University of Bonn

CAISA lab

Introduction to Natural Language Processing

(Winter Semester 2023/2024)

Default Project

Final Report of Team # 14

Fact Verification using the FEVER Dataset

Group Members

Suyash Thapa	-	50205756
Rouaa Maadanli	-	3365771
Abhishek S Pillai	-	50010996
Hossam Fawzy Mohamed Elsafty	-	3306219
Kashan Uddin Zaigham Khan	-	50064238

Lecturer: Prof. Dr. Lucie Flek
Course coordinator: Vahid Sadiri Javadi

February 12, 2024

1. Introduction

In an era characterized by the proliferation of information through the internet and social media, the task of moderating the vast amount of data generated daily has become an overwhelming challenge for human reviewers alone. Addressing this need, our project is dedicated to the evaluation and enhancement of natural language processing (NLP) capabilities, particularly in the domain of fact-checking.

Project Background and Purpose

The driving force behind our endeavor is the utilization of the FEVER (Fact Extraction and VERification) dataset[1].The fundamental purpose of our project is to contribute to the development of an automated Fact Checking system that can assist in the verification of claims, a task that has become increasingly vital in the digital age[2].

Objectives and Problem Statement

Our project aims to achieve several key objectives. Firstly, we seek to enhance the precision and efficiency of language comprehension in NLP[3] models. This endeavor is underpinned by the need to address the classification of assertions into distinct categories such as "Supported," "Refuted," or "NotEnoughInfo"[1]. By leveraging FEVER's structured documentation and labeled dataset[1], we aspire to facilitate the training of accurate models for automated fact-checking systems[2].

Research Question or Hypothesis

The central research question guiding our efforts is how we can leverage the FEVER dataset to improve the precision[4] and efficiency of NLP models in the critical process of claim verification[2]. By exploring this question, we anticipate contributing to the advancement of automated fact-checking systems[5] and, in turn, addressing the challenges posed by the overwhelming volume of information in today's digital landscape.

In the subsequent sections, we will delve into the specifics of our methodology, drawing upon foundational works such as [6] and [7], provide an analysis based on these frameworks, and present our findings in light of existing literature.

2. Literature Review

The goal of this literature review is to explore existing research and methodologies in the field of natural language processing (NLP)[3] and fact-checking[5].

According to [8] some based modeling strategies are known for Fact checking in NLP. The first one is Claim detection, which is commonly presented as a classification challenge, wherein models determine the checkability or check-worthiness of claims. The second approach is Evidence Retrieval and Claim Verification, which are commonly addressed together.

While in [9] they have stated that the most fact checking approaches are supervised based classification approaches. This is what we are going to delve in more. In this method a classifier learns a text from some form of labeled data that is provided at training. This trait that is independent of the task input or what sources of evidence are considered.

The main limitation of text classification approaches is that fact checking a claim requires additional world knowledge, typically not provided with the claim itself.[10].

[11] modeled fact checking as a form of Recognizing Textual Entailment (RTE)[12], predicting whether a premise, typically (part of) a news article, is for, against, or observing a given claim.

The RTE-based models assume that the textual evidence to fact check a claim is given. Thus they are inapplicable in cases where this is not provided, or it is a rather large textual resource such as Wikipedia (as in FEVER[1]). For the latter[1], they developed a pipelined approach in which the RTE component is preceded by a document retrieval and a sentence selection component. There is also work focusing exclusively on retrieving sentence-level evidence from related documents for a given claim [13].

A popular type of model often employed by fact checking organizations in their process is that of matching a claim with existing, previously fact checked ones. This reduces the task to sentence level textual similarity as suggested by [14] and implemented in [10].

3. Data Exploration and Preprocessing[1]

3.1. FEVER Dataset

The Fever (Fact Extraction and VERification) is a publicly available dataset that was introduced in 2018 for the task of fact extraction and verification.[1]. The sources for the dataset include Wikipedia[15] articles and a set of claims (statements) extracted from these articles. These claims are collected from Wikipedia articles by selecting sentences or phrases that can be fact-checked.

The dataset consists of 185,445 claims although only 165,447 data points can be used for the classification task as 19,998 data points have been reserved and labels for which are not made available publically. The labelled dataset is divided into training set (145,449 data points) and test set (19,998 data points). The claims are classified as SUPPORTED, REFUTED, or NOT ENOUGH INFO (henceforth referred to as NEI).

The structure of a data-point is as follows:

- id: The ID of the claim
- verifiable: Whether the claim can be verified by evidence (SUPPORTS or REFUTES) or not (NOT ENOUGH INFO)
- label: The annotated label for the claim. Can be one of SUPPORTS, REFUTES, NOT ENOUGH INFO.
- claim: The text of the claim.
- evidence: A list of evidence sets (lists of [Annotation ID, Evidence ID, Wikipedia URL, sentence ID] tuples) or a [Annotation ID, Evidence ID, null, null] tuple if the label is NOT ENOUGH INFO.

Below are the examples of a data-point for each of the classes:

"SUPPORTS" Example

```
{
    "id": 62037,
    "verifiable": "VERIFIABLE",
    "label": "SUPPORTS",
    "claim": "Oliver Reed was a film actor.",
    "evidence": [
        [
            [<annotation_id>, <evidence_id>, "Oliver_Reed", 0]
        ],
        [
            [<annotation_id>, <evidence_id>, "Oliver_Reed", 3],
            [<annotation_id>, <evidence_id>, "Gladiator_-LRB-2000_film-RRB-", 0]
        ],
        [
            [<annotation_id>, <evidence_id>, "Oliver_Reed", 2],
            [<annotation_id>, <evidence_id>, "Castaway_-LRB-film-RRB-", 0]
        ],
        [
            [<annotation_id>, <evidence_id>, "Oliver_Reed", 1]
        ],
        [
            [<annotation_id>, <evidence_id>, "Oliver_Reed", 6]
        ]
    ]
}
```

"REFUTES" Example

```
{
    "id": 78526,
    "verifiable": "VERIFIABLE",
    "label": "REFUTES",
    "claim": "Lorelai Gilmore's father is named Robert.",
    "evidence": [
        [
            [<annotation_id>, <evidence_id>, "Lorelai_Gilmore", 3]
        ]
    ]
}
```

"NEI" Example

```
{
    "id": 137637,
    "verifiable": "NOT VERIFIABLE",
    "label": "NOT ENOUGH INFO",
    "claim": "Henri Christophe is recognized for building a palace in Milot.",
    "evidence": [
        [
            [<annotation_id>, <evidence_id>, null, null]
        ]
    ]
}
```

The evidence reference given in the dataset can be extracted from the pre-processed 2017 Wikipedia data dump provided along with the dataset.

3.2. Data Analysis

We analysed the training data for distribution of samples between the three labels and found that the data is heavily inclined towards "SUPPORTS" label, accounting for 55% of the train set. The split between REFUTES and NEI is much more balanced (Figure 1).

