University of Milano-Bicocca
Department of Informatics, Systems and Communication
Master's degree program in Data Science

Multi-domain claim detection: a coreset and an external feature based approach for automated fact-checking

Supervisor: Prof. Gabriella Pasi

Co-supervisor: Prof. Marco Viviani

Co-supervisor: Dr. Sandip Modha

Co-supervisor: Dr. Marco Beltrame

Master's degree thesis by
Tommaso Redaelli
ID number 830442

Academic Year 2023-2024

A Ivano Sala per il pensiero,
A Roberto Saviano per la complessità,
A Fabrizio Tarducci per le chiavi.

E soprattutto ai miei amici
Eki, Emi, Ste, Tilo, Paolo, Leo, Ire,
Giuly, Mati, Leo, Yassine, Albi.

Abstract

This study focuses on the initial phase of automated fact-checking, the claim detection task, with the aim of developing a model capable of effectively generalizing across different linguistic styles and thematic domains while maintaining a sustainable computational complexity. By exploiting the *CheckThat!* competition datasets published in the last five years, we propose a coreset-based approach to reduce the size of the datasets without compromising their representativeness. This approach allows combining information from different domains, creating a smaller but equally representative training dataset. The use of coresets then allows the adoption of more complex models, integrating not only the representations generated by Large Language Models (LLM), but also external features, such as key concept definitions and syntactic structures. The results demonstrate the effectiveness of coresets in maintaining the representativeness of the original data, reducing noise and allowing an average performance equivalent (or even better) than the one obtained with the full datasets. Furthermore, an increase in the average performance on different datasets is observed when the model incorporates external features, improving the claim detection capabilities compared to the exclusive use of LLM representations.

Contents

1 Introduction to automated fact checking

Fact-checking has recently become a fundamental process in the field of journalism and communication. It consists of verifying the truthfulness of statements in written texts or oral speeches. This process is based on the collection and analysis of evidence to determine whether a statement is supported by reliable facts and data, clarifying whether it is true, partially true or false. The contexts in which fact-checking is applied are different, including political speeches, newspaper articles, and content published on social media.

In the digital age, the need for fact-checking has become more urgent, thanks to the speed and scale with which information can be transmitted and shared online. The spread of fake news, six times faster than that of verified content [1], and misleading information can have serious consequences, influencing public opinion, distorting political debate, and, in extreme cases, posing a risk to public safety [2] [3]. The growth of social media platforms, as well as the democratization of access to information, has facilitated access to the channels through which disinformation can spread, making it more difficult for the public to distinguish between verified facts and falsehoods. For these reasons, fact-checking is essential not only for the purpose of protecting the veracity of information, but also for safeguarding democracy and public discourse [4].

Traditionally, fact-checking has been done manually by professional journalists. It is a time-consuming and resource-intensive task when considering aspects such as finding reliable sources, analyzing evidence, and assessing the context in which claims are made [4]. The automation of fact-checking is an attempt to respond to the growing demand for fast and accurate verification of information. Automated fact-checking systems are inspired by the methodologies that professionals use to complete an entire information verification process and are usually structured in several steps:

1. **Claim Detection:**
   This stage consists of identifying claims that need to be verified. Not all claims are equally relevant or verifiable; it is therefore important to consider factors such as the *verifiability* and *check-worthiness* of the claims [5]. For example, a statement about a current event with potentially significant implications for

society is generally considered more relevant for fact-checking than a trivial or personal observation [6].

2. **Evidence Retrieval:**
   Once you have identified a claim to test, the next step is to find information and data that can confirm or deny it. This may involve searching for articles, documents, databases, or other sources that can provide concrete evidence. Automated systems often rely on advanced information retrieval techniques, such as keyword matching or the use of machine learning models to identify relevant sources [7].

3. **Claim Verification:**
   At this stage, the claim is compared to the evidence gathered to determine its veracity. This may include the use of logical inference techniques or natural language processing (NLP) models to assess whether the evidence supports or contradicts the claim [8]. For example, if a statement says that "*The population of Paris is 3 million*", the system might compare it to official demographic data to confirm or deny this figure.

4. **Justification Production:**
   Finally, it is important that the automated system not only gives a verdict on the veracity of the statement, but also provides a clear and understandable justification for its decision. This is essential to ensure transparency of the process and to convince the user of the correctness of the verification [9].

The use of automated fact-checking systems is not generally seen as a complete replacement for the work of human fact-checkers, but rather as a powerful supporting tool. Despite significant progress, fact-checking automation presents several challenges that require further research and development:

- **Understanding the context:**
  One of the most challenging problems is the ability of automated systems to understand the context in which a statement is made. Statements can be ambiguous, ironic, or context-dependent, making it difficult for an automated system to interpret them correctly without a thorough understanding of the context [4].

- **Source reliability:**
  Another important issue is the assessment of the reliability of sources. Not all information found online is equally reliable, and automated systems must be able to distinguish between authoritative sources and less reliable sources [10].

- **Produce coherent justifications:**
  It is essential that fact-checking systems not only provide a verdict, but also produce convincing and user-friendly justifications. This requires that systems be able to explain their decision-making process clearly and transparently, a task that is still under development [11].
- **Biases and impartiality:**
  Algorithms can be subject to bias, both because of the data they are trained on and because of their internal structures. It is therefore crucial to develop methods to reduce or eliminate these biases to ensure that fact-checking systems are fair and impartial [12].

1.1 Claim detection overview

The claim detection task is a crucial step in the automated fact-checking pipeline. It consists of identifying statements (claims) in a text that require verification, allowing verification resources to be focused on potentially false or misleading information. This task precedes the verification of the content and the assignment of a truthfulness label [13]. According to [14], claim detection represents an essential filter that allows fact-checkers to reduce the volume of content to be verified, focusing only on those that can have a significant impact.

A "claim" in the context of fact-checking is defined as a factual statement that can be verified as true or false. It must concern concrete facts rather than opinions, beliefs, or personal preferences [13]. The concept of check-worthy claims, as described by [15], refers to those statements that, due to their form and content, are of interest to public opinion and that, if not verified, could significantly influence the society and the public debate, and therefore deserve particular attention.

The main challenges of the claim detection task include linguistic complexity, context variability and the difference between explicit and implicit claims. In [16] the authors highlight how linguistic and cultural diversity can influence the formulation and interpretation of claims, making it difficult to develop models that are effective globally. Furthermore, according to [14], identifying claims that are implicitly linked to verifiable facts but that do not directly state them, represents a significant challenge. Semantic complexities, such as sarcasm or irony, further complicate the detection process [15]. These challenges are amplified when trying to apply models developed in one linguistic or cultural context to another.

1.2 Project's objectives and contributions

The contributions of the thesis work are summarized in the following:

1. **Generalization between datasets and therefore between different topics and textual styles:**
   A first goal of the research is to develop a model that is able to effectively generalize across different topics, while maintaining a manageable computational complexity. To achieve this, we propose using coresets to create smaller but representative datasets, which allow to merge data from various domains without losing fidelity to the original datasets.

2. **Explicit use of additional features (concepts and syntax):**
   Rather than relying exclusively on the representations automatically generated by LLMs, we propose to integrate these representations with additional information, in particular a representation of the syntactic structure of the sentence and the definitions of key concepts described in the texts.

3. **Comparison and evaluation of results between different models and approaches:**
   Finally, we report a comparison of the results obtained by different models and approaches resulting from the experiments carried out in this research. The performances obtained on datasets addressed individually and on a dataset representative of different topics are compared, both with state-of-the-art models and with custom models.

1.2.1 Thesis structure

The purpose of this first chapter [1] was to introduce the scope of the research by giving a general description of the fact-checking process, its automation and outlining in more detail the process of claim detection on which the research focuses, providing the necessary definitions and describing the current challenges in order to understand the reasons for the implementation choices in the implemented experiments. The second chapter [2] focuses on a description of the methodologies implemented to date for the claim detection task, and also provides an introductory overview of some previous methods and works that represent inspiration for the methodologies applied in this work. The third chapter [3] describes in full the procedures implemented in the various research steps, starting with the selection of the datasets, the selection of the models used and the customisation techniques adopted, attaching the results for each of them. Finally, the last chapter [4] aims to discuss the reasons for the results obtained, identifying the limitations of the used techniques and possible future improvements that can be adopted.

2 Background and related works

Below we provide a summary of the evolution of claim detection approaches; then we describe in detail the *CheckThat!* competition within the *CLEF ("Conference and Labs of the Evaluation Forum")* whose datasets has been used for our experiments and evaluations. Finally, a brief reference to some works that have been a source of inspiration for some key steps during the realization of the proposed solutions.

2.1 Evolution of claim detection systems

In the past, the claim detection task has been approached using traditional machine learning techniques, such as models based on manually extracted linguistic features [13]. These approaches included the use of keywords and other linguistic features that indicated the presence of factual claims. However, in recent years, there has been a shift towards deep learning approaches, which use neural networks to learn more complex representations from the given texts. For example, in [17] a neural network-based method for ranking statements based on their check-worthiness is proposed, using weak supervision to determine the relevance of sentences. At the same time, Support Vector Machines (SVM) models have been used in combination with feature engineering techniques, as in the case of [18], where syntactic and contextual features have been fused to improve the detection of claims in social media. In the context of the ClaimBuster system [13], the use of deep neural networks and more recently transformers, such as BERT [19], has enabled the recognition of complex claims within political debates. In [16] the use of multilingual transformers such as XLM-R [20] is explored to improve claim detection in multilingual and cross-lingual contexts. These models enable learning common representations of sentences across languages, improving the ability to detect claims in contexts where training data is limited or unevenly distributed. Finally, in [14] an annotation scheme has been developed to create a consistent benchmark for automated claims detection, using both supervised and unsupervised approaches to reliably identify and prioritize claims in various contexts.

These developments represent the evolution in the techniques in the field of claims detection, highlighting the shift from traditional techniques to more advanced methods based on deep learning and transformers, with an increasing emphasis on multilingual techniques, data augmentation and multilingual transformers. However, despite the progress, there are still significant limitations in current claim detection models. One of the main ones is the dependence on training data, which is often limited to a single domain or language, limiting the generalizability of models [16]. In [13] the authors highlight how deep learning models, while powerful, can lack transparency and interpretability, making it difficult to understand why a particular claim has been identified as worthy of verification. There is also the problem of the difficulty of detecting implicit or ambiguous claims, which require a deeper understanding of the context and linguistic nuances [3]. These limitations indicate that, despite the progress, the claim detection task remains complex and requires further research to improve the accuracy and reliability of the models.

