Documentation

1. Experiment Setup

1.1 Overview

The goal is to develop a small-scale LLM-based application for text topic categorization. Using DistilBERT and DistilGPT2, we compare which base model is more suitable for the task.

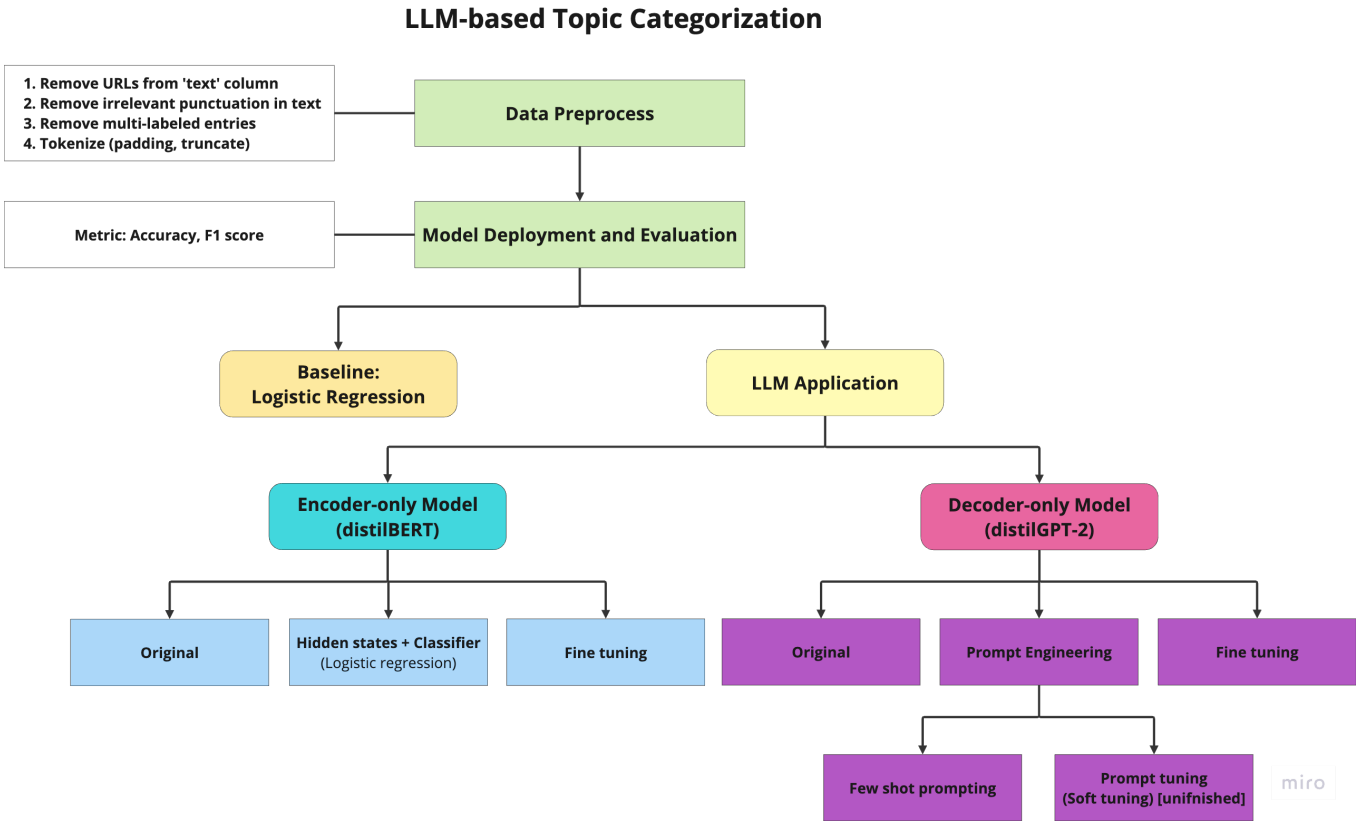

Figure1: Methodology

1.2 Data

I. Description:

We utilized the Twitter Financial News dataset, sourced from Twitter Financial News/Kaggle. This English-language dataset comprises an annotated corpus of finance-related tweets, making it suitable for tasks such as sentiment analysis or financial text classification. The data is multi-labeled and imbalanced.