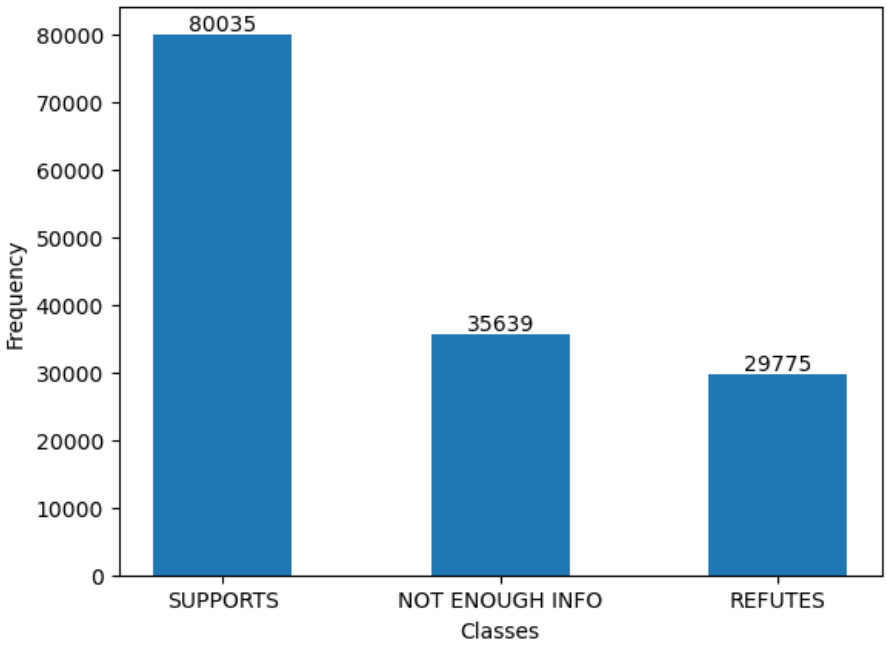


Figure 1: The number of data points for each label in the train dataset

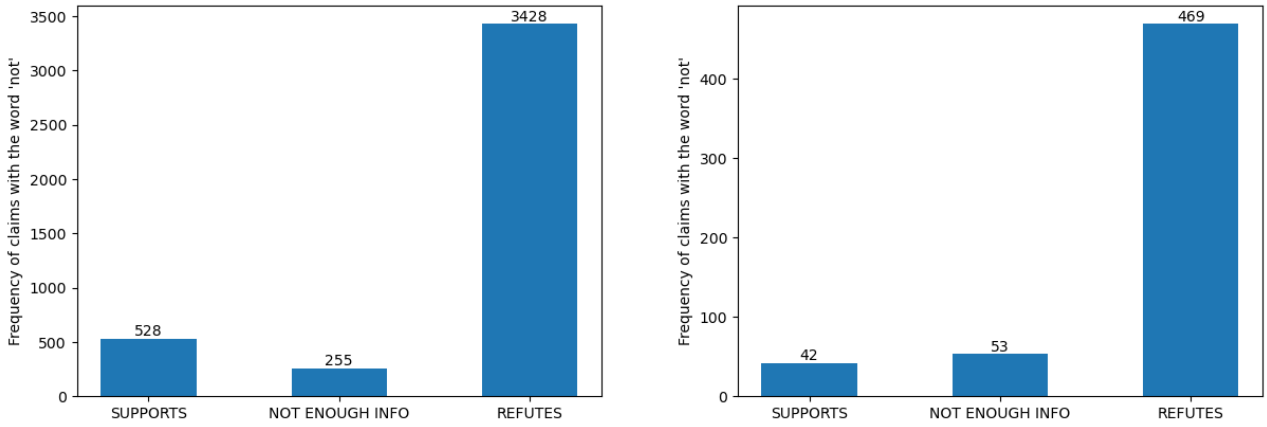
Further we analyse the length of the 'claim' statements (characterised by number of words) for each label. We found that nearly 70% of claims were sentences with a word length between [5, 9] for all 3 labels. The longest claim statements were 55 words long in the SUPPORTS label, 32 words long in the REFUTES label, and 33 words long in the NEI label. The vocabulary (number of unique words) for 'claim' statements stands at 21483 words for the train dataset and 9668 words for test dataset.

While analysing the dataset we noticed a relation between the number of 'claim' statements containing a word and the label of the claim. One such relation was between the word "not" and label REFUTES. For the train dataset the number of claims containing the word "not" was rather high in REFUTES class as compared to other classes. Figure 2 gives a clear picture of this imbalance. This might lead to a bias given the difference between the number of SUPPPORTS and REFUTES samples.

The model uses Wikipedia data dump from 2017 provided by Thorne et.al[1]. The dataset contains 109 json files with a total of 5 million wikipedia articles with the first paragraph of the article and it's senetnces in a jsonlines format. This dataset was used to retrieve evidences in the end-to-end models. The format of the Wikipedia datadump is as follows:

- id: name of the wikipedia page, also a unique id
- test: the first paragraph of the article
- lines: "next line" separated sentences with metadata. Each item consists of a sentence number, the sentence, and noun phrases in the given sentence, tab separated.

The training samples contain evidences only from 7999 Wikipedia documents out of 5 million.



(a) Train Dataset

(b) Test Dataset

Figure 2: The imbalance between the instances of the word "not" in the claim statements by label in train (2a) and test (2b) datasets.

3.3. Data Preprocessing

During our study on FEVER[1] dataset, we have tried to implement several models tackle fact extraction and verification challenges. To make these models work effectively, we've applied different preprocessing techniques tailored to each model's needs. We will summarise these techniques depending on each model as follows:

3.3.1. Random Forest(RF)[16] and Support Vector Machine(SVM)[17]

- Text data is vectorized using TF-IDF (Term Frequency-Inverse Document Frequency)[18].
- Labels are encoded using LabelEncoder.
- Linear Support Vector Classifier (LinearSVC) is trained on the vectorized data.

3.3.2. Recurrent Neural Network (RNN)[19]

- Tokenization[20] and padding to a maximum length[21] are performed using Tokenizer and pad_sequences from TensorFlow's Keras API.
- Labels are encoded using LabelEncoder.
- Sequences are converted to a fixed length (max_len) suitable for input to the RNN model.

3.3.3. tf-iDF based Sentence Retrieval with MLP Classifier

The pre-processing steps for this model involved removing redundant $< evidence\_id >, < annotation\_id >, id, verifiable$ fields from the training samples. Furthermore, the evidence list for each sample was flattened. The 7999 wikipedia pages were filtered using the page id given in the evidence. We found that 81 evidences containing non-English characters in their ids were not getting matched by simple string matching. Hence, given the already large size of the dataset, we chose to discard those samples which had those documents as evidences. This process affected only 719 samples from "SUPPORTS" and "REFUTES" classes combined.

We also decided to limit our Wikipedia database. We chose the 7918 documents which contain the evidences and selected 110 random documents from each wikipedia json file apart from the ones already selected. This process reduced our searchable database to nearly 20k documents while ensuring that all evidences were still available within the dataset. Given the process involved, this method is unlikely to affect the performance in a negative capacity.

