Estimating Argumentative Class Prevalence in Textual Data for Automatic Analysis of Scientific Literature

NLP Course Project & Project Work

Antonio Morelli, Edoardo Conca, Fabio Zanotti
Master's Degree in Artificial Intelligence, University of Bologna
{ antonio.morelli3, edoardo.conca, fabio.zanotti2 }@studio.unibo.it

Abstract

Peer review is an essential yet resource-intensive process. Recent studies have investigated the relationship between a research paper's relevance and its argumentative content using classification-based approaches (Brambilla et al., 2022). However, such methods have proven to be suboptimal (Esuli et al., 2022), as they rely on the intermediate classification steps rather than directly estimating class frequencies.

In this work, we tackle the task of quantification—estimating class prevalence in datasets—by leveraging methodologies from the LeQua 2022 challenge (Esuli et al., 2022) and the accompanying Python library, *QuaPy*. Using the techniques proposed by (Mayer et al., 2020, 2021) as a baseline, which focus on argumentative structure extraction in healthcare-related papers, and employing the dataset produced in that study, we evaluate a network-based approach, *QuaNet* (Esuli et al., 2018).

The results obtained demonstrate the effectiveness of these methodologies, suggesting their potential applicability to other datasets for further validation and exploration.

1 Introduction

Peer review plays a central role in modern research, serving as a key mechanism to uphold the quality and trustworthiness of published studies. However, it is a demanding and time-intensive task, and the growing interest in emerging fields often leads to a substantial review burden, particularly for experienced researchers in these areas.

In a recent work (Brambilla et al., 2022), it was hypothesized and partly demonstrated that a research paper relevance might be correlated with its argumentative content; in that study, researchers aimed to classify the arguments within a document and compute a series of indicators to quantify their presence and extent. Nevertheless, as per (Esuli et al., 2022), breaking this task in two steps leads more than often to sub-optimal results, which could be seen as a direct consequence of *Vapnik's principle* (Vapnik, 1999):

> If you possess a restricted amount of information for solving some problem, try to solve the problem directly and never solve a more general problem as an intermediate step. It is possible that the available information is sufficient for a direct solution but is insufficient for solving a more general intermediate problem.

Therefore, we aim to directly solve quantification in place of classification. Quantification refers to estimating class prevalence in datasets. This task is distinguished from classification, as it emphasizes predicting the distribution of class frequencies rather than assigning specific labels to individual items. LeQua 2022 (Esuli et al., 2022) was a lab for the evaluation of methods for "learning to quantify" in textual datasets, in which a quantification challenge was proposed and a Python library, *QuaPy*, created to offer a plethora of methodologies for solving the task; in our work, we inspected those methodologies and tried to determine if they were employable for the problem at hand.

In contrast to (Brambilla et al., 2022), which centers on analyzing entire papers to identify their argumentative content, our work focuses on datasets composed of abstracts (Mayer et al., 2020) (Mayer et al., 2021) and extends the task by automating the determination of the number of arguments present within them.

2 Background

The works we followed to build the starting baseline (Mayer et al., 2020, 2021) focus on the automatic extraction of argumentative structures from Healthcare related papers, which involves two main tasks: **Argument Component Detection** (*Claims*,

Premises) and **Relation Classification** between these components (*Support*, *Attack*, *Partial-attack*). They developed a system that combines a sequence tagging model that uses BIO tagging schemes[1] (Ramshaw and Marcus, 1995) to label components with a classification model to determine the multiclass categorization of argumentative relations.

Although not directly addressing the problem of Argument Quantification, their work in building a dataset with component and relation labels came in handy to our purpose: the study involves the annotation of 500 RCT abstracts from the MEDLINE database, yielding a dataset with over 4,000 argumentative components and 2,600 relations across diseases like neoplasms, glaucoma, and diabetes.

The work by (Esuli et al., 2022) instead introduces the LeQua 2022 lab, designed to evaluate methods for learning to quantify within textual datasets. They organized tasks in binary and single-label multiclass quantification, providing participants with data in both raw and vectorized formats. This structure allowed for evaluations tailored to different expertise levels, from text preprocessing to quantification optimization. Participants were required to estimate class prevalence without external datasets, emphasizing the robustness of their methodologies.

This initiative marked the first dedicated comparative evaluation of quantification techniques, contributing insights into the relative performance of state-of-the-art methodologies. Among all the entries, *QuaNet* (Esuli et al., 2018) was proposed as a neural network-based approach tailored for quantification, that performed competitively across several tasks. It leverages an architecture optimized for estimating class prevalence in text datasets and operates by directly predicting class frequencies rather than relying on intermediate classification and counting steps, addressing known limitations of traditional classifiers when repurposed for quantification.

3 System description

3.1 Baseline

The baseline we implemented is based on the open-source architecture proposed by (Mayer et al., 2020). The source code for this architecture is publicly available[2]. We retained both the Argument Component Detection (ACD) and Relation Classification (RC) models as they were originally structured, applied the optimal hyperparameter configuration described in the paper, and reproduced the experiments from scratch.

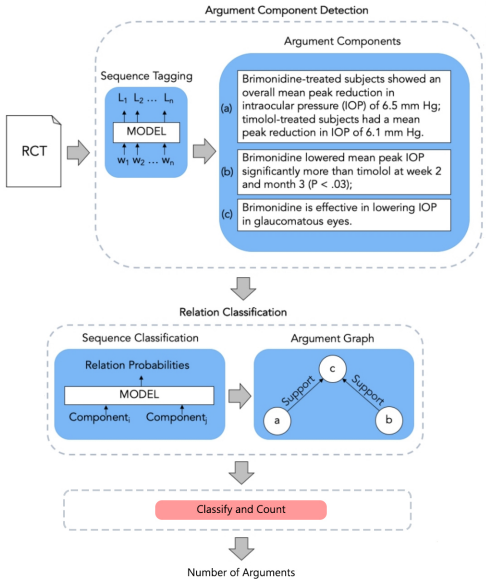

Figure 1: Baseline architecture (Mayer et al., 2020).

