BeaSku at CheckThat! 2021: Fine-Tuning Sentence BERT with Triplet Loss and Limited Data

Beata Skuczyńska[1], Shaden Shaar[2], Jennifer Spenader[1] and Preslav Nakov[2]

[1]*Artificial Intelligence Department, University of Groningen, The Netherlands*
[2]*Qatar Computing Research Institute, HBKU, Doha, Qatar*

Abstract
Misinformation and disinformation are growing problems online. The negative consequences of the proliferation of false claims became especially apparent during the COVID-19 pandemic. Thus, there is a need to detect and to track false claims. However, this is a slow and time-consuming process, especially when done manually. At the same time, the same claims, with some small variations, spread simultaneously across many accounts and even on different platforms. One promising approach is to develop systems for detecting new instances of claims that have been previously fact-checked online, as in the CLEF-2021 CheckThat! Lab Task-2b. Here we describe our system for this task. We fine-tuned sentence BERT using triplet loss, and we experimented with two types of augmented datasets. We further combined BM25 scores with language model similarity scores as features in a reranker. The official evaluation results have put our BeaSku system at the second place.

Keywords
previously fact-checked claim retrieval, sentence BERT, fake news, triplet loss

1. Introduction

As misinformation and disinformation has grown online, the need for reliable fact-checking has also grown. Consider for example the 2016 US Presidential Election, where fake news affected political decisions and opinions [1]. Since then, multiple initiatives and organisations (such as PolitiFact[1] in USA and EUvsDisinfo[2] in the EU) have put efforts into fact-checking claims. According to the Duke Reporters' Lab, in 2020 the global number of fact-check organisations crossed 300[3]. These organisations use a variety of approaches, typically relying on manual verification of claims, and less often using partial or full automation. The former approach is accurate but time-consuming, while the latter is resource-intensive; thus, it might be a good idea to combine them.

One of the first key steps in the fact-checking pipeline is to check whether the claim was previously fact-checked. This step is especially important for efficiency and tracking, as the same claims are often repeated across platforms with little or no change in the content.

CLEF 2021 – Conference and Labs of the Evaluation Forum, September 21–24, 2021, Bucharest, Romania
beata.skuczynska@gmail.com (B. Skuczyńska); sshaar@hbku.edu.qa (S. Shaar); j.spenader@gmail.com (J. Spenader); pnakov@hbku.edu.qa (P. Nakov)

[1]http://www.politifact.com/
[2]http://euvsdisinfo.eu/
[3]The Duke Reporters' Lab keeps track of media verification initiatives since 2014: http://reporterslab.org/fact-checking-count-tops-300-for-the-first-time/

Thus, before going for full fact-checking, it is useful to check whether the claim was verified before, which can make fact-checkers more efficient and could also help track the spread of false claims in different variations. Thus, verified claim retrieval could increase the impact of fact-checking without increasing the amount of manual work required.

The CLEF-2021 CheckThat! Lab featured a shared task designed to increase researchers' interest in verified claim retrieval. Task 2B of the lab is dedicated to determining whether a claim from a political debate or a speech was previously fact-checked. The task uses data from PolitiFact, which is one of the biggest fact-checking organizations. After a major political event (mostly in USA), PolitiFact verifies some of the claims that were made during the event. Often, they also link such claims to articles about already checked claims, e.g., when politicians keep repeating the same thing. The dataset thus contains a list of claims (*Input*) matched with previously fact-checked claims (*VerClaim*).

Below, we describe our approach to the task. We focused mainly on experimenting with triplet loss and other data augmentation methods to optimally exploit the provided data, and on fine-tuning sentence BERT. The produced cosine similarity scores between a candidate and previously fact-checked claims. We then used these features, together with BM25 scores, in a ranking model. Our submission was ranked second in the shared task, and our score was very close to that of the winner. The next sections provide more detail.