The Claims and 20k documents were then cleaned of any foreign words or punctuation and evidence were populated by cross referring the Wikipedia articles. Numbers were intentionally left in the Dataset and Wikipedia Database because dates are an important part of some claims, like birth dates, year of establishment of an organisation, etc.

Given that we have nearly 145k training samples, we decided to limit our pre-processing to the train samples alone and divided the train samples into test and train datasets later on.

3.3.4. BERT Baseline Model[22]

- Tokenization[20] is performed using the BERT tokenizer (bert-base-uncased). The input text is tokenized and converted into tensors[23] suitable for BERT model input.
- Padding and truncation are applied to ensure that all sequences have the same length (max_length=512).
- Labels are encoded using LabelEncoder and converted to PyTorch[24] tensors[23].

To summerize, each model has its specific requirements, and preprocessing is tailored accordingly. BERT requires tokenization and special tensor format, RNN needs tokenization and padding, and SVM relies on TF-IDF vectorization. These preprocessing steps are crucial to ensure that the input data is in a suitable format for each model's training and evaluation.

4. Methodology

As we have mentioned in the previous section, we have implemented a variety of models to assess the capabilities of FEVER dataset in claim verification and extraction. Our primary focus has been on applying the recommended pipeline outlined in [1] for claim verification and evidence retrieval. Which was detailed in [6]. Additionally, we have employed a diverse set of classifiers[25], including SVM (Support Vector Machine) [17], RF (Random Forest) [16], RNN /Recurrent Neural Networks) [19], and the base BERT model [22], to address the problem of multi-class classification in claim verification.

4.1. Support Vector Machine

Support vector machines (SVMs) are particular linear classifiers that are based on the margin maximization principle. They perform structural risk minimization, which improves the complexity of the classifier to achieve excellent generalization performance. The SVM accomplishes the classification task by constructing, in a higher dimensional space, the hyperplane that optimally separates the data into two categories[17]. In our FEVER dataset study, Support Vector Machines (SVMs) play an important role due to their efficient linear classification and margin maximization. SVMs prioritize excellent generalization, aligning well with FEVER's complexities for robust claim verification and evidence retrieval. Their ability to construct optimal hyperplanes in higher-dimensional spaces enhances accuracy, making SVMs suitable to our approach.

4.2. Random Forest

Random Forests is an ensemble learning technique. It is a hybrid of the Bagging algorithm and the random subspace method, and uses decision trees as the base classifier. Each tree is constructed from a bootstrap sample from the original dataset. An important point is that the trees are not subjected to pruning after construction, enabling them to be partially overfitted to their own sample of the data[16]. To this end, these characteristics of Random Forests have led us to consider implementing it as a robust approach for tasks like claim verification on the FEVER dataset. The adaptability and generalization capabilities of Random Forest align well with the complexities present in such datasets.

4.3. Recurrent Neural Network[19]

RNNs featuring gated recurrent cells, notably Long Short-Term Memory (LSTM) [26] and Gated Recurrent Units (GRU) [27], demonstrate an enhanced ability to capture long dependencies and extract comprehensive global information. In light of these capabilities, our objective is to evaluate the performance of RNNs on the FEVER dataset. We specifically focus on LSTM due to its unique architecture, which incorporates memory cells and gates designed to address the vanishing gradient problem inherent in conventional RNNs. This architectural refinement positions LSTM as a promising choice for tasks requiring the retention and utilization of contextual information over extended sequences.

4.4. End-to-end tf-iDF based Claim Verification with MLP Classifier

This is a base end-to-end model with a pipeline containing tf-iDF cosine similarity for Document Retrieval, Sentence Retrieval, and Textual Entailment with MLP as a classifier.

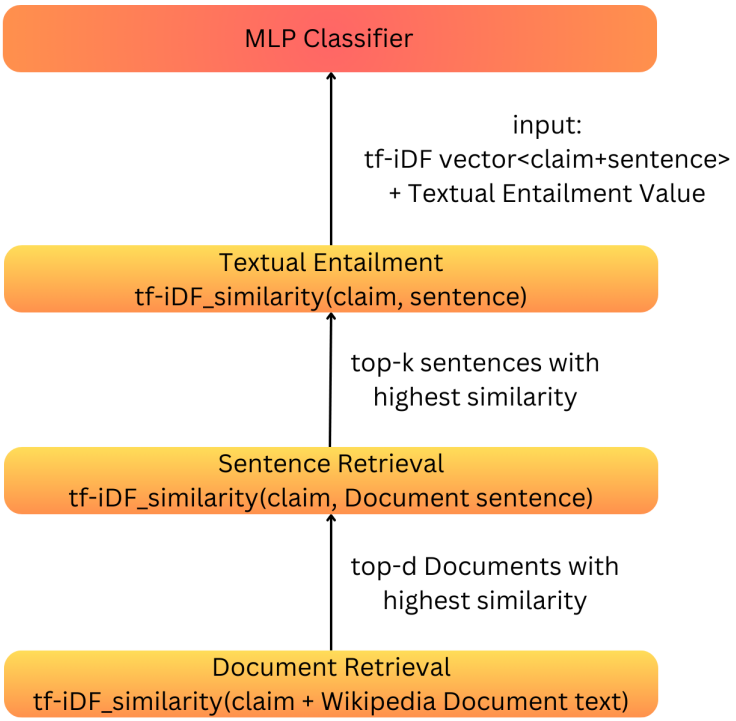


Figure 3: The tf-iDf based Sentence Retrieval and Claim Verification Pipeline

4.4.1. Document Retrieval

Using the 20k Wikipedia text as the base, a tf-iDF vectorization was performed for all claims from the preprocessed dataset. A tf-iDF based similarity measure between the claims and the 20k docs were used to select top d documents with the highest similarity[28].

4.4.2. Sentence Retrieval

A tf-iDF similarity measure between the claim and the sentences from the Wikipedia dataset of d documents selected in the previous step were used to select top k sentences with the highest similarity. These sentences were considered as probable evidences for the downstream task. We found that keeping $d = 5$ and $k = 5$ gave the best Sentence Retrieval accuracy at 36.03%.

4.4.3. Textual Entailment

Similarity measure between the claims and top k sentences retrieved in the Sentence Retrieval process was taken as the RTE value between the claim and the evidence. This RTE value was taken as input along with the tf-iDF vector[28] of the combined claim-evidence pair statement as an input to the MLP classifier.The input to the classifier was stored as a sparse array of $[tf\_iDF < claim + evidence > + RTE\_value]$.

4.4.4. Training Dataset

For the training dataset, we chose to pair the claims with the already provided evidences and perform textual entailment + tf-iDF vector based MLP classification.

However, the class "NOT ENOUGH INFO" has no evidences provided with it as examples of "irrelevant" evidences. Hence we decided to populate the evidence for "NOT ENOUGH INFO" using a semi-random Sentence Retrieval procedure. Herein we would first select top d documents based on similarity with the claim. However, during the

Sentence Retrieval procedure instead of selecting the top similarity sentences, we would just pick up the middle statement of each previously selected top document as an "irrelevant" evidence.