2.2 *CheckThat!* competition

One of the reference points for the automation of fact-checking is the *CheckThat!* competition, in particular its first task is dedicated to claim detection and over the years it has evolved starting from an initial phase of development of traditional machine learning models up to the integration of more advanced technologies such as those related to deep learning. The 2020 task involved classifying texts related to the US elections, mainly from social media platforms. In 2021, the texts provided mainly came from speeches and statements by political leaders. In 2022, there was a greater emphasis on the topic of Covid-19. In 2023, the sources expanded to online news and newspaper articles, while in 2024 the focus shifted to generalization on new topics, especially regarding global emergencies, political disinformation, and environmental issues.

2.2.1 State of the art techniques

In this section, we provide an overview of the solutions adopted by participants in Task 1 of *CheckThat!* 2023 and 2024. Most participants used pre-trained Transformer models, combining them with standard data augmentation and preprocessing techniques. Teams have explored the use of large-scale models such as Llama and GPT, and some have also combined data refinement techniques with linguistic transformations, such as multilingual translation or the use of additional training data to improve model generalizability.

1. **FactFinders at CheckThat! 2024:** [21]
   The FactFinders team focused on applying eight open-source LLM models, such as Llama2 and Mistral, to detect fact-checkable claims in political transcripts. They used fine-tuning and prompt engineering to optimize these models, but the real innovation was their use of a two-stage data selection

strategy based on verb types in the texts. This technique allowed to filter the training data, reducing it by 44%, but maintaining a competitive performance, while reducing training costs. This approach allowed the team to achieve first place in the check-worthiness task for the English language, demonstrating that it is possible to achieve excellent results with a small, high-quality dataset.

2. **IAI Group at CheckThat! 2024:** [22]
   The IAI group took a multilingual approach to the statement detection task, working on English, Dutch, and Arabic datasets. They used pre-trained Transformer models such as RoBERTa, and combined fine-tuning techniques with data augmentation and transfer learning. An interesting part of their methodology is the use of machine translation for the Arabic dataset: the Arabic test data was translated into English, allowing the team to use a GPT-3.5 model already fine-tuned in English for classification. This approach yielded impressive results, placing the team first in Arabic and third in Dutch.

3. **Adapter fusion for check-worthiness detection:** [23]
   This work proposes an adapter fusion model, a novel approach that combines a task adapter with a Named Entity Recognition adapter. The use of adapters represents a more resource-efficient alternative to the complete fine-tuning of Transformer models. The authors demonstrate that named entities, such as people's names, dates, or historical events, play a crucial role in determining the merit of a claim, significantly improving the model's accuracy.

4. **TurQUaz at CheckThat! 2024:** [24]
   The TurQUaz team adopted a hybrid approach based on a fine-tuned XLM-R classifier, combined with in-context learning using trained models such as GPT-3.5 and GPT-4. For the English and Arabic datasets, they used a combination of predictions from an XLM-R classifier and LLM models, aggregating the results with a weighted averaging method based on the F1 score. For Dutch, however, they relied exclusively on in-context learning, using a majority voting system among the models to decide the final label.

Although the proposed solutions represent a significant advancement in the state of the art for the claim detection task, they are tested on a single evaluation dataset provided in the competition, thus limiting the versatiliy of their evaluation. In particular, most of the methods used are based on classification models that operate exclusively on the analysis of the provided text, without including additional features that could potentially improve the performance of the system, such as metadata, contextual information or advanced linguistic structures. The focus on a single dataset and the lack of a broader exploration of other potential variables beyond the single text under consideration represents a risk in limiting the understanding of the generalization capacity of the proposed solutions. In particular, the robustness and versatility of the models with respect to different thematic topics or linguistic variations that may emerge in real contexts are not adequately assessed. Differences in tone, linguistic style, or even register of speech, which can vary significantly depending on the context in which the language is applied (e.g., political speeches, social media, or public debates), are not explored in depth. Consequently, the ability of the models to adapt and generalize to heterogeneous linguistic contexts remains an open question, leaving room for further research that can evaluate these solutions on a wider variety of data and contexts.

2.3 Our motivations for the research gap

The methodology proposed in this work aims to address more extensively the limitations emerged in current solutions, introducing an innovative approach to the construction and management of the dataset for the claim detection task. The main goal is to develop a database that includes texts from a variety of topics and linguistic styles, thus allowing a more in-depth evaluation of the generalization ability of the models across diverse contexts. At the same time, it will be necessary to keep the size of the dataset small enough to ensure manageable training times, favoring the possibility of conducting multiple tests on different architectures without compromising computational efficiency. Another central aspect of the methodology concerns the introduction of techniques aimed at making the classification models used more interpretable. In particular, syntactic features and complementary information to the text will be integrated, with the aim of clearly motivating their use for this specific task. This choice not only aims to improve overall performance, but also to produce more transparent models, whose structure is explanatory of which textual characteristics are decisive in recognizing a content as worthy of verification. In this way, the model's decision-making process becomes more interpretable and understandable, providing useful insights both for improving algorithms and for more precise identification of critical statements.

To achieve these goals, the proposed methodology is based on the use of coreset extraction techniques to build representative datasets, limiting the volume of data while maintaining its representativeness in the different topics. Subsequently, methodologies will be adopted for the integration of additional features within structures based on large language models, thus allowing to enrich the textual analysis with more detailed information regarding both syntactic structures and contextual information. The following subsections describe the theoretical basis and motivations that support these methodological choices, presenting examples of relevant studies that have anticipated and demonstrated their effectiveness.

2.3.1 Extracting coresets for representative data subsets

The effectiveness in extracting a representative subset from large text datasets used for training pre-trained language models is analyzed in [25]. Training these large-scale models requires enormous amounts of data and computational resources, limiting accessibility to researchers with more modest resources. The study therefore investigates the possibility of significantly reducing the amount of data without compromising performance. The authors conducted experiments on 17 natural language processing datasets, evaluating performance on 24 metrics. The resulting representative subset achieved 90% of the performance achievable using the entire dataset, but with a reduction in dimensionality of up to two orders of magnitude. This demonstrates that it is possible to identify a representative subset capable of providing similar performance, with significant savings in resources, thus opening new opportunities to broaden access to language modeling research.

2.3.2 Integration of syntactic information into LLMs

The use of syntactic features to make LLMs more robust is also demonstrated in several projects. In [26] and [27] the use of syntactic dependency trees is proposed to better understand the linguistic structure and the relationships of terms within sentences by modifying the attention mechanisms through specific masks derived from the dependency representation. The authors have demonstrated how these measures can be more effective in cases where there are sentences with complex structures for which taking into account the syntax is an important aspect of the task.

2.3.3 Integration of external concepts into LLMs

The use of external contextual information together with LLMs representations has been a strategy used to improve performance in classification tasks, especially in cases where the texts to be classified are quite short. One of the most intuitive methods is the one proposed in [28] for which the text to be classified is composed of the original target text concatenated with the contextual textual information. Alternatively, in [29] contextual representations are used for stance detection, in particular their solution involves the injection of contextual representations into the target text representations via cross-attention mechanisms. Both works demonstrate how adding context allows for the construction of richer input, allowing for greater understanding of the text precisely because key information is made explicit.

3 Proposed methodology

In this chapter, the data and methodologies applied in the research work are discussed, first briefly describing the pipeline of the experiments conducted and then going into the details of each of the steps.

3.1 Pipeline of the performed analyses

Our research is developed following a pipeline divided into six main phases, each of which contributes to the understanding and optimization of the claim detection task, answering the research questions described in the introductory chapter. Below is a description of the steps:

1. **Dataset collection:**
   The first step required the identification of multiple datasets, each representing different subject areas and linguistic styles, in order to conduct experiments across a variety of domains. This process was fundamental for two reasons: on the one hand, to evaluate the initial ability of the models to generalize in different contexts; on the other, as described in the research objectives, to build an aggregated and complete database, able to support the training of models capable at this point of effectively addressing the problem of generalization between heterogeneous contexts, characterized by different themes and linguistic forms.

2. **Pretrained LLM selection:**
   The second step consists in evaluating several pre-trained language models. The goal is to identify the model that offers the best trade-off between performance and complexity, taking into account the specific requirements of our research. The final choice of the model is driven by metrics such as accuracy (especially on the positive class), generalization ability, and computational efficiency.

3. **Empirical evaluation of the representativeness of the datasets:**
   Once the model has been selected, we proceed with an empirical assessment of the level of representativeness of each dataset compared to the others. This is done through a cross-dataset evaluation, where we train a model on each individual dataset and test it on all the others, including itself. This

approach allows us to understand how well models trained on one dataset can generalize and perform on different datasets, revealing the compatibility and diversity between different data sources.

4. **Reducing the amount of data through coreset selection:**
   Next, we use coreset selection methods to extract representative subsets from each dataset. The goal is to verify whether the joint analysis of these subsets leads to an increase in the average performance of the selected model. This is a key step to optimize the efficiency of the training process, reducing the volume of data required without compromising the overall performance of the model.

5. **Increasing model complexity:**
   Exploiting the effectiveness of the reduced but representative coreset, we increase the complexity of the model by introducing external features. In particular, we use definitions of key concepts described in the texts and representations of syntactic structures. For each of the two additional information proposed, the extraction method of these is described, and therefore the consequent representation and integration within architectures based on LLMs.

6. **Evaluation of the different proposed architectures:**
   In the final phase, the results obtained in the different experiments are analyzed. During the description of the various phases of the research, the corresponding results are presented, allowing a comparison between the trainings performed both on the original datasets and on those built through the extraction of coresets. Starting from large language models in their basic form, up to the version that integrates both syntactic dependencies and complementary information on key concepts, it is possible to evaluate the contribution of each component of the proposed architecture, thus verifying the impact of each element on the final results.

3.2 Datasets selection

This research examines five distinct datasets from the *CheckThat!* competition over the past five years. This choice is motivated by the need to address the problem of generalization. Generalization refers to the ability of a model to apply knowledge gained from a specific data set to other, unseen data while maintaining a high level of accuracy.

Using only one dataset can lead to a model that performs well only on that particular dataset, but fails to generalize to new data from different sources or containing different topics. This is particularly problematic in the context of fact-checking, where claims can vary greatly in terms of linguistic style, topics covered, and source of the information. Working across multiple datasets that include different sources and text topics allows to capture a wider range of variation in the data, including differences in vocabulary, syntactic structure, and argumentative complexity.

Furthermore, the variety of sources and topics should help to build a more robust model, reducing the risk of overfitting on a particular style or a single thematic domain. This is essential to develop an effective claim detection system in the real world, where information comes from a variety of sources and covers a wide range of topics.

The characteristics of the datasets are shown in the table 1.

