# NLP Project
# Fake News Challenge Stage 1 — Stance Detection

Sihan Xie, Ilyas Lebleu, Elycheva Dray, Lauréline Charret

May 14, 2023

**Abstract**

This report focuses on our work for the Natural Language Processing course, specifically on the stance detection task for the Fake News Challenge Stage 1 (FNC-1) project. We begin by introducing the FNC-1 competition and providing an overview of the key concepts related to stance detection. We then conduct a literature review to identify the current state-of-the-art method for stance detection. Based on our analysis of the existing literature, we devise a working plan for our approaches. We conclude by presenting and comparing the performance of various models that we implemented.

## 1 Preliminaries

### 1.1 The Fake News Challenge

The proliferation of social media contents has led to an increasing attention on the research of **Fake News Detection**. The Fake News Challenge Stage 1 (FNC-1) was a competition held in 2017 that aimed to explore how artificial intelligence technologies, particularly machine learning and natural language processing, can be used to combat the issue of fake news. Evaluating the veracity of news stories is a complex task, even for human experts. However, understanding the relationship between the news body and the news topic is a helpful first step in identifying fake news. Thus, the FNC-1 competition focused solely on automating this first step, called **Stance Detection**, which could serve as a valuable component in an AI-assisted fake-news-detection pipeline.

### 1.2 Formal task definition

The goal of stance detection is to automatically determine the attitude of a piece of text towards a particular topic. More formally, we denote $X = \{b, h\}_{i=1}^{N}$ as the input data, where $b$ is the news body and $h$ is the news headline. Stance detection involves predicting a stance label $s$ for the news body $b$ towards the given headline $h$. The FNC-1 competition proposed to classify the stance of the news body text relative to the claim made in the news headline into one of four categories:

- **agree** : the body text agrees with the headline
- **disagree** : the body text disagrees with the headline.
- **discuss** : the body text covers the same topic but does not take position
- **unrelated** : the body text is about a different topic

## 1.3 Dataset

The quality of the training data and the annotation process can have a significant impact on the performance of stance detection models. The dataset provided by the challenge is composed of 2587 headlines and bodies which were extracted from articles on 300 different topics. Topics were not shared between the test and training data split: 200 topics were reserved for a 49972-instance training set and 100 topics for a 25413-instance test set. Each topic is represented by about 5–20 news article documents. Each news article document was summarized into a headline that reflects the stance of the whole document. Each pair of headline-document was matched with one of the four stance labels (*agree*, *disagree*, *discuss*, *unrelated*). To generate the *unrelated* class, some headlines and bodies were randomly associated.

In 2017, before the emergence of transformer-based models [10], stance detection for news was viewed as a challenging task because it required a deep understanding of the context and the relationship between a short phrase and a long text, which could be difficult for models to capture accurately. Another challenge in FNC-1 was the imbalanced training data, as shown in Figure 1. The dataset was highly imbalanced with 73.1% of body-headline pairs belonging to the *unrelated* category, 17.8% to the *discuss* class, 7.4% to *agree*, and only 1.7% to the *disagree* class.

