ELECTRA and GPT-4o: Cost-Effective Partners for Sentiment Analysis

James P. Beno
Stanford Engineering CGOE
jim@jimbeno.net

Abstract

Bidirectional transformers like ELECTRA excel at sentiment analysis, and Large Language Models (LLM) like ChatGPT are effective zero-shot learners. Might they perform better as a team? This paper explores collaborative approaches between ELECTRA and GPT-4o for three-way sentiment classification. We fine-tuned (FT) four models (ELECTRA Base/Large, GPT-4o/4o-mini) using a mix of reviews from Stanford Sentiment Treebank (SST) and DynaSent. We provided input from ELECTRA to GPT as: predicted label, probabilities, and retrieved examples. Sharing ELECTRA Base FT predictions with GPT-4o-mini significantly improved performance over either model alone (82.74 macro F1 vs. 79.29 ELECTRA Base FT, 79.52 GPT-4o-mini) and yielded the lowest cost/performance ratio ($0.12/F1 point). However, when GPT models were fine-tuned, including ELECTRA's predictions decreased performance. GPT-4o FT was the top overall performer (86.99), with GPT-4o-mini FT close behind (86.77) at much less cost ($0.38 vs. $1.59/F1 point). On individual datasets, GPT-4o FT was best on DynaSent R1 (90.57) and R2 (89.00), while GPT-4o-mini FT was best on SST-3 (75.68). Our results show: (1) augmenting LLM prompts with predictions from fine-tuned local models is an efficient way to boost performance, and (2) a fine-tuned GPT-4o-mini is nearly as good as GPT-4o FT at 76% less cost, providing affordable options for projects with limited resources.

1 Introduction

Sentiment analysis—the computational study of opinions, attitudes, and emotions in text (Medhat et al., 2014)—has seen remarkable advances from transformer architectures (Vaswani et al., 2017), which have outperformed traditional machine learning approaches. Bidirectional transformers like BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and ELECTRA (Clark et al., 2020) excel at sentiment classification when fine-tuned, and Large Language Models (LLM) like GPT (Radford et al., 2018) demonstrate strong zero-shot and few-shot learning ability—they can perform sentiment analysis by prompting alone, or with just a few examples (Kheiri and Karimi, 2023).

Recent work has explored opportunities for collaboration between these models, such as escalating to open LLMs like Llama 2 (Touvron et al., 2023) when RoBERTa classification confidence was low (Andrade et al., 2024), using GPT to augment data of minority classes before fine-tuning with RoBERTa (Kok-Shun et al., 2023), and using GPT for aspect extraction and RoBERTa for sentiment scoring (Qian et al., 2024; Andrade et al., 2024). However, a number of collaborative scenarios have yet to be explored.

This paper investigates collaborative approaches between ELECTRA and GPT-4o models (OpenAI, 2024b,c) for three-way sentiment classification (negative, neutral, positive) of reviews. Our research was centered on the following hypotheses:

- **H1:** Providing predictions from a fine-tuned ELECTRA model as context to GPT-4o/4o-mini (not fine-tuned) will improve sentiment classification performance.
- **H2:** The improvement in performance provided by the ELECTRA predictions will be less for a fine-tuned GPT-4o/4o-mini model.
- **H3:** The format of ELECTRA predictions in the prompt (text label vs. probabilities) will affect GPT-4o/4o-mini's performance.
- **H4:** Including semantically similar examples with true labels in the prompt will improve GPT-4o/4o-mini's classification performance.

These hypotheses are based on the following observations:

- **ELECTRA's bidirectional transformer architecture**, unique use of replaced token detection (rather than masked token prediction), and discriminator-based pre-training (Clark et al., 2020) make it well-suited for capturing nuanced sentiment patterns when fine-tuned (B et al., 2023).
- **GPT models are versatile learners** (Radford et al., 2019; Liu et al., 2019; Kocoń et al., 2023; OpenAI, 2024a)—they can perform well across diverse tasks when given the appropriate context through prompting (Liu et al., 2023; Khattab et al., 2024). Although they may struggle with emotion and nuance (Kocoń et al., 2023), retrieved examples can improve performance (Zhang et al., 2023).
- **ELECTRA can be fine-tuned locally on available GPUs**, making it a cost-effective option compared to the fees required to fine-tune GPT-4o/4o-mini via an API.