Their best-performing ACD model employs a fine-tuned BERT model trained on scientific texts, `SciBERT`) (Beltagy et al., 2019), as the foundational encoder for contextual embeddings. On top of the BERT output, a bidirectional GRU is added to model sequential relationships in token-level embeddings. The output layer consists of a Conditional Random Field (CRF)[3] for structured sequence prediction, which enforces valid label transitions. Finally, a fully connected linear layer maps the RNN output to the predicted labels (O, B-Claim, I-Claim, B-Premise, I-Premise).

The best-performing RC model also employs `SciBERT` as the base encoder for text pairs. A linear layer is used for relation classification (Support, Attack), with dropout applied before the classifier to reduce overfitting. We followed the original evaluation approach, which involves calculating relation classification metrics only for predicted components with an overlap of at least 75% with

[1] BIO tagging is a method for labeling tokens as the beginning (B), inside (I), or outside (O) of specific entities.

[2] The source code for the architecture proposed by (Mayer et al., 2020) can be accessed at `https://gitlab.com/tomaye/ecai2020-transformer_based_am`.

[3] Conditional Random Fields (CRFs) are statistical models used for structured prediction, accounting for context unlike classifiers, which label samples independently.

the ground truth.

To achieve our goal of determining the number of arguments within an abstract, we grouped the detected components and their relations into argument graphs, then compared these graphs with the ground truth arguments using a straightforward *classify and count* (CC) strategy. For this purpose, we established—and consistently applied across all experiments—that an argument is considered valid if its graph contains at least one Claim and one Premise.

In addition to reproducing this architecture, we addressed the limitations of the outdated codebase and aligned the architecture more closely to the one described in (Mayer et al., 2021), whose code is unavailable. To this end, we rewrote the implementation from scratch and explored the impact of several modifications. Following the suggestions of (Mayer et al., 2021), we introduced a weighted cross-entropy loss function to better handle class imbalances, and replaced the multi-class relation classification head with a binary classification head, as our primary focus was to determine the presence of a relation rather than its specific type. However, the results obtained from this modified architecture were not better than those of the original baseline and have therefore been omitted from the report.

3.2 QuaNets Ensemble

3.2.1 QuaNet Module

The core components of our architecture are the QuaNet modules: they observe the classification predictions to learn higher-order *quantification embeddings*, which are refined by incorporating quantification predictions of simple classify-and-count-like methods.

CNN-Based Probabilistic Classifier QuaNet requires a classifier that can provide embedded representations of the inputs. For this purpose, we use a `CNN` (Convolutional Neural Network) as a probabilistic classifier. The CNN processes input text through a series of convolutional and pooling layers, extracting hierarchical features that represent the content of the documents. The final layer of the CNN produces a dense vector, referred to as the *document embedding* $\vec{x}$.

The CNN-based classifier starts with an *Embedding Layer* that converts input tokens into dense vector representations. Then the *Convolutional Layers* apply multiple filters of varying sizes to capture n-gram features; a *Max-Pooling Layer* aggregates the most salient features from the convolutional outputs before the *Fully Connected Layer* combines the pooled features into a fixed-length representation $\vec{x}$.

The CNN is trained to output class probabilities $Pr(c|x)$, where c is the predicted class and x is the input document. These probabilities, along with the document embeddings $\vec{x}$, serve as inputs to the QuaNet module.

Recurrent Component: Bidirectional LSTM The core of the model is a *Bidirectional LSTM* (Long Short-Term Memory), a type of recurrent neural network. It receives as input a list of pairs:

$$\langle Pr(c|x), \vec{x} \rangle,$$

where:

- $Pr(c|x)$: The probability that a classifier h assigns class c to document x.
- $\vec{x}$: The document embedding, a vector representing the document's content.

The list is sorted by the value of $Pr(c|x)$, meaning the documents are arranged from least to most likely to belong to class c.

The intuition behind this approach is that the LSTM "learns to count" positive and negative examples. By observing the ordered sequence of probabilities, the LSTM identifies the point where the documents switch from negative to positive examples. The document embedding $\vec{x}$ helps the LSTM assign different importance to each document when making its prediction.

The output of the LSTM is called a quantification embedding—a dense vector representing information about the quantification task learned from the input data.

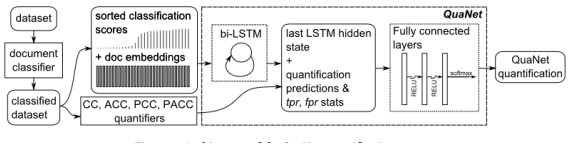

Figure 2: QuaNet module architecture (image from (Esuli et al., 2018)).

Fully Connected Component The vector returned by the Bi-LSTM is combined with quantification-related statistics:

- $\hat{p}_c^{CC}(D), \hat{p}_c^{ACC}(D), \hat{p}_c^{PCC}(D), \hat{p}_c^{PACC}(D)$: Quantification predictions from different methods.

- $tpr_b, fpr_b, tpr_s, fpr_s$: Aggregate statistics related to true positive and false positive rates, computed using a validation set.

This combined vector passes through fully connected layers with *ReLU* activations. These layers adjust the quantification embedding using the additional statistics to improve accuracy.

The final output is a prediction $\hat{p}_c^{QuaNet}(c|D)$, representing the probability of class c for dataset D, produced by a *softmax* layer.

QuaNet focuses on computationally efficient methods (like CC, ACC, PCC, and PACC) to ensure the process remains fast while providing accurate predictions.