The dataset consists of two primary columns:

- text: The content of the tweet.
- label: The corresponding annotation or classification for each tweet (category). There are 20 distinct labels, the match is shown below.

For data splitting:

- Training Dataset: Contains 16,990 unique records with two columns (text and label).
- Test Dataset: Contains 4,117 unique records with the same structure.

```
{0: 'Analyst Update',
 1: 'Fed and Central Banks',
 2: 'Company and Product News',
 3: 'Treasuries and Corporate Debt',
 4: 'Dividend',
 5: 'Earnings',
 6: 'Energy and Oil',
 7: 'Financials',
 8: 'Currencies',
 9: 'General News and Opinion',
 10: 'Gold and Metals and Materials',
 11: 'IPO',
 12: 'Legal and Regulation',
 13: 'M&A and Investments',
 14: 'Macro',
 15: 'Markets',
 16: 'Politics',
 17: 'Personnel Change',
 18: 'Stock Commentary',
 19: 'Stock Movement'}
```

Figure2: ID2label

text	label
Here are Thursday's biggest analyst calls: Apple, Amazon, Tesla, Palantir, DocuSign, Exxon & more https://t.co/QPN8Gwl7Uh	0
Buy Las Vegas Sands as travel to Singapore builds, Wells Fargo says https://t.co/fLS2w57iCz	0
Piper Sandler downgrades DocuSign to sell, citing elevated risks amid CEO transition https://t.co/1EmtywmYpr	0
Analysts react to Tesla's latest earnings, break down what's next for electric car maker https://t.co/kwhoE6W06u	0
Netflix and its peers are set for a 'return to growth,' analysts say, giving one stock 120% upside https://t.co/jPpdl0D9s4	0
Barclays believes earnings for these underperforming stocks may surprise Wall Street https://t.co/PHbsyVGAyE	0
Bernstein upgrades Alibaba, says shares can rally more than 20% from here https://t.co/m3ApoPRGU0	0
Analysts react to Netflix's strong quarter, with some pointing to a potential bottom for the stock https://t.co/cQngJsyefD	0
Buy Chevron as shares look attractive at these levels, HSBC says https://t.co/GkDpFvxjEP	0
Morgan Stanley says these global stocks are set for earnings beats — and gives one over 45% upside https://t.co/GeWxa5YoWr	0

Figure3: training data sample

II. Data preprocessing:

1. Separate URLs from 'text' column:

Each tweet entry contains at least one URL. These URLs are extracted and stored in a new column called url. The intention behind this separation is to leverage the URLs as potential sources of additional information, as the content linked in the URLs often provides the full context of the tweets. This external data can be utilized for improving the model's performance in future iterations.

2. Remove duplicate cleaned text entries:

After extracting and cleaning the text column (removing URLs), we observed that some tweets share identical cleaned text but have different URLs. For instance, certain tweets form a series with the same name and category, differing only by issue dates, which are not reflected in the text column. To ensure data quality, rows with duplicate cleaned text and labels are removed, retaining only unique entries.

3. Remove special punctuation:

While large language models (LLMs) can process punctuation, certain special characters commonly used in tweets (e.g., @ and #) may not contribute meaningfully to the classification task. These characters are removed to simplify the input data and reduce potential noise.

4. Focus on single-labeled data:

To streamline the experiment as a proof of concept (POC), we focused on single-labeled data, where each tweet is assigned only one label. This approach enhances time efficiency during training and testing while allowing for clearer insights into the model's performance.

5. Tokenization:

The text data was tokenized using a pre-trained tokenizer from the Hugging Face library corresponding to the specific model used (e.g., BERT or GPT-2). Tokenization included:

- Converting text into numerical tokens while preserving special tokens, e.g. [CLS], [SEP].
- Padding sequences to a fixed length for batch processing.
- Truncating longer sequences beyond the model's maximum input length.

1.3 Testing Cycles

I. Baseline

To establish a benchmark for evaluating the effectiveness of LLM-based models, a logistic regression classifier is used with TF-IDF vectorization.

Steps are as follows:

1. Preprocess cleaned text data by converting it into TF-IDF features using TfidfVectorizer.
2. Train a logistic regression model using the TF-IDF features.
3. Evaluate the model on the test dataset using standard metrics: accuracy, and F1 score.
4. Analyze the baseline results to compare them with subsequent models.