4.4.5. Test Dataset

We prepared two datasets for testing our model.

- Type-I dataset was created by making claim-evidence pairs from the available train samples just like the training set and calculating the accuracy of the classifier to accurately predict the labels. Evidences for "NOT ENOUGH INFO" were selected using the same semi-random method as discussed above.

  This dataset was designed to test the ability of the MLP classifier to correctly predict the labels provided the scenario that the upstream Pipeline works perfectly. FEVER Score is not applicable for this dataset.

- Type-II dataset (Retrieved Evidence Dataset) was created on the same samples as the first one but here the entire Document and Sentence Retrieval pipeline was employed to select the top 5 most similar statements as evidences. Hence each claim was paired with 5 different evidences in the second test dataset. This dataset can be evaluated on the FEVER Score.

  This dataset was designed to calculate the overall robustness of the model such that the model should ignore (classify as "NOT ENOUGH INFO") any wrong claim-evidence pairs produced by the pipeline.

For both the training and the two test datasets, we generated two versions of datasets, one small and one large. Due to the data preparation techniques used each claim was paired with several evidences to generate multiple data points. This lead to much more samples being generated than the size of the dataset chosen.

For training we chose 13750 samples for small dataset giving 32.2k training data points, and for the large model we chose 123k samples giving 285k training data points. For testing we chose 7.2k data points for the small dataset giving 16.7k data points for Type-I dataset and 36.1k data points for Type-II data. For the large dataset we chose 21.7k samples which resulted in 50k and 108k data points for Type-I and Type-II datasets respectively.

4.5. Claim Verification and Evidence Extraction using Bert

Pre-trained language models like Bidirectional Encoder Representations from Transformers (BERT) have revolutionized the field of natural language processing (NLP).BERT has significantly advanced the performance in a wide variety of information retrieval and natural language processing tasks including passage re-ranking, question answering, and question retrieval.Developed by Google, BERT is a deep learning model that has set new standards for a variety of NLP tasks, including but not limited to text classification, question answering, named entity recognition, and sentiment analysis.

The pipeline Figure 4 has adopted a three-step pipeline.

- Document Retrieval: a set of documents, which possibly contain relevant information to support or reject a claim, are retrieved;
- Sentence Retrieval: five sentences are extracted from the retrieved documents;
- Claim Verification: the claim is verified against the retrieved sentences.

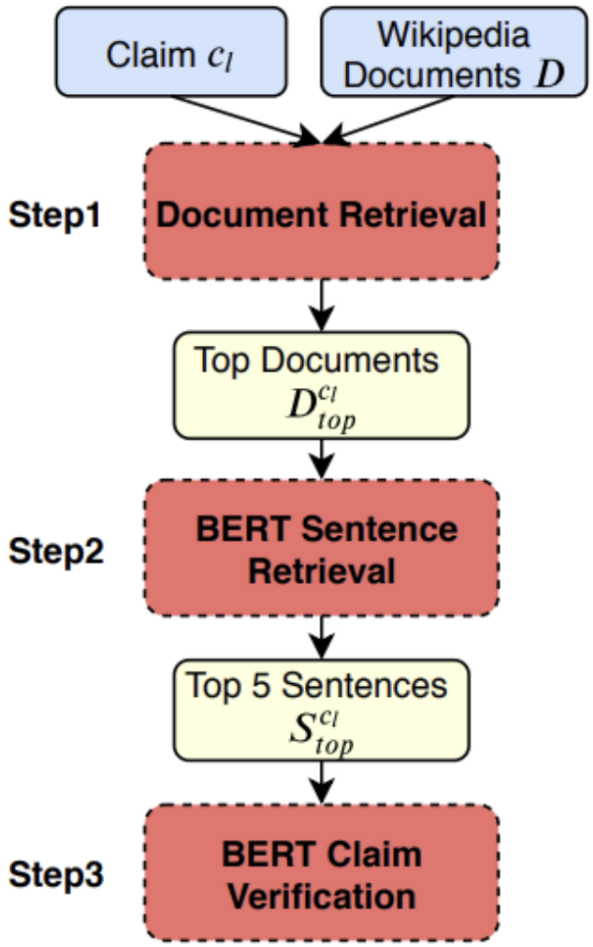


Figure 4: Pipeline for BERT [29]

4.5.1. Document Retrieval and Top Documents

We are refining our document retrieval approach based on the methodologies outlined in the paper UKP-athene [30]. UKP-athene [30] had used an entity linking approach [31] for document retrieval. We followed the same methodology done in the paper UKP-athene [30] for Document retrieval with few changes, where 3 steps are defined for document retrieval:

1. **Mention Extraction [30]**
   Mention extraction takes the claim from the FEVER dataset and extracts the noun phrases. The UKP-athene [30] uses a constituency parser from AllenNLP [32] to find noun phrases from the claim. Example:

   Claim: "The Mona Lisa, a famous painting by Leonardo da Vinci."

   Noun Phrases:

   - "The Mona Lisa"
   - "a famous painting by Leonardo da Vinci"

2. **Loading document**
   We meticulously curated a comprehensive database utilizing the Wikipedia dump available until 2017, sourced from the esteemed Fever website. Leveraging this extensive dataset, we meticulously constructed a robust SQLite database encapsulating the entirety of Wikipedia's content up to the specified cutoff date. By incorporating data up to 2017, our database encompasses a wealth of knowledge spanning various domains and topics. This meticulously crafted repository serves as a foundational resource, enabling efficient retrieval of pertinent documents for a diverse array of research and analysis endeavors.

3. **Top-documents** We used the noun phrases we got for the claim and used them to query it with the SQL database we made using the Wikipedia dump until 2017. From where the top 7 overlapping articles with the query are chosen as top documents for the claims.

4.5.2. Sentence retrieval

After selecting the top documents, we further refine our analysis by utilizing a pre-trained BERT model to extract the top 3 sentences from these documents, as discussed in Soleimani et al. [29]. This process begins by tokenizing both the claim sentence and each evidence sentence using the BERT tokenizer. This step ensures that the text is converted into a format that the BERT model comprehends, with special tokens such as [CLS] and [SEP] added to demarcate the beginning and end of each sequence.

Subsequently, the tokenized sequences are passed through the pre-trained BERT model, allowing us to obtain contextual embeddings. These embeddings encapsulate the semantic essence of the entire sentence, enabling us to capture nuanced meanings and relationships within the text.

To assess the relevance of each sentence to the claim, we employ cosine similarity—a widely used metric for measuring the similarity between two vectors. By computing the cosine similarity between the embedding of the claim sentence and those of the top documents or sentences, we can quantify the degree of alignment in their semantic representations.

$$\text{cosine\_similarity}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}$$

The resulting cosine similarity scores facilitate the ranking of sentences, with higher scores indicating a greater alignment with the claim. This ranking mechanism enables us to identify the most pertinent documents for either supporting, refuting, or indicating insufficient evidence for the claim.