Dataset	**Sources**	**Topics**	**Size**
CLEF 2020	Twitter	U.S. election	808 records
CLEF 2021	Speech transcriptions	Politics	1150 records
CLEF 2022	Twitter	Covid-19	3568 records
CLEF 2023	Newspapers	Politics	17184 records
CLEF 2024	Newspaper, speech transcriptions	Politics, global emergencies	22832 records

Table 1: Description of the datasets used with the source of the texts contained, the topics covered and the size of the training partition.

3.3 LLMs selection

The model selection process was conducted carefully to ensure that the chosen model was the most suitable for the claim detection task. We considered and tested five different pre-trained language models:

- **BERT (Bidirectional Encoder Representations from Transformers):** [19] [30]
  A model that uses bidirectional attention to understand the context of a word based on all surrounding words in a sentence, both before and after it.
- **XML-RoBERTa:** [20] [31]
  Robust multilingual version of BERT, pre-trained on large amounts of text data in different languages, improved to understand different and complex linguistic contexts.
- **mBERT (Multilingual BERT):** [19] [32]
  A variant of BERT designed to handle text in different languages, without

specific adaptations for each language, but able to generalize well across a variety of languages.

- **BART (Bidirectional and Auto-Regressive Transformers):** [33] [34] Model that combines bidirectional encoding and autoregressive decoding techniques, proving particularly effective in text generation.
- **GPT-2 (Generative Pre-trained Transformer 2):** [35] [36] Autoregressive model that generates text by predicting the next word, based solely on the previous context, known for its ability to generate coherent and contextually relevant text.

3.3.1 Experiments setup

Model implementations were obtained from *Hugging Face*. For each model, inputs were processed through the corresponding tokenizers, which produce token-IDs along with attention masks. For text representation, we used the model's pooler-output, an aggregation of information extracted from the text, to which we added a dropout layer with a rate of 0.2 to prevent overfitting, followed by a dense layer of one unit with sigmoid activation for binary classification. The structure of a generic model trained according to these settings is shown in the figure 3.1.

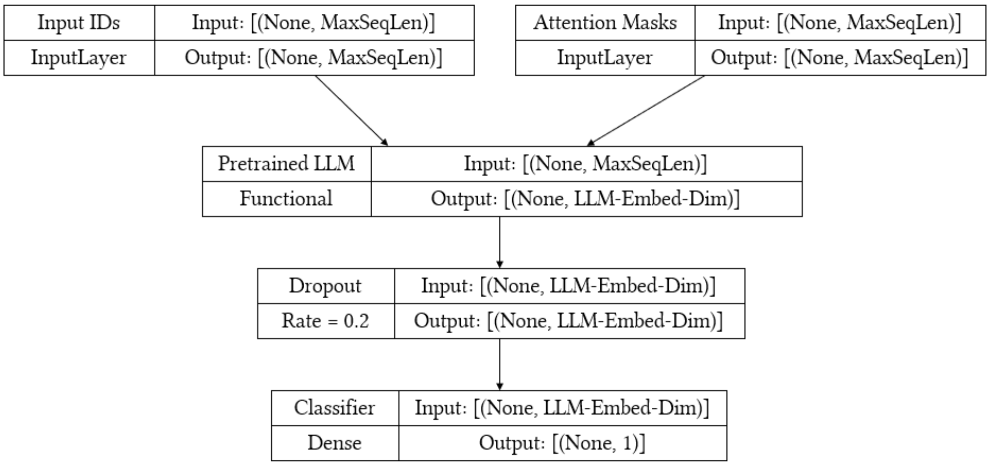
Input IDs
Input: [(None, MaxSeqLen)]
InputLayer
Output: [(None, MaxSeqLen)]
Attention Masks
Input: [(None, MaxSeqLen)]
InputLayer
Output: [(None, MaxSeqLen)]
Pretrained LLM
Input: [(None, MaxSeqLen)]
Functional
Output: [(None, LLM-Embed-Dim)]
Dropout
Input: [(None, LLM-Embed-Dim)]
Rate = 0.2
Output: [(None, LLM-Embed-Dim)]
Classifier
Input: [(None, LLM-Embed-Dim)]
Dense
Output: [(None, 1)]

Figure 3.1: Generic structure of a model that uses a pretrained language model (PLM) to obtain textual representations. The parameter *MaxSeqLen* indicates the maximum number of tokens allowed for each sentence and is set to *512* while *LLM-Embed-Dim* refers to the size of the vectors obtained from the PLM that represent the content of the texts and its value depends on the PLM used (the values related to each model are shown in the table 2)

Model	**Parameters**	**Embed dim**	**Training time**
BERT	110 M	768	3.5 h
XLM-RoBERTa	278 M	768	4.5 h
mBERT	177 M	768	4 h
BART	610 M	1024	5 h
GPT-2	108 M	768	3.5 h

Table 2: The tests were all performed on a Google-Colab environment with a T4 GPU with 12 GB of dedicated RAM

The models were trained using a learning rate of 1e-5, a batch size of 8, and a maximum of 4 epochs with early stopping to prevent overtraining. Each model was trained on each dataset individually and tested on its respective test partition.

3.3.2 Results and considerations

The metric used for the comparison was the average of the F1 scores on the positive class of each dataset. This choice was dictated by the fact that the datasets are strongly unbalanced and, in the context of claim detection, we believe it is more important to identify the greatest number of claims possible, accepting a greater number of false positives rather than incurring false negatives. The results are listed in the table 3.

Model	**2020 F1**	**2021 F1**	**2022 F1**	**2023 F1**	**2024 F1**	**Avg. F1**
BERT	0.892	**0.794**	0.786	0.875	**0.747**	0.819
XLM-RoBERTa	0.872	0.781	0.780	0.856	0.725	0.803
mBERT	0.851	0.755	0.752	0.831	0.711	0.780
BART	0.833	0.731	0.746	0.810	0.703	0.765
GPT-2	**0.906**	0.789	**0.790**	**0.885**	0.738	**<u>0.822</u>**

Table 3: Performance of the various models trained and tested on each of the five datasets examined.

We chose BERT and GPT-2 as the models on which to conduct the next experiments of the research because they achieved the best average performances among the different datasets, showing a high generalization capacity in the claim detection task. Furthermore, these models are also lighter than the others, allowing significantly less training time. This combination of high accuracy and higher efficiency made these models the best choice for balancing performance and computational costs, allowing us to conduct experiments more quickly and sustainably without compromising results.

3.4 Empirical evaluation on dataset generalization

To empirically assess the representativeness of the datasets, we trained BERT and GPT-2 on each of the five available datasets. Subsequently, for each training, we tested the model on the test partitions of the other four datasets, as well as on the test partition belonging to the same training dataset. This approach allows us to assess the transferability of the model: that is, how effective a model trained on data from specific sources and topics can be when used in different contexts.

The result of this experiment is a 5x5 matrix of F1 scores, reported in the tables 4 and 5, where the rows represent the training dataset and the columns the testing datasets. It highlights the relationships between datasets, showing how some are representative of others and how certain datasets are more unique and difficult to generalize. This analysis is useful for understanding the compatibilities between datasets and for identifying potential difficulties when a model trained in one context is applied in different domains.

	2020 F1	**2021 F1**	**2022 F1**	**2023 F1**	**2024 F1**	**Avg. F1**
Trained 2020	0.892	0.774	0.783	0.717	0.625	0.758
Trained 2021	**0.910**	0.794	**0.808**	0.683	0.572	0.753
Trained 2022	0.843	**0.957**	0.796	0.686	0.604	0.775
Trained 2023	0.721	0.793	0.730	**0.875**	**0.755**	0.775
Trained 2024	0.752	0.810	0.722	0.873	0.747	**<u>0.781</u>**

Table 4: Performance of the BERT model trained and tested on each of the five datasets under consideration.

	2020 F1	**2021 F1**	**2022 F1**	**2023 F1**	**2024 F1**	**Avg. F1**
Trained 2020	0.906	0.767	0.788	0.728	0.614	0.761
Trained 2021	**0.922**	0.789	**0.811**	0.690	0.566	0.756
Trained 2022	0.854	**0.952**	0.790	0.694	0.591	0.776
Trained 2023	0.731	0.789	0.732	**0.885**	0.742	0.776
Trained 2024	0.758	0.806	0.728	0.882	**0.738**	**<u>0.782</u>**

Table 5: Performance of the GPT model trained and tested on each of the five datasets under consideration.

3.5 Coreset extraction

A coreset is a reduced weighted subset of an original dataset, selected so as to approximate the behavior of the entire dataset with respect to a specific objective function. This subset is chosen such that the learning algorithm, when run on it, produces results similar to those that would be obtained by working on the entire dataset. Coresets are particularly useful in machine learning contexts when the size of the dataset can make processing computationally expensive. The idea behind a coreset is to reduce computational complexity without losing the quality of the predictions the model can make. In practice, this means that the computing time, memory and resources required to process the dataset can be drastically reduced, making it possible to use more complex algorithms even on less powerful hardware.

In our specific case, we work on five distinct datasets, each of which has different characteristics in terms of text sources and topics covered. By merging the five datasets into one, we could address the data variety problem, but at the cost of significantly increasing computational time and introducing noise into the dataset. This noise could arise from the presence of irrelevant or non-representative data from each of the datasets, which could negatively influence the model learning. The use of coresets in this context allows us to create a reduced dataset, composed of a subset of the original data, which is representative of the originals. Through this method, the goal is therefore to obtain a more efficient and performing model, capable of managing the diversity of texts without being conditioned by redundant or irrelevant data.

3.5.1 Datapoints selection method

The coreset construction in our project follows a naive procedure, inspired by the method described in chapter 2.1 of [37]. The procedure is described step by step below:

1. **Calculating the average of the datapoint representations:**
   Given a dataset, the first step is to calculate the average of the representations (embeddings) of each datapoint. This average serves as a central reference for the entire dataset, and is used in subsequent steps to evaluate how much each individual datapoint deviates from this average.

2. **Calculating weights based on Euclidean distance:**
   Once the mean is calculated, a weight is determined for each embedding. This weight is given by the Euclidean distance of an embedding from the mean representation obtained in the previous step. The idea behind this is that

data points that are further away from the mean may be more representative of the variations within the dataset.

3. **Calculation of extraction probabilities:**
   The weights calculated in the previous step are then normalized to obtain a probability of extraction for each embedding. The normalization of the weights serves to convert the distances into probabilities, ensuring that the total sum of these is equal to 1.

4. **Extraction of the coreset's datapoints**
   Finally, an arbitrary size is chosen for the coreset, for which data points are extracted from the original dataset according to the probabilities calculated in the previous step. This sampling process, based on weighted probabilities, ensures that the data points selected for the coreset are representative of the entire dataset, but small in number.

