UNIPI-NLE at CheckThat! 2020: Approaching Fact Checking from a Sentence Similarity Perspective Through the Lens of Transformers

Lucia C. Passaro[1], Alessandro Bondielli[2,1], Alessandro Lenci[1], and Francesco Marcelloni[1]

[1] University of Pisa, Italy
lucia.passaro@fileli.unipi.it
{alessandro.lenci,francesco.marcelloni}@unipi.it
[2] University of Florence, Italy
alessandro.bondielli@unifi.it

Abstract. This paper describes a Fact Checking system based on a combination of Information Extraction and Deep Learning strategies to approach the task named "Verified Claim Retrieval" (Task 2) for the CheckThat! 2020 evaluation campaign. The system is based on two main assumptions: a claim that verifies a tweet is expected i) to mention the same entities and keyphrases, and ii) to have a similar meaning. The former assumption has been addressed by exploiting an Information Extraction module capable of determining the pairs in which the tweet and the claim share at least a named entity or a relevant keyword. To address the latter, we exploited Deep Learning to refine the computation of the text similarity between a tweet and a claim, and to actually classify the pairs as correct matches or not. In particular, the system has been built starting from a pre-trained Sentence-BERT model, on which two cascade fine-tuning steps have been applied in order to i) assign a higher cosine similarity to gold pairs, and ii) classify a pair as correct or not. The final ranking produced by the system is the probability of the pair labelled as correct. Overall, the system reached a 0.91 MAP@5 on the test set.

1 Introduction

The great proliferation of online misinformation and fake news in the last few years encouraged the development of several fact-checking initiatives by various actors including journalists, governments, organizations, and companies. In the past, fact-checking was typically performed manually, resulting in the collection of large amounts of annotated resources for this specific task. More recently, researchers have started to use such resources with the aim of training automatic fact-checking systems [19, 25, 33]. A common phenomenon in social media is that viral claims often come back after a while [25], increasing the probability that a particular claim has been previously fact-checked by a trusted organization. Therefore, systems able to decide whether a claim has been already fact-checked have become particularly relevant, because they contribute to breaking down the costs of verifying both old and new viral claims. In this scenario, the CLEF2020-CheckThat! task 2 [1, 2, 8, 26] has been organized with the goal of supporting journalists and fact-checkers when trying to determine whether a claim has been already fact-checked.

The goal of the task is specified as follows: "Given a check-worthy claim and a dataset of verified claims, rank the verified claims, so that those that verify the input claim (or a sub-claim in it) are ranked on top".[3]

This paper describes a system that approaches such task by exploiting a combination of Information Extraction (IE) and Deep Learning (DL) strategies to associate a tweet with the *most probable* claim that verifies it. The task is indeed strongly related to the concepts of information extraction and text similarity. Intuitively, to guess if two claims are related to each other, it is important to establish whether i.) they share some linguistic properties (e.g., mentioned entities) and ii.) they are in general semantically similar. In order to deal with i.), traditional IE methods are very useful and accurate [21] when extracting information such as Named Entities (e.g., persons, locations and organizations) and content words (e.g., nouns, verbs). On the other hand, ii.) requires a deeper representation of text meaning, which can be obtained with Neural Language Models (NLMs) [3]. State-of-the-art NLMs [12, 22] based on *Transformer* architectures and *attention mechanisms* [31] have become increasingly popular in the last couple of years, thanks to their ability to model whole text sequences and generate pre-trained representations that can be fine-tuned for different tasks. An important feature of the word representations produced by such models is that they are contextualized (i.e., they differ depending on the word context), thereby improving model performance in tasks based on word [23] and sentence [24] similarity.

The rest of the paper is organized as follows: Section 2 presents an overview of both fact-checking frameworks and NLP resources relevant for the task. Section 3 describes the proposed approach to solve the fact-checking task, consisting in the creation of two fine-tuned models to handle the claim semantic relatedness. Sections 4 and 5 focus on results and discussion, respectively. Finally, Section 6 draws some conclusions and describes future research directions.

