COMP90042 Report

Tue_10AM_Group3

Abstract

In this project, we developed an automated fact-checking system to address the challenges of manual fact-checking. Our system includes two main components: evidence retrieval and claim classification. For evidence retrieval, we used a Siamese network architecture with a listwise loss function and leveraged TF-IDF, Word2Vec, and BM25 for initial retrieval and negative sample generation. For claim classification, we employed GRU and Transformer models, with GRUs enhanced by an attention mechanism performing best. Despite not using pretrained models, our system demonstrated effective performance in handling long sequences and focusing on relevant text, providing valuable insights into automated fact-checking capabilities and limitations.

1 Introduction

Automated fact-checking is crucial in today's media ecosystem due to the rapid spread of information and misinformation, especially for complex climate-related claims (Guo et al., 2022). Manual fact-checking cannot keep up with the volume of new information, prompting interest in automated systems. These systems have two main components: evidence retrieval, which finds relevant information to support or refute claims, and claim verification, which assesses the truthfulness of claims based on the retrieved evidence (Thorne et al., 2018).

One major challenge in evidence retrieval is filtering relevant evidence from large databases. While traditional methods such as BM25 and TF-IDF vectors are effective for initial retrieval, they often lack precision required for verifying claims (Robertson and Zaragoza, 2009). To improve the retrieval accuracy, we adopted the recent research using dense retrievers with learned representations and dot-product similarities which showed competitive result (Karpukhin et al., 2020). Dense retrieval is promising because it captures deeper semantic relationships between texts through continuous vector space representations, allowing for more nuanced matching beyond keyword overlap, potentially improving the retrieval of highly relevant evidence even in complex queries.

For claim verification, Bowman et al. (2015) regard it as a form of Recognizing Textual Entailment, where the task is to determine if evidence supports or refutes a claim. However, in our project, apart from the `SUPPORTS`, `REFUTES`, `NOT ENOUGH INFO` that utilized to label individual evidence, our claim label also include `DISPUTE` label when both `SUPPORTS` and `REFUTES` presented in the evidence. Thus we experimented with both evidence-level and claim level classification to explore which approach could better capture the nuance of the `DISPUTE` label. The evaluation revealed that the evidence level approach heavily relies on the quality of evidences retrieved, as it perform well when utilizing the ground truth label from the dev set but performance suffered significantly when using the evidence retrieved from our neural ranking model. The claim level classification shown more robust performance against irrelevant evidences, could attributed to the combined representation enhance the model's ability to focus on the overall context and ignore less relevant details.

For the sequence processing components, we evaluate the effectiveness of Gated Recurrent Units (GRU) and Transformer architecture. We choose GRU over LSTM was because the similar performance between these two, GRU model is more efficient without using the context vector and output gate, resulting in fewer parameters and more robust to vanishing gradient problem (Shewalkar et al., 2019). On the other hand, many state of the art performance in the verification task has been achieved through the use of pre-trained Transformer model (Soleimani et al., 2020), we want to explore the effectiveness of the Transformer model when pre-trained model is not available. After evaluating the performance on the development set, the GRU model with attention mechanisms consistently outperformed the Transformer with much faster training process, likely due to its simpler and more computationally efficient structure, which is particularly advantageous in settings with limited training data and computational resources.

2 Approach

2.1 Evidence Retrieval

We formulate the Evidence Retrieval as dense passage retrieval task, where we encode claim c_i and a list of evidence $\{e_1, e_2, \ldots, e_N\}$ into dense vectors using Siamese network (Mitra and Craswell, 2018; Das et al., 2016). Let $f(c_i)$ denote the vector representation of the claim c_i encoded by the model, and let $g(e_j)$ denote the vector representation of the evidence text e_j encoded by the model. The similarity $\text{sim}(c_i, e_j)$ between the claim c_i and the evidence document e_j is computed as the dot product (Mussmann and Ermon, 2016) of their vector representations:

$$\text{sim}(c_i, e_j) = f(c_i) \cdot g(e_j)$$

The goal of our neural ranking model is to maximize the similarity score $\text{sim}(c_i, e_j^+)$ for relevant evidence documents e_j^+ while minimize the similarity score $\text{sim}(c_i, e_k^-)$ for irrelevant evidence documents e_k^-. To achieve this, we utilized the loss function proposed by (Karpukhin et al., 2020), which focus on optimizing the negative log-likelihood of the positive evidence. The loss function used is defined as:

$$\begin{aligned} &\mathcal{L}(c_i, e_j^+, e_{i,1}^-, \ldots, e_{i,n}^-) = \\ &\quad -\log \frac{e^{\text{sim}(c_i, e_j^+)}}{e^{\text{sim}(c_i, e_j^+)} + \sum_{k=1}^{n} e^{\text{sim}(c_i, e_{i,k}^-)}} \end{aligned} \tag{1}$$

To utilize the defined loss function, we use the ground truth evidences for each claim c_i as positive samples. For negative samples, we include both in-batch negatives and top negatives from initial filtering. In-batch negatives are other claims' positive evidences, providing $B - 1$ in-batch negatives for each claim, where B is the batch size. For top negatives, we evaluate the effectiveness of TF-IDF (Thorne et al., 2018), Word2Vec (Mikolov et al., 2013) (using cosine similarity), and BM25 (Zhan et al., 2021) algorithms. We manually implemented BM25 with an inverted index in Python to compute similarity scores and rerank evidences for each claim.

2.2 Claim Verification

For the claim verification task, our objective is to classify a given claim c_i into one of four categories: `SUPPORTS`, `REFUTES`, `NOT ENOUGH INFO`, or `DISPUTED`. We explored two different strategies to achieve this classification: the evidence-level aggregation approach and the direct claim-level classification approach. Both strategies employ the same underlying model architectures but differ in how the evidence is aggregated and utilized to make the final prediction.

Evidence-Level Classification: In the evidence-level aggregation approach, each piece of evidence related to a claim is individually classified into one of three categories: `SUPPORTS`, `REFUTES`, or `NOT ENOUGH INFO`. These individual classifications are then aggregated to produce the final claim label. The aggregation method involves tallying the individual evidence predictions. If both `SUPPORTS` and `REFUTES` evidence are present, the claim is classified as `DISPUTED`. If only `SUPPORTS` or only `REFUTES` evidence is present, the claim is classified accordingly. If neither `SUPPORTS` nor `REFUTES` evidence is found, the claim is classified as `NOT ENOUGH INFO`.

Direct Claim-Level Classification: The direct claim-level classification approach treats the claim and its related evidence as a single combined sequence. This combined text is input to the model, which then directly classifies the claim into one of the four categories. By processing the combined sequence of the claim and all related evidence documents as a single text input, the model can capture the semantic relationships between the claim and the evidence, allowing for a more informed prediction based on the integrated context of all the evidence provided for the claim.

2.3 Model Architectures

We employed two primary model architectures: a GRU-based model with attention and a Transformer-based model. Both architectures were used in the context of the two tasks mentioned above: evidence retrieval and claim verification.

The **Bi-Directional GRU** (Tang et al., 2015) model with attention uses a Gated Recurrent Unit (GRU) to encode the sequence of words in both the claim and the evidence documents. We implemented an attention mechanism applied to the GRU outputs to ensure model focus on most important part of the text during training. The encoded representations from the GRU are then used differently depending on the task. For evidence retrieval, the dot product is computed in a Siamese network setting to measure similarity between claims and evidence vector. For claim verification, the encoded representations are passed through fully connected layers to produce the final classification logits. The attention mechanism computes attention weights that highlight important parts of the evidence when making a decision about the claim.