To test these hypotheses, we established four baselines and conducted 23 experiments across three sentiment classification datasets: Stanford Sentiment Treebank (SST), and DynaSent Rounds 1 and 2. We used ELECTRA Base/Large and GPT-4o/4o-mini, each of which were fine-tuned on a merge of SST and DynaSent reviews. In addition, we investigated the effects of different prompt augmentation scenarios using DSPy (Khattab et al., 2024): sharing the predicted class text label, the probabilities of each class as percentages, retrieving similar reviews with their class labels (few-shot examples), and combinations thereof. We evaluated classification performance with the macro average F1 score, and cost-effectiveness by dividing the total fine-tuning costs by the F1 score.

Our results provided the following insights:

- **Sharing Predictions Boosted Base Models:** Augmenting GPT-4o-mini with predictions from ELECTRA Base FT significantly improved performance over either model alone, achieving a macro F1 score of 82.74 on the merged test set (vs. 79.29 ELECTRA Base FT, 79.52 GPT-4o-mini). It also yielded the lowest cost/performance ratio ($0.12/F1 point). A similar boost was seen by sharing the ELECTRA Large FT prediction with GPT-4o-mini: 83.49 macro F1 (vs. 82.36 ELECTRA Large FT, 79.52 GPT-4o-mini), but at a higher cost ($0.64/F1 point).
- **Fine-tuned GPTs Performed Best:** GPT-4o FT achieved the highest overall performance on the merged test set (86.99 macro F1), with GPT-4o-mini FT closely following (86.77) at significantly lower cost ($0.38 vs. $1.59/F1 point). If overall cost/performance is the main driver, a fine-tuned gpt-4o-mini is the clear choice. However, there were slight differences in performance across the datasets.
- **Top Performers Varied by Dataset:** GPT-4o FT was the top performer on the DynaSent Round 1 (90.57 macro F1) and Round 2 (89.00) test sets. Surprisingly, a fine-tuned GPT-4o-mini—the smaller model—was the top performer on SST-3 (75.68).
- **Sharing Predictions Hurt Fine-Tuned Models:** When both GPT models were fine-tuned, incorporating ELECTRA predictions led to a significant decrease in performance. GPT-4o-mini FT's macro F1 on the merged test set was 86.99, but it fell to 83.82 when the ELECTRA prediction was included in the prompt. Similarly, GPT-4o FT's macro F1 for the merged test set was 86.77, but it fell to 81.42. This decrease persisted (but was less) when the GPT models were fine-tuned on the same prompt template used at inference time, which included the ELECTRA prediction.
- **Fine-tuned ELECTRA Large Outperformed Base GPTs:** The fine-tuned ELECTRA Large model was the best performing local model, with a macro F1 on the merged test set of 82.36, vs. 79.29 for ELECTRA Base FT. However, the cost/performance ratio was higher ($0.65 vs. $0.12/F1 point). ELECTRA Large FT was better than both GPT-4o and GPT-4o-mini base models (80.14 and 79.52).

These findings show that there are affordable options for sentiment classification projects with limited resources. First, if fine-tuning via API is an option, a fine-tuned GPT-4o-mini is nearly as good as GPT-4o FT at 76% less cost. Alternatively, augmenting LLM prompts with predictions from fine-tuned local models is an efficient way to boost performance. And lastly, for projects that want to stay local, a fine-tuned ELECTRA Large model is quite capable with sentiment classification, and better than the GPT models prior to fine-tuning.

2 Prior Literature

Sentiment analysis is the "computational study of people's opinions, attitudes and emotions toward an entity," (Medhat et al., 2014) and is often applied to reviews (products or businesses), financial investments, politics, and news. There are two main approaches to classifying sentiment: the machine learning approach, and the lexicon approach. This research focuses on evaluating machine-learning methods, specifically those that involve the interplay between transformer-based neural networks like ELECTRA and ChatGPT.

2.1 Masked Language Models

BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) is a Masked Language Model (MLM) based on the transformer architecture (Vaswani et al., 2017). Its bidirectional encoding allows it to develop holistic representations of the entire sentence or document, which makes it well-suited for sentiment analysis. RoBERTa (A Robustly Optimized BERT Pre-training Approach) (Liu et al., 2019) optimized the pre-training approach and hyper-parameters for a performance boost. However, these models were inefficient due to their use of the [MASK] token. Learning was only occurring in about 15% of the tokens that were masked.

