LSTM-based Language Models for Downstream Fact Checking

COMP90020 Natural Language Processing

Jonathan Lätti 108337
Zihang Xu 1446938
Flynn Schneider 982143
Abbey He 949219

Abstract

The pervasive nature of misinformation poses significant real-world consequences. This project aims to tackle this challenge in the context of dubious climate science claims, by developing an NLP pipeline for fact-checking. Our implementation consists of a bi-directional LSTM language model, using cosine similarity for evidence retrieval; an FFNN model is subsequently used for claim classification. We found that although the language model adeptly handles token prediction, it struggles to produce semantically rich sentence embeddings, which limits performance in downstream tasks.

1 Introduction

The spread of unverified claims, or *misinformation*, on social media platforms and the internet more widely, has played a critical role in shaping information on key events across recent decades. This is particularly relevant for the topic of climate change, where a significant proportion of the public hold views contrary to scientific consensus (Mooney, 2011; Sarathchandra and Haltinner, 2020). Findings from a 2017 MIT study suggest false claims are 70% more likely to be retweeted than truthful ones (Vosoughi et al., 2018), underscoring the necessity for automated fact-checking systems that can effectively counteract misinformation online.

Recent literature on such claim-verification systems has focused on the use of fine-tuned language models for downstream classification tasks, including the popular BERT model (Devlin et al., 2019) and other pre-trained transformer models. In this paper, we propose a pipeline of NLP-based models for this task, by first sourcing evidence from a general information corpus, and then verifying specific climate-change related claims using this retrieved evidence.

The dataset for this project includes labelled training and development sets, and an unlabelled test set, all of which contain claims to be fact-checked against a large evidence corpus. Each claim in the labelled sets has an associated label (SUPPORTS, REFUTES, NOT_ENOUGH_INFO, DISPUTED) and a list of linked evidences. It is worth noting that the evidence corpus is significantly larger than the labelled sets, containing over 1.2 million passages, as opposed to the mere 1-2k labelled train pairs.

2 Approach

2.1 Baselines

We used a TF-IDF model trained on the entire set of evidence sentences (as *documents*) as a baseline model for the evidence retrieval task (Pedregosa et al., 2011). As opposed to our other approaches, this baseline directly performs evidence retrieval without the use of a pre-trained language model. Tf-IDF was selected particularly because it is effective in identifying important tokens that are less common across a broader corpus.

For claim classification, majority class prediction (ie: naively predicting "SUPPORT" for every claim) was used as a baseline, given that the 'SUPPORT' class has around 42% representation in the development set.

2.2 Pre-processing

To prepare texts from the *evidence* and *train-claims* datasets for our language model, we first convert all sentences to lowercase and then tokenize using the "basic_english" tokenizer from the TorchText library. We retain all words, including stopwords, to preserve word order information which is informative for tasks such as next word prediction. These pre-processing steps ensure that the data is clean and consistent, which facilitates efficient numerical representations whilst still retaining contextual nuances.

2.3 Language Model

Our objective in training a language model in this context is to derive dense vector representations of sentences that encapsulate both semantic and syntactic relationships. The aim is for sentences that are related - in both meaning and structure - to cluster together when represented in high-dimensional space. As (Cer et al., 2018) demonstrate, high quality sentence em-

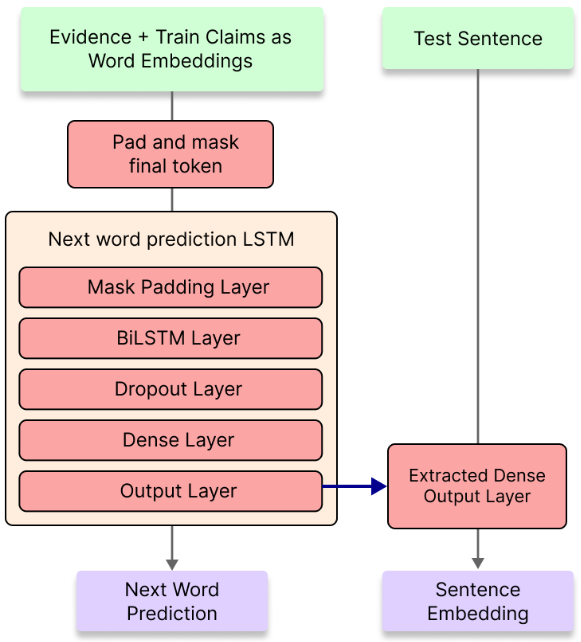

Figure 1: Next Word Prediction Model Architecture

beddings transfer particularly well to downstream classification tasks.

