Claim Detection: Detecting whether the tweet is a claim or not

Himanshi Sethi (himanshi20510@iiitd.ac.in), Rupin Oberoi(rupin20571@iiitd.ac.in) ,
Aniket Goel (aniket20281@iiitd.ac.in)

Abstract

Social Media and online newsgroups have become comfortable places for people to write their views and consume a lot of content online. They can post without any filtering and even anonymously if they want. The downside of this comfort is that people make false claims and use false identities as well. This behaviour is not limited to Social Media, but if you see the following telltale signs of falsities, you are probably reading a piece of misinformation or a hoax. It becomes very important to filter any content/comment posted online and detect the presence of claims, if any. The task of claim detection is helpful in various fields like Fraud Detection, Complaint analysis or Claim validation.

This kind of analysis may track the progression, frequency and reputation of a claim. It is very common to identify false or misleading trends by looking at their temporal or geographic distribution. The results obtained from this analysis are often called "Claim Streams". These streams will be influenced by many factors, such as content popularity and user participation.

1 Problem Statement

The task is to determine whether a tweet is a claim or not given an input tweet, using NLP and associated language models. It is a binary classification task. A claim is an assertive statement that may or may not be proven. The term 'claim' here is subjective and requires careful consideration. A claim is described as "an assertion that deserves our attention" by Toulmin(2003) in his argumentation theory. Given the variance in conceptualization and the absence of a clear definition of a claim, the claim detection task is in a difficult situation. Since Twitter is a significant Networking platform, it serves as the ideal playground for many viewpoints and ideologies.

Twitter is the place to test and build the model, and it is helpful in detecting detect claims from people's viewpoints. Humans, sometimes unknowingly, tend to make baseless claims in tweets/conversations. Choosing twitter for data has its disadvantages as well. The data coming from Twitter is human-generated and really noisy, which makes our task more challenging.

2 Related Work

2.1

LESA: Linguistic Encapsulation and Semantic Amalgamation Based Generalised Claim Detection from Online Content[1]. The research aims at Claim Detection from online posts. The paper proposes a deep neural framework LESA. The major difference between the rest research and this paper is that most works in this domain don't work on unstructured data, but LESA claims to be capable of accounting for different text distributions. The broad methodology used here is: using a pre-trained language model and accounting for two linguistic properties, namely POS and dependency tree for the classification. The paper handles text effectively by categorising the models separately as noisy, non-noisy, and semi-noisy based on their sources. The attention module is used for using these two properties together.

2.2

Check square at CheckThat! 2020: Claim Detection in Social Media via Fusion of Transformer and Syntactic Features [2]. The research aims to detect fake news claims in online comments. The task is divided into two steps, firstly detecting the claim and secondly checking validity. The first task is achieved by categorising a tweet's check-worthiness; they investigate the integration of syntactic characteristics with deep transformer BERT embeddings. As syntactic , they employ POS tags, named entities, and dependency

relations in BERT to compute twitter embedding.

2.3

Claim Detection in Biomedical Twitter Posts [3]: The research aims at using Biomedical natural language processing (BioNLP) to detect the presence of claims in tweets if any. The data is extracted from TwitterAPI which uses tweets from the year 2020. Unlike other researches, classical ML models were also tried, for example Naive Bayes and Logistic Regression were both applied with an average vector of the token embeddings as input. Furthermore, core NLP models were also used, such as bi-directional LSTM and BERT. The research concluded that the comparably small amount of the dataset is the reason why the more sophisticated models (LSTM, BERT) do not consistently beat the linear models.

3 Methodology

3.1 Data Preprocessing

The data used is tweets from real people. People often tend to make mistakes both knowingly and unknowingly. We all know how easily our data can become tainted when we don't take care of it or make mistakes unknowingly while inputting it. There are many ways to clean the data and make it noise free. Another reason we want to pre-process the raw data would be that we don't want our model to train on unnecessary words which won't affect our classification and make our code slower. The training dataset given to us has tweets and a label coresponding to them 1 or 0, where 1 represents claim and 0 represents non- claim.

As the dataset consists of noisy data and tweets contain text which does not provide any relevant information, we have done preprocessing on the data by removing usernames, URLs and hashtags. We also removed stopwords, extraspace in the preprocessinng of training data. The text was converted to lower case and punctuations were removed. The text data was stemmed and lemmatized as well.

3.2 Dataset Sampling

The dataset comprises Of 6986 tweets in total, out of which 6105 are labelled 1 (claim) and rest are are non-claim. The skewed data gave us an idea of oversampling/undersampling.

From the class distribution of the dataset, we observe that it is very skewed, and the ratio of claims to non-claims is nearly 7:1, which caused our models to predict nearly all test samples as claims. To tackle this, we tried undersampling and oversampling. In undersampling, we considered a part of samples belonging to the claim class, which were randomly selected, the best results were obtained by using 2000 samples with around 1100 claims. The validation set is prepared as one-fifth of the training dataset Data augmentation or oversampling involves synthetically generating samples of the class with less number of samples. We created such samples for the non-claim class using the 'nlpaug' library, by replacing maximum of 3 words from the tweets with label 0 with their synonyms. We generate 5 similar sentences for each sentence in the training data of minority class.

3.3 Models

We have tried two vastly different methodologies: One is using pre-trained sentence transformers along with extracting some features from the given tweets, and the other is finetuning various BERT-based models to get better results on the given downstream task. The former involved extracting certain features from sentences such as part-of-speech tags, dependency relationships and used named entity recognition. We compute POS encodings using the nltk library for postag. We create the encodings as the number of words for the respective postag in the tweet.
For Named entity recognition (NER), we used spacy to import the displacy. We generated generated ner encoding with the help of it.
In the dependency parser, we examine a sentence's grammatical structure to identify related words and the nature of their relationships. A dependency Parser is used to analyse the dependency structures of sentences. The encoding generated was a bucketed array with each index representing a type of relationship that can exist, and the value of the array at that index tells us the number of words with that particular relation.