- **CC (Classify & Count)**: the simplest aggregative quantifier that relies on the label predictions of a classifier to deliver class prevalence estimates;

- **ACC (Adjusted Classify & Count)** (Forman, 2008): the "adjusted" variant of CC that corrects the predictions of CC according to the "misclassification rates" of the classifier;

- **PCC (Probabilistic Classify & Count)** (Bella et al., 2010): the probabilistic variant of CC that relies on the posterior probabilities returned by a probabilistic classifier;

- **PACC (Probabilistic Adjusted Classify & Count)** (Bella et al., 2010): which stands to PCC as ACC stands to CC.

Layer	Type	Dimensions	Activation	Dropout
Input	LSTM	128	N/A	N/A
Dense 1	Dense	1024	ReLU	0.5
Dense 2	Dense	512	ReLU	0.5
Output	Dense	2	Softmax	N/A

Table 1: QuaNet module architecture details

3.2.2 Ensemble

Our final architecture is an ensemble of three QuaNet modules, the first two trained to perform Sentence-Wise Claims and Premises Quantification, the other used for Relations Quantification. Our first implementation of the Ensemble used a single QuaNet module to quantify Components, later replaced with the two Claims and Premises distinct modules.

The classification heads of these modules are removed, and their quantification embeddings are extracted. These embeddings, encapsulating the features learned by the models, are concatenated to form a unified representation:

$$\mathbf{z}_{ensemble} = \mathbf{z}_{claims} \oplus \mathbf{z}_{premises} \oplus \mathbf{z}_{relations}$$

where $\mathbf{z}_{claims}$, $\mathbf{z}_{premises}$ and $\mathbf{z}_{relations}$ are the quantification embeddings from the three QuaNet modules, and $\oplus$ denotes concatenation.

Each abstract $\mathbf{D}$ is processed through the three QuaNet modules, resulting in a concatenated embedding $\mathbf{z}_{ensemble}$. Once this feature extraction step is complete, a new classification head g is trained to predict the number of arguments:

$$\hat{y} = g(\mathbf{z}_{ensemble}).$$

The F1-score is monitored to to handle imbalanced data more effectively and improve the training strategy (e.g. early stopping regularization) .

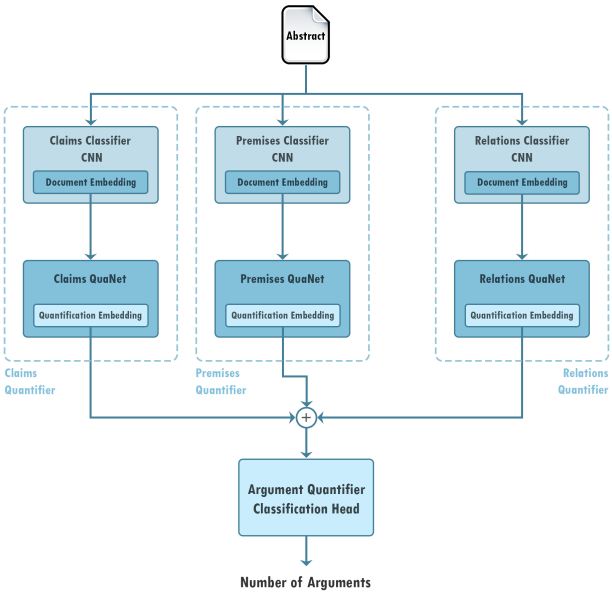

Figure 3: Architecture of our final Ensemble.

4 Data

We focus on the pre-processing routine adopted for our ensemble method, as the two baselines follow the steps outlined in (Mayer et al., 2020) and (Mayer et al., 2021).

4.1 AbstRCT Argument Mining Dataset

The AbstRCT (Mayer et al., 2020) dataset consists of randomized controlled trials (RCTs) retrieved from the MEDLINE database via PubMed search. The trials are annotated with argument components and argumentative relations. The data is split in different subsets:

Subset Name	Number of Abstracts
`neoplasm_train`	350
`neoplasm_test`	100
`glaucoma_test`	100
`mixed_test`	100

Table 2: Overview of the AbstRCT dataset splits. The `mixed_test` subset includes 20 abstracts each on glaucoma, neoplasm, diabetes, hypertension, and hepatitis.

4.2 Pre-processing

Following our *sentence-wise* approach, we preprocess our data by splitting the abstracts into sentences instead of word-wise tokens: labels were then adapted accodingly to each subtask.

4.2.1 Component Quantification task

We split each abstract into individual sentences and label them as either *Non-component* or *Component*, corresponding to `0` and `1`, respectively; abstracts are first split according to the dataset subdivision, then Non-components are further split using `sent_tokenize`, a function from the *Natural Language Toolkit* (nltk) in Python used for sentence tokenization.

Metric	Train Set	Test Set
Label 0 (samples)	2378	1948
Label 1 (samples)	2267	1880
Average number of sentences/file	13	14
Max sentence length	107	91
Average components/file	6.48	6.99
Average non-components/file	6.79	7.24

Table 3: Dataset statistics for Components Quantification.

During evaluation and inference, we apply a sampling technique to process all sentences within an abstract, as our goal is to determine the percentage of argument components each abstract contains; splitting is here conducted through `sent_tokenize` only, as we are aware of the previous knowledge injected into the model.

When instead of labeling as positive components, we chose to label as positive either Claims or Premises, we follow the same process, but inevitably obtain less balanced splits, as we can observe in Tables 4 and 5.

Metric	Train Set	Test Set
Label 0 (samples)	3924	3188
Label 1 (samples)	730	650
Average number of sentences/file	13	14
Max sentence length	107	91
Average components/file	2.09	2.43
Average non-components/file	11.21	11.85

Table 4: Dataset statistics for Claims Quantification.

Metric	Train Set	Test Set
Label 0 (samples)	3108	2599
Label 1 (samples)	1537	1230
Average number of sentences/file	13	14
Max sentence length	107	91
Average components/file	4.39	4.57
Average non-components/file	8.88	9.66

Table 5: Dataset statistics for Premises Quantification.