2.3.1 Word Embeddings

A precondition for our language model is to first generate individual word embeddings in order to represent sentences as 2D arrays. We used the Continuous Bag of Words-based Word2Vec model from Gensim (Rehurek and Sojka, 2011), training on the entire corpus of tokens identified in *evidence* sentences. To ensure dimensional consistency, shorter sentences are padded to a fixed length, and conversely, longer sentences are truncated.

2.3.2 Sentence Embeddings

The function of the language model is to retrieve sentence embeddings which capture semantic information beyond the basic word-level embedding vectors offered by Word2Vec. To achieve this, we used a Bidirectional LSTM model (Chollet et al., 2015; Hochreiter and Schmidhuber, 1997), trialling two language tasks for training - next word prediction, and an autoencoder model.

As shown in Figure 1, the next word prediction model masks the final token of each sentence, providing this as a train label, with loss calculated as the distance between the predicted vector and the vector of the masked label word. The second last layer of the model is extracted as a standalone model for creating sentence embeddings from unseen sentences.

In the autoencoder model, the goal of the model is to reconstruct the original sentence in word embedding form. We used three dense layers of varying sizes (25, 50, 100) in order to provide the complexity required to reproduce the input sentence. The final dense layer was subsequently extracted as a standalone model for unseen prediction. In some ways, this autoencoder is most similar to a compression algorithm, converting the larger 2-dimensional input embedding into a 1-dimensional output embedding whilst *theoretically* maintaining semantic information.

2.4 Filtering

The *evidence* dataset contains over 1.2 million evidence sentences (174.2 MB) and is overwhelmingly comprised of contextually irrelevant texts. It's volume makes it impractical to perform tasks with linear time complexity, particularly the *K most relevant* evidence retrieval sub-task which would require pairwise comparison with all evidences for each claim.

To reduce the size of the dataset, we used a Logistic Regression classifier with approximated (*mean*) sentence embeddings engineered for each evidence text to serve as features. These embeddings were calculated as the average of the individual word embeddings for all tokens within a sentence. In training, each would also be labeled with the class value **1** if linked to a training claim and **0** otherwise. By retaining only the sentences predicted to belong to class 1, this classifier effectively filtered out over 90% of the total evidence set, while maintaining approximately 92% recall.

2.5 Evidence Retrieval

As a base strategy, we ran all filtered evidence through our pre-trained language model to retrieve comparable embeddings, selecting the evidence which maximises cosine similarity with an unseen claim's embedding (from the same language model).We used Annoy Index (ann) to make this more efficient, where *N* approximate nearest neighbours are first found, before running cosine similarity on only a reduced number of claim/evidence pairs.

As an extension to simple cosine similarity, we also implemented a subsequent Siamese neural network, which makes use of the labelled claim/evidence pair data. To train this model, pairs were created by marking all the known tuples of claims and evidences with the label '1' - representing a valid pair - as well as creating an equal number of tuples with randomly selected evidences and labelling these with '0' - representing a mismatched pair.

Both evidence and claims are passed through the same sets of weights, with binary cross entropy loss calculated using the labels and both output vectors. The intuition behind this approach is that the model should learn to transform the two vectors towards a

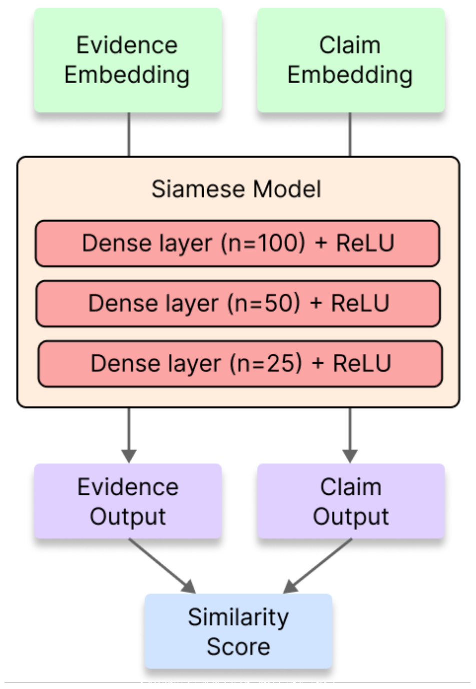

Figure 2: Siamese Model Architecture

similar output vector, which would result in a high similarity score at test time.