2 Related work

A key aspect of the process of building trustworthy data sets of fake and reliable news is actually how the Fact-Checking process is performed [5]. In the last years, several approaches have been proposed for different purposes. For example, Fact Check Explorer, developed by Google, [4] browses and searches for fact checks

[3] `https://github.com/sshaar/clef2020-factchecking-task2`
[4] `http://toolbox.google.com/factcheck/explorer`

by exploiting mentions and topics and by offering several filters to refine the queries. Similarly, ClaimsKG [29] offers a Knowledge Graph to search the claims containing particular named entities or keyphrases.

Over many years, fact-checking has been performed manually by journalists, by exploiting available tools online [25]. Earliest work on automated fact-checking define the task as *the assignment of a truth value to a claim made in a particular context* [32]. Most of the approaches on automated fact-checking exploit the reliability of a source and the stance of its claims with respect to other claims and already verified information. The assignment of the truth value is often based on the way in which particular claims (or rumors) are spread on social media [7, 9, 13, 28] or on the Web [17, 20]. Other approaches use Wikipedia [18, 30] or other knowledge graphs [11, 27] to fact-check claims. More recently, a novel approach has been proposed that exploits Sentence-BERT [24] to re-rank claims [25] in order to predict whether a claim has been fact-checked before.

Indeed, DL models have proven to be among the most effective techniques for Language Modelling. In addition, the availability of new DL architectures such as Transformers [12] has led to a significant performance improvement in a wide range of NLP tasks. Transformers have two main advantages over previous Language Modelling architectures. First, thanks to the attention mechanism each element of a text sequence can access information of all the other elements. Thus, the meaning of words (and sentences) in context can be modelled more effectively. Second, the Transformer architecture is geared towards exploiting the full potentialities of transfer learning for NLP tasks. The idea behind transfer learning is that the knowledge learnt on a more general task can be exploited to specialize a model on new problems for which the amount of data is much more limited. One particular instance of transfer learning consists of two different training paradigms, namely *pre-training* and *fine-tuning*. During pre-training, language models are typically trained with an unsupervised learning tasks on vast collections of textual data. For example, models can be trained to predict specific words in a sequence based on their surrounding context, and to predict whether two sentences are sequential or not [12]. During fine-tuning, the pre-trained model is further trained, this time for a limited number of epochs, on supervised learning tasks such as for example sequence labeling or sequence pair classification. The main idea is that, the initial weights (or a subset thereof) of the pre-trained model, are further adjusted to model the fine-tuning task. Typically, the pre-training step is very time-consuming and computationally expensive, but the same resulting model can be used as a starting point to solve a wide range of tasks. On the other hand, the fine-tuning step is less resource-demanding and requires less labelled data.

Transformer-based architectures such as BERT [12] and XLNet [35] have obtained state-of-the-art results in most NLP tasks they have been applied to. One advantage of such architectures is that they learn contextualized representations that allow models to capture word polysemy. Conversely, more traditional language models such as Skip-Gram and Continuous-Bag-of-Words algorithms [4, 15, 16] learn non-contextualized embeddings and store a single vector for each word type belonging to the training set, independently of its context. Moreover, Transformers have been also exploited to obtain context-aware sentence representations that have been proven to enable semantic comparison of sentences with promising performances [24].

3 The UNIPI-NLE approach

Given a tweet and a set of already verified claims (*vclaims*), the goal of the task is to predict, for every *target tweet-vclaim pair*, the likelihood of the vclaim verifying the tweet. Indeed, among the target tweet-vclaim pairs, there exists only a *gold pair* whose vclaim verifies the tweet, which therefore is a correct match. The goal is achieved by ranking, for each tweet, the claims that are more likely to verify it. The dataset is composed of three elements:

1. the verified claims used for fact checking, each of them provided with an identifier, a title, and the actual claim;
2. the training tweets, associated with an identifier and a textual content;
3. the correct pairing between tweets and verified claims.

The training set consists of $1,003$ tweets (803 for training, 200 for development) and $10,373$ already verified claims. The test set consists of 200 additional tweets.