Finally, we select the top 3 sentences with the highest similarity scores.

4.5.3. Bert Claim Verification

Each claim c_l within the Fever dataset is accompanied by its corresponding top sentences $S_{\text{top}}^{c_l}$, which represent the most salient evidence extracted for that particular claim. To harness the predictive power of these claim-evidence pairs, we leverage a BERT-based model trained on the Fever dataset, encompassing claims, evidence, and their respective labels.

Following the training phase, we utilize the trained BERT model to infer the textual entailment relationship between each claim and its associated top sentences. This involves presenting the model with the concatenated representation of the claim and its top sentences. The model's predictions are then categorized into one of three classes: "SUPPORTS," "REFUTES," or "NOT ENOUGH."

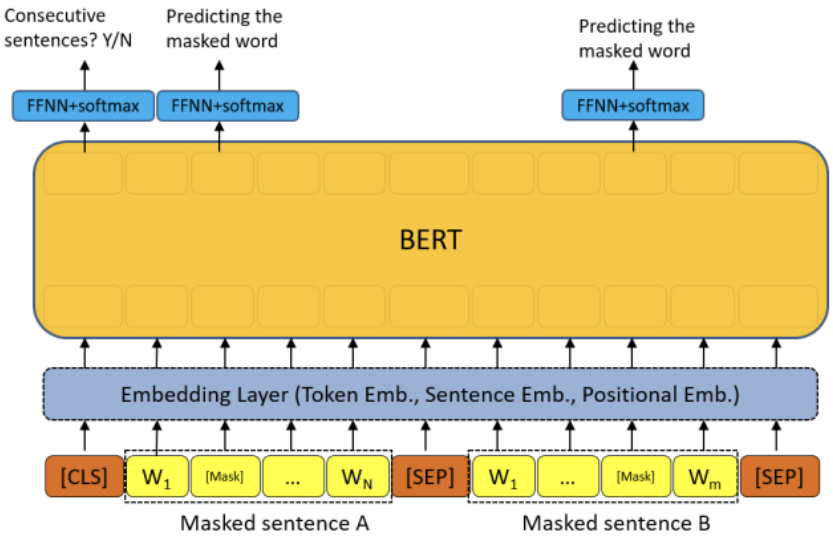


Figure 5: BERT-architecture [33]

4.6. Libraries, Frameworks, and APIs Choice Discussion

This section delves into the toolbox we've employed for our classification task. We've strategically chosen libraries and frameworks like scikit-learn for SVM and RF, TensorFlow for RNN, and Hugging Face Transformers for BERT. Each tool plays a crucial role in addressing the unique demands of its respective model. Let's explore how these choices contribute to the efficiency and effectiveness of our classification solution:

4.6.1. Random Forest (RF)

- Library: scikit-learn
- Reasoning: Scikit-learn's implementation of Random Forest, provided by the RandomForestRegressor class, is a robust and widely-used tool for classification tasks. It is known for its simplicity, scalability, and effectiveness.

4.6.2. SVM (Support Vector Machine)

- Library: scikit-learn
- Reasoning: Scikit-learn is a widely-used machine learning library in Python, known for its simplicity and efficiency. It provides an easy-to-use implementation of SVM through the LinearSVC class.

4.6.3. RNN (Recurrent Neural Network)

- Library: TensorFlow (tf.keras)
- Reasoning: TensorFlow is a powerful and widely-used open-source deep learning framework. The Sequential model in tf.keras provides an easy-to-use interface for building and training neural networks, including RNNs.

4.6.4. tf-iDF based Claim Verification with MLP Classifier

- Library: Scikit-learn
- Reasoning: As this was a baseline end-to-end model we decided to keep the implementation as simple as possible to allow the ability to quickly modify the model for repeated testing.

4.6.5. BERT (Bidirectional Encoder Representations from Transformers)

- Library: Hugging Face Transformers
- Reasoning: The Hugging Face Transformers library provides pre-trained transformer models, including BERT. It simplifies working with state-of-the-art language models for various natural language processing tasks, where the Hugging Face Transformers is a pytorch library.

4.6.6. Additional Libraries

- Pandas: Used for data manipulation and analysis. It provides data structures like DataFrames, which are useful for organizing and processing tabular data.
- datasets (Hugging Face): Used for loading datasets. This library facilitates easy access to various datasets for experimentation and evaluation.

5. Experimental Setup

For hyperparameters optimization we have used ADAM[34] optimization method. Which is suitable for problems that are large in terms of data and/or parameters[34].

5.1. Simple Models (SVM, Random Forest, RNN)

These models were simple claim-label prediction models mostly to figure out the nature of the dataset and find any correlations within the claim and labels that could potentially affect the end-to-end models. Hence, no special tuning was used on these models. The training and Testing sets were obtained for these models using sickit-learn library, particularly using train_test_split.

5.2. End-to-end tf-iDF based Claim Verification with MLP

The MLP classifier was evaluated on two sets of datasets, each having one small and one large sample collection. For the small dataset we chose a single hidden layer of size 40, with early stopping, batch size of 256 and max iterations of 200. For the large datasets we chose two hidden layers of size 40 each with batch size of 1024, early stopping enabled and max iterations of 200. The validation split was chosen at 10% for both the datasets.

These values were chosen based on the validation scores of the MLP model and provide the best results for the given dataset. In our experiments iterations never went beyond 75 and loss/validation-score converged/stabilised within reasonable time.

5.3. Claim Verification and Evidence Extraction using BERT

We utilized a pre-trained BERT model for sentence retrieval, a task where we aimed to identify relevant sentences from a large corpus. Given computational constraints, we limited our analysis to the first 30 rows of the top documents.

The BERT model, trained on both the train and test datasets from FEVER, was enhanced with a special feature: the selection of top sentences based on cosine similarity. This approach allowed us to identify sentences that are most similar to a given query, thereby streamlining the retrieval process.

6. Evaluation and Results

6.1. Evaluation Metrics

6.1.1. Precision

It denotes the proportion of Predicted Positive cases that are correctly Real Positives.[35] Precision reflects the accuracy of the positive predictions made by the model.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

6.1.2. Recall

Recall is the proportion of Real Positive cases that are correctly Predicted Positive. This measures the coverage of the Real Positive cases by the Predicted Positive rule. Its desirable feature is that it reflects how many of the relevant cases the Predicted Positive rule picks up.[35]

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

6.1.3. F1 Score

The harmonic mean of precision and recall.[35] It describes system performance using a scale from zero to one.[36] F1 score provides a balance between precision and recall.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

6.1.4. Accuracy

A measure of the closeness of an estimate to the true mean or variance of a population[37]. Measures the accuracy of predicting (Supported/Refuted/Not enough info) info for a claim.[38]

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Predictions}}$$

6.1.5. FEVER Score

Label Accuracy is a general evaluation metric, which calculates claim classification accuracy rate without considering retrieved evidence. The FEVER score considers whether one complete set of evidences is provided and better reflects the inference ability.[39] It further measures the percentage of correct retrieved evidence for (Supported/Refuted/Not enough info) categories[40]. In other words, FEVER score is specifically designed for fact-checking tasks. It considers the ability of a system not only to classify claims accurately but also to provide relevant and accurate supporting evidence.