In our project, the coreset extraction algorithm is applied following two distinct approaches:

1. **Application on the union of the original datasets:**
   In this first approach, the five original datasets are merged into a single dataset. The algorithm described above is then run on this merged dataset to extract a coreset representative of the entire data collection.

2. **Application on the original single datasets:**
   In the second approach, the algorithm is run separately on each of the five original datasets. From each dataset, a specific coreset is extracted, which represents the peculiarities of that particular dataset. Subsequently, the coresets obtained from the individual datasets are merged together to form a new composite coreset, which retains the distinctive features of each original dataset.

For both approaches the coreset size is set to 50% of the original dataset size. The coresets extracted with the two methods therefore have the same final size.

3.5.2 Coreset applications and results

The model is trained both on the coreset obtained by the union of the original datasets and on the coreset composed by the union of the individually extracted coresets. This will allow us to evaluate the effectiveness of each approach and determine which one produces the best results in terms of generalization and accuracy. Furthermore, one of the properties of coresets is that the union of two coresets is itself a coreset. By applying the algorithm separately on the individual datasets and then merging the resulting coresets, we can empirically verify this property. If the coreset resulting from the union of the coresets of the individual datasets produces similar results to the coreset obtained from the algorithm run on the unified dataset, this would confirm the validity of this property in the context of claim detection.

In table 6 we present the results obtained using the models that proved to be the best in the analysis in section section 3.3.

Dataset	**Records**	**Time**	**2020 F1**	**2021 F1**	**2022 F1**	**2023 F1**	**2024 F1**	**Avg. F1**
BERT + Whole dataset	44 K	3.5 h	0.830	0.943	0.750	0.860	0.712	0.819
GPT-2 + Whole dataset	44 K	3.5 h	0.885	0.932	0.763	0.854	0.701	0.827
BERT + Whole-Coreset	22 K	2 h	0.848	0.880	0.772	0.848	0.691	0.808
GPT-2 + Whole-Coreset	22 K	2 h	0.911	0.873	0.792	0.845	0.682	0.821
BERT + Union-Coreset	22 K	2 h	0.840	0.884	0.773	0.864	0.746	**0.822**
GPT-2 + Union-Coreset	22 K	2 h	0.928	0.868	0.808	0.853	0.710	**0.833**

Table 6: The results described in each row refer respectively to a model (BERT or GPT-2) trained on the union of the five datasets; on a coreset extracted from the union of the five datasets; and on a dataset given by the union of the coresets extracted individually from each of the five datasets.

The results show that working on coreset is functional, confirming that it is possible to obtain performances on average comparable to those obtained with the entire dataset, but with a significant reduction in computation times and resources required. Furthermore, the property that the union of two coresets is itself a coreset has been empirically verified. The performance obtained by the union of the coresets of the individual datasets was found to be very similar to that obtained by the coreset extracted from the entire unified dataset.

Additionally, it is interesting to note that the model trained on the union of the coresets achieved on average higher performance not only than the coreset extracted from the unified dataset, but also than the model trained on the entire original dataset. This tells us that the proportion of noisy or redundant data points has probably been reduced, causing an improvement in the quality of the model, which can be said to have been trained on truly informative and representative data. For this reason, this last dataset is used as a training dataset for the next experiments that see an increase in the complexity of the model with the explicit use of new features.

3.6 Exploiting syntactic dependencies

This section describes the technique by which we use syntactic features explicitly together with an LLM for the claim detection task. The description of the methodology is divided into an introduction to syntactic dependencies with a description of what they represent and why they can be useful for our investigation. The implementation process within a model is then defined, ending with a discussion of the results obtained following this approach.

3.6.1 Brief description

Syntactic dependencies are a fundamental concept in natural language processing and refer to the grammatical relationships that exist between words in a sentence. These relationships establish a hierarchical structure that allows us to understand how words connect to each other to express meaning. A syntactic dependency is a direct relationship between two words, where the first word is considered the "head" and the second is called the "dependent".

There are different types of syntactic dependencies, each one expresses a different logical-grammatical function that the term assumes within the sentence. Some examples are:

- **Subject (*nsubj*):** Indicates who or what performs the action.
- **Object complements (*obj*, *iobj*, *dobj*, *pobj*):** They indicate who or what undergoes the action.
- **Clauses (*ccomp*, *xcomp*, *acl*, *advcl*, *mark*):** They indicate terms with the function of enriching or specifying the meaning of others, often expressing the meaning of causality or condition.
- **Negation modifier (*neg*):** Negates or inverts the meaning of another word.

- **Nominal or numeric modifier (*amod*, *nummod*):** They indicate respectively a quality or a quantity associated with a noun.

3.6.2 Motivation for using syntactic dependencies

Syntactic dependencies represent the grammatical relationships between words in a sentence, a factor that LLMs often only address implicitly. By using dependency parsing, it is possible to explicitly describe the hierarchical structure of the sentence, highlighting the main subjects that contribute to outlining the overall meaning of the text as well as defining which subjects are related, above all, with which function they relate within the sentence, allowing to reduce ambiguity, especially in the presence of long and complex texts.

In the literature, syntactic dependencies have already been used for sentence classification tasks, for example in [38] syntactic dependencies allowed to identify specific relations between aspects and sentiment terms, making an improvement in the model's ability to distinguish towards which aspects certain sentiments were addressed. In [39], among others features of a text, syntactic information is also used through dependency parsing to classify the topics of the texts.

Within the specific task of claim detection, the use of syntactic dependencies can help distinguish which are and which are not the statements to pay attention to. A claim is an objective, verifiable and free of personal judgments assertion, in order to detect it, in addition to the semantic meaning of the words it can be useful to consider the function of the relationships between them. For example, understanding whether the action performed or undergone by the subject is a predicate that refers to something subjective, arbitrary or not, helps us understand the degree of impartiality of a sentence. Furthermore, in the categorization of claim types proposed in [14] two of the major types are described as *"Quantity"* and *"Causal"*, two concepts that can be well represented by syntactic dependencies.

3.6.3 Syntactic dependency representation

Syntactic dependencies represent hierarchical relationships between words in a sentence, where a central word, often a verb, acts as the root, while other words are attached to it as dependents. This hierarchical structure can be naturally modeled as a dependency tree. In a dependency tree, each node represents a word, and the arcs between the nodes represent the directional syntactic relations that connect the head to its dependents. A tree structure, as in the figure 3.2, is therefore the most immediate representation, however it can be limited as it does not fully exploit the potential of neural networks that tipically work with different structures (vectors and matrices).

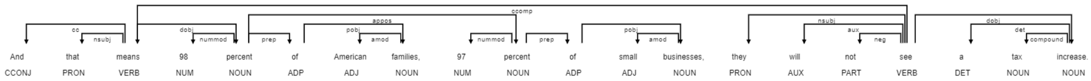

Figure 3.2: Visualization of syntactic dependencies in a sentence, showing the relationships and their type between the terms in the sentence.

Transforming the tree into a graph allows us to represent dependencies by an adjacency matrix. The adjacency matrix is a mathematical representation that captures the connectivity of the graph. An adiacency matrix is a square matrix in which both rows and columns correspond to the words of the sentence. An element of the matrix $A_{i,j}$ is non-zero if there is a direct dependency relation from word i to word j in the original tree.

Binary adiacency matrix

This connectivity criterion may be too stringent, thus, for this reason, we propose a representation in which we consider not only direct relations but also relations within subtrees. In this approach, an element $A_{i,j}$ of the adjacency matrix will be equal to 1 if one of the word i or j is part of the subtree rooted by the other word. This means that the adjacency matrix will not only reflect the immediate connections between words, but will also capture deeper relationships within the hierarchical structure of the dependency tree.

Path-length-corrected adiacency matrix

As proposed in [27], we can modify the value associated with the connection between two terms based on the distance between them. In this variant, the element $A_{i,j}$ of the adjacency matrix will not be simply binary (0 or 1), but will have a score that reflects the degree of connectivity between words i and j. The idea is to keep a weight of 1 for direct relationships, but to decrease the weight as the length of the path connecting the two terms increases, using the following modified sigmoid function:

$$f(d) = \frac{1}{1 + e^{1.25(d-4)}} \quad \text{where } d \text{ indicates the distance between two terms.}$$

As visible in the figure 3.3 the function assumes the value 1 or close to 1 if the terms i and j are the same term or the relationship between them is direct, otherwise it decreases until it provides 0 as a result for distances greater than 6, arbitrarily considered as not relevant.

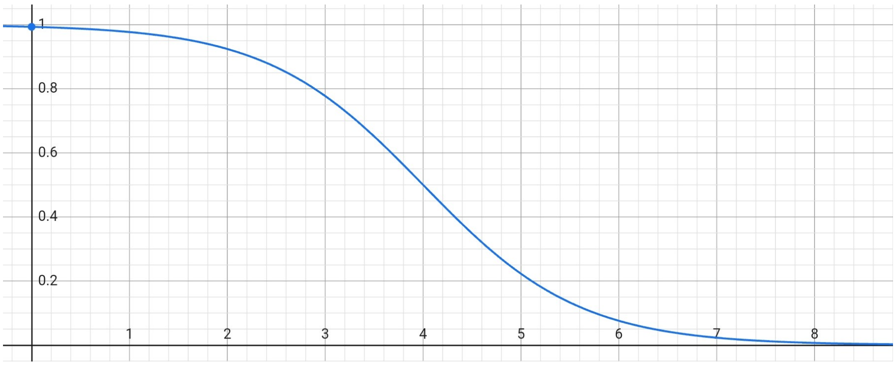

Figure 3.3: Graph of the modified sigmoid function.

3.6.4 Explicit use in LLMs

Syntactic dependencies capture the grammatical relationships between words in a sentence, providing a structured view of the connections between terms. In the context of an LLM model, the semantic meaning of words is already well represented by the embeddings generated during the pre-training process. However, these embeddings may not explicitly exploit the value of the intrinsic syntactic relations present in a sentence.

The intuition behind our approach is to exploit the adjacency matrix derived from syntactic dependencies as an attention mask. This mask serves to further refine the word embeddings, by directing the model's attention more specifically to the relevant syntactic relations, rather than considering all possible relations indiscriminately. However, we cannot simply replace the traditional LLM approach, where the attention mask relates all terms to all terms. Doing so would limit the model, depriving it of its ability to capture less obvious but still important semantic connections. Therefore, our method involves a combination of the two approaches.

After getting the word embeddings from the LLM, they are used as input to a second module. This module consists of a series of Transformer-Encoder [40] blocks, which use the syntactic dependency adjacency matrix as an attention mask. In this way, the module does not simply capture the general semantics of words, but refines embeddings based on specific syntactic relationships, such as those between subject and verb, object and verb, and so on.