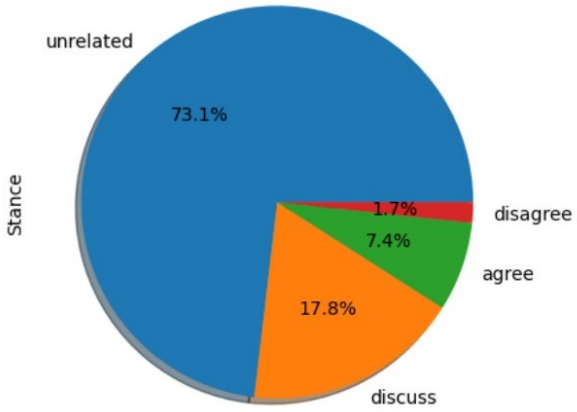


Figure 1: Distribution of the stances

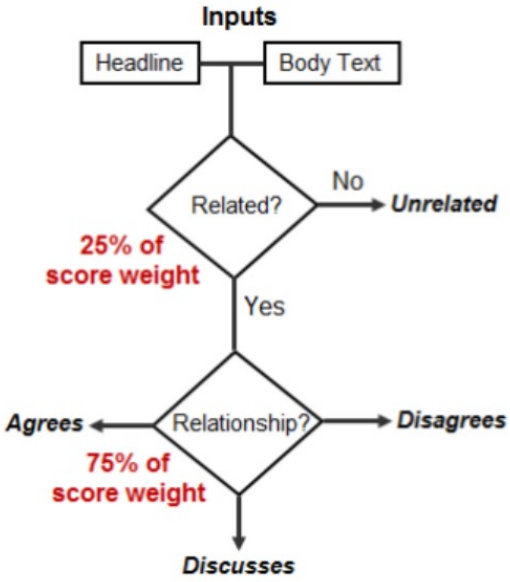


Figure 2: Scoring process from the Fake New Challenge

## 1.4 Evaluation

The official evaluation method, called FNC score, is presented in Figure 2. The models were evaluated based on a weighted, two-level scoring system:

- Classifying the headline and body text as *unrelated* or related(*discuss*, *agree*, *disagree*) represents 25% of the final score
- Classifying related pairs as *agree*, *disagree*, or *discuss* represents 75% of the final score

The purpose of this weighted schema was to balance the large number of *unrelated* instances. However, as explained in [3], this scoring system is actually unfair towards minorities as it fails to take into account the highly imbalanced class distribution of the three related classes (*agree*, *disagree*, *discuss*). Thus, a classifier that just randomly predicts one of the three related

classes would already achieve a very high FNC score. A more appropriated metric to evaluate document-level stance detection task has been proposed in [3]. It consists in computing the class-wise $F_1$ scores and average them to make the macro-averaged $F_1$ scores ($F_1m$). This metric is not biased by the large size of the majority class *unrelated*. As a reminder, the F1-score is computed as follows:

$$F_1\_score = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{1}$$

### 1.5 FNC-1 Result

Following the introduced evaluation system in Section 1.4, the participant team obtained both a raw score and a normalized relative score that accounts for the maximum possible score achievable on the test set. The top-3 models in the FNC-1 competition were:

| **Rank** | **Team Name** | **Affiliation** | **Score** | **Relative Score** |
|---|---|---|---|---|
| 1 | SOLAT in the SWEN | Talos Intelligence | 9556.50 | 82.02 |
| 2 | Athene (UKP Lab) | TU Darmstadt | 9550.75 | 81.97 |
| 3 | UCL Machine Reading | UCL | 9521.50 | 81.72 |

To gain an understanding of successful solutions, it was mandatory for the winning teams to share their solutions under an Apache 2.0 license, which ensures us to have access to the details of the winning solutions. SOLAT in the SWEN team used an ensemble based on an 50/50 weighted average between gradient-boosted decision trees and a deep convolutional neural network. Athene team used an ensemble method consisting of 5 multi-layer perceptron, whereby the stances had been predicted by hard voting. The solution of UCL Machine Reading team was based on a single, end-to-end system consisting of lexical as well as similarity features passed through an one-layer perceptron. We can see that the solutions developed by all the winning teams exhibit fairly close performance, reflecting the prevailing trend in 2017 towards addressing downstream NLP tasks by using sophisticated feature engineering in conjunction with a relatively straightforward machine learning or deep learning model.

## 2 Literature review and our work plan

### 2.1 Literature review

The problem of stance detection, along with other NLP downstream tasks, has been approached through various methods, including traditional machine learning algorithms and deep learning techniques. Indeed, NLP tasks were initially divided into smaller and more manageable tasks, such as Sentiment Analysis, Named Entity Recognition and Machine Translation. This was done to facilitate the development of specialized models and techniques for each task, as each task requires a different type of language understanding. However, the development of solutions to these downstream tasks has followed a similar trend as the evolution of NLP itself.