The UNIPI-NLE system is based on two main assumptions: *the claims that verify a tweet are expected to mention the same entities and keyphrases and should have a similar meaning.* To address the first point, among target tweet-vclaim pairs, we identify the subset of *candidate pairs* (also referred as *potential pairs*) in which the tweet and the vclaim share at least a named entity or a content word. We refer to the step of identifying candidate pairs among target ones as the *IE step*. The DL modules described below have been fed only with the portion of the dataset consisting of such candidate pairs. In order to estimate the text similarity between a tweet and a vclaim, we exploit Siamese BERT networks [24] to create a language model that is able to better deal with sentence-level textual similarity. This model is then used to learn if a claim can be used to verify a tweet. In particular, we perform two cascade fine-tuning steps aimed at i.) assigning a higher cosine similarity to gold tweet-vclaim pairs and ii.) actually classifying a target tweet-vclaim pair, and more specifically a candidate tweet-vclaim pair, as a correct match (gold) or not.

Figure 1 shows the neural components of the system architecture. The first white stack (`bert-base-uncased` + `bert-base-nli-mean-tokens`) consists of the pre-trained model released by [24] and trained on SNLI [6] and MultiNLI dataset [34] to create universal sentence embeddings. The black boxes show our two fine-tuned models: Our Sentence-BERT model (`bert-base-nli-factcheck-cos`) follows a training paradigm similar to the one described in [24], but it is specifically geared to assigning a higher cosine similarity to gold tweet-vclaim pairs. The last level of the architecture represents the final Transformer-based classifier trained to decide, given a candidate tweet-vclaim pair, whether the tweet is actually verified by that claim or not. The

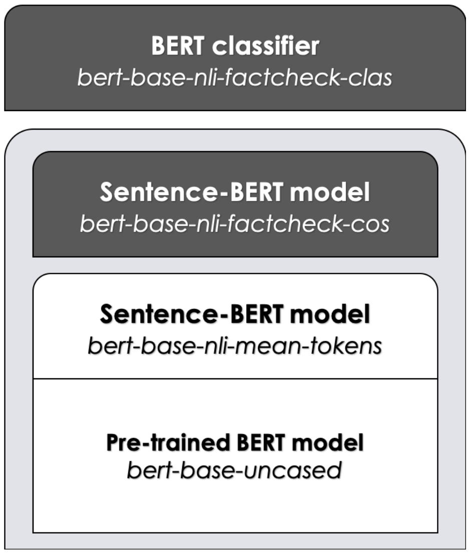

Fig. 1. Neural modules of the UNIPI-NLE system.

classifier fine-tunes `bert-base-nli-factcheck-cos` on the fact-checking task, by labelling candidate tweet-vclaim pairs as correct matches (gold) or not.

3.1 Information Extraction (IE) step

Starting from the assumption that similar claims tend to mention the same entities and keyphrases, we developed an IE module to find potential tweet-vclaim pairs. Such a module is based on Stanza [21], a state of the art natural language analysis package. We processed each text fragment (i.e., a tweet, a vclaim or a vclaim title) with Sentence Splitting, PoS-tagging, Lemmatization, and Named Entity Recognition. Thus, each text is associated with its keywords, consisting of its content words (nouns, verbs, and adjectives) and named entities.

Given a tweet, in order to retrieve potential claims that verify it, we used two different functions based on the keywords:

IE function – the overlapping score is simply computed by counting the number of elements (cf. the keywords field in Table 1 and Table 2) shared by the tweet and the claim. Candidate tweet-vclaim pairs are required to share at least one lowercased element (named entity or content word).

IEElastic function – it exploits Elasticsearch[5] to find the potential candidate pairs. Specifically, for each tweet, candidate claims consist of the top $1,000$ matches ranked by relevance, using the scoring function provided by

[5] `https://www.elastic.co/`

vclaim and title	**keywords**
title: Was Sen. Chuck Schumer a Client of ‘Hollywood Madam’ Heidi Fleiss? **vclaim:** Sen. Chuck Schumer’s name and/or phone number were found in ”Hollywood Madam” Heidi Fleiss’s black book of clients.	[‘chuck schumer’, ‘hollywood’, ‘heidi fleiss’s’, ‘chuck schumer’, ‘hollywood’, ‘heidi fleiss’, ‘sen.’, ‘chuck’, ‘schumer’, ‘phone’, ‘number’, ‘find’, ‘hollywood’, ‘madam’, ‘heidi’, ‘fleiss’, ‘black’, ‘book’, ‘client’, ‘sen.’, ‘chuck’, ‘schumer’, ‘client’, ‘hollywood’, ‘madam’, ‘heidi’, ‘fleiss’]