2.6 Claim Classification

Our approach to claim classification was to predict a class label for each of the **k** evidences retrieved in relation to the claim, before performing an ensemble voting strategy on these results.

As seen in Figure 3 we used a fully connected feed forward network (Paszke et al., 2019) to learn not only the label for each claim group but also the underlying hidden labels of each one-to-one evidence-claim concatenated vector (abstracted away within a hidden layer), and consequently the voting strategy that connects these layers.

This entire process trained all at once, with back-propagation through the final model as well as each *claim label prediction unit*. Intuitively, such a configuration would eventually allow the label prediction units to predict labels for individual evidence/claim pairs, despite no labelled data existing for this component.

Given that the model requires *exactly 5* evidence embeddings as input, we padded smaller evidence sets with an arbitrary value of *–99999999*. Additional dropout layers are included to prevent overfitting, and the final output is a tensor representing the class scores for all four classes - "SUPPORTS", "REFUTES", "DISPUTED" and "NOT_ENOUGH_INFO".

The network is trained with an Adam optimiser according to cross entropy loss of the final layers output (after applying a softmax operation internally). In order to handle class imbalances, class weights are

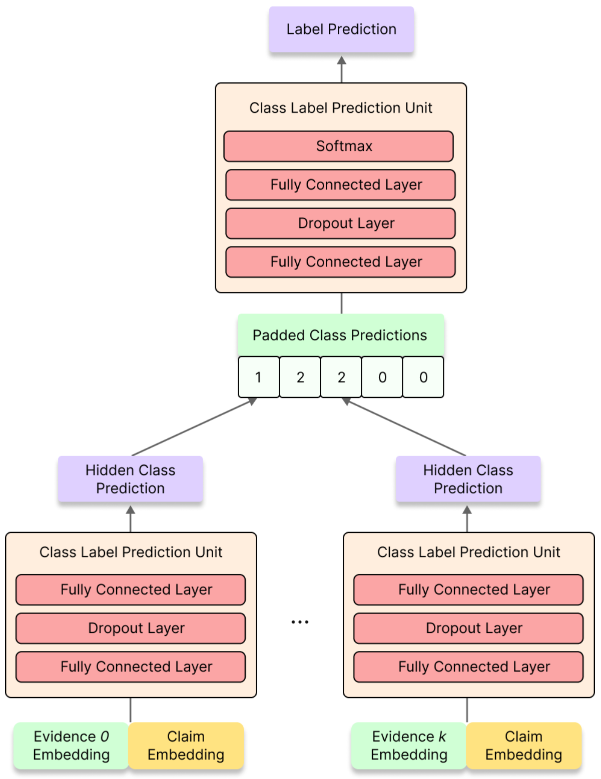

Figure 3: Claim Classification

calculated according to the inverse training set class proportions. This is necessary to avoid the model from naively choosing the majority class as the label for all instances.

3 Experiments

3.1 Evaluation Methods

3.1.1 Evidence Retrieval

2The primary quantitative metrics used are F-Score, Recall, and Precision, given that the primary goal is predicting a set of evidence sentences; intuitively, a good model should care about minimising both false negatives and false positives.

Qualitative inspection was also used in understanding what kind of semantic information was being learnt by the language model, by analysing claim and predicted evidence pairs.

3.1.2 Claim Classification

The performance of our feed forward classifier was evaluated using multiple metrics, including accuracy, precision, recall, and F1 score, to provide a comprehensive assessment of the model's effectiveness. These metrics were calculated for both the training and validation datasets over a series of epochs (10 in our experiment).

Hyperparameter	**Value**
Word Embedding Vector Size	100
N Nearest Neighbours (Annoy)	10
Dropout Rate	0.5
BiLSTM/LSTM Units	200
BiLSTM/LSTM Output Vector Size	(1, 100)
k Evidences to Predict	3
Classification learning rate	0.01

Table 1: Hyperparameter Settings

3.2 Experimental Details

The hyper-parameters tuned during experimentation included sentence length, model type (next word vs autoencoder), layer specifications, and class weights. Given the computational resources required to fully train the BiLSTM model, we varied each hyper-parameter independently to understand its effects on model performance. Optimised fixed parameters were set as seen in Table 1.