The earliest approaches to stance detection from the 1950s used rule-based systems, which were heavily relied on *feature engineering* to identify the stance of a piece of text. However, these methods often suffer from low accuracy due to their reliance on hand-crafted features and the inability to generalize well to different domains and languages. Since the 1990s, a transition into traditional machine learning based classifiers was made. Some of the most used algorithms were Support Vector Machine (SVM), Naive Bayes, Decision Tree.

With the advent of deep learning and the availability of large amounts of labeled data in the 2010s, deep learning models quickly became the mainstream techniques for stance detection, leading to remarkable advancements in performance. Salient features were learned jointly with the training of the model itself, and hence focus shifted to *architecture engineering*. The deep neural networks of different structures, such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), attention-based models and Graph Convolutional Network (GCN), have

been utilized to develop the desired stance classifier. Attention-based models primarily leverage target-specific information as the attention query and implement an attention mechanism to infer the stance polarity, while GCN aims to model the relation between the target and text. Long short-term memory (LSTM) was once widely used for sequence classification tasks as it has the advantage of being able to handle variable-length sequences and capture long-term dependencies in the input data.

However, from 2017-2019 there was a significant change in the learning of NLP models. With the great success of Google's bidirectional encoder representations from transformers (BERT) [2] model in 2018, the field of transfer learning for NLP gained significant attention and became more prevalent. Following BERT, numerous Large Language Models (LLMs) have been proposed with different specialties and purposes. Specifically, the standard shifted to using pre-trained LLMs together with a fine-tuning process [7], entirely removing the need for task-specific architectures. LLMs are trained on massive amounts of text data in an unsupervised way and are able to capture rich semantic and syntactic information about language. LLMs use a transformer-based architecture, which allows them to effectively capture long-range dependencies and contextual information in text. Furthermore, LLMs can be fine-tuned on specific downstream tasks by adding task-specific layers on top of the pre-trained model to achieve effective transfer while requiring minimal changes to the model architecture. This *generative* pre-train and *discriminative* finetune paradigm provides exceptional performance for most NLP downstream tasks including stance detection. To fine-tune a LLM for stance classification objective, we can adapt our LLM by building a stance classification head, for example, a linear+softmax layer on top of the [CLS] token, then update the weights of the pre-trained LLM by training on a supervised dataset. Typically thousands to hundreds of thousands of labeled stances are needed.

Although the pre-train and fine-tune paradigm has been shown to achieve state-of-the-art performance, it has conceptual limitations. As a matter of fact, humans do not require large supervised datasets to learn most language tasks. Rather, a brief directive in natural language or, at most, a small number of demonstrations is often sufficient to enable a human to perform a new task to a reasonable degree of competence. Inspired by this observation and popularized by the release of GPT-3 [1], more recently a new paradigm called "pre-train, prompt and predict" has gained widespread attention. The core idea behind this prompt-based paradigm is mimicking LLMs to design prompts suitable for different tasks. Obviously it can alleviates the demand for large amount of training data and the tedious training process as no gradient updates are performed. Another advantage of this method is that, given a suite of appropriate prompts, a single LLM can be used to solve a great number of tasks. However, as most conceptually enticing prospects, there is a catch. This method introduces the necessity for *prompt engineering*, which involves finding the most appropriate prompt to allow a LLM to solve the task at hand. They can be divided into 3 categories:

(a) **zero-shot learning**: No demonstrations are allowed and only an instruction in natural language is given to the model.
(b) **one-shot learning**: We allow only one demonstration.
(c) **few-shot learning**: We allow as many demonstrations as will fit into the model's context window (typically 10 to 100).