4.2.2 Relation Quantification task

Initially, we considered splitting each abstract into sentences and labeling all possible combinations of sentences as either *No related* or *Related*, corresponding to `0` and `1`, respectively. However, this led to an imbalanced and overly difficult task, as the majority of sentence pairs were unrelated.

To address this, we approached the problem differently. Instead of trying to predict relationships between all possible sentence pairs, we reformulated the task to focus on the components directly. Specifically, we label each component as either *In a relationship* or *Not in a relationship*. A component is considered In a relationship if it serves as the first member (source) of at least one relation, and Not in a relationship otherwise.

Consider the following annotation associated to a randomly chosen abstract:

```
R1    Partial-Attack Arg1:T10 Arg2:T11
R2    Support Arg1:T8 Arg2:T9
R3    Support Arg1:T7 Arg2:T10
R4    Support Arg1:T6 Arg2:T10
R5    Support Arg1:T5 Arg2:T9
R6    Support Arg1:T2 Arg2:T4
R7    Partial-Attack Arg1:T3 Arg2:T4
R8    Partial-Attack Arg1:T1 Arg2:T4
```

In this annotation, there are eight relations, corresponding to the `R1` through `R8` entries. Each relation has a source, marked as `Arg1`, and a target, marked as `Arg2`. For example, in `R1`, `T10` is the source, and `T11` is the target. Using this information, we label components based on whether they appear as `Arg1` in any relation. Components like `T1`, `T2`, `T3`, `T5`, `T6`, `T7`, `T8`, and `T10` are labeled as `In a Relationship` (1), while components like `T4`, `T9`, and `T11`, as well as non-components, are labeled as `Not in a relationship` (0). This reformulation sim-

plifies the task from predicting relationships between pairs of sentences to predicting whether a given component participates in an outgoing relationship. Although it underestimates the actual number of relationships, we ensured that the likelihood of the same node being a source more than once is minimal in our dataset.

Metric	Train Set	Test Set
Label 0 (samples)	3251	2743
Label 1 (samples)	1394	1085
Average number of sentences/file	13	14
Max sentence length	107	91
Average relationships/file	4.06	4.09
Average no relationships/file	9.29	10.20

Table 6: Dataset statistics for Relations Quantification.

5 Experimental setup and results

We structured our work as a series of experiments to incrementally build our final ensemble and its variants.

5.1 Experiment 1: Components Quantification through Sentence-wise Approach

The data pre-process described in section 4.2.1 is first applied: we then conducted a hyperparameter study on the first CNN using *Optuna*, an automatic hyperparameter optimization framework, to determine optimal values. The values shown in Table 14 refer to the best trial, scoring ≈ 0.85 in Macro F1 on the Validation Set.

We then trained the CNN with those hyperparameters: the evaluation conduced on the test sets is provided in Table 7.

Next, we trained the QuaNet model used for Component Quantification, using a random sampling technique with a number of samples per batch matching the average number of sentences per abstract. During evaluation, we modified the batch selection process to select sentences from a single abstract per batch. We also experimented with this latter technique during training, but since the results were slightly worse than the standard random sampling, we chose not to use this approach in subsequent experiments.

The metrics used in the evaluation include *Absolute Error* (AE), *Relative Absolute Error* (RAE), *Mean Squared Error* (MSE), *Mean Absolute Error* (MAE), and *Modified Kullback-Leibler Divergence* (MKLD) (Agahi, 2019), although we consider AE and RAE to be the most relevant ones (Sebastiani, 2018). The results of this evaluation are presented in Table 8; while they appear promising, the differences observed between the random sampling technique and the modified sampling strategy were significant. This prompted us to investigate the effects of increasing the number of elements per batch so that each batch contained sentences from multiple abstracts. This adjustment led to a decrease in error values, which tended to align more closely with those obtained using the standard random sampling technique.

	Experiment 1 Components		**Experiment 1-A Claims only**		**Experiment 1-B Premises only**	
Metric	**BL**	**CNN**	**BL**	**CNN**	**BL**	**CNN**
Glaucoma Test Set						
Macro F1	0.883	0.861	0.838	0.764	0.919	0.825
Acc. (%)	_	86.10	_	89.62	_	84.41
Neoplasm Test Set						
Macro F1	0.858	0.835	0.788	0.757	0.918	0.829
Acc. (%)	_	83.48	_	87.14	_	84.91
Mixed Test Set						
Macro F1	0.859	0.846	0.793	0.756	0.917	0.829
Acc. (%)	_	84.60	_	87.34	_	85.02

Table 7: Comparison of Evaluation Results on components classification between the Baseline classifier (BL) and the CNN classifiers in our model across Experiments 1, 1-A, and 1-B.

5.1.1 Experiment 1-A & 1-B: Claims & Premises Quantification through Sentence-wise Approach

These two experiments are variants of 5.1 in which we consider as positive only `Claims` (and `MajorClaims`) in the first case and only `Premises` in the latter. The optimal hyperparameter configurations are once again determined using Optuna and are shown in 14.

In the preprocessing phase, we adjusted the labels accordingly to the tasks, setting to `1` only the components of our need and to `0` everything else. This different labeling strategy led to more unbalanced datasets with respect to Experiment 1: when evaluating the trained CNNs, it reflects on more misclassification of minority samples (class `1`) that has little impact on accuracy but a significant impact on the F1 score because both precision and recall are affected, as observable in Table 7.

The respective *QuaNets* were again trained and evaluated with the same method applied to 5.1. The evaluations of the Claims Quantifier QuaNet and Premises Quantifier QuaNet (Table 8) returned similar results to the generic Component Quantifier.