2. Related Work

Fact-checked claim matching has not received much attention from the NLP community, and has only recently been explored as a separate task. Research done within Shared Task 2 on Verified Claim Retrieval at CLEF 2020 CheckThat! Lab [2] helps define and refine the scope of claim matching. In this earlier shared task, the organizers provided a dataset from Snopes, where the focus was on tweets rather than on political debates. The baseline used Elasticsearch[4], and achieved a MAP@5 of 0.609. Eight teams submitted systems using diverse approaches, and the top two teams exceeded a score of 0.9. The organizers then created, in a similar fashion another, a more challenging dataset based on data from PolitiFact [3], for which the Elasticsearch baseline achieved only around 0.5 MAP. This latter dataset is an earlier version of the one used in this year's shared task.

Shaar et al. [3] achieved their best performance on the Snopes and on the PolitiFact datasets by combining both semantic similarity and classic information retrieval techniques. Many of the high-performing models from the CheckThat! 2020 Lab shared task also developed novel methods to obtain both simple negative examples as well as challenging ones, which imitated positive examples well, so they can be used to make models more sensitive to semantic subtleties. For example, the UNIPI-NLE team [4] added negative candidates from the pool of *Input–VerClaim* pairs that reached high scores (but were wrong) in an information retrieval module, which was based on keyword matching. In another innovation, in the process of training, the Buster.AI team [5] looked for similar claims (but unmatching) as the ones that the model already predicted correctly and added them to the training data.

[4]http://www.elastic.co/

In the present work, we experiment with using a triplet loss representation of the data, both with the aim to increase the amount of information we can use to train our models, to deal with data imbalance, and to encourage the model to learn the subtleties of the data. The mathematical formula for triplet loss is presented in Equation 1. A triplet consist of an anchor (A), plus a positive (P) and a negative (N) example. The training objective of the triplet loss is to increase the distance between an anchor and a negative example, while decreasing the distance between an anchor and a positive example. α is a margin between these two distances and it a hyperparameter of the model.

$$\mathcal{L}(A, P, N) = max(\|f(A) - f(P)\| - \|f(A) - f(N)\| + \alpha, 0) \tag{1}$$

The triplet loss was first successfully implemented in computer vision for face recognition ([6]), and has more recently also been applied in Natural Language Processing tasks [7, 8]. What is particularly interesting about triplet loss is that it is possible to create triplets in an online manner, i.e., after each training round, the next most difficult (as of yet unused) negative example can be chosen to create a triplet for training. In this way, the model can learn more nuanced representation.

3. Data Preparation

The PolitiFact dataset for the CLEF-2021 CheckThat! lab task 2B consist of 700 claims (*Input*) from political debates or speeches, matched against 19,250 previously fact-checked claims (*VerClaim*). All the data for this subtask was in English (however, the lab featured tasks in five languages). Each *VerClaim* comes with a *Title* and a *Body* of an article discussing the veracity of the claim. The dataset is divided into training and development parts, with 561 input claims for training, and 139 for testing. In order to emulate the desired use of the system, the smaller set (which we treated as a test set) consists of events that happened later in time than the events that were used to produce input claims for the training set.

3.1. Triplet Construction

The dataset is severely imbalanced: for each positive pair of *Input*–*VerClaim* there are almost 20,000 negative examples. Thus, the triplet loss can be used as a strategy to compensate for this imbalance when fine-tuning sentence BERT. We converted the training into a collection of triplets, i.e., an anchor, a positive, and a negative claim. The anchor is an *Input* claim, while the positive and the negative claims are taken from *VerClaims*. Positive claims matched to the anchor were already provided by the shared task organisers. For the negative claims, we ranked all the previously fact-checked claims against the anchor using the BM25 algorithm [9]. The highest scoring claim that was not a positive match to the anchor was chosen. The advantage of this method is that the difference between the positive and the negative claims in a triplet is not trivial, and it can really challenge the system.

3.2. Dataset Augmentation

The triplet dataset is quite small: the training set contains just 561 input claims. Thus, we explored two methods for data augmentation. The key problem is the limited number of positive examples, and thus our goal was to extend those.

Potential sentences that could expand the narrow pool of matched claims were mined from the *Title* and the *Body* of the articles.

Our first method simply created additional triplets, where the positive example was the article title. Examples are shown in Table 1. This doubled the size of the dataset to 1,124 triplets.