Here, we give an illustration in Figure 3, showing the differences between the "pre-train, prompt and predict" paradigm and the "pre-train, fine-tune" paradigm for language translation tasks. Some recent experiments [13] have demonstrated that utilizing the GPT-3.5-0301 prompt can result in state-of-the-art or comparable performance for frequently used stance detection datasets, such as SemEval-2016 [6] and P-Stance [4]. Although this prompt-based approach shows some initial promise on several NLP tasks, it still achieves results that are far inferior to the fine-tuning paradigm. There are some tasks, such as reading comprehension, on which few-shot performance struggles even at the scale of GPT-3. Also this type of learning can be unstable: the choice of prompt format, training examples, and even the order of the training examples can cause accuracy to vary from near chance to near state-of-the-art. It has been shown that this instability arises from the bias of language models towards predicting certain answers(e.g., those that are placed near the end of the prompt or are common in the pre-training

data) [14].

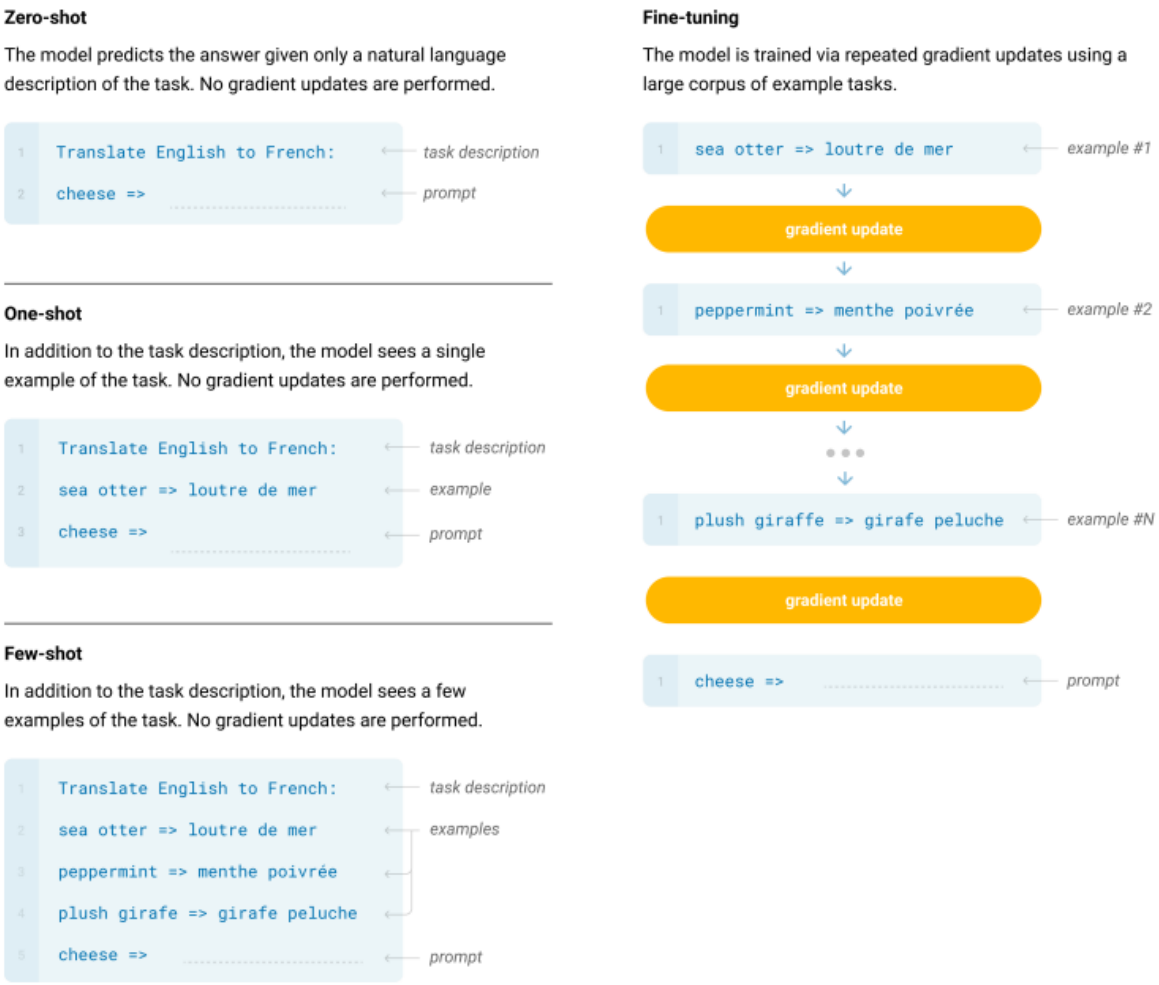


Figure 3: Zero-shot, one-shot and few-shot, contrasted with traditional fine-tuning for translation task [1]