II. Decoder-only Model (distilGPT2) with Different Strategies

1. **Direct Classification with Pre-trained distilGPT2:**
   - Use HuggingFace pipelines for initial evaluation.
   - Record metrics without any additional training.
2. **Few-shot Prompting:**
   - Use a basic prompt template to feed the input text to distilGPT2.
   - Modify the prompt to include labeled examples.
   - Evaluate how few-shot prompts affect the performance compared to basic prompt.
3. **Fine-tuning:**
   - Fine-tune distilGPT2 for classification using AutoModelForSequenceClassification.
   - Split the training dataset into training and validation sets for fine-tuning.
   - Optimize hyperparameters to maximize validation performance (unfinished).
   - Evaluate the fine-tuned model on the test dataset.
4. **Prompt Tuning (Unfinished):**
   - Implement soft prompting

III. Encoder-only Model (distilBERT) with Different Strategies

1. **Feature Extraction with distilBERT:**
   - Use distilBERT to extract hidden state embeddings for each input text.
   - Pass the embeddings to a logistic regression model for classification.
2. **Direct Classification with Pre-trained distilBERT:**
   - Use AutoModelForSequenceClassification with pre-trained weights to directly classify input text.
   - Test the model on the validation dataset without further training.
3. **Fine-tuning distilBERT:**
   - Fine-tune distilBERT on the classification task.
   - Split the training dataset into training and validation sets for fine-tuning.
   - Optimize hyperparameters to maximize validation performance (unfinished).

1.4 Code Structure

Note: While the final code is intended to be in the form of Python scripts (.py), due to time constraints, only Jupyter Notebook files (.ipynb) have been uploaded. The notebooks are organized to walk through the different stages of the project, including data preprocessing, model training, evaluation.

2. Observed Results and Insights

2.1 Results

I. Accuracy and F1 Score

Base Model	Method	Accuracy	F1
-	Logistic regression	0.79	0.78
DistilBERT	Direct Classification	0.14	0.07
DistilBERT	Feature Extraction	0.82	0.82
DistilBERT	Fine tune (best so far)	0.90	0.90
DistilGPT2	Direct Classification	0.09	0.07
DistilGPT2	Few shot prompting	-	-
DistilGPT2	Fine tune (one trail only)	0.89	0.89

Note:

1. Missing data:

Few-shot prompting with DistilGPT2 has incomplete results due to ongoing work on optimizing the prompt design. Current efforts focus on refining the prompt to produce better-structured and more accurate outputs. Further experimentation is needed to evaluate the model's potential fully.

2. Relatively low metric value:

The accuracy and F1 scores for both models using the direct classification method are relatively low. This could be due to the baseline model essentially making random predictions without any training.

II. Fine Tuning DistilBERT Model

Different runs were done by varying number of epochs, initial learning rates and weight decay. Visualization can be seen in Figure 4 and 5. All fine-tuning results are saved in WanDB.

Run	Number of epochs	Initial learning rate	weight_decay
2	10	5e-6	1e-2
3	10	2e-6	1e-2
4	4	5e-5	5e-2
5	4	6e-5	5e-3
6	4	8e-6	5e-3

Table: Parameters for fine tuning DistilBERT

Figure 4: Evaluation metrics

Concern:

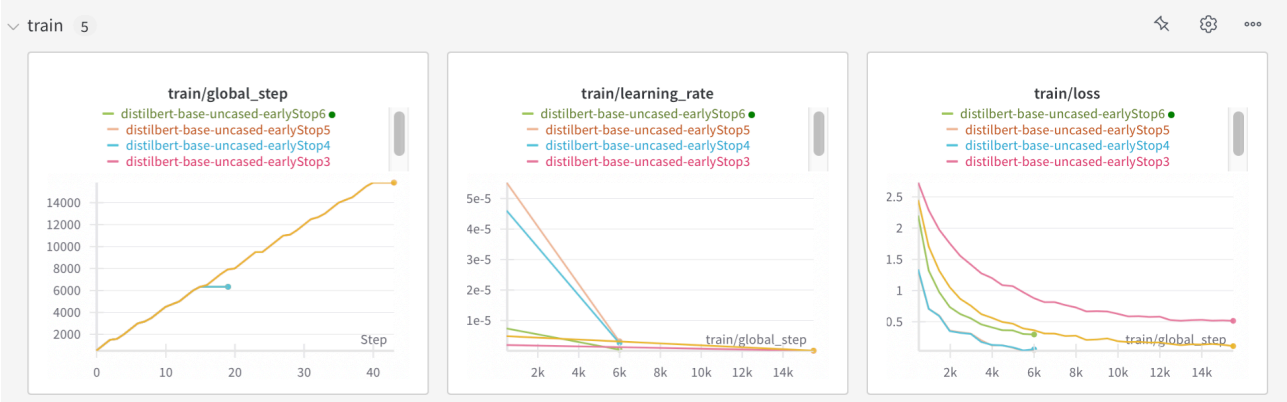

Figure 5: Training metrics

Although the run labeled 'earlyStop5' demonstrates the highest accuracy and F1 score on the test set, the evaluation and training loss curves suggest a potential risk of overfitting.

III. Performance by Class

Ideally, testing results should be analyzed in depth to identify specific classes with the highest error rates and gain insights into model performance across different categories. This step is crucial for understanding the model's weaknesses and tailoring improvements accordingly. However, due to time constraints and the absence of hyperparameter tuning in the current phase, this detailed analysis has not yet been conducted.

2.2 Insights

I. Comparing with Baseline: While direct classification with LLMs underperforms, leveraging LLMs for feature extraction or fine-tuning yields better performance.

The baseline model, Logistic Regression, delivers a strong performance with an accuracy of 0.79 and F1 score of 0.78. However, LLM-based models show mixed results. Direct classification with DistilBERT and DistilGPT2 significantly underperforms, suggesting that LLMs require refinement. When used for feature extraction or fine-tuned, both models surpass the baseline, emphasizing the need for further tuning to optimize LLM performance for classification tasks.

II. Comparing the Methods Used with LLM: Feature Extraction with DistilBERT Shows Strong Performance, but Fine-Tuning Holds Greater Promise.

Leveraging DistilBERT for feature extraction combined with Logistic Regression yields a solid performance, with an accuracy of 0.82 and an F1 score of 0.82. Fine-tuning both DistilBERT and DistilGPT2 shows significant promise, with DistilBERT reaching an accuracy and F1 score of 0.90, while DistilGPT2 achieves 0.89. These results suggest that LLM-based models, especially when fine-tuned or used for feature extraction, can effectively compete with traditional models, with potential for further improvement as fine-tuning progresses.

III. Comparing Pre-trained LLM Models: The Potential of DistilBERT and DistilGPT2 Remains to Be Fully Explored.

In theory, encoder-only models like DistilBERT are generally better suited for classification tasks, which focuses on understanding and encoding the entire input sequence. While DistilGPT2, a decoder-only model, may perform well in generative tasks. As the fine-tuning process for both models is ongoing, it is too early to draw definitive conclusions, but DistilBERT shows more promise for classification at this stage.

IV. Challenges with Few-Shot Prompting: More Work to Be Done.

The few-shot prompting approach with DistilGPT2 is still under development and has not yet yielded meaningful results. This method holds promise for improving performance by utilizing prompt engineering, but more work is needed to refine the prompts and structure the inputs in a way that maximizes the model's potential.

3. Recommendations for Future Improvement or Scalability

3.1 Improvement

I. Continue Prompt Optimization for distilGPT2

1. Suggestions for refining prompts based on observed results.
2. Exploration of advanced prompt techniques (e.g., task-specific templates, ensemble prompting).

II. Model Fine-Tuning

1. Continue hyperparameter tuning.
2. Experiment with adjusting training (e.g., unfreezing additional layers, using LoRA).

3.1 Scalability

I. Dataset expansion

Currently we only used the title data for the tweets, the tweets body can be used as added data to train the model.

II. Model scaling

In this project, DistilBERT and DistilGPT2 were chosen as base models due to their smaller size and efficiency. However, upon confirming the potential of this approach, we can scale to larger architectures, such as GPT-3.5 or GPT-4.

Authored by: Xin (Kiana) Wang

Date: 2025.01.14