Table 1. Example of relevant lemmas and named entities in a claim.

tweet	**keywords**
Chuck Schumer was one of Hedil Fleiss’ top clients. Look it up. Doug Masters (@protestertrophy) January 23, 2019	[‘chuck schumer’, ‘hedil fleiss’, ‘doug masters’, ‘january 23, 2019’, ‘chuck’, ‘schumer’, ‘hedil’, ‘fleiss’, ‘client’, ‘look’, ‘doug’, ‘masters’, ‘@protestertrophy’, ‘january’]

Table 2. Example of relevant lemmas and named entities in a tweet.

the task organizers for the baseline. Such scoring function is an Elasticsearch multi-match query based on both the vclaim and its title and the tweet itself.

Candidate tweet-claim pairs obtained with the IE overlapping function were used to train the model `bert-base-nli-factcheck-cos`. Candidate tweet-claim pairs obtained with the IE and IEElastic functions have been used at inference time to obtain the final predictions submitted for evaluation, namely respectively T2-EN-UNIPI-NLE-BERT2IE (contrastive run) and T2-EN-UNIPI-NLE-BERT2IEElastic (primary run). Moreover, we used both the IE and IEElastic functions to simply rank the potential claims associated with each tweet according to the overlapping score. The score provided by IEElastic corresponds to the task official baseline. The results of these rankings are reported in Section 4.

3.2 Fine-tuning of Transformer models

Siamese BERT networks [24] can be used to create language models specialized on tasks related to Semantic Textual Similarity (STS) [10]. In order to train our system to recognize gold tweet-vclaim pairs, we first model the textual similarity between tweets and claims belonging to the same candidate tweet-vclaim pairs. To this purpose, we started from one of the available fine-tuned Sentence-BERT models,[6] namely `bert-base-nli-mean-tokens` [24]. Such model was originally

[6] `https://github.com/UKPLab/sentence-transformers`

trained on SNLI [6] and MultiNLI dataset [34] and tested on the STSbenchmark [10]. The training phase was performed by classifying a pair of sentences with the labels *entail*, *contradict*, and *neutral* while evaluation was performed on the STSbenchmark [10] dataset, which contains sentence pairs and their similarity score. The trained model was exploited to infer sentence pair similarity via cosine. The `bert-base-nli-mean-tokens` achieved 77.12 Pearson correlation with gold scores on the STSbenchmark test set.

We added two levels of fine-tuning to this model, in order to adapt the sentence pair similarity task to the fact-checking one (i.e., gold tweet-vclaim pairs are associated with the maximum cosine similarity), as well as to fact-check a pair with a classification layer (i.e., gold tweet-vclaim pairs are associated with the positive label). To this end, we exploited both the vclaim text content and its title, namely the *vclaim_title*.

In fact, each vclaim is provided with a title that can be considered as a summary of the vclaim itself and therefore, very similar to it. The usage of both the vclaim and vclaim_title for training has two main advantages. First, it allows to increase the size of the dataset so that the model is shown more positive examples, that are under-represented. Second, it helps to add variability to the training examples, both positive and negative. For example, a title may contain an acronym such as "KKK", whereas the claim may contain its extended form, in this case "Ku Klux Klan". In our experiments we noticed that such a variability was very helpful to improve the overall performances of our models.

Sentence pair similarity We modeled the fine-tuning step like Reimers and Gurevych [24] to estimate the semantic similarity between two sentences. The authors used the STSbenchmark [10] dataset, containing pairs of sentences with a similarity score ranging from 0 (no similarity) to 5 (maximum similarity). The Sentence-BERT model was fine-tuned using the regression objective function on the training set [24]. Therefore, for each epoch, loss was computed by considering the correlation between the gold similarity judgments and the predicted cosine similarity between sentence embeddings.