## 2.2 Work plan

Although the winning solutions of the FNC-1 challenge presented in Section 1.5 may not align with today's trend for solving downstream tasks, it is noteworthy that a simple solution could achieve high performance, as demonstrated by the UCL Machine Reading team (UCLMR) team's solution using a single-layer MLP[8]. Thus, in Section 3, we present our baseline solution by replicating the UCLMR team's model. As identified in Section 2.1, the current state-of-the-art approach for solving stance detection task is the "pre-train and fine-tune" paradigm. Hence, in Section 4, we compare the performance of different fine-tuned LLM models that we implemented on the FNC-1 dataset. In addition, since prompt engineering is a hot topic after the launch of chatGPT and it has several practical advantages over the fine-tuning approach, we present the results of the "pre-train, prompt and predict" paradigm in Section 5.

# 3 Reproduce UCLMR's model

## 3.1 UCLMR's model and training process description

The UCLMR team proposed a MLP classifier with only a single hidden layer. Their features were composed of two simple bag-of-words (BOW) representations for the text inputs: term frequency (TF) and term frequency-inverse document frequency (TF-IDF). More precisely, for the headlines and the bodies, they used term frequency vectors of unigrams of the 5,000 most frequent words. They also computed the cosine similarity between the TF-IDF vectors of the headline and body. The resulting term frequency feature vectors of headline and document were concatenated along with the cosine similarity of the two TF-IDF vectors.

We reproduced their model with the hyperparameters given in [8] and trained it with an Adam optimizer, $L_2$ regularisation, dropout, gradient clipping and learnign rate decay.

## 3.2 Result discussion

We managed to obtain similar results : 81.91% while UCLMR team's FNC score in the challenge was 81.72%

We also evaluated it using the class-wise $F_1$ and the $F_1m$ metric described in Section 1.4, the results are presented in Table 1. We notice that the model gets a much lower score when we use the $F_1m$ metric that is not affected by the size of the majority class. As expected, the model's $F_1$ score is very different for each class and the model favorites the most represented class.

| Model | FNC score | $F_1m$ | $F_{1_{Agree}}$ | $F_{1_{Disagree}}$ | $F_{1_{Discuss}}$ | $F_{1_{Unrelated}}$ |
|---|---|---|---|---|---|---|
| UCLMR | 81.91% | 55.83% | 47.35% | 3.862% | 74.77% | 97.34% |

Table 1: Different evaluations of the UCLMR's model

We can conclude from Table 1 and the confusion matrix (Figure 5 in the Appendix A) that the very high performance of the UCLMR model comes from the close to perfect classification of the instances into related and *unrelated* headline/body pairs and the more or less default *discuss* classification of the related instances. However, the classifier's performance with respect to the *agree* label is quite average, whereas the classifier's accuracy on the *disagree* test examples is very poor. It is quite a shame since these two labels are the most relevant to the goal of automating the stance evaluation process and ultimately detect fake news.

This shows that the metric chosen by the organisers of the challenge is not very appropriated and does not balance out enough the large number of unrelated instances.

# 4 Fine-tune pretrained LLMs

## 4.1 Model and fine-tuning process description

We began by processing the input data for stance detection. Typically, the input to the LLM model would be a combination of the news headline and body. To concatenate these two, we inserted a [SEP] token between them. To ensure a fixed-length input sequence, we applied a simple padding and truncation strategy. However, a possible improvement could be to use a text summarization algorithm to extract a few key sentences or phrases from the news body, thereby avoiding the loss of important information due to truncation. After this, we tokenized the input texts and encoded the stance labels into numerical form.