Our second augmentation method added new triplets with positives examples made of sentences in the body of the articles. Table 1 also shows examples of such artificial positives for this second approach. From the articles, we chose sentences that had the biggest cosine similarity between their embedding and the *Input* claim embedding, relying on the `stsb-roberta-large` model [5]. Augmenting the training set with article sentences almost quadrupled (2,212 triplets) the number of triplets.

4. Experiments

Our experiments focused on fine-tuning sentence BERT. We used the `sentence-transformers` package[6]. Following common practice [3], we ran experiments using both a smaller language model (245M parameters: `distilbert-base-nli-mean-tokens`) and a larger one (1.31G parameters: `stsb-roberta-large`)[7]. We used a training batch size of 30, warm up steps set to 10% of the total, and triplet margin of 5. With the distilled version of sentence-BERT, we trained for 5 epochs, and with RoBERTa, we trained for 8 epochs. For evaluation, fact-checked claims were ranked against input claims based on cosine similarity between their BERT embeddings.

4.1. Online Mining of Triplets

By choosing negative claims that are still quite similar to the anchor claim in our triplets, our negative claims are quite challenging. We can also use the learned representation to create additional challenging claims. After each training epoch, the *Input* claim and the collection of all (*VerClaims*) are again encoded by currently fine-tuned model. Because the model was trained for one epoch, the encoded representations are different than before that epoch. We then computed the cosine similarity between the *Input* claim and (*VerClaims*) again. Then we constructed new triplets, where we chose the most similar fact-checked claim that is not a positive as a new negative. These triplets replaced the ones from the previous epoch. Therefore, as the model learns the dataset better, we iteratively select the most challenging unmatching *VerClaims*. This is an online mining of triplets, and it yields consistent improvement of around 0.05 MAP points (Table 2). Unfortunately it also substantially increases training time (from 8 minutes to around 30 minutes using Google Colab GPU[8] and distilBERT).

[5]http://huggingface.co/sentence-transformers/stsb-roberta-large
[6]http://www.sbert.net/index.html
[7]http://public.ukp.informatik.tu-darmstadt.de/reimers/sentence-transformers/v0.2/
[8]http://colab.research.google.com/

Table 1
Positive examples from our augmented dataset.

Input	She voted for the War in Iraq.
Vclaim	Hillary Clinton "agreed with (John McCain) on voting for the war in Iraq."
Title	Clinton and McCain had same vote on Iraq war
Top−3 sentences of *Text*	1. Obama is right: Clinton and McCain were on the same side in voting for the use of force in Iraq. 2. Clinton voted for the measure, as did McCain. 3. In making his argument, Obama attacked Clinton for voting with Republicans on national security issues.. .
Input	Governor Romney, I'm glad that you recognize that alQaida's a threat because a few months ago when you were asked, what's the biggest geopolitical threat facing America, you said Russia not alQaida, you said Russia.
Vclaim	"A few months ago when you were asked what's the biggest geopolitical threat facing America, you said Russia."
Title	Obama: Romney called Russia our top geopolitical threat
Top−3 sentences of *Text*	1. In the debate, Obama said Romney called Russia "the biggest geopolitical threat facing America.". 2. He has called Russia the biggest geopolitical foe or enemy for the U.S., but he has said the biggest threat is Iran.. . 3. Blitzer asked Romney if he thought Russia is a bigger foe than Iran, China or North Korea.. .
Input	You were for a singlepayer system, a Canadian−style system.
Vclaim	Says Donald Trump is for a single-payer health care system.
Title	Is Donald Trump still 'for single-payer' health care?
Top−3 sentences of *Text*	1. Perry said Trump is "for single-payer health care.". 2. Fifteen years ago, Trump was decidedly for a universal healthcare system that resembled Canada's system, in which the government pays for care for all citizens. 3. "How can anyone who's a conservative stand up and say I am for single-payer health care?".

4.2. Model Robustness

Our experimental results with augmenting the triplet dataset were mixed. Augmenting with sentences from the articles actually resulted in performance that was worse than the baseline performance (i.e., not fine-tuned) achieved with distilBERT (see Table 3). This might be due to the article sentences being more context-dependent and having more pronouns rather than named entities. Moreover, our inspection showed that the top-3 sentences did not always say exactly the same thing as the *Vclaim.* An example is the third sentence for the third input in Table 1, where the augmenting claim is much more general and only vaguely refers to the former President of USA.