5.2 Experiment 2: Relations Quantification

After having a model capable of quantifying components, we moved on to estimating the number of

Metric	Ex. 1	Ex. 1-A	Ex. 1-B	Ex. 2
Glaucoma Test Set				
AE	0.0920	0.0828	0.0931	0.1404
RAE	0.1804	0.2587	0.2146	0.4652
Neoplasm Test Set				
AE	0.0971	0.1008	0.0838	0.1104
RAE	0.1943	0.2828	0.1806	0.2594
Mixed Test Set				
AE	0.0829	0.1004	0.0882	0.1364
RAE	0.1648	0.2868	0.1974	0.4005

Table 8: Error Metrics in Test Sets Evaluation for components and relations quantifications QuaNets in experiments 1, 1-A, 1-B and 2, using a single abstract as input per batch. Full results are available in section A.2.

relations: the preprocessing procedure described in section 4.2.2 was followed and the training data fed to another CNN Probabilistic Classifier. The hyperparameters of the latter network has been again optimized through Optuna: the best set (scoring ≈ 0.81 in Macro F1 on the Validation Set) is shown in Table 15.

We obtain worse results if compared with the components model (Table 9): this is expected, as the number of positive labels is now a subgroup of the positive labels used to train the previous model, leading to an imbalanced dataset and a more difficult task.

The QuaNet training followed the same procedure seen in the previous experiments. The evaluation results (Table 8) are once again slightly worse, reflecting the increased difficulty of the task.

	Experiment 2 Relations	
Metric	**BL**	**CNN**
Glaucoma Test Set		
Macro F1	0.694	0.731
Acc. (%)	93.00	76.48
Neoplasm Test Set		
Macro F1	0.729	0.788
Acc. (%)	92.00	82.06
Mixed Test Set		
Macro F1	0.744	0.742
Acc. (%)	91.00	78.56

Table 9: Comparison of evaluation results on relation classification between the baseline classifier (BL) and the CNN classifier in Experiment 2. Baseline results are based on components detected by the sequence tagging model with over 75% overlap between ground truth and prediction.

5.3 Experiment 3: Arguments Quantification

After training our QuaNets, we attempted to leverage their acquired knowledge by creating an ensemble. As already described in 3.2.2, we used the trained models as backbones, removing their classification heads and extracting their quantification embeddings, which we then concatenated to form a unified representation. In particular, in this first variant, we leverage the Components and Relations architectures described respectively in Experiment 1 (5.1) and Experiment 2 (5.2) .

The preprocessing for this experiment follows the exact methodology of the previous experiments, ensuring consistency in how the text is tokenized and structured before being fed into the models. Each abstract is processed through the two QuaNets, and the resulting quantification embeddings from both models are concatenated into a single feature vector; the backbones' layers are progressively frozen based on a percentage, with the optimal value determined through optimization.

Once this feature extraction step is complete, a new classification head is trained. The objective of this head is to predict the number of arguments occurring in an abstract. Initially, we approached this as a regression problem, but saw an improvement when we changed it to a classification one, where we have four total classes as targets: 0, 1, 2, and 3 arguments. This approach resulted in strongly imbalanced sets (see Table 10), with distributions shifting towards the middle classes, 1 and 2.

Set/Metric	N.Args 0	N.Args 1	N.Args 2	N.Args 3
Training Set	10	209	41	4
Validation Set	1	103	28	4
Glaucoma Test Set	4	61	33	2
Neoplasm Test Set	1	73	20	6
Mixed Test Set	3	74	18	5

Table 10: Label statistics for arguments quantification.

To counteract the imbalance of the dataset we computed *class weights* and used them to weight CrossEntropy and to perform *Weighted Random Sampling* (Hughes, 2022); these two techniques are not performed together, meaning that we don't weight the loss criterion if we already counteract the imbalance using the random sampling technique.

Optuna was once again used for hyperparameter optimization. The best set of parameters is shown in Table 16.

5.3.1 Experiment 3-A: Arguments Quantification - Claims and Premises quantifiers as backbones

This experiment differs from Experiment 3 as it replaces the components QuaNet with the two models trained in 5.1.1, one trained to quantify only claims, the other only premises; we hope that this gives the resulting ensemble more expressiveness in determining how many arguments an abstract

contains.

We optimized again the set of hyperparameters (Table 16) with Optuna and trained the new classification head using the same labeling scheme followed in 5.3. The results are shown in Table 11.

Metric	Baseline	Ex. 3	Ex. 3-A
Glaucoma Test Set			
Accuracy (%)	68.00	59.38	62.50
Macro F1	0.440	0.280	0.265
Weighted F1	0.650	0.575	0.575
Neoplasm Test Set			
Accuracy (%)	65.00	47.92	57.29
Macro F1	0.382	0.197	0.268
Weighted F1	0.670	0.498	0.559
Mixed Test Set			
Accuracy (%)	65.00	62.50	60.42
Macro F1	0.432	0.268	0.468
Weighted F1	0.680	0.625	0.625

Table 11: Comparison of Evaluation Results across Baseline A, Baseline B, Experiment 3, and Experiment 3-A.

5.3.2 Experiment 3-B: Argument Detector with New Dataset Split

Prompted by the better results obtained on the mixed test set, we decided to follow the intuition that exposing the model to documents related to a diverse range of diseases could improve generalization, hence we designed this experiment. Here, we retained the ensemble structure employed in Experiment 3-A but utilized a new split of the datasets.

The distribution of labels in the newly created splits is shown in Table 12. While this re-splitting process resulted in a more balanced dataset from a content point of view, a strong numerical imbalance is still present in the arguments count, similar to the original subdivision.

Set/Metric	N.Args 0	N.Args 1	N.Args 2	N.Args 3
Training Set	13	322	81	12
Validation Set	1	80	22	4
Test Set	5	100	26	3

Table 12: Label distribution for Experiment 3-B.