2.2 ELECTRA

Masked Language Models (MLMs) like BERT and RoBERTA were eventually outperformed by models that were not constrained by the [MASK] token: XLNet (Yang et al., 2020) and ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately) (Clark et al., 2020). ELECTRA was pre-trained with two models using replaced token detection. An MLM generator "corrupts" some tokens in the input sequence, and a discriminator detects which tokens are "real" (in the training data) vs. "fake" (replaced by generator samples). As a result, it learned from all tokens and exhibited comparable or better performance in a variety of tasks with less compute.

ELECTRA was found to be a top performer in sentiment classification tasks such as the Stanford Sentiment Treebank (SST) (Clark et al., 2020), DynaSent (Potts et al., 2021), and the IMBD movie review dataset (B et al., 2023). It was also found to be better suited for prompt-based learning due to its use of a discrimnator (Xia et al., 2022). It's for these reasons that we chose to use ELECTRA as the fine-tuned local model in our research, in addition to observing a performance gain relative to RoBERTa in early trials.

2.3 GPT Models

The bidirectional nature of these models seems to give them an edge over early autoregressive (unidirectional) models like GPT (Radford et al., 2018). But that edge is being whittled away by the successors of GPT that are pre-trained at a massive scale: GPT-3, GPT-3.5, GPT-4, and GPT-4o (OpenAI, 2024a,b,c).

For sentiment analysis of social media posts, Kheiri and Karimi (2023) found that the GPT models significantly outperformed a number of prior models on the SemEval 2017 dataset. In contrast, Kocoń et al. (2023) found that, although ChatGPT is versatile and competent across a wide range of tasks, it did not perform as well as state-of-the-art (SOTA) models–especially for pragmatic tasks involving detection of emotional and contextual nuances. In particular, the SOTA model for TweetSent was RoBERTa. They propose that fine-tuning ChatGPT may be a key ingredient for success, but were unable to explore this.

In the current research, we did explore fine-tuning GPT-4o and GPT-4o-mini, in addition to evaluating their default performance (through in-context prompting alone).

2.4 Collaborative Approaches

Recent work has revealed several promising approaches for collaboration between bidirectional transformer models and LLMs.

Kok-Shun et al. (2023) explored a unique framework that chains GPT and RoBERTa for emotion detection. They used GPT's generative capabilities to augment training data for minority classes, addressing the class imbalance. The augmented dataset is then used to fine-tune RoBERTa on emotion detection. By harnessing the generative capabilities of GPT, they were able to improve RoBERTa's ability to detect emotions with the GoEmotions dataset. In this research, we are not using any generative aspects of GPT.

Qian et al. (2024) took a collaborative approach to analyzing stadium reviews. One GPT-3.5 model was fine-tuned to extract experience aspects, while another classified these aspects into experience categories. A RoBERTa model then performed sentiment scoring on the extracted aspects. This

shows how different model architectures can perform tasks that align with their strengths in a Natural Language Processing (NLP) pipeline. We are chaining ELECTRA and GPT-4o in a similar manner here, but in a different sequence, and with different tasks.

Andrade et al. (2024) investigated the benefits of collaboration between MLMs and open LLMs for sentiment classification, similar to the current research. In their "Call-My-Big-Sibling" (CMBS) approach, the initial classification is done with a calibrated RoBERTa model. If RoBERTa has low confidence on the classification, an open LLM like Llama 2 (Touvron et al., 2023) is invoked to perform the classification task instead (typically zero-shot or few-shot).

In CMBS, the final prediction is either made by RoBERTa or Llama 2—it's a decision tree. In contrast, our approach always passes the ELECTRA prediction to the LLM. If we had to come up with a similar analogy, it would be "Show-Me-Your-Answers" (SMYA). And then it's up to the LLM to decide if it follows the ELECTRA prediction, or decides to classify the review differently.

Table 1: Label Distribution for SST-3

Split	Negative	Neutral	Positive
Train	3,310	1,624	3,610
Validation	428	229	444
Test	912	389	909

Table 2: Label Distribution for DynaSent R1

Split	Negative	Neutral	Positive
Train	14,021	45,076	21,391
Validation	1200	1200	1200
Test	1200	1200	1200

Table 3: Label Distribution for DynaSent R2

Split	Negative	Neutral	Positive
Train	4,579	2,448	6,038
Validation	240	240	240
Test	240	240	240

3 Data

In this research, all models were trained and evaluated on a Merged dataset comprised of the Stanford Sentiment Treebank (SST) and DynaSent Rounds 1 and 2. We'll now describe these in detail.