These were then encoded into arrays of size equal to the number of distinct POS tags/ dependency relations with each element containing the number of occurrences of that particular relation in the sentence. We then used pre-trained sentence transformers embeddings for all sentences that were computed. We used the Hugging Face

'all-MiniLM-L6-v2' BERT model as it gives the best combination for performance and speed. We then trained various traditional machine learning classification models such as SVM, random forest and logistic regression.

3.4 Fine tuning

We start by pre-training the modeland then fine-tune it using transfer learning. We work with the Hugging face library "transformers". We import the AutoTokenizer from pre-trained "distilbert-base-cased" model, and use it to tokenize our training data. After this we load the model and perform sequence classification on it. We train the model with 3 epochs and save it. Further, using this model we make our predictions the test data. As an alternate to "distilbert-base-cased" model, we use the "all-mpnet-base-v2" model and observe better results.

4 Experimental Results

We evaluate the F1 score (macro) on the model used. Using the postag, NER and dependency parser encoding and getting embedding from the unsupervised learning algorithm Glove. We trained the model on ML classification models SVM, random forest on the non-augmented data. This gave 0.466 F1 score. Applying 'all-MiniLM-L6-v2' BERT model from the hugging face library instead of Glove, we get the same score. Since the training data given to us is skewed with 1:7 non-claim-to-claim ratio, we apply undersampling and oversampling or data augmentation. After undersampling the data to total number of rows 2000, with around 1100 claims, we get the F1 score of 0.613. After augmenting the data from 6987 rows to 11392 rows with 5286 0 labels and 6105 1- label, we get an improved F1 score of 0.636.

After this we fine-tuned different models, first we tried 'distilbert-base-cased' model and got F1 score of 0.681. On fine-tuning the "all-mpnet-base-v2" model, we get F1 score of 0.694. However, training the same model using original training dataset gave score of 0.51. This is our best result so far. On the validation set, we obtain an F1-score of 0.93 using the original training dataset on the 'all-MiniLM-L6-v2' BERT model. Using the augmented training dataset and 'all-MiniLM-L6-v2' BERT model we obtain F1-score of 0.77. Training on the undersampled data using the same model, we obtain the F1-score on the validation set is 0.76. On the fine-tuned 'distilbert-base-cased model trained using augmented data, the validation set score is 0.721. On the fine-tuned 'all-mpnet-base-v2' model trained using augmented data, the validation set score is 0.73. On the fine-tuned 'dall-mpnet-base-v2' model trained using original data, the validation set score is 0.83. Using glove and logistic regression using provided original training dataset score achieved on the validation set is 0.81.

5 Analysis

From the Data undersampling and oversampling, we analysed that oversampling gave better results as compared to under-sampling. This could be because undersampling deletes the data of the majority class, and we could lose potentially important information. Significantly better results on the validation set than test set when using the original dataset could be explained because of similar class distribution as train set in validation set.

The results drastically improved even after undersampling the training data as it was skewed. When the data were skewed, improving the models did not lead to better results. Even with fine-tuning, we could obtain the best score of 0.51 on test data using the original training database provided, however, training on the augmented data, we obtained a 0.694 score on the test dataset. Fine-tuning the data also improved the score by a significant number. We observe that fine-tuning gave the best results, and the " all-mpnet-base-v2" model gave a better result than 'distilbert-base-cased' on the same training set and the number of epochs.

6 Individual Contribution

Data Augmentation and Fine tuning the 'distilbert-base-cased' and 'all-mpnet-base-v2' using augmented and original training dataset was done by Himanshi. Under-sampling and training the data using BERT models with logistic regression and Random Forest was done by Rupin. The encodings using postag were generated by Himanshi. The encodings for NER were generated by Rupin. The encodings for the dependency parser were generated by Aniket. In report, Models, Experimental Results, and Analysis were written by Rupin and Himanshi equally. The abstract, problem statement and related work was written by Aniket.

Method and Training Dataset	F1 score (Test Set)	F1 score (Validation Set)
Fine tune 'all-mpnet-base-v2' (Aug. Dataset)	0.694	0.73
Fine tune 'distilbert-base-cased' (Aug. Dataset)	0.681	0.721
'all-MiniLM-L6-v2'+ Random Forest (Aug. Dataset)	0.636	0.773
'all-MiniLM-L6-v2'+ Random Forest (Undersampled Data)	0.613	0.761
Fine-tune 'all-mpnet-base-v2' (Original Dataset)	0.517	0.83
Glove + Logistic Regression (Original Dataset)	0.466	0.81
BERT 'all-MiniLM-L6-v2'+ Random Forest (Org. Dataset)	0.466	0.931

Table 1: F1-score obtained on test set and Validation set using different pre-trained models and sampling the training dataset.

7 References

[1] Gupta, S., Singh, P., Sundriyal, M., Akhtar, M.S. and Chakraborty, T., 2021. Lesa: Linguistic encapsulation and semantic amalgamation based generalised claim detection from online content. arXiv preprint arXiv:2101.11891.

[2] Cheema, G.S., Hakimov, S. and Ewerth, R., 2020. Checksquare at checkthat! 2020: Claim detection in social media via fusion of transformer and syntactic features. arXiv preprint arXiv:2007.10534.

[3] Wührl, A. and Klinger, R., 2021. Claim detection in biomedical Twitter posts. arXiv preprint arXiv:2104.11639.