotations, with which improving the effectiveness of machine and deep learning-based argument mining models.

In this sense, several development and research lines may be addressed in the future. With regard to argument annotation, and more specifically, the flexibility of ARGAEEL to define and use different argumentation models, we should test and maybe extend ARGAEEL according to existing theories of argument identification [37], such as Walton's taxonomy of argument schemes [38], and Wagemans' periodic table of arguments [39]. We could also explore (semi-automatic) efficient methods to assist with the manual establishing of the annotations, e.g., by recommending potential text to annotate or labels to use [22,23], or by following

an interactive label choice mechanism [21]. We could also incorporate into ARGAEEL the functionality of exporting and loading argument annotations in different formats (e.g., AIF [20]), thus increasing its compatibility and allowing its comparison with other tools.

With respect to the evaluation of argument annotations, we could integrate into ARGAEEL diverse evaluation methodologies and procedures [40], and metrics, such as the fairness and diversity of arguments [41], which allow measuring the quality of identified arguments beyond the accuracy and topic relevance, as it is commonly done [2].

Moreover, based on [42], in ARGAEEL graphical interface, it would be desirable to have an additional view showing an analysis of results achieved in the (collaborative) annotation generation and evaluation processes, such as annotation statistics and inter-annotator agreement measures.

In addition, it would be convenient to conduct an evaluation and comparison of ARGAEEL with respect to related tools, likely through a user study on different aspects, such as the effectiveness and efficiency of the collaborative annotation process, the use and support of different argument schemes proposed in the literature, and the compatibility with existing exchange annotation formats, to name a few.

Finally, we contemplate the future implementation of a web version of ARGAEEL that would extend and improve the desktop version presented herein. The web application would not take away the transversal advantages of its desktop client, such as the lack of need for a web server in which deploying the application, the possibility of working without internet connection, and the opportunity for any researcher or developer to modify ARGAEEL code in the public repository, adding new functionalities to the tool.

CRedit authorship contribution statement

Andrés Segura-Tinoco: Conceptualization, Software, Data curation, Writing – original draft, Visualization. **Iván Cantador:** Conceptualization, Methodology, Validation, Data curation, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing

interests: Andres Segura-Tinoco reports a relationship with Autonomous University of Madrid that includes: employment. Ivan Cantador reports a relationship with Autonomous University of Madrid that includes: employment.

Data availability

The data used in the article are public and links to them can be found in the article (as footnotes).

Acknowledgments

This research was supported by the Spanish Ministry of Science and Innovation (PID2019-108965GB-I00). We sincerely thank the anonymous reviewers of the paper for their valuable comments and suggestions.