To obtain a comprehensive representation of the sentence, the original embed representations generated by the LLM are then combined with the refined ones through the attention mask-based Transformer module. This combination is done by a weighted sum, governed by a λ parameter that varies in the interval [0;1].

More specifically, the original LLM representation is multiplied by λ, while the syntactic dependency-based representation is multiplied by 1-λ. The sum of these two representations gives rise to the final representation. This weighted combination mechanism makes it possible to dynamically adjust the level of influence that each of the two techniques exerts on the final representation. The structure of the model is represented in the figure 3.4.

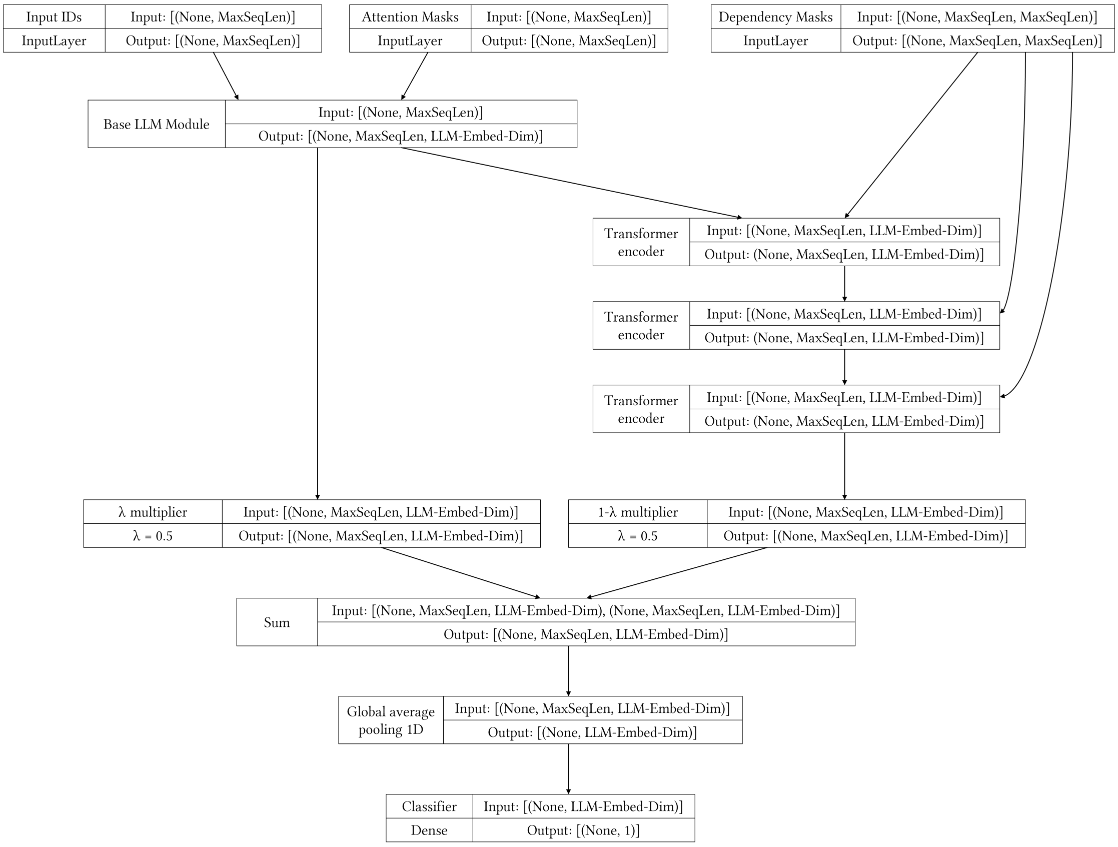
Input IDs
Input: [(None, MaxSeqLen)]
InputLayer
Output: [(None, MaxSeqLen)]
Attention Masks
Input: [(None, MaxSeqLen)]
InputLayer
Output: [(None, MaxSeqLen)]
Dependency Masks
Input: [(None, MaxSeqLen, MaxSeqLen)]
InputLayer
Output: [(None, MaxSeqLen, MaxSeqLen)]
Base LLM Module
Input: [(None, MaxSeqLen)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Transformer encoder
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Transformer encoder
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Transformer encoder
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
λ multiplier
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
λ = 0.5
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
1-λ multiplier
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
λ = 0.5
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Sum
Input: [(None, MaxSeqLen, LLM-Embed-Dim), (None, MaxSeqLen, LLM-Embed-Dim)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Global average pooling 1D
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: [(None, LLM-Embed-Dim)]
Classifier
Input: [(None, LLM-Embed-Dim)]
Dense
Output: [(None, 1)]

Figure 3.4: Model structure using syntactic dependencies.

3.6.5 Evaluation and results

The model then is trained on the dataset constructed by the union of the coresets extracted from each of the original *CheckThat!* competition datasets obtained in the 3.5.2 section. Once it is trained, the model is evaluated on the original test partitions of each of the five datasets. The results are described in the table 7.

Model	Parameters	Time	2020 F1	2021 F1	2022 F1	2023 F1	2024 F1	Avg. F1
BERT	112 M	2.5 h	0.851	0.900	0.778	0.892	0.734	0.8313
GPT-2	110 M	2.5 h	0.891	0.871	0.787	0.904	0.743	0.8393

Table 7: Performance of syntactic dependency integrating models trained and tested on each of the five datasets under consideration.

3.7 Integration of contextual informations

In the next chapter, the integration of contextual information to enrich the meaning of the main concepts of a sentence will be examined. The process of identifying key concepts, extracting their definitions, and how this information can be combined with an LLM to improve the claim detection task will be outlined. Finally, the obtained results are described and discussed.

3.7.1 Motivation for concept integration

Integrating contextual information into a pre-trained language model (LLM) can offer significant benefits, especially when dealing with short text classification. LLMs are certainly good at capturing the meaning of words and sentences within a given context, but their ability to fully understand the meaning of a sentence may be limited when the context is restricted or even not explicitly present in the text. Adding external contextual information allows to enrich the semantic interpretation of key concepts, with the consequent possibility of improving the classification accuracy.

In the literature, as demonstrated in [29], the integration of contextual information allows to obtain a more robust representation of the text, overcoming the limitations that emerge when considering only the contents of the sentence in an isolated way. It is shown that, for tasks such as cross-target stance detection, the incorporation of external context improves the model's ability to correctly understand and classify the expressed stance in relation to different targets, even when the sentences are short and ambiguous. Recalling one of the definitions of a check-worthy claim as *"an assertion whose truthfulness is of interest to the public or which may have dangerous implications for public opinion"* then the accuracy in detecting such claims may depend not only on the isolated processing of the text but also on the in-depth understanding of what the key concepts described in it represent.

3.7.2 Collecting concept information

The next section illustrates the process of obtaining information on key concepts within a sentence. The process is divided into two stages. In the first, an analysis integrating syntactic and semantic aspects is carried out, with the aim of identifying the key concepts in the text. This combination allows us to recognize which elements of the sentence represent the main ideas. In the second phase, you proceed to obtain the definitions of these key concepts, through access to external resources of information.

Keyphrases extraction

The approach used to identify key concepts in a sentence falls into the family of keyphrase extraction tasks. Keyphrases are expressions or keywords that concisely represent the essential content of a text, capturing the most relevant concepts. Their extraction process, inspired to [41], is structured in two main phases: a syntactic analysis and a semantic analysis, and is described in the following points:

1. **Syntactic analysis:**

   1.1. **Obtaining PosTags:**
   PosTags, (part-of-speech tags), are labels that identify the grammatical category of each word in a sentence, such as nouns, verbs, adjectives, etc. They are obtained for each term in a sentence.

   1.2. **PosTag pattern search:**
   Specific patterns are searched for which often correspond to key concepts. This is because key concepts are frequently found in the form of nouns or combinations of nouns with verbs and adjectives. The goal is to identify portions of text that could represent important concepts, such as noun phrases or adjective-verb combinations. The searched patterns are listed below:

   - **(NOUN | PROPN) * :** A common noun or proper noun or a succession of common noun and/or proper noun.
   - **(ADJ) . (NOUN | PROPN) :** An adjective followed by a common or proper noun.

   The text portions identified through syntactic analysis are considered as candidate keyphrases. However, these can be numerous and not all of them can be relevant to the overall meaning of the sentence. They are then filtered through the next semantic analysis procedure.

2. **Semantic analysis:**

    2.1. **Embedding representation:**
    We use BERT to generate embeddings, for the whole phrase and for each of its candidate keyphrases.

    2.2. **Top-K selection:**
    The cosine distance between the sentence embedding and those of the candidate keyphrases is measured, selecting the *top-k* most representative ones, where k is arbitrarily chosen equal to 2. During this process, we give precedence to proper names and discard candidates that are contained in other candidates that are longer and multi-term names.

Some examples of keyphrases obtained according to this procedure are presented in the table 8, while in the table 9 some statistics on the frequency of the extracted keyphrases are reported.

Sentence	**Extracted concepts**
Urgent need to ensure fair and equitable distribution of #COVID19 vaccines. Our @ifrc plan focuses on ensuring that, once received, vaccines reach those who need them most. We need to quickly fill this deadly gap in global immunization policy and funding.	global immunization ; covid19 vaccines
I supported him on NAFTA and GATT.	NAFTA ; GATT
DEVELOPING: Secret Service COVID-19 Phishing alert @CBSNews Cyber criminal emails pose as legitimate medical or health groups. Unsuspecting victims open attachment “causing malware to infect their system” or compromised “login credentials”	Phishing alert ; login credentials
Antifa is an idea, not an organization...	Antifa ; idea
And if there is any inequities in oil or any other commodity, then I would vote to close that loophole, I have voted in the past to reduce the depletion allowance for the largest producers; for those from five million dollars down, to maintain it at twenty-seven and a half per cent.	inequities ; depletion allowance
The inflation rate under Kennedy and Johnson was about 2 percent - one-third what it is under this administration.	inflation rate

Table 8: Examples of key concepts extracted from some training texts.

	No concepts	**At least 1 concept**	**Total**
Non claim	4115	9197	13312
Claim	1317	7405	8722
Total	5432	16602	22034

Table 9: Frequencies of sentences for which key concepts have or have not been extracted, divided by class to which they belong.

Gathering keyprhases definition

Once the key concepts within a sentence have been identified, the next step is to obtain precise definitions for each of them. This process takes advantage of access to external information resources, such as Wikipedia and Google, using web scraping techniques. The goal is to ensure that each concept is accompanied by a definition that clarifies its meaning in context. The method followed to obtain definitions of keyphrases is divided into the following steps:

1. **Search for the term on Google Search:**
   For each identified keyphrase, a Google search is performed using a query consisting of the term itself.