Our goal was to train the model to identify the gold tweet-vclaim pair among the set of potential ones (i.e., those filtered with the IE step). To this aim, we tried to separate the gold tweet-vclaim pairs from the other candidates. In particular, given the assumption that a claim that verifies a tweet is semantically similar to it, we built our training set as follows:

1. we created two positive examples from a gold tweet-vclaim pair, the first one composed by the tweet and the vclaim itself (tweet-vclaim pair), and the second one composed by the tweet and the title of the claim (tweet-vclaim_title). Both the positive pairs were assigned with a cosine similarity value of 1.0. This forces the model to boost the similarity between the texts belonging to gold pairs;
2. for each gold tweet-vclaim pair, 20 other tweet-vclaim pairs were randomly selected as negative examples from the list of candidate pairs obtained with the IE overlapping function (cf. Section 3.1). The similarity of the negative

examples was computed as the cosine similarity between vectors predicted by `bert-base-nli-mean-tokens`, modified by the *tanh* function. This has the effect of decreasing the cosine similarity, thus effectively penalising negative examples.

The `bert-base-nli-factcheck-cos` model was trained for 4 epochs with a batch size of 8, and 10% of data was used for the model warm up.

Classification The `bert-base-nli-factcheck-cos` model is used to initialize the weights for the classifier. In this case, the model is trained on a simple binary classification task to distinguish between matching (gold) pairs, labelled as 1, and non-matching ones, labelled as 0. Similarly to the previous fine-tuning step, we selected negative examples among candidate tweet-vclaim pairs returned by the IE module. Like for the sentence pair similarity model, for each tweet, the tweet-vclaim and the tweet-vclaim_title pairs were used as positive examples. However, in this case 2 negative examples were selected among the tweet-vclaim candidate pairs, in order to better balance the training data for the classification.

Our model `bert-base-nli-factcheck-clas` is therefore a Transformer with a classification head on top of it, implemented with the Huggingface library.[7] The model was trained for 3 epochs with a batch size of 8. We used the AdamW optimizer with a learning rate of $2e-5$ [14].

3.3 Inference step

The inference step was performed by classifying the candidate tweet-vclaim pairs. To retrieve the potential candidates, in fact, we applied the functions IE and IEElastic described in section 3.1 to obtain, respectively, the T2-EN-UNIPI-NLE-BERT2IE and the T2-EN-UNIPI-NLE-BERT2IEElastic predictions. Moreover, we also tested a run in which we classified all the target tweet-vclaim pairs with no-preselection. The results of this additional experiment are shown in Table 4. In all cases, we used the probability of the class 1 predicted by the `bert-base-nli-factcheck-clas` model to rank the vclaims for each tweet.

4 Results

The evaluation metric used in the competition is the Mean Average Precision @5 (MAP@5) calculated over the gold ranking. The overall performance of our models is reported in Table 3. For the sake of comparison, we also show the performances obtained by the top models and by the official baseline as well.

Our systems, namely the T2-EN-UNIPI-NLE-BERT2IE and the T2-EN-UNIPI-NLE-BERT2IEElastic, which differ for the overlapping function used at inference time, obtained respectively 0.9160 and 0.9120 (cf. tables 4 and 5).

Moreover, in order to explain the effectiveness of each module for the final predictions, we computed their performances on the test set. Table 6 shows the

[7] `https://huggingface.co`

Team	**type**	**MAP@1**	**MAP@3**	**MAP@5**
Buster.ai	primary	0.897	0.926	0.929
Buster.ai	contr.-2	0.907	0.937	0.938
UNIPI-NLE	primary	0.877	0.907	0.912
UNIPI-NLE	contr.-1	0.877	0.913	0.916
Task Organizers	baseline	0.767	0.812	0.815

Table 3. Performance of the UNIPI-NLE models against the top performing models and the official baseline.

performances of each module obtained on the task by ranking the claim for a tweet according to several measures. More specifically, for each module, we report the model name, the type of the fine-tuning we applied, the function used at inference time for selecting candidates and the MAP@5 obtained with the official scorer.

As for the IE step, given a tweet, we ranked the claims according to the overlapping function for both the IE and the IEElastic methods. The IEElastic method coincides actually with the baseline provided by the task organizers. To assess the performances of the Sentence-BERT model fine-tuned on cosine similarity, namely the `bert-base-nli-factcheck-cos`, we ranked the claim according to the adjusted cosine similarity. Finally, we show the final submitted results. At inference time, the model `bert-base-nli-factcheck-clas` was fed with the candidate tweet-vclaim pairs calculated with both the IE and the IEElastic methods. In addition, we also report the results obtained by making the predictions for all the target tweet-vclaim pairs.