Table 2
Comparison of triplet loss training with and without online mining when fine-tuning pre-trained language models of different sizes. The absence/presence of online mining in the model is indicated with *no*/*yes* after the model name.

Model	MAP@k						HasPositives@k					
	1	3	5	10	20	all	1	3	5	10	20	50
distilBERT:no	.236	.27	.273	.279	.281	.286	.279	.35	.364	.393	.45	.55
distilBERT:yes	**.264**	**.305**	**.312**	**.323**	**.327**	**.33**	**.307**	**.414**	**.429**	**.507**	**.564**	**.593**
RoBERTa:large:no	.25	.277	.289	.298	.303	.308	.307	.35	.393	.457	.493	.586
RoBERTa:large:yes	**.293**	**.324**	**.333**	**.342**	**.347**	**.352**	**.35**	**.414**	**.45**	**.507**	**.571**	**.629**

Table 3
Impact of training *without online mining* on augmented and unaugmented datasets, using distilBERT. The number of training examples is shown in parentheses.

Model	MAP@k						HasPositives@k					
	1	3	5	10	20	all	1	3	5	10	20	50
Baseline (no training)	.193	.213	.216	.223	.226	.229	.207	.236	.257	.336	.364	.429
Unaugmented (561)	.236	.27	.273	.279	.281	.286	.279	.35	.364	.393	.45	.55
Titles (1,124)	.243	.258	.26	.267	.271	.275	.286	.314	.329	.386	.421	.486
Top–3 article sentences (2,212)	.171	.204	.207	.209	.213	.218	.207	.271	.286	.307	.393	.464

Using the article titles as additional positive examples to double the number of triplets (see Table 3) did not bring the expected boost in performance on the test set: for some of the evaluation measures, there was a slight improvement, but for other, there was a decrease. This could be because article titles and previously fact-checked claims differ stylistically: titles are usually shorter, contain less quotations, and are much better formulated.

However, even though models trained only on the dataset provided by the task organisers and models trained on the dataset augmented with article titles both exhibited roughly similar performance, in closer analysis, we noticed that data augmentation did have a positive impact. We constructed a new test set consisting of input claims from the test set matched on the article titles instead of the corresponding fact-checked claims. The comparison of the performance is shown in Table 4. There we can see that both fine-tuned models improve on both test sets. However the one trained on the augmented dataset performs considerably better. These results suggest that using the dataset extended with titles is better in abstracting from specific formulations of previously checked claims and is therefore more robust. In the final submission, the use of the RankSVM model further improved the results. The features used in training are not only concerned with *VerClaims*, but also with the *Titles* of the articles.

Table 4
Evaluation of the model robustness when using data augmentation with online mining and distilBERT. The number of training examples is shown in parentheses.

Model *Evaluated on*	MAP@k						HasPositives@k					
	1	3	5	10	20	all	1	3	5	10	20	50
Baseline (no training) *VerClaims*	.193	.213	.216	.223	.226	.229	.207	.236	.257	.336	.364	.429
Unaugmented (561) *VerClaims*	.264	.305	.312	.323	.327	.33	.307	.414	.429	.507	.564	.593
Titles (1,124) *VerClaims*	.271	.302	.314	.324	.327	.33	.329	.436	.464	.507	.536	.586
Baseline (no training) *Titles*	.029	.042	.045	.047	.048	.052	.029	.057	.071	.107	.121	.179
Unaugmented (561) *Titles*	.086	.095	.105	.11	.114	.118	.1	.121	.157	.2	.257	.3
Titles (1,124) *Titles*	.129	.173	.181	.183	.187	.192	.136	.257	.293	.307	.321	.414

4.3. RankSVM

As shown in [3], strong results for the task of detecting previously fact-checked claims are achieved when combining BM25 features with similarity scores obtained from sentence BERT. The former method is good at detecting exact matches of words between compared claims, which is especially useful when the same phrases (e.g., named entities) are used. On the other hand, the latter method provides more semantic matching, grasping similarities of paraphrases. In order to get the best of both worlds, we decided to train a rankSVM model with an RBF kernel using a pairwise loss. The features used were scores and reciprocal ranks from BM25 and from sentence BERT for matching input claims against *VerClaims* and against *Titles*, i.e., a total of eight features. We obtained the BM25 scores using the BM25Plus algorithm from the rank-bm25[9] Python package. The cosine similarity scores were produced by matching embeddings created by a distilBERT model fine-tuned with online mining on a dataset augmented with *Triplets*.