2. **Getting the Wiki summary:**
   The featured summary proposed by Google, usually derived from Wikipedia, is extracted first. This summary often provides a concise and accurate definition of the term.

3. **Explicit search for definitions:**
   If the initial search produces no results, we proceed with a more specific query in the form *"Definition of <TERM>"*.

4. **Getting definitions proposed by Google:**
   If Google features a prominent definition of the term, often derived from third-party sites such as online dictionaries or encyclopedias, this is then collected.

Examples of the collected definitions are presented in the table 10, while the table 11 describes the percentage of successfully collected definitions compared to the total number of extracted concepts.

Concept	**Definition**
Global immunization	CDC's Global Immunization Division (GID) works to protect people worldwide from vaccine-preventable diseases, disabilities, and death. CDC's Work in Global Immunization. Vaccine-Preventable Diseases.
NAFTA	The North American Free Trade Agreement was an agreement signed by Canada, Mexico, and the United States that created a trilateral trade bloc in North America.
GATT	The General Agreement on Tariffs and Trade is a legal agreement between many countries, whose overall purpose was to promote international trade by reducing or eliminating trade barriers such as tariffs or quotas.
Phishing alert	Phishing is a form of social engineering and a scam where attackers deceive people into revealing sensitive information or installing malware such as viruses, worms, adware, or ransomware.
Antifa	Antifa is a left-wing anti-fascist and anti-racist political movement in the United States. It consists of a highly decentralized array of autonomous groups that use nonviolent direct action [...] to achieve their aims.
Depletion allowance	The oil depletion allowance in American tax law is an allowance claimable by anyone with an economic interest in a mineral deposit or standing timber. The principle is that the asset is a capital investment that is a wasting asset, and therefore depreciation can reasonably be offset against income.
Inflation rate	Inflation is the rate of increase in prices over a given period of time. Inflation is typically a broad measure, such as the overall increase in prices or the increase in the cost of living in a country.

Table 10: Examples of definition extracted for some key concepts.

	No definitions	**At least 1 definition**	**Total**
Non claim	6602	2595	9197
Claim	5298	2107	7405
Total	11900	4702	16602

Table 11: Frequencies of sentences from which at least one concept has been extracted for which at least one definition has or has not been obtained through access to external resources, divided by class.

3.7.3 Injecting concept definitions into LLM

To integrate key concept definitions directly into the target sentence embeddings, we follow a process inspired to [29] based on a cross-attention injection mechanism, which allows the model to enrich its semantic understanding of the sentence using the additional conceptual information. The method consists of the following steps:

1. **Getting definition embeddings:**
   A parallel instance of the same LLM architecture is used to obtain the embeddings of the key concept definitions. The set of definitions related to the key concepts of a sentence is provided to the model in a textual format structured as follows:

   “ Definition of <Concept-1>: <Definition of the first concept>
   ...
   Definition of <Concept-N>: <Definition of the N-th concept> ”

2. **Cross-attention based injection:**
   Once the target sentence and definition embeddings are obtained, they are combined through the following steps:

   2.1. **Self-attention calculation:**
   For both embeddings (target sentence and definitions), self-attention is calculated.

   2.2. **Attentioning concepts to the target sentence**:
   The target sentence embeddings are used as *Key* and *Value* for the cross-attention calculation with the definition embeddings, which act as *Queries*. This should allow to calibrate the concept representations based on those of the target sentence.

   2.3. **Attentioning the target sentence to concepts:**
   The embeddings related to the definitions obtained in the previous point are used as *Key* and *Value* for a second cross-attention, with the target sentence used as *Query*. This second step would allow to modify the representations of the terms of the target sentence enriching them with the meaning of the concepts whose representation was previously focused on the context provided by the target sentence itself.

   2.4. **Embedding for final classification:**
   If no definitions of key concepts are available for a target sentence, the model is equipped with a conditional layer before the classification stage. This layer automatically selects the representation to be used. It outputs

embeddings enriched through the injection mechanism if definitions are available, otherwise the original representation of the target sentence.

The overall structure of the model built with this mechanism is represented by the figure 3.5.

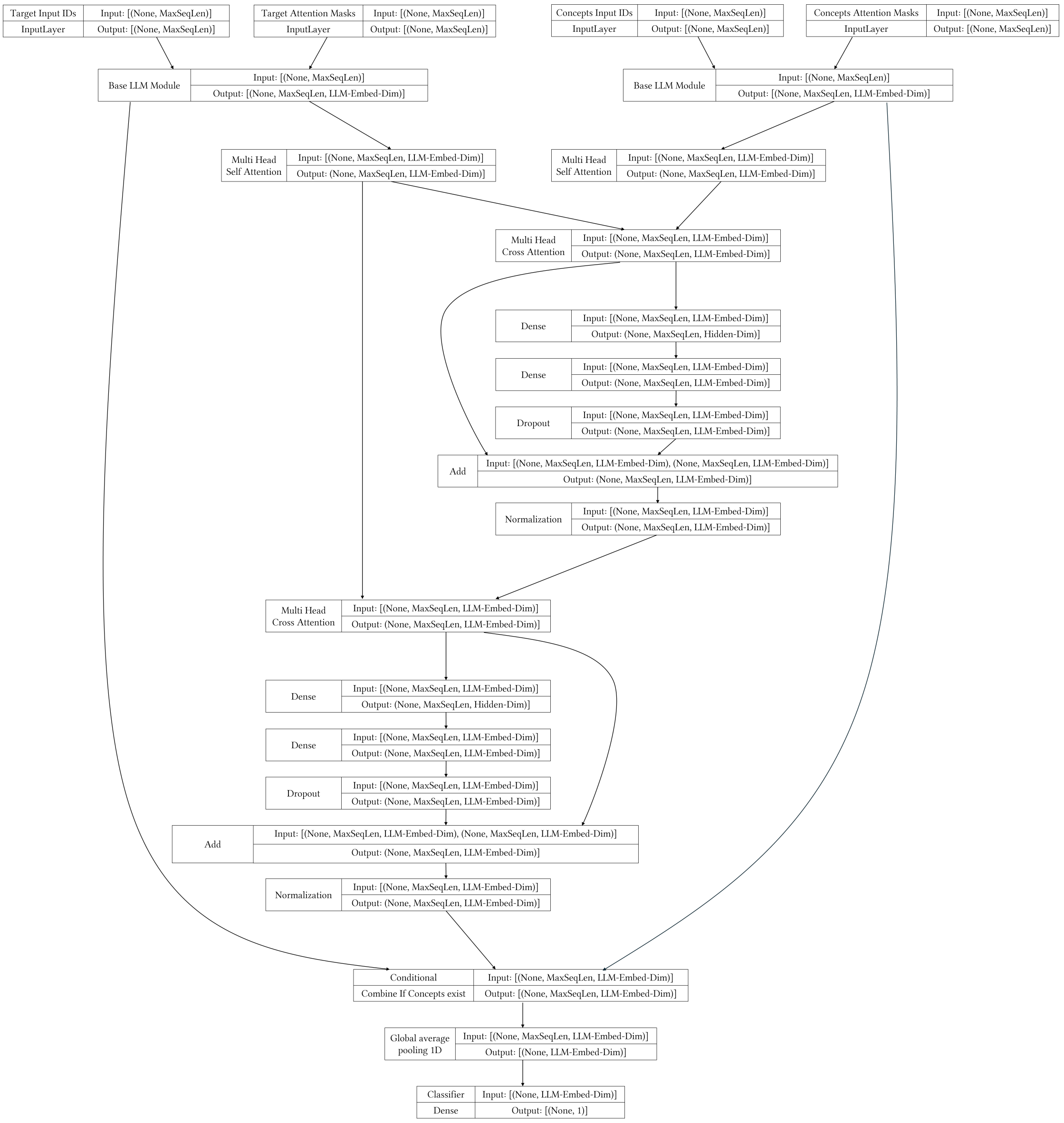
Target Input IDs
InputLayer
Input: [(None, MaxSeqLen)]
Output: [(None, MaxSeqLen)]
Target Attention Masks
InputLayer
Input: [(None, MaxSeqLen)]
Output: [(None, MaxSeqLen)]
Concepts Input IDs
InputLayer
Input: [(None, MaxSeqLen)]
Output: [(None, MaxSeqLen)]
Concepts Attention Masks
InputLayer
Input: [(None, MaxSeqLen)]
Output: [(None, MaxSeqLen)]
Base LLM Module
Input: [(None, MaxSeqLen)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Base LLM Module
Input: [(None, MaxSeqLen)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Multi Head Self Attention
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Multi Head Self Attention
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Multi Head Cross Attention
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Dense
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, Hidden-Dim)]
Dense
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Dropout
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Add
Input: [(None, MaxSeqLen, LLM-Embed-Dim), (None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Normalization
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Multi Head Cross Attention
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Dense
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, Hidden-Dim)]
Dense
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Dropout
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Add
Input: [(None, MaxSeqLen, LLM-Embed-Dim), (None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Normalization
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Conditional
Combine If Concepts exist
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Global average pooling 1D
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: [(None, LLM-Embed-Dim)]
Classifier
Dense
Input: [(None, LLM-Embed-Dim)]
Output: [(None, 1)]

Figure 3.5: Model structure using key concept definition injection.

3.7.4 Evaluation and results

As for the training procedure adopted for the model using syntactic dependencies presented in section 3.6.5, the model is trained on the dataset produced by the union of the coresets extracted from each of the original datasets, then, it is evaluated on the test partitions of each of them. The results are described in table 12.

Model	**Parameters**	**Time**	**2020 F1**	**2021 F1**	**2022 F1**	**2023 F1**	**2024 F1**	**Avg. F1**
BERT	220 M	2.5 h	0.877	0.880	0.773	0.880	0.742	0.8301
GPT-2	216 M	2.5 h	0.890	0.897	0.782	0.899	0.735	0.8407

Table 12: Performance of models with injection of key concept definitions trained and tested on each of the five datasets under consideration.

3.8 Union of the syntactic and the conceptual module

The final model combines two distinct modules: the syntactic relations representation module and the concept injection module to create a system that bases classification on both syntactic structure and content that expands the major conceptual information of sentences.

The model consists of three main blocks:

1. **Base model for the representation of the target sentence:**
   The base model, a pre-trained LLM in the structure illustrated in the figure 3.1 of the section 3.3.1, provides an initial representation of the target sentence that will serve as a starting point for both syntactic analysis and concept injection.

2. **Syntactic relations representation module:**
   As described in the 3.6.4 section, the target sentence embeddings are refined using Transformers blocks with masks given by the syntactic dependency adjacency matrix, allowing the model to focus on the most relevant grammatical relations.