4 Results

4.1 Evidence Retrieval

Model	**F-Score**	**Precision**	**Recall**
LSTM	0.0119	0.0108	0.0152
BiLSTM	0.0366	0.0444	0.0364
BiLSTM + Dense Layer	0.0423	0.0519	0.0416
BiLSTM (Sen_len = 80)	0.0113	0.0091	0.0183
BiLSTM (Autoencoder)	0.0019	0.0022	0.0016
BiLSTM (Siamese Classifier)	0.0000	0.0000	0.0000
TF-IDF Baseline	**0.0767**	**0.0500**	**0.1900**

Table 2: Claim Classification Results (Dev Set)

As seen in Table 2, the best performing non-baseline model was the BiLSTM model with an added dense layer before the final output layer. We typically observed in our experiments that adding more layers to the model improved performance, albeit somewhat marginally. This included dropout layers, further dense layers, and the Bidirectional approach to LSTM.

In general, these LSTM-style models perform somewhat worse than a TF-IDF baseline, largely as a result of poor sentence embeddings. The underwhelming quality of the language model component can be understood as the result of a number of likely factors:

1. **Non-Semantic Hidden Layer:** Theoretically, extracting the penultimate hidden layer from the BiLSTM model should provide a semantic embedding representing the *meaning* expressed by a given sentence. In practice, we have found that these embeddings are capturing mostly lexical information; as seen in Figure 4 and in a number of similar instances, the prediction matches a specific topic, *energy*, as well as the fact that a *country* is mentioned, but is unable to understand the actual semantics of the sentence. This suggests that the knowledge learnt by the hidden layer in order to predict tokens - or reconstruct the input in the case of the autoencoder - does not transfer well as a semantic representation of the entire sentence.

2. **Model Complexity:** In training the best BiLSTM model, the *mean squared error* loss flattens out after 10 epochs, with a value of 1.3702. This indicates that at some point the model is unable to take in more information, limiting the semantic information capturable in sentence embeddings.

3. **Data Limitations:** Models such as BERT train on both *masked token prediction* and *next sentence prediction*, which allows for deep learning conceptual representations of sentences. Given that the data used in this project was in the form of individual sentences as opposed to documents, only token-level prediction was possible, which has likely contributed to lexical dominance in our hidden embeddings.

"South Australia has the most expensive electricity in the world."

(a) Claim

"The United States is the second-largest single consumer of energy in the world."

(b) Top predicted evidence

Figure 4: Claim and predicted evidence using BiLSTM with cosine similarity

4.1.1 Siamese Model

The Siamese model that extends the base BiLSTM model underperformed simple cosine similarity, typically predicting no correct evidences for a given claim. Beyond the inherently low quality of the sentence embeddings, one key cause of this is overfitting; as seen in Figure 5, the validation loss largely stays high whilst train loss improves. This could be due to the fairly small train dataset used for training the supervised model, as well as the inherent challenge of learning with unclean data and a poorly optimised embeddings model.

4.2 Claim Classification

Since the sentence embeddings produced by the BiLSTM language model were sub-optimal, the classifi-

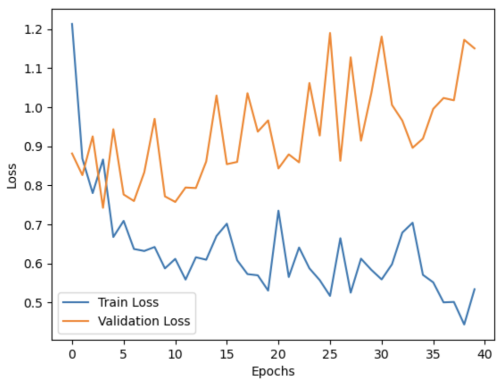

Figure 5: Siamese Model Overfitting Analysis

Model	F-Score	Precision	Recall	Accuracy
FF Retrieved Evi.	0.1950	0.1672	0.2348	0.3506
FF True Evi.	0.3969	0.3442	0.4706	0.6558
Baseline	0.1532	0.1104	0.2500	0.4416

Table 3: Claim Classification Results (Dev Set)

cation model's performance was evaluated not only with evidences retrieved from the BiLSTM and Cosine similarity, but also on ground truth evidence from the development set. As seen in Table 3, the classification model performs significantly better when classifying claims using ground truth evidence as opposed to retrieved evidence.