5 Discussion

Several remarks can be made to comment our results. By looking at the model scores, we see that our approach is able to outperform the baseline by a wide margin, despite the fact that the Elasticsearch based approach proposed by the task organizer was shown to be very effective nonetheless. In addition, our system ranked second among the participants of the task, obtaining performances that are only slightly worse than the winning system, which obtained a MAP@5 of 0.938.

Moreover, we can draw some interesting insights by considering the various steps and data selection strategies. We notice that the IE baseline appears to be less effective as a standalone tool for selecting the best candidates among claims for each tweet, with results well below the IEElastic one. However, the IE method performs optimally when used as a selection criterion at inference time. In fact, we experimented three methods for selecting candidate pairs at inference time. In addition to the IE method and the IEElastic one, we assessed the

metric	**@depth**	**score**
MAP	1	0.877
MAP	3	0.913
MAP	**5**	**0.916**
MAP	10	0.917
MAP	20	0.917
MAP	all	0.917
Precision	1	0.879
Precision	3	0.322
Precision	5	0.195
Precision	10	0.098
Precision	20	0.049
Precision	all	0.000
Rec_Rank	1	0.879
Rec_Rank	3	0.915
Rec_Rank	5	0.918
Rec_Rank	10	0.919
Rec_Rank	20	0.919
Rec_Rank	all	0.919

Table 4. T2-EN-UNIPI-NLE-IE

metric	**@depth**	**score**
MAP	1	0.877
MAP	3	0.907
MAP	**5**	**0.912**
MAP	10	0.913
MAP	20	0.913
MAP	all	0.913
Precision	1	0.879
Precision	3	0.317
Precision	5	0.194
Precision	10	0.098
Precision	20	0.049
Precision	all	0.000
Rec_Rank	1	0.879
Rec_Rank	3	0.909
Rec_Rank	5	0.914
Rec_Rank	10	0.915
Rec_Rank	20	0.915
Rec_Rank	all	0.915

Table 5. T2-EN-UNIPI-NLE-IEElastic

performance with no pre-selection as well. In this case, the classifier was shown with all possible tweet-vclaim pairs. We see that the IE method performs best, but only slightly better than IEElastic. However, both pre-selection methods outperform the model for which no pre-selection is made. We can argue that this is because, during training, our objective was to enable the classifier to distinguish between the gold tweet-vclaim pair and other pairs that share similar features but are in fact incorrect. Therefore, the classifier may be more prone to errors when tweet-vclaim pairs, which differ greatly from each other, are shown as it never saw such examples during training. Experimental results seem to confirm such hypothesis. This could be seen as a potential shortcoming for the classifier itself. However, we can argue that considering the system as a whole, it can bring two potential advantages. First, the classifier needs less negative examples for an effective training. We can argue that it is more difficult to decide between two similar claims for a tweet, rather than between two very different ones. Therefore, we chose to train the classifier to solve the “harder”

Model	**Fine-tuining**	**Inference**	**MAP@5**
`bert-base-nli-factcheck-clas`	classification	IE	0.916
`bert-base-nli-factcheck-clas`	classification	IEElastic	0.912
`bert-base-nli-factcheck-clas`	classification	-	0.89
IEElastic baseline	-	IEElastic	0.815
IE baseline	-	IE	0.74
`bert-base-nli-factcheck-cos`	cosine similarity	IE	0.41
`bert-base-nli-factcheck-cos`	cosine similarity	IEElastic	0.35

Table 6. Results calculated for each module of the architecture.

problem, and addressed the "simpler" one with a less sophisticated, yet effective, approach. Second, the classification of each tweet-vclaim pair is time consuming. On our machine, equipped with a Nvidia TitanXp graphic card, the inference step considering all pairs took around ten hours. When performing the pre-selection of pairs with our IE method, the inference step took less than four hours. The IE method is efficient because it only needs to extract content words and named entities for each pair, a task that is almost trivial in terms of time complexity with modern NLP toolkits and current hardware.