We evaluated the performance metrics of our implemented models using the scikit-learn metrics library.

6.2. Results

Task	**top-5**	**top-10**	**top-20**	**top-50**	**top-100**
Document Retrieval	84.65%	89.73%	92.93%	95.38%	96.60%
Sentence Retrieval	36.03%	35.17%	34.25%	33.38%	32.88%

Table 1: tf-iDF Retrieval Accuracies for different values of d and k

Method	Accuracy	Precision	Recall	F1 Score	FEVER Score
SVM	0.800	0.803	0.800	0.752	-
RF	0.720	0.668	0.720	0.686	-
RNN	0.660	0.435	0.660	0.524	-
BERT Baseline	0.640	0.253	0.296	0.273	-
BERT claim and top sentences	0.4285	0.183	0.428	0.257	-
BERT claim and evidence(test)	0.3975	0.3035	0.3975	0.3305	-
MLP (Small, T-I)	67.63%	66.50%	67.63%	66.89%	NA
MLP (Large, T-I)	83.20%	82.87%	83.20%	82.96%	NA
MLP (Small, T-II)	48.59%	49.13%	48.59%	48.75%	16.42%
MLP (Large, T-II)	50.77%	52.05%	50.77%	51.32%	17.45%
DrQa with MLP[1] (comparision)	-	-	-	-	19.42%
BERT & UKP Athene[6] (comparison)	-	-	-	-	69%

Table 2: Performance Metrics for Different Methods

7. Analysis and Discussion

7.1. Simple Models

The simple models of SVM, RF, and RNN display that even though no evidence was provided to the model, based on the claims alone the model was able to predict the label with high accuracy for all three models. This confirms our hypothesis regarding high correlation between certain words in a claim and the assigned label as seen with the word 'not' and the label 'REFUTES'. More thorough analysis should be able to figure out more such corelations within the dataset.

7.2. End-to-end tf-idf based Claim Verification with MLP

Below we thoroughly analyse the performance of the pipeline

7.2.1. High Document Retrieval Performance

As shown in Table 1 the Document Retrieval accuracy increases when we increase the number of top documents fetched. This happens because the evidence may have high similarity with a few words in an unrelated document and comparatively low similarity with important words in the relevant document. Increasing the number of top documents overcomes this problem and achieves a very high accuracy.

7.2.2. But why is Sentence Retrieval Performance decreasing...

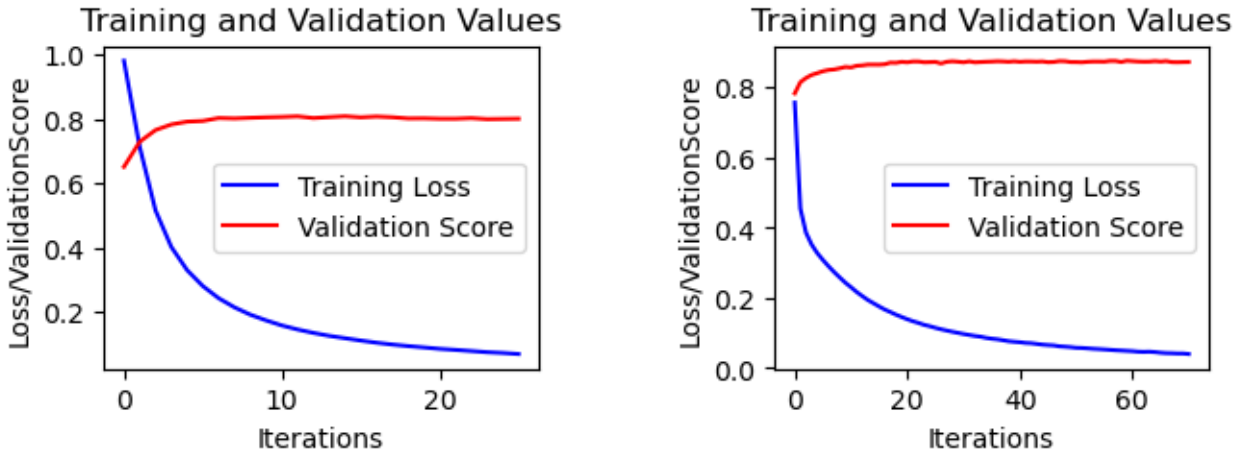
The Sentence Retrieval Performance unexpectedly decreases when we have more documents to retrieve sentences. This can be explained through the observation that by selecting more documents, we're providing our pipeline with more statements that have comparable similarity values thus leading to more irrelevant statements being selected as evidences.

This also implies that tf-iDF based similarity is perhaps not the best of metrics for the Sentence Retrieval task.

7.2.3. tf-iDF Similarity as Textual Entailment

tf-iDF based similairty values are perhaps, as seen in the Sentence Retrieval Task, not the best of metrics. However, we chose to stick with this pipeline as it provided a very good baseline for an end-to-end model. The RTE value is the same as similarity value in the previous task of the pipeline and hence was reused allowing for quick preparation of data.

7.2.4. MLP Training



(a) Small Dataset

(b) Large Dataset

Figure 6: The loss and Validation Score values of the small (a) and large (b) datasets.

The Loss and Validation score graphs for both the datasets were fairly stable and showed no signs of over or under fitting. The validation score of small dataset was 30% higher than than testing accuracy but for the large dataset the Validation score and accuracy for were very similar for Type-I dataset. The difference in accuracy and Validation scores for Type-II dataset were expected as a lot of negative examples existed in test as compared training.

7.2.5. Analysis on the test datasets

Below we analyse and detail all the experiments we performed with the 4 test datasets

- **High accuracy for True Evidences (Type - I Datasets)**: Type-I Datasets were designed to test solely the ability of the MLP classifier to correctly predict the labels provided the correct claim-evidence pair. And although the model achieves very high accuracy on this dataset, specially on large dataset, the fact can't be ignored that the underlying correlation between claims and labels might have played a huge role in the high accuracy.
- **Performance on the Type-II Dataset**: Type-II dataset is the real test of the end-to-end pipeline. Getting an accuracy of around 50% seems like a good baseline. However, this ignores the fact the only 36% of evidences were correctly retrieved by the pipeline. This means that the model is incorrectly predicting a wrong claim-evidence pair as either 'SUPPORTS' or 'REFUTES' instead of predicting 'NOT ENOUGH INFO'. Hence, we performed further tests to confirm our hypothesis.
- **How Robust is our RTE+Classifier model? Can it correct the errors of the upstream Pipeline?** The Type-II Dataset comes with two labels, a true label which tells the actual status of the claim, and a negative label which tells whether the given claim-evidence pair has been generated correctly by the upstream Pipeline.