The Feed Forward classifier comfortably surpasses the majority class baseline for both accuracy and F1 score, sacrificing neither recall or precision in its approach. These results suggest that our model is capturing some essential aspects necessary for making informed label predictions. However, it remains unclear whether these outcomes truly reflect the model's nuanced understanding of the evidence-claim relationships according to the sentence embeddings features provided, or if it conversely reflects a moderately sophisticated understanding of the underlying majority voting strategy and/or how the number of input evidences a claim is classified with affects that strategy.

As seen in Figure 6, the training graphs reveal a significant disparity between precision and recall. Precision fluctuates notably, indicating that the classifier is prone to false positives, while recall steadily increases, suggesting improved true positive identification. This pattern is typical in imbalanced datasets, where the model learns dominant classes better but struggles with minority classes. To address this in future research, techniques such as class re-balancing, advanced loss functions such as Focal Loss, and regularization could be implemented to enhance the classifier's balance and overall performance.

We also noticed the classifier tend to pre-

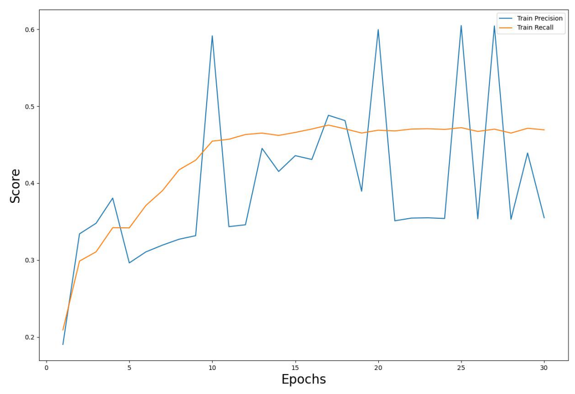

Figure 6: Precision and Recall During Training of the Claim Classification FFNN

dict the majority classes of "SUPPORTS" and "NOT_ENOUGH_INFO" for almost all inputs. As a test, we manually set the label array to all 0s and 1s respectively; note that we do not necessarily know what '0' and '1' represent in the hidden state of the class label unit, as we are allowing the model to learn these classes independently. However, as seen in Table 4, both input arrays predict *Not Enough Information* as the final class label, despite having strictly different inputs. This indicates that the model has not effectively learnt the underlying voting strategy correctly, potentially due to the limited training data available and class imbalance.

Class Label	Score(0s)	Score(1s)
SUPPORTS	-0.1482	-0.9440
REFUTES	-0.8033	-3.5171
DISPUTED	-0.8138	-3.3578
NOT_ENOUGH_INFO	1.2808	4.1092

Table 4: Sample Voting Results for Classification

5 Conclusion

Ultimately, this research indicates that whilst LSTM and BiLSTM models can be effective in predicting next tokens, they transfer poorly as sentence embedding generators for downstream evidence retrieval and claim verification. Data quality and complexity, including the presence of consecutive sentence documents, is also a major influence in final performance. In terms of future directions, state-of-the-art models such as transformer-based approaches are certainly worth exploring for fact-checking applications, especially fine-tuning BERT-like models.

6 Team Contributions

- **Abbey He:** Worked on evidence retrieval
- **Jonathan Latti:** Worked on the BiLSTM language model for evidence retrieval
- **Flynn Schneider:** Worked on pre-processing pipeline, LSTM Model, filtering methods, code refactoring
- **Zihang Xu:** Worked on claim classification

7 References

References

ANNOY library. https://github.com/spotify/annoy. Accessed: 2017-08-01.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder. *CoRR*, abs/1803.11175.

Francois Chollet et al. 2015. Keras.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*.

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

Chris Mooney. 2011. The science of why we don't believe science. *Mother Jones*, 11.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Radim Rehurek and Petr Sojka. 2011. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2).

Dilshani Sarathchandra and Kristin Haltinner. 2020. Trust/distrust judgments and perceptions of climate science: A research note on skeptics' rationalizations. *Public Understanding of Science*, 29(1):53–60. PMID: 31691642.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science*, 359:1146–1151.