We trained the RankSVM reranker on the original unaugmented training dataset from PolitiFact. The evaluation results of the reranking are shown in the top half of Table 5. We can see that using rankSVM improved the scores by around 0.02 points absolute across all evaluation measures.

[9]https://pypi.org/project/rank-bm25/

5. Official Evaluation Results on the Test Set

Our official submission used rankSVM and the features described in the previous section. In the bottom half of Table 5 are shown the official evaluation results of our BeaSku team submission. Our model was ranked second in the competition, just 0.001 points absolute behind the winning team, based on the official evaluation measure: MAP@5.

Table 5
Evaluation scores for our official submission for the shared task.

Official evaluation results on the test set.						
	MAP@k					
MRR	1	3	5	10	20	1000
.320	.266	.308	**.327**	.332	.332	.333

6. Conclusion and Future Work

We have described the submission of our BeaSku team for the CLEF-2021 CheckThat! Lab Task 2B on detecting previously fact-checked claims. Our main focus was on fine-tuning sentence BERT on the training dataset. For that, we used triplet loss and we augmented the dataset to make the model more robust. We then used scores and reciprocal ranks produced by this model, together with BM25 scores, to train a reranker. We further discussed a number of experiments and improvements introduced in the model such as online mining of triplets, different methods for data augmentation and various extensions of sentence BERT. Our official submission was ranked second in the competition with a MAP@5 score of 0.327.

As dataset augmentation yielded only a small performance boost, increasing the dataset size is a good direction for future work. This can be done through adding paraphrases of *VerClaims*. There are not many satisfactory paraphrase generators, but fine-tuning GPT-2 has shown promising results [10]. Moreover, the additional features used in the reranker, such as matching against the article sentences or Natural Language Inference metrics, could be included when training the reranker.

References

[1] H. Allcott, M. Gentzkow, Social media and fake news in the 2016 election, Journal of Economic Perspectives 31 (2017) 211–236.

[2] A. Barrón-Cedeno, T. Elsayed, P. Nakov, G. Da San Martino, M. Hasanain, R. Suwaileh, F. Haouari, N. Babulkov, B. Hamdan, A. Nikolov, et al., Overview of CheckThat! 2020: Automatic identification and verification of claims in social media, in: Proceedings of the International Conference of the Cross-Language Evaluation Forum for European Languages, CLEF '20, 2020, pp. 215–236.

[3] S. Shaar, N. Babulkov, G. Da San Martino, P. Nakov, That is a known lie: Detecting previously fact-checked claims, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL '20, 2020, pp. 3607–3618.

[4] L. Passaro, A. Bondielli, A. Lenci, F. Marcelloni, UNIPI-NLE at CheckThat! 2020: Approaching fact checking from a sentence similarity perspective through the lens of transformers, Cappellato et al.[10] (2020).

[5] M. Bouziane, H. Perrin, A. Cluzeau, J. Mardas, A. Sadeq, Team Buster.ai at CheckThat! 2020: Insights and recommendations to improve fact-checking (2020).

[6] F. Schroff, D. Kalenichenko, J. Philbin, FaceNet: A unified embedding for face recognition and clustering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR '15, 2015.

[7] F. Ren, S. Xue, Intention detection based on siamese neural network with triplet loss, IEEE Access 8 (2020) 82242–82254.

[8] J. M. Coria, S. Ghannay, S. Rosset, H. Bredin, A metric learning approach to misogyny categorization, in: Proceedings of the 5th Workshop on Representation Learning for NLP, 2020, pp. 89–94.

[9] S. Robertson, H. Zaragoza, The probabilistic relevance framework: BM25 and beyond, Now Publishers Inc, 2009.

[10] S. Witteveen, M. Andrews, Paraphrasing with large language models, in: Proceedings of the 3rd Workshop on Neural Generation and Translation, Hong Kong, China, 2019, pp. 215–220.