The **Transformer-based** (Vaswani et al., 2017) model leverages self-attention mechanisms to capture long-range dependencies within the text. Both the claim and evidence documents are processed using multiple layers of self-attention and feedforward networks. Positional encodings are added to the word embeddings to retain the order of the tokens. The encoded representations from the Transformer are used differently depending on the task. For evidence retrieval, the similarity is computed in a Siamese network setting. For claim verification, the encoded representations are combined to make the final classification. The self-attention mechanisms in the Transformer allow the model to better understand the relationships between different parts of the text, making it well-suited for tasks that require deep semantic understanding.

3 Experiments

3.1 Evaluation method

Evidence Retrieval Recall at k: This metric evaluates the proportion of relevant evidence passages retrieved in the top k predictions compared to the ground truth. It measures the system's recall in retrieving the correct evidence passages. We use this metric to evaluate the initial retrieval models—TF-IDF, Word2Vec, and BM25. The goal is to achieve high recall to narrow the search space from over 1 million evidence passages and provide negative samples for efficient training of the neural ranking model.

$$\text{Recall@}k = \frac{\text{Relevant Passages Retrieved@}k}{\text{Total Relevant Passages}}$$

Evidence Retrieval F-score at k: This metric evaluates how well the system retrieves evidence passages compared to the ground truth, considering the top k predictions. By setting k to 5, we focus on the most confident predictions, balancing precision and recall to measure both completeness and relevance. This metric is used to assess the effectiveness of our neural retrieval model, which aims for high F1 scores given that the average number of evidence for a claim in the training data is between 3 and 4.

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

Claim Classification Evaluation: These metric assesses the accuracy and macro F1 score of multi-class claim predictions, independent of the evidence retrieved. Accuracy is the ratio of correct to total predictions, and macro F1 averages Precision and Recall across all classes, providing a balanced evaluation of the model's performance in diverse classification scenarios.

$$\text{A} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

$$\text{F1} = \frac{1}{N}\sum_{i=1}^{N} \frac{2 \times \text{Precision}_i \times \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i}$$

3.2 Experimental details

3.2.1 Preprocessing

In the preprocessing process of evidence retrieval, we refine the text by tokenizing, removing stop words and special characters, and applying the Porter Stemmer to reduce words to their root forms, aiding in effective matching. For claim verification, we tokenize the text while retaining stop words, special characters, and numbers, and use lemmatization to convert words to their base forms, preserving context for accurate claim verification.

3.2.2 Model Configuration

The model parameters were tuned through testing with different settings to identify the configurations that produced the best results.

For the **GRU-based model**, we use an embedding layer initialized with random weights, mapping input tokens to 256-dimensional dense vectors. The core of the model is a bidirectional GRU with 512 hidden units, which captures dependencies in both forward and backward directions. An attention mechanism computes attention scores over the GRU hidden states, allowing the model to focus on the most relevant parts of the sequences. A dropout rate of 0.7 is applied for regularization, preventing overfitting by randomly dropping units during training.

For the **Transformer-based model**, the embedding layer maps input tokens to 256-dimensional dense vectors, initialized with random weights. Positional encoding is added to retain the order of the tokens. The Transformer encoder consists of 6 layers, each with multi-head attention comprising 8 heads, a feedforward network with a hidden size of 1024, ReLU activation, a dropout rate of 0.8 for regularization, and layer normalization. This configuration captures complex dependencies and stabilizes the training process by maintaining a consistent representation of the input data.

3.2.3 Training Details

Loss Function: For evidence retrieval, we utilized the Negative Log-Likelihood of the positive passage discussed in Equation 1. For claim verification, we used Weighted Cross-Entropy Loss, which is appropriate for multiclass classification as it effectively handles multiple classes by comparing the predicted probability distribution across all classes to the actual class labels, ensuring accurate and stable training.

Optimizer: The Adam optimizer is used for training with a learning rate of 0.001. We also applied gradient clipping to prevent exploding gradients. Additionally, we used a learning rate scheduler, `torch.optim.lr_scheduler.ReduceLROnPlateau`, to adjust the learning rate, reducing it by a factor of 0.5 if the validation loss did not improve for 3 consecutive epochs

Epochs: Our experiments used 15 epochs for the GRU-based model and 30 epochs for the Transformer-based model. The model state was saved when it reached the best F1 score on the evaluation set for retrieval and the best accuracy for classification.