3. **Concept injection module:**
   In parallel, the concept injection mechanism exploits cross-attention to combine the embeddings of the key concept definitions with those of the target sentence. This module enriches the semantic representation of the sentence by incorporating external information that expands the meaning of the key terms.

The representations obtained from the two syntactic and conceptual modules are combined through a weighted sum, governed by a λ parameter as initially described in the section 3.6.4. Finally, before the last classification layer, the combined representations are processed by an additional dense layer, equipped with dropouts. This block provides the model with an additional stage for integrating the information produced by the two representations.

Figure 3.6 illustrates the final architecture of the complete model.

3.8.1 Evaluation and results

Again, the model is retrained on the *Union-Coreset* dataset described in the 3.5.1 section. During the training process, the weights obtained in the training of the individual modules are used and freezed, leaving as *trainable* only the final block composed of the last dense hidden layer and the classification layer. Once the training is completed, the model is then evaluated on the five test datasets separately as done in the previous experiments. The results are reported in the table 13.

Model	**Parameters**	**Time**	**2020 F1**	**2021 F1**	**2022 F1**	**2023 F1**	**2024 F1**	**Avg. F1**
BERT	430 M	3 h	0.862	0.903	0.780	0.893	0.738	0.8353
GPT-2	426 M	3 h	0.901	0.891	0.791	0.905	0.747	0.8470

Table 13: Performance of models composed of the syntactic dependency module and the key concept injection module trained and tested on each of the five datasets under consideration.

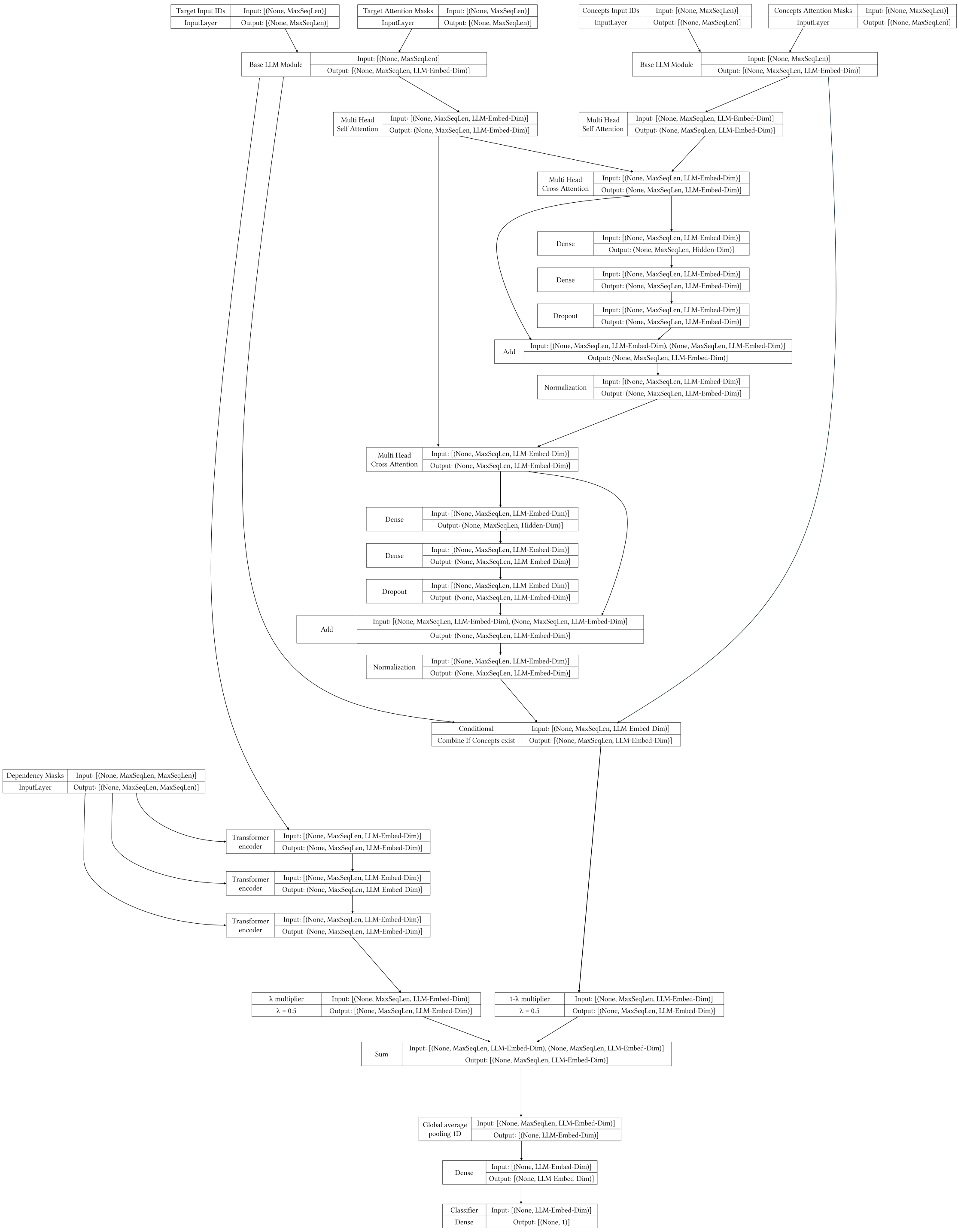
Target Input IDs
Input: [(None, MaxSeqLen)]
InputLayer
Output: [(None, MaxSeqLen)]
Target Attention Masks
Input: [(None, MaxSeqLen)]
InputLayer
Output: [(None, MaxSeqLen)]
Concepts Input IDs
Input: [(None, MaxSeqLen)]
InputLayer
Output: [(None, MaxSeqLen)]
Concepts Attention Masks
Input: [(None, MaxSeqLen)]
InputLayer
Output: [(None, MaxSeqLen)]
Base LLM Module
Input: [(None, MaxSeqLen)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Base LLM Module
Input: [(None, MaxSeqLen)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Multi Head Self Attention
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Multi Head Self Attention
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Multi Head Cross Attention
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Dense
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, Hidden-Dim)]
Dense
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Dropout
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Add
Input: [(None, MaxSeqLen, LLM-Embed-Dim), (None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Normalization
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Multi Head Cross Attention
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Dense
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, Hidden-Dim)]
Dense
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Dropout
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Add
Input: [(None, MaxSeqLen, LLM-Embed-Dim), (None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Normalization
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Conditional
Combine If Concepts exist
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Dependency Masks
Input: [(None, MaxSeqLen, MaxSeqLen)]
InputLayer
Output: [(None, MaxSeqLen, MaxSeqLen)]
Transformer encoder
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Transformer encoder
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
Transformer encoder
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: (None, MaxSeqLen, LLM-Embed-Dim)]
λ multiplier
λ = 0.5
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
1-λ multiplier
λ = 0.5
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Sum
Input: [(None, MaxSeqLen, LLM-Embed-Dim), (None, MaxSeqLen, LLM-Embed-Dim)]
Output: [(None, MaxSeqLen, LLM-Embed-Dim)]
Global average pooling 1D
Input: [(None, MaxSeqLen, LLM-Embed-Dim)]
Output: [(None, LLM-Embed-Dim)]
Dense
Input: [(None, LLM-Embed-Dim)]
Output: [(None, LLM-Embed-Dim)]
Classifier
Dense
Input: [(None, LLM-Embed-Dim)]
Output: [(None, 1)]

Figure 3.6: Model structure that uses both syntactic dependencies and injection of key concept definitions.

4 Conclusions and future works

In this study we addressed the challenge of automated claim detection in the fact-checking process, evaluating the effectiveness of a multidomain approach based on the use of coresets and the integration of external features. The analysis across multiple domains, both in terms of topics and linguistic styles, was made possible by using multiple datasets from the *CheckThat!* competition. Our research therefore explored diverse topics such as politics, Covid-19, and broader social issues, including international conflicts and environmental crises, drawing on sources with varying linguistic styles, such as public speeches, newspaper articles, and social media posts. The integration of heterogeneous sources allowed us to address one of the main obstacles in the field of claim detection: the ability to generalise models across different thematic and stylistic domains. Compared to building models that focus on a single type of texts, generalizing across a wide range of sources and topics is a more complex task. However, this challenge is fundamental to broaden the applicability of the claim detection process, allowing the model to operate effectively in different contexts without having to implement new ad hoc solutions for each specific need.

Working with different datasets has highlighted some key challenges. The first is the feasibility of developing and training a model in a reasonable amount of time, given the limited hardware resources available. The second difficulty is related to the heterogeneous nature of the training texts, which requires a comprehensive analysis of the data, going beyond the simple meaning of the individual texts. To address these obstacles, we adopted two main strategies respectively: the use of coresets, and the integration of external features derived from text preprocessing, subsequently incorporated into the model training process.

The results obtained with this methodology, as well as their limitations and possible improvements are discussed in the next two sections.

4.1 Discussion of the results

The extraction of coresets from each of the examined datasets allowed us to work with reduced datasets, but equally representative of the originals. Training on the union of these coresets resulted in superior performance on individual test partitions compared to training on a single dataset. While this result was expected, it is particularly significant that training on the coreset performed better even than training on the full union of the datasets. This result has a twofold benefit: first, the amount of data required for training has been significantly reduced, resulting in a significant decrease in training time. Second, the improved performance can be explained by the fact that the coresets allowed to select only highly representative examples, eliminating those potentially causing bias or noise. This contributed to a more effective generalization of the model, avoiding the influence of less relevant or biased data.

The integration of external features into the model, in addition to textual representations, increased the complexity of the system, increasing the number of parameters and, consequently, the training time for the same dataset size. However, this choice was made possible and justified by the preliminary use of coresets, which reduced the number of training examples, allowing to manage the increase in model complexity efficiently. The integrated external features, such as syntactic structures and definitions of key concepts, were selected for their ability to capture specific and relevant characteristics of the investigated phenomenon, improving the accuracy of the model in detecting claims.

Specifically, the use of syntactic features is motivated by the need to evaluate the objectivity of claims. Understanding the relationships between the terms of a sentence and the force with which these manifest themselves helps to discern whether the meaning of a text allows a greater understanding of the fact that it represents an objective assertion or a personal and arbitrary interpretation, especially when the form of this is complex or ambiguous due to linguistic overtones such as sarcasm or irony. The obtained results confirm that this strategy led to significant improvements compared to the use of LLM representations alone. This effect was observed using both BERT-based and GPT-based models, demonstrating the effectiveness, within the claim detection task, of explicitly considering the syntactic aspect of texts in a way that can then refine the overall its semantic interpretation.