Finally, it is interesting to point out the contribution of the cosine similarity adaptation performed with Sentence-BERT. Clearly, the model itself does not perform well on the present task. However, two observations can be made. First, during development we noticed that, by using a standard BERT model such as BERT-base-uncased for representing sentences (i.e., by averaging word-level representations obtained from the model), ranking claims based on cosine similarity was completely ineffective, obtaining a very low MAP@5. Instead, by exploiting a pre-trained Sentence-BERT model, we obtained much more encouraging results, that were subsequently improved thanks to our cascade fine-tuning strategy. This serves as additional evidence for the fact that standard BERT models are not able to represent sentences in a semantically proper way, as already claimed in the literature [24]. Second, by exploiting the fine-tuned Sentence-BERT model (`bert-base-nli-factcheck-cos`) for obtaining the initial weights for the classifier (`bert-base-nli-factcheck-clas`), we clearly outperformed a model based on BERT-base-uncased and trained in the same way. More specifically, on the development set we obtained a 0.72 MAP@5 for a `bert-base-uncased` fine-tuned model and a MAP@5 of 0.78 for the `bert-base-nli-factcheck-cos`.

6 Conclusion and future directions

The approach to Fact Checking performed by the UNIPI-NLE team is based on a combination of IE and DL strategies. The choice has been led by the assumptions that supporting claims tend to mention the same entities and keywords of the target tweet, and are semantically similar to it. On the one hand, a standard IE module extracts relevant words and entities from texts and is used to construct the training set for the following DL modules and to constrain the inference process. In fact, the IE step is also crucial to turn down the processing time by filtering the candidate pairs to be classified. Transformers, on the other hand, are very useful to carry out effective transfer learning, by fine-tuning large pre-trained models for specific tasks such as the fact-checking one. In this paper, fine-tuning is exploited both for modeling textual similarity and for classifying text pairs to decide if a member of the pair verifies the other.

The UNIPI-NLE approach strongly outperforms the baseline and ranked second among the primary submissions of the task. In the future, we plan to perform some additional hyperparameter tuning on the models. Moreover, we would like to test this approach in similar tasks such as Fake News identification. We are confident that by exploiting the dynamic selection of training data in addition to an effective and efficient information extraction strategy, we will obtain strong performances also to solve this harder task.

7 Acknowledgements

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. This work was partially supported by the University of Pisa in the context of the project "Event Extraction for Fake News Detection" in the framework of the MIT-Unipi program, and by the Italian Ministry of Education and Research (MIUR) in the framework of the CrossLab project (Departments of Excellence).

References

1. Arampatzis, A., Kanoulas, E., Tsikrika, T., Vrochidis, S., Joho, H., Lioma, C., Eickhoff, C., Névéol, A., Cappellato, L., Ferro, N. (eds.): Experimental IR Meets Multilinguality, Multimodality, and Interaction Proceedings of the Eleventh International Conference of the CLEF Association (CLEF 2020). LNCS (12260), Springer (2020)
2. Barrón-Cedeño, A., Elsayed, T., Nakov, P., Da San Martino, G., Hasanain, M., Suwaileh, R., Haouari, F., Babulkov, N., Hamdan, B., Nikolov, A., Shaar, S., Sheikh Ali, Z.: Overview of CheckThat! 2020: Automatic identification and verification of claims in social media. In: Arampatzis et al. [1]
3. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. Journal of machine learning research **3**(Feb), 1137–1155 (2003)
4. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)
5. Bondielli, A., Marcelloni, F.: A survey on fake news and rumour detection techniques. Information Sciences **497**, 38–55 (2019)