Changing the dataset split led to noticeable improvements in the results, with performance metrics for the ensemble becoming comparable to those of the baseline: this new split helped reduce some of the variance and class imbalance issues that hindered previous experiments. The results are shown in Table 13.

6 Discussion

The experiments conducted demonstrate varied performance across tasks, with promising results approaching but generally not surpassing the baseline performance.

Metric	Baseline	Ex. 3-B
Accuracy (%)	63.43	69.70
Macro F1	0.447	0.420
Weighted F1	0.650	0.673

Table 13: Comparisons of Evaluation Results on arguments quantification between Baseline and Experiment 3-B.

Both the CNNs responsible for detecting components and relations achieve results close to it, as shown in Tables 7 and 9. As CNNs require substantially less training time and exhibit much simpler architectures, they represent a valid alternative to the Probabilistic Classifiers employed in the Baseline.

A direct comparison of Argument Quantification errors is not feasible due to the completely different strategy for counting the argument in the baseline. Nevertheless, performance insights can be gleaned from Tables 11 and 13: in these experiments, the strategy of classifying and counting consistently performed better. The learned quantification embeddings, further refined through the Argument Quantifier training, probably do not differentiate abstracts containing one or two arguments in the proper way; moreover, abstracts with zero or three arguments are underrepresented, leading to higher error rates. These issues are evident in Figures 4 and 5: when the model correctly classifies abstracts with one argument, it often misclassifies those with two arguments and vice versa. Furthermore, when the model eventually learns to predict these cases correctly, it tends to classify abstracts with zero or three arguments as one of the more frequent categories. The imbalance in class distribution likely undermines performance, as suggested by those results: using a larger dataset, more balanced on the number of arguments that each abstract contatins might yield better results, along with using full text papers instead of abstracts alone. Expanding and diversifying the dataset represents a promising direction for future research: this hypothesis is supported by the improved results in Experiment 3-B, where abstracts from RCTs on different diseases are mixed.

Another consideration is on the architecture itself: although the CNNs perform reasonably well, incorporating the probabilistic classifiers in the Baseline into our architecture could enhance performances. For example, replacing the relation classi-

fier in the CNN with the more powerful classifier used in the Baseline might address the tendency of the current architecture to underestimate the true number of relations, at the cost of higher computational costs. Since the model classifies only a subset of relations, this limitation may affect the overall performance.

7 Conclusion

With an emphasis on abstracts from randomized controlled trials, this study investigated the quantification of arguments in textual datasets. We assessed the performance of QuaNet-based models and ensembles for component, relation, and argument quantification through a series of experiments: while the models achieved results close to the baseline, they did not surpass it. The performance of the CNN-based architectures demonstrated the feasibility of using simpler models for lightweight applications, though challenges remain in addressing class imbalance and distinguishing between closely related classes.

The primary limitations of this work are the reliance on imbalanced datasets and the limited expressiveness of the current model architecture. Promising directions for future research include augmenting datasets to address class imbalance, integrating more sophisticated probabilistic classifiers like the ones employed in the Baseline, and expanding the scope to include full-text papers to improve contextual understanding and generalization. By addressing these limitations, future work can build upon our findings to achieve more accurate and robust argument quantification in diverse textual datasets.

8 Links to external resources

All associated Jupyter notebooks, source code, and supplementary materials for this paper can be accessed on GitHub. The dataset used in this work is available on GitLab.

References

Hamzeh Agahi. 2019. A modified kullback–leibler divergence for non-additive measures based on choquet integral. *Fuzzy Sets and Systems*, 367:107–117. Theme: Uncertainty Management.

Antonio Bella, Cesar Ferri, José Hernández-Orallo, and María José Ramírez-Quintana. 2010. Quantification via probability estimators. In *2010 IEEE International Conference on Data Mining*, pages 737–742.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. In *EMNLP*. Association for Computational Linguistics.

G. Brambilla, A. Rosi, F. Antici, A. Galassi, D. Giansanti, F. Magurano, F. Ruggeri, P. Torroni, E. Cisbani, and M. Lippi. 2022. Argument mining as rapid screening tool of covid-19 literature quality: Preliminary evidence. *Frontiers in Public Health*, 10:945181.

Andrea Esuli, Alejandro Moreo, Fabrizio Sebastiani, and Gianluca Sperduti. 2022. A detailed overview of lequa@ clef 2022: Learning to quantify. *CLEF (Working Notes)*, pages 1849–1868.

Andrea Esuli, Alejandro Moreo Fernández, and Fabrizio Sebastiani. 2018. A recurrent neural network for sentiment quantification. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 1775–1778.

George Forman. 2008. Quantifying counts and costs via classification. *Data Mining and Knowledge Discovery*, 17:164–206.

Chris Hughes. 2022. Demystifying pytorch's weightedrandomsampler by example.

Tobias Mayer, Elena Cabrio, and Serena Villata. 2020. Transformer-based argument mining for healthcare applications. In *ECAI 2020*, pages 2108–2115. IOS Press.

Tobias Mayer, Santiago Marro, Elena Cabrio, and Serena Villata. 2021. Enhancing evidence-based medicine with natural language argumentative analysis of clinical trials. *Artificial Intelligence in Medicine*, 118:102098.

Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text chunking using transformation-based learning. *CoRR*, cmp-lg/9505040.

Fabrizio Sebastiani. 2018. Evaluation measures for quantification: An axiomatic approach. *CoRR*, abs/1809.01991.

V.N. Vapnik. 1999. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988–999.

A Appendix

A.1 Optimal Hyperparameters Configurations

Parameter	Value
Embedding Size	180
Hidden Size	269
Optimizer	Adam
Learning Rate (lr)	0.001
Scheduler	CosineAnnealingLR
$T_{\textbf{max}}$	2

Table 14: Optimal Hyperparameters for the component classification CNN found with Optuna.