On the other hand, the choice of providing specific definitions of the representative concepts in the text is linked to another characteristic of claims, that is, that the statements of real interest are those relating to topics of interest not to the individual, but to the community. The objective of fact-checking is to contrast misinformation on topics that could lead to a distortion of public opinion, for which it is necessary to provide a basis of knowledge that is built on truthful and certified information. Providing the model with this type of information helps expand the classification process to a broader context that considers in detail the themes to which a given text refers.

Again, the integration of this strategy into the training process led to improvements on both models examined, thus proving the usefulness of enriching the model with additional contextual information.

4.2 Limitations and possible future improvements

The results of the research, although being satisfiying, can however be questioned by evaluating some critical aspects within each of the main steps of the methodology. In this section, the limitations of these are discussed and then some ideas that can be adopted in the future to seek improvements are proposed.

A first area for future efforts concerns the extraction of coresets. The criterion by which the amount of records extracted from each dataset is calculated is a key point for optimization. A next step could be to conduct experiments to reduce the number of examples used to the minimum necessary, testing different coreset sizes to further optimize the overall dataset size without compromising performance. Furthermore, the approach adopted so far, although it has led to positive results, remains a naive method. There are more advanced methods, such as record selection based on the importance of each example in relation to its contribution in reducing the cost of the objective function. By adopting these more sophisticated approaches, we can more confidently ensure that the data selected for training are those that actually contribute to improving the model optimization, further eliminating noise and maximizing training quality.

Subsequently, the use of syntactic information presents some limitations that deserve consideration. In the proposed solution, the syntactic connections between the terms of the text are made explicit, together with their strength. However, the current approach does not fully exploit the logical function of these links, which could be described through syntactic dependency labels. Integrating this information would require more complex methods than the simple attention mask. One possible approach would be to integrate the labels of syntactic dependencies, using advanced techniques such as graph networks. These networks would allow to represent in a richer way the logical relations between words in the text. However, adopting this solution brings additional challenges, as graph networks should be integrated with Large Language Models, making the model more complex and introducing new optimization and complexity management issues.

The integration of key concepts in the text also presents some important considerations. The extraction of keyphrases, currently limited to a maximum of two per text, while allowing one to focus on the main concepts without introducing irrelevant terms, can be restrictive in cases where a sentence contains several important concepts. A possible improvement would be to explore an approach that combines a maximum limit with a threshold based on the relevance of the keyphrases. This would allow us to expand the collection of concepts while still maintaining meaningful and relevant extractions. Another limitation concerns the process of collecting concept definitions, which is currently based on web scraping. This approach can be problematic because it requires constant updates to stay aligned with changes in external sources of information. Furthermore, there is the issue of disambiguation: some terms may have different meanings depending on the context, and the integration of incorrect or imprecise definitions may negatively impact the model's performance. A future improvement could consist in the introduction of more advanced disambiguation techniques, which ensure that the definitions collected are actually appropriate for the context of the text.

Finally, the integration of additional datasets containing texts from sources of different style and subject matter certainly represents one of the most effective methods to improve the generalization capacity of the model. The expansion of the thematic spectrum of the texts allows the model to manage a wide range of claims with greater flexibility, making it more robust and versatile. Consequently, as the variety of data increases, even exploring experiments with more advanced Large Language Models, such as the latest releases of GPT, LLaMA, or Mistral, could offer significant benefits, despite requiring more computational resources. The use of these advanced models could be fundamental in pushing the boundaries of performance in the field of automated claim detection.

Bibliography

[1] Soroush Vosoughi, Deb Roy, and Sinan Aral. "The spread of true and false news online". In: *science* 359.6380 (2018), pp. 1146–1151.

[2] David MJ Lazer et al. "The science of fake news". In: *Science* 359.6380 (2018), pp. 1094–1096.

[3] Xinyi Zhou and Reza Zafarani. "A survey of fake news: Fundamental theories, detection methods, and opportunities". In: *ACM Computing Surveys (CSUR)* 53.5 (2020), pp. 1–40.

[4] Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. "A survey on automated fact-checking". In: *Transactions of the Association for Computational Linguistics* 10 (2022), pp. 178–206.

[5] Naeemul Hassan, Chengkai Li, and Mark Tremayne. "Detecting check-worthy factual claims in presidential debates". In: *Proceedings of the 24th acm international on conference on information and knowledge management.* 2015, pp. 1835–1838.

[6] Pepa Atanasova et al. "Overview of the CLEF-2018 CheckThat! lab on automatic identification and verification of political claims. Task 1: Check-worthiness". In: *arXiv preprint arXiv:1808.05542* (2018).

[7] William Ferreira and Andreas Vlachos. "Emergent: a novel data-set for stance classification". In: *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Human language technologies.* ACL. 2016.

[8] James Thorne et al. "FEVER: a large-scale dataset for fact extraction and VERification". In: *arXiv preprint arXiv:1803.05355* (2018).

[9] Kashyap Popat et al. "Declare: Debunking fake news and false claims using evidence-aware deep learning". In: *arXiv preprint arXiv:1809.06416* (2018).

[10] Brooke Borel. *The Chicago guide to fact-checking.* University of Chicago Press, 2023.

[11] Neema Kotonya and Francesca Toni. "Explainable automated fact-checking: A survey". In: *arXiv preprint arXiv:2011.03870* (2020).

[12] Aviv Barnoy and Zvi Reich.
"The when, why, how and so-what of verifications".
In: *Journalism Studies* 20.16 (2019), pp. 2312–2330.

[13] Naeemul Hassan et al.
"Claimbuster: The first-ever end-to-end fact-checking system".
In: *Proceedings of the VLDB Endowment* 10.12 (2017), pp. 1945–1948.

[14] Lev Konstantinovskiy et al.
"Toward automated factchecking: Developing an annotation schema and benchmark for consistent automated claim detection".
In: *Digital threats: research and practice* 2.2 (2021), pp. 1–16.

[15] Chaoyuan Zuo, Ayla Karakas, and Ritwik Banerjee. "A hybrid recognition system for check-worthy claims using heuristics and supervised learning".
In: *CEUR workshop proceedings*. Vol. 2125. 2018.

[16] Rrubaa Panchendrarajan and Arkaitz Zubiaga.
"Claim detection for automated fact-checking: A survey on monolingual, multilingual and cross-lingual research".
In: *Natural Language Processing Journal* 7 (2024), p. 100066.

[17] Casper Hansen et al. "Neural Weakly Supervised Fact Check-Worthiness Detection with Contrastive Sampling-Based Ranking Loss."
In: *CLEF (Working Notes)*. 2019.

[18] Gullal S Cheema, Sherzod Hakimov, and Ralph Ewerth.
"Check_square at checkthat! 2020: Claim detection in social media via fusion of transformer and syntactic features".
In: *arXiv preprint arXiv:2007.10534* (2020).

[19] Jacob Devlin. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[20] Alexis Conneau et al.
"Unsupervised cross-lingual representation learning at scale".
In: *arXiv preprint arXiv:1911.02116* (2019).

[21] Yufeng Li, Rrubaa Panchendrarajan, and Arkaitz Zubiaga.
"FactFinders at CheckThat! 2024: refining check-worthy statement detection with LLMs through data pruning".
In: *arXiv preprint arXiv:2406.18297* (2024).

[22] Peter Røysland Aarnes, Vinay Setty, and Petra Galuščáková.
"IAI group at CheckThat! 2024: transformer models and data augmentation for checkworthy claim detection".
In: *arXiv preprint arXiv:2408.01118* (2024).

[23] Inna Vogel et al. "Adapter fusion for check-worthiness detection-combining a task adapter with a NER adapter". In: (2024).

[24] Mehmet Eren Bulut, Kaan Efe Keleş, and Mucahid Kutlu. "TurQUaz at CheckThat! 2024: a hybrid approach of fine-tuning and in-context learning for check-worthiness estimation". In: *Faggioli et al.[22]* (2024).

[25] Jun Suzuki, Heiga Zen, and Hideto Kazawa. "Extracting representative subset from extensive text data for training pre-trained language models". In: *Information Processing & Management* 60.3 (2023), p. 103249.

[26] Zhongli Li et al. "Improving BERT with syntax-aware local attention". In: *arXiv preprint arXiv:2012.15150* (2020).

[27] Xing Zhang et al. "Detecting dependency-related sentiment features for aspect-level sentiment classification". In: *IEEE Transactions on Affective Computing* 14.1 (2021), pp. 196–210.

[28] Umer Mushtaq and Jérémie Cabessa. "Argument classification with bert plus contextual, structural and syntactic features as text". In: *International Conference on Neural Information Processing.* Springer. 2022, pp. 622–633.

[29] Tilman Beck, Andreas Waldis, and Iryna Gurevych. "Robust integration of contextual information for cross-target stance detection". In: *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (* SEM 2023).* 2023, pp. 494–511.

[30] *google-bert/bert-base-uncased · Hugging Face — huggingface.co.* `https://huggingface.co/google-bert/bert-base-uncased`. [Accessed 22-08-2024].

[31] *FacebookAI/xlm-roberta-base · Hugging Face — huggingface.co.* `https://huggingface.co/FacebookAI/xlm-roberta-base`. [Accessed 22-08-2024].

[32] *google-bert/bert-base-multilingual-cased · Hugging Face — huggingface.co.* `https://huggingface.co/google-bert/bert-base-multilingual-cased`. [Accessed 22-08-2024].

[33] Mike Lewis et al. "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension". In: *arXiv preprint arXiv:1910.13461* (2019).

[34] *facebook/bart-base · Hugging Face — huggingface.co.* `https://huggingface.co/facebook/bart-base`. [Accessed 22-08-2024].

[35] Alec Radford et al. "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8 (2019), p. 9.

[36] *openai-community/gpt2 · Hugging Face — huggingface.co.* `https://huggingface.co/openai-community/gpt2`. [Accessed 22-08-2024].

[37] Olivier Bachem, Mario Lucic, and Andreas Krause. “Practical coreset constructions for machine learning”. In: *arXiv preprint arXiv:1703.06476* (2017).

[38] Lan You et al. “DRGAT: Dual-relational graph attention networks for aspect-based sentiment classification”. In: *Information Sciences* 668 (2024), p. 120531.

[39] S Supraja and Andy WH Khong. “Quad-Faceted Feature-Based Graph Network for Domain-Agnostic Text Classification to Enhance Learning Effectiveness”. In: *IEEE Transactions on Computational Social Systems* (2024).

[40] A Vaswani. “Attention is all you need”. In: *Advances in Neural Information Processing Systems* (2017).

[41] Tim Schopf, Simon Klimek, and Florian Matthes. “Patternrank: Leveraging pretrained language models and part of speech for unsupervised keyphrase extraction”. In: *arXiv preprint arXiv:2210.05245* (2022).