Batch Size: A batch size of 32 is used to train the model, which is chosen to ensure efficient use of computational resources while maintaining sufficient data variability within each batch.

3.2.4 Data Sampling

For the evidence retrieval part, we experimented with different negative sampling strategies to evaluate their effectiveness:

Random: Randomly samples negatives from the top filtered set provided by the initial retriever. This baseline introduces diverse negative examples but may include easily distinguishable negatives.

In Batch: Uses positive evidence from other claims within the same batch as negatives. Each claim has one positive sample and $B - 1$ in-batch negatives, improving the model's ability to distinguish between similar evidence.

In-Batch + Gold Negatives: For each claim, this strategy uses one positive sample, $B - 1$ in-batch negatives, and one top negative from the top 50 reranked non-ground truth evidence as the gold negative. All claims in the same batch share these top negatives (Karpukhin et al., 2020).

4 Result

4.1 Initial Retrieval Model

Model	Recall @ 100
BM25	**0.5128**
TF-IDF	0.4104
Word2Vec	0.4203

Table 1: Comparison of Recall @ k = 100 on dev set for different models

As shown in Table 1, BM25 outperforms both TF-IDF and Word2Vec in terms of recall @k = 100. Despite BM25 being an older, non-ML-based retrieval algorithm, it excels due to its ability to effectively incorporate document-specific term frequency and inverse document frequency, adjusting the weight of terms based on their document length and rarity across documents (Robertson and Zaragoza, 2009). TF-IDF, while useful, does not normalize for document length and can favor longer documents, leading to less effective retrieval results (Mitra and Craswell, 2018). Word2Vec captures semantic relationships between words but lacks the term frequency and document relevance considerations that are crucial for precise information retrieval (Mikolov et al., 2013). Therefore, BM25's design makes it more suitable for initial retrieval tasks, resulting in higher recall compared to TF-IDF and Word2Vec.

4.2 Negative Sampling Strategies

Figure 1 shows that InBatchGold yields the lowest validation loss, followed by InBatch and Random sampling. InBatchGold, which selects top negatives from the top 50 reranked non-ground truth evidence, provides challenging samples, helping the model develop precise decision boundaries. Random sampling results in the highest loss due to

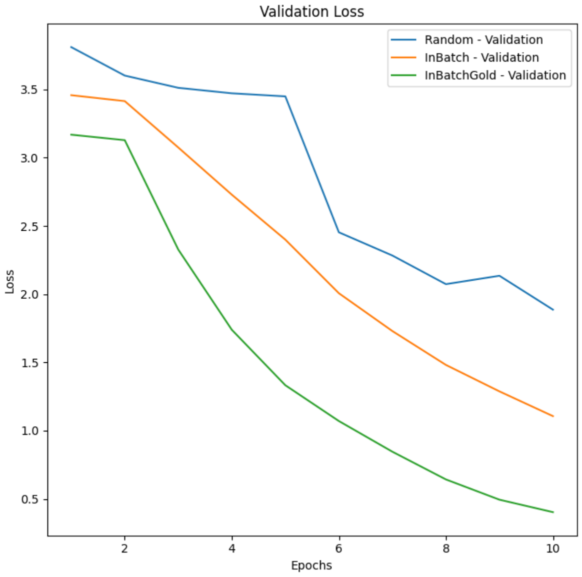

Figure 1: Validation Loss for Different Sampling Strategies

insufficiently challenging negatives, causing overfitting. InBatch offers intermediate performance with moderately challenging negatives. These results highlight that using more informative and challenging negatives, as in InBatchGold, improves the model's ability to distinguish between relevant and irrelevant evidence (Zhan et al., 2021).