We selected the following 4 LLMs to fine-tune:

(a) **BERT [2]**: It's based on a bidirectional transformer encoder because this architecture allows the model to capture contextual information from both left and right contexts of a word.

(b) **DistillBERT** [9]: It is a compressed version of the BERT model. It is designed to be smaller and faster than the original BERT model by leveraging knowledge distillation during the pre-training phase, while still maintaining comparable performance(e.g., reduce the size of BERT by 40%, while retaining 97% of its language understanding capabilities and being 60% faster).
(c) **RoBERTa** [5]: It builds on BERT and adjusts key hyperparameters. It improves upon BERT in several ways, including using a byte-level BPE as a tokenizer, removing the next-sentence pretraining objective and training on a larger corpus with much larger mini-batches and learning rates.
(d) **XLNet** [12]: XLNet employs a novel pre-training method based on permutation, which allows it to model dependencies among all positions in a sequence, unlike BERT's bidirectional approach that models only the left and right positions of each token. This is achieved by generating a sequence of tokens with randomly masked tokens and randomly shuffled order, and training the model to predict the original order of the tokens, which allows it to learn more robust and generalized representations.

We trained each LLM for 4 epochs using the AdamW optimizer and set each epoch as a checkpoint. After the training process, we selected the best model to ensure optimal performance. Our implementation was carried out using Hugging Face's Transformers library which supports a wide range of tasks from data processing to model deployment.

## 4.2 Result comparison

In Table 2 and Figure 4, we present a comparative analysis of different LLMs on various evaluation metrics. The results demonstrate that RoBERTa outperforms all other models on each metric. Although BERT performs better than XLNet in classifying the majority class (*unrelated* and *discuss*), XLNet excels in classifying the minority class (*agree* and *disagree*). We also observe that all fine-tuned LLMs significantly surpass the UCLMR's model on every metric, without much effort on feature engineering and optimization for the fine-tuning process.

| Model | FNC score | $F_1m$ | $F_{1_{Agree}}$ | $F_{1_{Disagree}}$ | $F_{1_{Discuss}}$ | $F_{1_{Unrelated}}$ |
|---|---|---|---|---|---|---|
| BERT | 88.64% | 74.82% | 67.77% | 48.65% | 84.03% | 98.83% |
| DistillBERT | 84.25% | 61.81% | 58.92% | 11.78% | 78.49% | 98.04% |
| RoBERTa | **90.42%** | **78.59%** | **72.18%** | **55.97%** | **86.88%** | **99.33%** |
| XLNet | 88.52% | 75.02% | 68.96% | 48.91% | 83.43% | 98.77% |

Table 2: Performance comparison of different LLMs

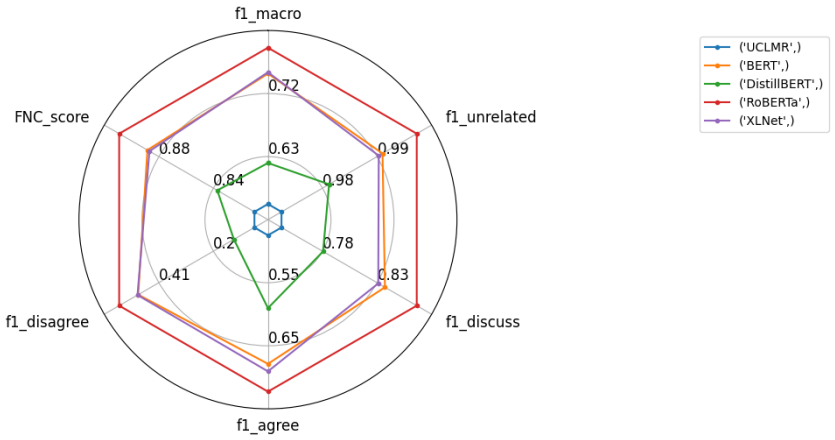


Figure 4: Performance comparison of different LLMs through a radar plot

# 5 Prompt-based in-context learning

## 5.1 Prompt design

Recent studies have shown that scaling up language models greatly improves task-agnostic few-shot performance [1]. Therefore, we chose to use GPT-3.5 for zero-shot and few-shot learning in our stance detection task. We utilized the OpenAI's Chat Completion API to generate text based on prompts. However, since OpenAI restricts API access for normal users to only three requests per minute, we were only able to test our approach on a smaller dataset of 1000 instances. This dataset follows the same class distribution as the original dataset.