Parameter	Value
Embedding Size	195
Hidden Size	278
Optimizer	Adam
Learning Rate (lr)	0.0005
Scheduler	CosineAnnealingLR
$T_{\textbf{max}}$	11

Table 15: Optimal Hyperparameters for the relation classification CNN found with Optuna.

Parameter	Ex. 3	Ex. 3-A	Ex. 3-B
ap_ff_layers	[128, 64]	[256, 128, 128]	[128, 64, 32]
apdrop_p	0.3410	0.1216	0.2402
p_frozen_layers (%)	_	0	25
c_frozen_layers (%)	50	0	0
r_frozen_layers (%)	100	50	100
optimizer	AdamW	Adam	AdamW
weight_decay	4.4896e-05	4.4740e-05	5.3774e-05
beta1	0.9238	0.9270	0.8405
beta2	0.9928	0.9819	0.9074
lr	6e-4	2e-4	7e-4
batch_size	8	8	4
WRS	True	True	False
Weighted CE	False	False	True

Table 16: Best Hyperparameters for final architecture in Experiments 3, 3-A, and 3-B.

A.2 Error Metrics Tables of QuaNets evaluations in Experiments 1, 1-A, 1-B and 2

Error Metric	Standard	ByDoc (n=1)	ByDoc (n=3)	ByDoc (n=5)	ByDoc (n=10)
Glaucoma Test Set					
AE	0.0363	0.0920	0.0638	0.0514	0.0412
RAE	0.0678	0.1804	0.1212	0.0969	0.0775
MSE	0.0013	0.0138	0.0070	0.0038	0.0026
MAE	0.0363	0.0920	0.0638	0.0514	0.0412
MRAE	0.0678	0.1804	0.1212	0.0969	0.0775
MKLD	0.0023	0.0251	0.0128	0.0067	0.0046
Neoplasm Test Set					
AE	0.0031	0.0971	0.0667	0.0524	0.0316
RAE	0.0059	0.1943	0.1281	0.0997	0.0591
MSE	0.0000	0.0164	0.0062	0.0041	0.0014
MAE	0.0031	0.0971	0.0667	0.0524	0.0316
MRAE	0.0059	0.1943	0.1281	0.0997	0.0591
MKLD	0.0000	0.0316	0.0108	0.0071	0.0023
Mixed Test Set					
AE	0.0101	0.0829	0.0571	0.0407	0.0259
RAE	0.0188	0.1648	0.1095	0.0767	0.0483
MSE	0.0001	0.0110	0.0050	0.0025	0.0008
MAE	0.0101	0.0829	0.0571	0.0407	0.0259
MRAE	0.0188	0.1648	0.1095	0.0767	0.0483
MKLD	0.0002	0.0201	0.0088	0.0043	0.0015

Table 17: Error Metrics in Test Sets Evaluations for the Components Quantifier QuaNet.

Error Metric	Standard	ByDoc (n=1)	ByDoc (n=3)	ByDoc (n=5)	ByDoc (n=10)
Glaucoma Test Set					
AE	0.0184	0.0828	0.0480	0.0309	0.0260
RAE	0.0599	0.2587	0.1494	0.0959	0.0820
MSE	0.0003	0.0108	0.0042	0.0017	0.0009
MAE	0.0184	0.0828	0.0480	0.0309	0.0260
MRAE	0.0599	0.2587	0.1494	0.0959	0.0820
MKLD	0.0011	0.0529	0.0182	0.0058	0.0029
Neoplasm Test Set					
AE	0.0401	0.1008	0.0554	0.0438	0.0399
RAE	0.1131	0.2828	0.1479	0.1194	0.1101
MSE	0.0016	0.0162	0.0050	0.0028	0.0020
MAE	0.0401	0.1008	0.0554	0.0438	0.0399
MRAE	0.1131	0.2828	0.1479	0.1194	0.1101
MKLD	0.0046	0.0674	0.0147	0.0082	0.0059
Mixed Test Set					
AE	0.0268	0.1004	0.0546	0.0480	0.0314
RAE	0.0781	0.2868	0.1523	0.1441	0.0904
MSE	0.0007	0.0166	0.0049	0.0031	0.0015
MAE	0.0268	0.1004	0.0546	0.0480	0.0314
MRAE	0.0781	0.2868	0.1523	0.1441	0.0904
MKLD	0.0021	0.0686	0.0180	0.0100	0.0045

Table 18: Error Metrics in Test Sets Evaluations for the Claims Quantifier QuaNet.

Error Metric	Standard	ByDoc (n=1)	ByDoc (n=3)	ByDoc (n=5)	ByDoc (n=10)
Glaucoma Test Set					
AE	0.0418	0.0931	0.0707	0.0601	0.0489
RAE	0.0881	0.2146	0.1575	0.1331	0.1068
MSE	0.0017	0.0153	0.0070	0.0050	0.0032
MAE	0.0418	0.0931	0.0707	0.0601	0.0489
MRAE	0.0881	0.2146	0.1575	0.1331	0.1068
MKLD	0.0033	0.0318	0.0134	0.0095	0.0062
Neoplasm Test Set					
AE	0.0055	0.0838	0.0537	0.0355	0.0284
RAE	0.0113	0.1806	0.1115	0.0740	0.0591
MSE	0.0000	0.0118	0.0047	0.0018	0.0012
MAE	0.0055	0.0838	0.0537	0.0355	0.0284
MRAE	0.0113	0.1806	0.1115	0.0740	0.0591
MKLD	0.0001	0.0258	0.0093	0.0035	0.0024
Mixed Test Set					
AE	0.0049	0.0882	0.0391	0.0371	0.0233
RAE	0.0103	0.1974	0.0839	0.0800	0.0492
MSE	0.0000	0.0131	0.0027	0.0022	0.0010
MAE	0.0049	0.0882	0.0391	0.0371	0.0233
MRAE	0.0103	0.1974	0.0839	0.0800	0.0492
MKLD	0.0000	0.0308	0.0057	0.0043	0.0019

Table 19: Error Metrics in Test Sets Evaluations for the Premises Quantifier QuaNet.