  To test this we changed the labels of all the incorrect claim-evidence pairs to 'NOT ENOUGH INFO' and predicted the labels using the MLP model. We found that the model can only discard the incorrect claim-evidence pairs with an accuracy of 28% for the small dataset and an accuracy of 27% for the large dataset.

  A simple explanation could be that the number of 'NOT ENOUGH INFO' samples is only about 25% of the dataset hence there are not enough training samples for the model to correctly learn the relation between such pairs. Another reason is that the RTE and tf-iDF values are unable to correctly represent the irrelevance of the claim-evidence pair.
- **Recall on the 'REFUTES' samples** To test the ability of the model to refute an evidence we again tested the model on the Type-II dataset with only the 'REFUTES' samples. The first test was done on the true

claim-evidence pairs in Type-II dataset with 'REFUTES' label. This gives a Recall of 34.03% and 58.17% for small and large datasets respectively.

For all the pairs, correct and incorrect, these values decrease to 32.05% & 44.53% respectively.

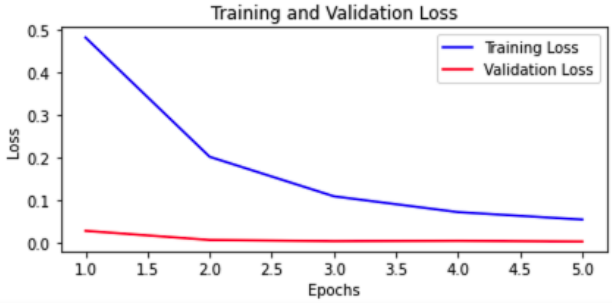
This shows that even though the overall performance of the model is high, most of it is coming from correctly predicting the 'SUPPORTS' label.

- **The FEVER Score** The FEVER score on the Type-II datasets is the most meaningful evaluation metric of the end-to-end model. The model achieves a FEVER score of 16.42% and 17.45% for the small and large datasets respectively.

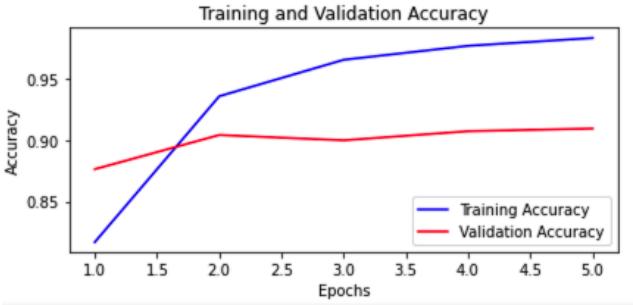
  The non-existent difference between the FEVER score of small and large datasets signifies that the reason for low performance is distributed throughout the model, and not just in the classifier.

  This shows that although this model is simple to implement and easy to interpret and works as a good baseline, it is not a good model overall for Evidence Retrieval and Claim Verification task.

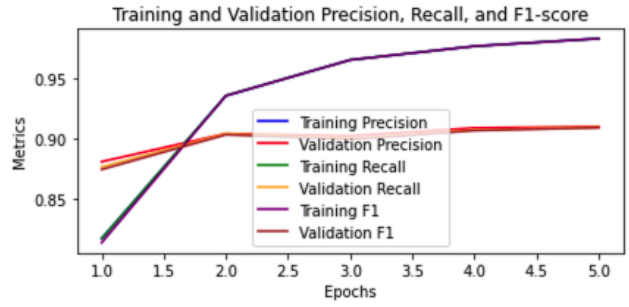
7.3. Claim Verification and Evidence Extraction using BERT



(a) Training Loss



(b) Accuracy



(c) F1-score,Precision, and Recall

7.3.1. Training and Validation Loss

In our FEVER training dataset comprising 32,179 rows, we employed 16,090 data points, accounting for 50% of the total dataset, chosen randomly to mitigate bias. Utilizing a learning rate of 2e-5, a batch size of 8, and executing 5 epochs, we applied the Adams optimizer to minimize training loss. During training, we observed a consistent decrease in the training loss over successive epochs, indicating effective optimization. Conversely, the validation loss remained consistently low throughout epochs 3 to 5, displaying minimal variation.

One potential reason for the low and stable validation loss could be the model's ability to generalize well to unseen data. The validation loss reflects the model's performance on a separate dataset not used for training, indicating its capability to make accurate predictions on new examples. The consistent and low validation loss suggests that the model has learned generalizable patterns from the training data, enabling it to perform consistently well on unseen data. Similarly, the decreasing training loss indicates that the model is effectively learning from the training data, resulting in improved performance over time.

7.4. Training and Validation Accuracy

In the graph 7b, we see that as training goes on, the model gets better at fitting the training data, which is shown by the rising training accuracy. But the validation accuracy also goes up, just not as much. This hints at over-fitting, where the model is learning too much from the training data and not enough from the overall patterns. So, while it's doing well on the training data, it might not perform as well on new data it hasn't seen before. To fix this, we can try techniques like dropout or stopping training early to help the model learn more general rules. This way, it can perform better on new, unseen data.

7.5. Precision, Recall and F1-score

In the graph 7c, we observe that as training continues, the training precision, recall, and F1-score gradually improve, indicating that the model is getting better at correctly identifying positive instances and minimizing false positives and false negatives on the training data.

However, similar to what we observed in the other graphs, the validation precision, recall, and F1-score also show improvement, but the rate of improvement is slower compared to the training metrics. This trend suggests over-fitting, where the model becomes too specialized in capturing the nuances of the training data, leading to limited generalization of unseen data.

In simpler terms, while the model becomes very good at fitting the training data perfectly, it struggles to generalize well to new, unseen data. To address this over-fitting issue, techniques like regularization or early stopping can be applied during training. By preventing the model from becoming overly complex and tailored to the training data, we can encourage it to learn more generalised patterns, ultimately leading to better performance on unseen data.

7.6. Test data metrics

The model's performance on unseen data is concerning. With an accuracy of only 39.75%, it frequently makes incorrect predictions. Additionally, its ability to identify true positives is limited, with a recall of 39.75%, while it tends to make many false positive errors, as indicated by its Recall of 30.35%. These results suggest potential issues with over-fitting or insufficient training data.

8. Conclusion

In this project we performed several experiments on the FEVER dataset using simple as well as end-to-end models. The dataset was challenging due to the huge amount of dataset involved as well as the application of several NLP techniques to Retrieve relevant evidences. We used Simple models to understand the dataset and figure out the correlations between the claims and labels and then used those understandings to critically analyse the two end-to-end models we developed.