Effective prompt design is crucial to provide the LLM with sufficient context and guidance for accurate stance classification. However, no standard implementation framework for prompt-learning has been proposed yet. To address this, we employed a fill-in-the-blank style, which involves providing the LLM with a partial sentence and asking it to fill in the blank with the appropriate stance classification. To eliminate random completions of the model, we also utilized a multiple-choice prompt style, whereby we provided the LLM with a set of options for the stance classification and asked it to choose the most appropriate one. For few-shot learning, we also included a *discuss*, a *unrelated*, an *agree* and a *disagree* stance example from the training set as input so that the model could learn to adapt to this context. Below, we present the prompt designs we used for zero-shot learning and few-shot learning:

**Zero-shot Learning**:
"Given the following news headline: $< Headline >$.
Given the following news body: $< Body >$.
What is the stance of this news body towards this news headline? Please choose one of the following stances: unrelated, discuss, agree, disagree. The stance is "

**Few-shot Learning**:
"Here are four examples to guide your classification:
News Headline: $< Headline >$, News Body: $< Body >$, Stance: unrelated
News Headline: $< Headline >$, News Body: $< Body >$, Stance: discuss
News Headline: $< Headline >$, News Body: $< Body >$, Stance: disagree
News Headline: $< Headline >$, News Body: $< Body >$, Stance: agree
Using these examples, given the following news headline: $< Headline >$, given the following news body: $< Body >$, what is the stance of this news body towards this news headline? Please choose one of the following stances: unrelated, discuss, agree, disagree. The stance is "

## 5.2 Result Discussion

With the prompt design described in Section 5.1, GPT-3.5 typically generated responses directly aligned with the defined stances(*unrelated*, *discuss*, *agree*, *disagree*). This eliminated the need for designing a reliable mapping function to associate GPT-3.5's generated response with a specific stance label, which can be a challenging task. In some cases, we only required simple mapping like lowercase conversion (e.g., mapping "Discuss" to "discuss").

There was an exceptional instance where GPT-3.5 provided a comprehensive response, offering reasoning to support its stance classification decision (e.g. GPT-3.5 returned "unrelated. (There is no mention of Obama ordering the Fed to adopt the Euro currency in this news body.)"). This highlights the potential of prompt-based learning as a valuable tool for gaining insights into the decision-making process of LLMs and understanding the rationale behind their stance choices. However, it's important to acknowledge that the explanations provided by LLMs are generated based on learned patterns from the training data and may not always align with the ground truth. Therefore, it is crucial to complement the insights from LLMs with critical evaluation and human judgment.

Furthermore, we observed that the stance detected by GPT-3.5 was not consistently deterministic. Naturally, we got the idea of collecting multiple responses from the LLM and applying

statistical tests to analyze the distribution of stances. This approach would enable us to assess the level of agreement among the responses, leading to more robust and consensus-based decisions.

The results of zero-shot learning and few-shot learning are presented in Table 3. The performance of the prompt-based methods fell short of our expectations. They only managed to outperform UCLMR's model in terms of $F_{1_{Disagree}}$ metric and they struggled to accurately predict the *unrelated* and *discuss* stances. This can be attributed to GPT-3.5's tendency to classify most instances as *discuss* without seeing the distribution of stance labels in the training set. When we transitioned to few-shot learning by incorporating additional examples, the performance further deteriorated. As discussed in Section 2.1, few-shot learning can be highly unstable and influenced by various factors. In addition to refining the prompt design, we believe that employing multi-round Chain-of-Thought prompting [11] could be a potential improvement. This approach would enable the LLM to generate more knowledgeable responses, building upon the context and previously generated knowledge(e.g. We first let LLM distill or summarize the key points of a news article then ask it the stance towards the news headline.).