References

- [1] Mochales Palau R, Moens M-F. Argumentation mining: The detection, classification and structure of arguments in text. In: Proceedings of the 12th international conference on artificial intelligence and law. ACM; 2009, p. 98–107.
- [2] Lawrence J, Reed C. Argument mining: A survey. *Comput Linguist* 2020;45(4):765–818.
- [3] Mochales Palau R, Moens M-F. Argumentation mining. *Artif Intell Law* 2011;19(1):1–22.
- [4] Eger S, Daxenberger J, Gurevych I. Neural end-to-end learning for computational argumentation mining. In: Proceedings of the 55th annual meeting of the association for computational linguistics. ACL; 2017, p. 11–22.
- [5] Reimers N, Schiller B, Beck T, Daxenberger J, Stab C, Gurevych I. Classification and clustering of arguments with contextualized word embeddings. In: Proceedings of the 57th annual meeting of the association for computational linguistics. ACL; 2019, p. 567–78.
- [6] Cunningham H, Maynard D, Bontcheva K, Tablan V. GATE: An architecture for development of robust HLT applications. In: Proceedings of the 40th annual meeting of the association for computational linguistics. ACL; 2002, p. 168–75.
- [7] Stenetorp P, Pyysalo S, Topić G, Ohta T, Ananiadou S, Tsujii J. BRAT: A web-based tool for NLP-assisted text annotation. In: Proceedings of the demonstrations at the 13th conference of the European chapter of the association for computational linguistics. ACL; 2012, p. 102–7.
- [8] Reed C, Rowe G. Araucaria: Software for argument analysis, diagramming and representation. *Int J Artif Intell Tools* 2004;13(04):961–79.
- [9] Gordon TF, Prakken H, Walton D. The Carneades model of argument and burden of proof. *Artificial Intelligence* 2007;171(10–15):875–96.
- [10] Manning C, Schütze H. Foundations of statistical natural language processing. MIT Press; 1999.
- [11] Cunningham DMH, Bontcheva K. Text processing with GATE (version 6). University of Sheffield; 2011.
- [12] Bontcheva K, Cunningham H, Roberts I, Roberts A, Tablan V, Aswani N, et al. GATE Teamware: A web-based, collaborative text annotation framework. *Lang Resour Eval* 2013;47(4):1007–29.
- [13] Maeda K, Strassel S. Annotation tools for large-scale corpus development: Using ACTK at the Linguistic Data Consortium. In: Proceedings of the 4th international conference on language resources and evaluation. ACL; 2004, p. 2077–80.
- [14] Klie J-C, Bugert M, Boulosa B, de Castilho RE, Gurevych I. The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In: Proceedings of the 27th international conference on computational linguistics: system demonstrations. ACL; 2018, p. 5–9.
- [15] Abrami G, Mehler A, Lücking A, Rieb E, Helfrich P. TextAnnotator: A flexible framework for semantic annotations. In: Proceedings of the 15th joint ACL-ISO workshop on interoperable semantic annotation. ALC; 2019, p. 1–12.
- [16] Wolf M, Ruiter D, D'Sa AG, Reiners L, Alexandersson J, Klakow D. HUMAN: Hierarchical universal modular ANnotator. In: Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations. ACL; 2020, p. 55–61.
- [17] de Souza E, Freitas C. ET: A workstation for querying, editing and evaluating annotated corpora. In: Proceedings of the 2021 conference on empirical methods in natural language processing: System demonstrations. ACL; 2021, p. 35–41.
- [18] Nivre J, De Marneffe M-C, Ginter F, Goldberg Y, Hajic J, Manning CD, et al. Universal dependencies v1: A multilingual treebank collection. In: Proceedings of the 10th international conference on language resources and evaluation. ACL; 2016, p. 1659–66.
- [19] Janier M, Lawrence J, Reed C. OVA+: An argument analysis interface. In: Computational models of argument. IOS Press; 2014, p. 463–4.
- [20] Chesñevar C, Modgil S, Rahwan I, Reed C, Simari G, South M, et al. Towards an argument interchange format. *Knowl Eng Rev* 2006;21(4):293–316.
- [21] Lawrence J, Visser J, Reed C. An online annotation assistant for argument schemes. In: Proceedings of the 13th linguistic annotation workshop. ALC; 2019, p. 100–7.
- [22] Sperrle F, Sevastjanova R, Kehlbeck R, El-Assady M. VIANA: Visual interactive annotation of argumentation. In: Proceedings of the 2019 IEEE conference on visual analytics science and technology. IEEE; 2019, p. 11–22.
- [23] Chernodub A, Oliynyk O, Heidenreich P, Bondarenko A, Hagen M, Biemann C, et al. TARGER: Neural argument mining at your fingertips. In: Proceedings of the 57th annual meeting of the association for computational linguistics: System demonstrations. ACL; 2019, p. 195–200.
- [24] Winkels R, Douw J, Veldhoen S. State of the ART: an argument reconstruction tool. In: Proceedings of the 5th international workshop on semantic processing of legal texts. 2014, p. 17.
- [25] Musi E, Stede M, Kriese L, Muresan S, Rocci A. A multi-layer annotated corpus of argumentative text: From argument schemes to discourse relations. In: Proceedings of the 11th international conference on language resources and evaluation. ACL; 2018, p. 1629–36.
- [26] Putra JWG, Teufel S, Matsumura K, Tokunaga T. TIARA: A tool for annotating discourse relations and sentence reordering. In: Proceedings of the 12th language resources and evaluation conference. ACL; 2020, p. 6912–20.
- [27] Putra JWG, Matsumura K, Teufel S, Tokunaga T. TIARA 2.0: An interactive tool for annotating discourse structure and text improvement. *Lang Resour Eval* 2021;1–25.
- [28] Morio G, Fujita K. Annotating online civic discussion threads for argument mining. In: Proceedings of the 2018 IEEE/WIC/ACM international conference on web intelligence. IEEE; 2018, p. 546–53.
- [29] Blessing A, Blokker N, Haunss S, Kuhn J, Lapesa G, Padó S. An environment for relational annotation of political debates. In: Proceedings of the 57th annual meeting of the association for computational linguistics: System demonstrations. ACL; 2019, p. 105–10.
- [30] Collins E, Rozanov N, Zhang B. LIDA: Lightweight interactive dialogue annotator. In: Proceedings of the 2019 conference on empirical methods in natural language processing: System demonstrations. ACL; 2019, p. 121–6.
- [31] Haddadan S, Cabrio E, Villata S. Yes, we can! Mining arguments in 50 years of us presidential campaign debates. In: Proceedings of the 57th annual meeting of the association for computational linguistics. ACL; 2019, p. 4684–90.
- [32] Suhartono D, Gema AP, Winton S, David T, Fanany MI, Arymurthy AM. Argument annotation and analysis using deep learning with attention mechanism in Bahasa Indonesia. *J Big Data* 2020;7(1):1–18.
- [33] Segura-Tinoco A, Cantador I. On the extraction and use of arguments in recommender systems: A case study in the e-participation domain. In: Proceedings of the 3rd edition of knowledge-aware and conversational recommender systems and the 5th edition of recommendation in complex environments workshops. Vol. 2960. CEUR-WS; 2021, p. 3:1–3:11.
- [34] Cantador I, Cortés-Cediel ME, Fernández M. Exploiting open data to analyze discussion and controversy in online citizen participation. *Inf Process Manage* 2020;57(5):102301.
- [35] Brooke J. SUS - A quick and dirty usability scale. *Usability Eval Ind* 1996;189(194):4–7.
- [36] Lewis JR, Sauro J. The factor structure of the System Usability Scale. In: Proceedings of the 1st international conference on human centered design. Springer; 2009, p. 94–103.
- [37] Visser J, Lawrence J, Reed C, Wagemans J, Walton D. Annotating argument schemes. In: Argumentation through languages and cultures. Springer; 2020, p. 101–39.
- [38] Walton D, Reed C, Macagno F. Argumentation schemes. Cambridge University Press; 2008.
- [39] Wagemans J. Constructing a periodic table of arguments. In: Proceedings of the 11th international conference of the ontario society for the study of argumentation. SSRN; 2016, p. 1–12.
- [40] Hinton M, Wagemans JHM. Evaluating reasoning in natural arguments: A procedural approach. *Argumentation* 2022;36(1):61–84.
- [41] Pathiyann Cherumanal S, Spina D, Scholer F, Croft WB. Evaluating fairness in argument retrieval. In: Proceedings of the 30th ACM international conference on information & knowledge management. ACM; 2021, p. 3363–7.
- [42] Stab C, Gurevych I. Annotating argument components and relations in persuasive essays. In: Proceedings of the 25th international conference on computational linguistics: Technical papers. ACL; 2014, p. 1501–10.