6. Bowman, S.R., Angeli, G., Potts, C., Manning, C.D.: A large annotated corpus for learning natural language inference. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 632–642. Association for Computational Linguistics, Lisbon, Portugal (2015)
7. Canini, K.R., Suh, B., Pirolli, P.L.: Finding credible information sources in social networks based on content and social structure. In: 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing. pp. 1–8. IEEE (2011)
8. Cappellato, L., Eickhoff, C., Ferro, N., Névéol, A. (eds.): Working Notes of CLEF 2020—Conference and Labs of the Evaluation Forum (2020)
9. Castillo, C., Mendoza, M., Poblete, B.: Information credibility on twitter. In: Proceedings of the 20th international conference on World wide web. pp. 675–684 (2011)
10. Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., Specia, L.: SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). pp. 1–14. Association for Computational Linguistics, Vancouver, Canada (2017)
11. Ciampaglia, G.L., Shiralkar, P., Rocha, L.M., Bollen, J., Menczer, F., Flammini, A.: Computational fact checking from knowledge networks. PloS one **10**(6), e0128193 (2015)
12. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (2019)
13. Gorrell, G., Kochkina, E., Liakata, M., Aker, A., Zubiaga, A., Bontcheva, K., Derczynski, L.: SemEval-2019 task 7: RumourEval, determining rumour veracity and support for rumours. In: Proceedings of the 13th International Workshop on Semantic Evaluation. pp. 845–854. Association for Computational Linguistics, Minneapolis, Minnesota, USA (2019)
14. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: In Proceedings of the 2019 International Conference on Learning Representations (2019)
15. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. CoRR **abs/1301.3781** (2013)
16. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2. p. 3111–3119. NIPS'13, Curran Associates Inc., Red Hook, NY, USA (2013)
17. Mukherjee, S., Weikum, G.: Leveraging joint interactions for credibility analysis in news communities. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. pp. 353–362 (2015)
18. Nie, Y., Chen, H., Bansal, M.: Combining fact extraction and verification with neural semantic matching networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 6859–6866 (2019)
19. Popat, K., Mukherjee, S., Strötgen, J., Weikum, G.: Where the truth lies: Explaining the credibility of emerging claims on the web and social media. In: Proceedings of the 26th International Conference on World Wide Web Companion. pp. 1003–1012 (2017)

20. Popat, K., Mukherjee, S., Strötgen, J., Weikum, G.: Where the truth lies: Explaining the credibility of emerging claims on the web and social media. In: Proceedings of the 26th International Conference on World Wide Web Companion. pp. 1003–1012 (2017)
21. Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C.D.: Stanza: A Python natural language processing toolkit for many human languages. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (2020)
22. Radford, A.: Improving language understanding by generative pre-training (2018)
23. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI Blog **1** (2019)
24. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using siamese BERT-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. pp. 3982–3992. Association for Computational Linguistics (2019)
25. Shaar, S., Babulkov, N., Da San Martino, G., Nakov, P.: That is a known lie: Detecting previously fact-checked claims. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 3607–3618. Association for Computational Linguistics, Online (2020)
26. Shaar, S., Nikolov, A., Babulkov, N., Alam, F., Barrón-Cedeño, A., Elsayed, T., Hasanain, M., Suwaileh, R., Haouari, F., Da San Martino, G., Nakov, P.: Overview of CheckThat! 2020 English: Automatic identification and verification of claims in social media. In: Cappellato et al. [8]
27. Shiralkar, P., Flammini, A., Menczer, F., Ciampaglia, G.L.: Finding streams in knowledge graphs to support fact checking. In: Proceedings of the 2017 IEEE International Conference on Data Mining (ICDM). pp. 859–864. IEEE (2017)
28. Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H.: Fake news detection on social media: A data mining perspective. ACM SIGKDD explorations newsletter **19**(1), 22–36 (2017)
29. Tchechmedjiev, A., Fafalios, P., Boland, K., Gasquet, M., Zloch, M., Zapilko, B., Dietze, S., Todorov, K.: Claimskg: a knowledge graph of fact-checked claims. In: International Semantic Web Conference. pp. 309–324. Springer (2019)
30. Thorne, J., Vlachos, A., Christodoulopoulos, C., Mittal, A.: FEVER: a large-scale dataset for fact extraction and VERification. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). pp. 809–819. Association for Computational Linguistics, New Orleans, Louisiana (2018)
31. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017)
32. Vlachos, A., Riedel, S.: Fact checking: Task definition and dataset construction. In: Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science. pp. 18–22. Association for Computational Linguistics, Baltimore, MD, USA (2014)
33. Wang, W.Y.: "liar, liar pants on fire": A new benchmark dataset for fake news detection. In: Barzilay, R., Kan, M.Y. (eds.) ACL (2). pp. 422–426. Association for Computational Linguistics (2017)
34. Williams, A., Nangia, N., Bowman, S.: A broad-coverage challenge corpus for sentence understanding through inference. In: Proceedings of the 2018 Conference of

the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). pp. 1112–1122. Association for Computational Linguistics (2018)

35. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. In: Advances in neural information processing systems. pp. 5753–5763 (2019)