| Approach | FNC score | $F_1m$ | $F_{1_{Agree}}$ | $F_{1_{Disagree}}$ | $F_{1_{Discuss}}$ | $F_{1_{Unrelated}}$ |
|---|---|---|---|---|---|---|
| Zero-shot | 50.41% | 27.82% | 21.93% | 19.75% | 31.81% | 37.79% |
| Few-shot | 42.25% | 22.72% | 27.45% | 23.08% | 23.12% | 17.24% |

Table 3: Performance comparison of zero-shot learning and few-shot learning

# Project Review

In this project, our main objective was to explore different approaches for stance detection task and evaluate their performance using various metrics. We successfully replicated one of the winning solutions from the FNC-1 challenge. Additionally, we fine-tuned different pre-trained LLMs, including BERT, DistillBERT, RoBERTa, and XLNet, to assess their suitability for stance detection. Our experiments demonstrated the power of fine-tuning LLMs, revealing their capability to interpret contextual information in text and achieve impressive performance. During our investigation, we also delved into the design of prompts for zero-shot and few-shot learning scenarios. We acknowledge the trade-offs associated with these different approaches. It is worth noting that fine-tuning LLMs can be a time-consuming and computationally demanding process, but the resulting performance is generally promising. On the other hand, zero-shot learning offers a faster solution without the need for retraining, although the achieved performance may not be as robust.

Throughout the project, we identified the strengths and limitations of each method, enabling a comprehensive analysis of their applicability and potential areas for improvement. Looking ahead, we believe that few-shot learning holds promise as a future direction for the stance detection task. This approach offers the potential to improve accuracy, enhance adaptability, address data scarcity challenges and reduce the burden of data annotation. Continued research and experimentation in this direction are likely to bring new insights and advancements in stance detection methodologies.

# References

[1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[3] Andreas Hanselowski, Avinesh P.V.S., Benjamin Schiller, Felix Caspelherr, Debanjan * Chaudhuri, Christian M. Meyer, and Iryna Gurevych. A retrospective analysis of the fake news challenge stance-detection task. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, Juni 2018.

[4] Yingjie Li, Tiberiu Sosea, Aditya Sawant, Ajith Jayaraman Nair, Diana Inkpen, and Cornelia Caragea. P-stance: A large dataset for stance detection in political domain. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2355–2365, Online, August 2021. Association for Computational Linguistics.

[5] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[6] Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California, June 2016. Association for Computational Linguistics.

[7] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[8] 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.

[9] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[11] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

[12] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding, 2020.

[13] Bowen Zhang, Daijun Ding, and Liwen Jing. How would stance detection techniques evolve after the launch of chatgpt?, 2023.

[14] Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models, 2021.

# A Appendix

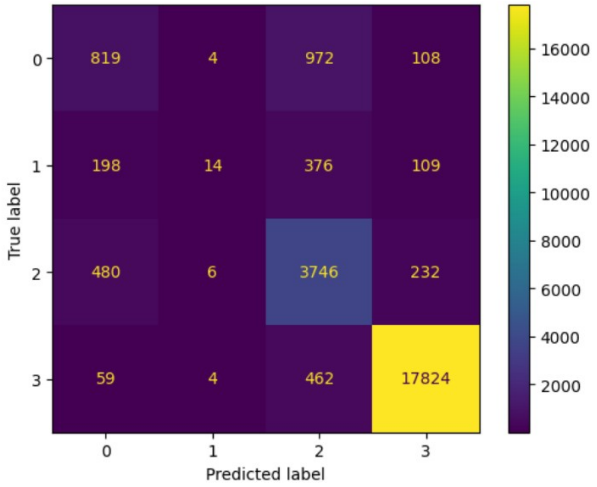


Figure 5: Confusion matrix of the UCLMR model
(0 = agree; 1 = disagree; 2 = discuss; 3 = unrelated)