- The Simple Models provide good idea of the nature of the dataset and help figure out probable issues within the dataset that help with the evaluation and analysis of the comples models
- The tf-iDF based MLP model provides a good baseline for the end-to-end model and helps understand the task much more clearly. Due to the simplicity and interpretability of the model it was very easy to perform repeated experiments and analysis and draw meaningful conclusions.
- The Bert model presents a comprehensive pipeline for claim and evidence retrieval. While leveraging the pre-trained Bert model facilitates the implementation of sentence retrieval, it demands significant computational resources due to its complexity. The incorporation of various factors into the BERT model initially posed challenges in understanding its implementation. Moreover, the model may exhibit overfitting tendencies, resulting in suboptimal performance. However, with access to robust computational resources, there is potential for refinement and enhancement of the model's effectiveness

At the start of the project our aim was to successfully implement an end-to-end model which would correctly retrieve relevant evidences and perform Claim verification. We successfully achieved that objective by implementing two end-to-models: one a tf-iDF based MLP and another BERT model. Although the accuracy of the tf-iDF model wasn't high it provided a good baseline and a deep understanding of the dataset and possible pitfalls and shortcomings in the data as well as in our theoretical approach. BERT model proved very challenging and provided a first hand experience of the difficulty and the learning curve involved with working on LLMs with large datasets.

9. Future Work

- The MLP model can be significantly improved by choosing a better Sentence Retrieval and RTE model. Another thing that can be improved upon is synthetically increasing the training data points of 'NOT ENOUGH INFO' label. The most difficult class to train remains 'REFUTES' as the data points for this class can't be generated synthetically.
- The presence of high correlation between claims and labels should be removed for a more meaningful dataset.
- As shown by ProoFVer[41], the best performing model on the FEVER dataset, rather than relying solely on LLMs and Data Retrieval solutions, Natural Logic and Linguistics may provide a fairly robust solution to the Fact Verification problem. This is a paradigm that should be explored further for improving fact verification models.

References

[1] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*, 2018.

[2] Naeemul Hassan, Bill Adair, James T Hamilton, Chengkai Li, Mark Tremayne, Jun Yang, and Cong Yu. The quest to automate fact-checking. In *Proceedings of the 2015 computation+ journalism symposium*. Citeseer, 2015.

[3] N. Moreira A. P. Tomás M. Filgueiras, L. Damas. Natural language processing. Citeseer, 1990.

[4] *Precision*, pages 1095–1095. Springer International Publishing, Cham, 2021.

[5] Sajjad Ahmed, Knut Hinkelmann, and Flavio Corradini. *Fact Checking: An Automatic End to End Fact Checking System*, pages 345–366. Springer International Publishing, Cham, 2022.

[6] Amir Soleimani, Christof Monz, and Marcel Worring. Bert for evidence retrieval and claim verification. In *Advances in Information Retrieval: 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14–17, 2020, Proceedings, Part II 42*, pages 359–366. Springer, 2020.

[7] Parth Patwa, Shivam Sharma, Srinivas Pykl, Vineeth Guptha, Gitanjali Kumari, Md Shad Akhtar, Asif Ekbal, Amitava Das, and Tanmoy Chakraborty. Fighting an infodemic: Covid-19 fake news dataset. In *Combating Online Hostile Posts in Regional Languages during Emergency Situation: First International Workshop, CONSTRAINT 2021, Collocated with AAAI 2021, Virtual Event, February 8, 2021, Revised Selected Papers 1*, pages 21–29. Springer, 2021.

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

[9] James Thorne and Andreas Vlachos. Automated fact checking: Task formulations, methods and future directions. *arXiv preprint arXiv:1806.07687*, 2018.

[10] Naeemul Hassan, Fatma Arslan, Chengkai Li, and Mark Tremayne. Toward automated fact-checking: Detecting check-worthy factual claims by claimbuster. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1803–1812, 2017.

[11] 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.

[12] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

[13] Mian Guo, Yong-Min Lee, Ranjana Gupta, Mi Sook Seo, Takehiro Ohta, Hua-Hua Wang, Hai-Yang Liu, Sunder N Dhuri, Ritimukta Sarangi, Shunichi Fukuzumi, et al. Dioxygen activation and o–o bond formation reactions by manganese corroles. *Journal of the American Chemical Society*, 139(44):15858–15867, 2017.

[14] Andreas Vlachos and Sebastian Riedel. Fact checking: Task definition and dataset construction. In *Proceedings of the ACL 2014 workshop on language technologies and computational social science*, pages 18–22, 2014.

[15] Reda Alhajj and Jon Rokne, editors. *Wikipedia*, pages 2406–2406. Springer New York, New York, NY, 2014.

[16] Claude Sammut and Geoffrey I. Webb, editors. *Random Forests*, pages 1054–1054. Springer US, Boston, MA, 2017.

[17] Mathias M. Adankon and Mohamed Cheriet. *Support Vector Machine*, pages 1303–1308. Springer US, Boston, MA, 2009.

[18] Claude Sammut and Geoffrey I. Webb, editors. *TF–IDF*, pages 1274–1274. Springer US, Boston, MA, 2017.

[19] Zhong-Yuan Zhang. *Recurrent Neural Network*, pages 1824–1825. Springer New York, New York, NY, 2013.

[20] Gregory Grefenstette. *Tokenization*, pages 117–133. Springer Netherlands, Dordrecht, 1999.

[21] Mahidhar Dwarampudi and NV Reddy. Effects of padding on lstms and cnns. *arXiv preprint arXiv:1903.07288*, 2019.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[23] Stan Z. Li and Anil Jain, editors. *Tensor*, pages 1328–1328. Springer US, Boston, MA, 2009.

[24] Sagar Imambi, Kolla Bhanu Prakash, and G. R. Kanagachidambaresan. *PyTorch*, pages 87–104. Springer International Publishing, Cham, 2021.

[25] Manfred Schwab, editor. *Classifier*, pages 879–879. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[26] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[27] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[28] Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis, and Sebastian Riedel. A simple but tough-to-beat baseline for the fake news challenge stance detection task, 2018.

[29] Amir Soleimani, Christof Monz, and Marcel Worring. Bert for evidence retrieval and claim verification. *arXiv preprint arXiv:1910.02655*, 2019.

[30] Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. Ukp-athene: Multi-sentence textual entailment for claim verification. *arXiv preprint arXiv:1809.01479*, 2018.

[31] Silviu Cucerzan. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 708–716, 2007.

[32] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*, 2018.

[33] Zhuolin Jiang, Amro El-Jaroudi, William Hartmann, Damianos Karakos, and Lingjun Zhao. Cross-lingual information retrieval with bert. *arXiv preprint arXiv:2004.13005*, 2020.

[34] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[35] David MW Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*, 2020.

[36] Leon Derczynski. Complementarity, f-score, and nlp evaluation. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 261–266, 2016.

[37] *Accuracy*, pages 19–19. Springer Netherlands, Dordrecht, 2005.

[38] Luca Di Liello, Siddhant Garg, Luca Soldaini, and Alessandro Moschitti. Paragraph-based transformer pre-training for multi-sentence inference. *arXiv preprint arXiv:2205.01228*, 2022.

[39] Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. Fine-grained fact verification with kernel graph attention network. *arXiv preprint arXiv:1910.09796*, 2019.

[40] Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. Reasoning over semantic-level graph for fact checking. *arXiv preprint arXiv:1909.03745*, 2019.

[41] Amrith Krishna, Sebastian Riedel, and Andreas Vlachos. Proofver: Natural logic theorem proving for fact verification, 2022.