Error Metric	Standard	ByDoc (n=1)	ByDoc (n=3)	ByDoc (n=5)	ByDoc (n=10)
Glaucoma Test Set					
AE	0.1110	0.1404	0.1196	0.1125	0.1097
RAE	0.2529	0.4652	0.2893	0.2655	0.2528
MSE	0.0123	0.0338	0.0197	0.0164	0.0139
MAE	0.1110	0.1404	0.1196	0.1125	0.1097
MRAE	0.2529	0.4652	0.2893	0.2655	0.2528
MKLD	0.0233	0.0652	0.0369	0.0307	0.0261
Neoplasm Test Set					
AE	0.0175	0.1104	0.0686	0.0525	0.0465
RAE	0.0368	0.2594	0.1497	0.1098	0.0987
MSE	0.0003	0.0194	0.0071	0.0046	0.0031
MAE	0.0175	0.1104	0.0686	0.0525	0.0465
MRAE	0.0368	0.2594	0.1497	0.1098	0.0987
MKLD	0.0006	0.0435	0.0146	0.0091	0.0064
Mixed Test Set					
AE	0.0655	0.1364	0.0858	0.0741	0.0633
RAE	0.1503	0.4005	0.2134	0.1782	0.151
MSE	0.0043	0.0309	0.0114	0.0079	0.0058
MAE	0.0655	0.1364	0.0858	0.0741	0.0633
MRAE	0.1503	0.4005	0.2134	0.1782	0.1511
MKLD	0.0085	0.0630	0.0225	0.0155	0.0115

Table 20: Error Metrics in Test Sets Evaluations for the Relations Quantifier QuaNet

A.3 Training Histories

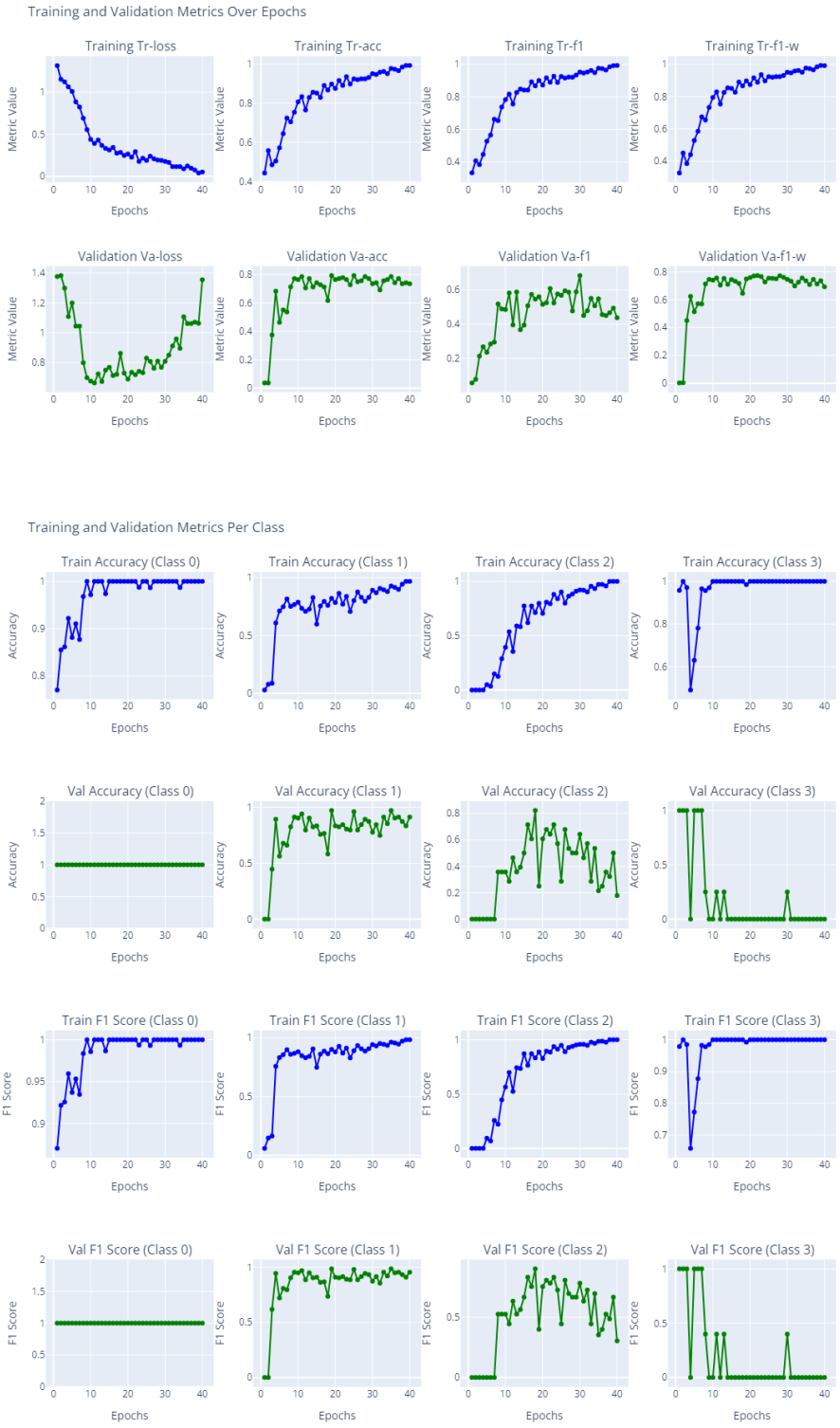

Figure 4: Training history for Experiment 3.

Figure 5: Training history for Experiment 3-A.