4.3 Neural Ranking Models

Model	Precision @ k=5	Recall @ k=5	F1 @ k=5
GRU (Baseline)	0.1211	0.0714	0.0841
GRU + Attention	**0.1727**	**0.0922**	**0.1122**
Transformer	0.1633	0.0896	0.1068

Table 2: Performance of Neural Retrievers on dev set(Precision, Recall, and F1 @ k=5)

The results in Table 2 show that the GRU with attention outperforms both the GRU baseline and the Transformer model in precision, recall, and F1 score at $k = 5$. The attention mechanism in the GRU enhances its ability to focus on relevant parts of the input, capturing key patterns and relationships more effectively. The Transformer's performance, while competitive, is limited by the lack of extensive pre-training, which is essential for fully leveraging its complex architecture (Vaswani et al., 2017). This highlights the effectiveness of attention mechanisms in GRUs for precise feature extraction and relevance determination in evidence retrieval.

4.4 Claim Verification

Performance Comparison of Different Models and Approaches			
Model	Approach	Macro Avg F1	Accuracy
Bi-Directional GRU	Evidence Level	0.34	0.40
	Claim Level	**0.42**	**0.49**
Transformer	Evidence Level	0.29	0.34
	Claim Level	0.39	0.44

Table 3: Performance Comparison of Different Models and Approaches

Test Set Evaluation Results		
Evaluation Phase	Evidence Retrieval F1 Score	CLS Accuracy
Ongoing Evaluation	0.1015	0.4737
Final Evaluation	0.0726	0.2208

Table 4: Performance of the Best Model from Development Set on the Test Set

While the Bi-Directional GRU generally outperforms the Transformer in both evidence-level and claim-level approaches as shown in Table 3. However, our model achieved only 0.22 accuracy in the final test set Table 4, indicating potential overfitting to the majority class seen during development. This performance discrepancy suggests that the separation of evidence retrieval and claim verification into distinct processes might be limiting. Integrating these stages could potentially leverage retrieval patterns to improve verification accuracy, pointing to a more cohesive approach as a promising direction for future enhancements (Nie et al., 2019).

5 Conclusion

In this project, we developed an automated fact-checking system focusing on evidence retrieval and claim verification for climate-related claims. Our findings show that the GRU-based model with attention outperformed both the GRU baseline and Transformer models, highlighting the importance of attention mechanisms for precise feature extraction. The InBatchGold sampling strategy provided the best results, emphasizing the need for challenging negative samples. The claim-level classification approach yielded better results than the evidence-level approach, demonstrating the value of holistic context. The primary limitation was the absence of pre-trained models, which are crucial for fully leveraging Transformer architectures. Future work should explore joint learning approaches to better utilize patterns learned during retrieval and improve overall system performance.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Arpita Das, Harish Yenala, Manoj Chinnakotla, and Manish Shrivastava. 2016. Together we stand: Siamese networks for similar question retrieval. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 378–387, Berlin, Germany. Association for Computational Linguistics.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A Survey on Automated Fact-Checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering. *Preprint*, arXiv:2004.04906.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *Preprint*, arXiv:1310.4546.

Bhaskar Mitra and Nick Craswell. 2018. An introduction to neural information retrieval. pages 77–81.

Stephen Mussmann and Stefano Ermon. 2016. Learning and inference via maximum inner product search. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2587–2596, New York, New York, USA. PMLR.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6859–6866.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3:333–389.

Apeksha Shewalkar, Deepika Nyavanandi, and Simone Ludwig. 2019. Performance evaluation of deep neural networks applied to speech recognition: Rnn, lstm and gru. *Journal of Artificial Intelligence and Soft Computing Research*, 9:235–245.

Amir Soleimani, Christof Monz, and Marcel Worring. 2020. Bert for evidence retrieval and claim verification. In *Advances in Information Retrieval*, pages 359–366, Cham. Springer International Publishing.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1432, Lisbon, Portugal. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018. The fact extraction and VERification (FEVER) shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1503–1512, New York, NY, USA. Association for Computing Machinery.