Table 4: Label Distribution for the Merged Dataset

Split	Negative	Neutral	Positive
Train	21,910	49,148	31,039
Validation	1,868	1,669	1,884
Test	2,352	1,829	2,349

Table 5: Contribution of Sources to the Merged Dataset

Dataset	Samples	Percent (%)
DynaSent R1 Train	80,488	78.83
DynaSent R2 Train	13,065	12.80
SST-3 Train	8,544	8.37
Total	102,097	100.00

3.1 SST

SST-5 is the Stanford Sentiment Treebank (Socher et al., 2013) 5-way classification (positive, somewhat positive, neutral, somewhat negative, negative). For this research, three classes of sentiment were used: positive, negative, neutral. To create this, the "somewhat positive" class was merged and treated as "positive." Similarly, the "somewhat negative class" was merged and treated as "negative." See Table 1 for the distribution of class labels in SST-3.

3.2 DynaSent

DynaSent (Potts et al., 2021) is a sentiment analysis dataset and dynamic benchmark with three classification labels: positive, negative, and neutral. The dataset was created in two rounds. First, a RoBERTa model was fine-tuned on a variety of datasets including SST-3, IMBD, and Yelp. They then extracted challenging sentences that fooled the model, and validated them with humans. For Round 2, a new RoBERTa model was trained on similar (but different) data, including the Round 1 dataset. The Dynabench platform was then used to create sentences written by workers that fooled the model. See Tables 2 and 3 for the distribution of class labels in DynaSent R1 and R2.

3.3 Merged Dataset

The SST-3, DynaSent R1, and DynaSent R2 datasets were randomly mixed to form a new dataset with 102,097 Train examples, 5,421 Validation examples, and 6,530 Test examples. See Table 4 for the distribution of labels within this merged dataset.

It's worth noting that the source datasets all have class imbalances. Merging the data helps mitigate

Table 6: Models Used in Research

Model	Provider	Access	Identifier
ELECTRA Base	Hugging Face	Local	google/electra-base-discriminator
ELECTRA Large	Hugging Face	Local	google/electra-large-discriminator
GPT-4o	OpenAI	API	gpt-4o-2024-08-06
GPT-4o-mini	OpenAI	API	gpt-4o-mini-2024-07-18

this imbalance. Although there is still a majority of neutral examples in the training dataset, the neutral to negative ratio in DynaSent R1 is 3.21, and this is improved to 2.24 in the merged dataset.

Another potential issue is that the models will learn the dominant dataset, which is DynaSent R1. And indeed, all model configurations consistently performed better on DynaSent R1 vs. R2 or SST-3. However, that may be due to the more challenging nature of the latter datasets. As a test, the minority classes were over-sampled to create a new balanced dataset. When this was evaluated, the performance did not improve.

The performance of the models will be evaluated against the Test split of the Merged dataset, as well as the Test splits of each individual dataset (DynaSent R1, DynaSent R2, SST-3). This will help us assess how well the models were able to learn from the different source datasets, and enable comparison with benchmarks.

4 Models

Four models were fine-tuned and evaluated in this research, both individually and in collaboration with each other: ELECTRA Base and Large, and GPT-4o and 4o-mini. See Table 6 for details.

4.1 ELECTRA

ELECTRA (Clark et al., 2020) was chosen as the bidirectional transformer because it's pre-training architecture gives it an advantage over MLMs. It also outperformed RoBERTa in early trials. We evaluated both the Base (110M parameters) and Large (335M parameters) variants.

To function as a classifier, ELECTRA's output is sent through a mean pooling layer. A classifier head is appended with 2 hidden layers of dimension 1024, and a final output dimension of 3. Swish GLU (Shazeer, 2020) was used as the hidden activation function, and dropout layers were added with a rate of 0.3. See Appendix A for more details on the model architecture and hyper-parameters.

4.2 GPT-4o/4o-mini

For comparison and collaboration, two GPT models were used via OpenAI's API: GPT-4o (OpenAI, 2024b) and its more efficient variant GPT-4o-mini (OpenAI, 2024c). Although the full specifications are not public, they are state-of-the-art autoregressive language models with strong zero-shot capabilities. GPT-4o is described as a "high-intelligence flagship model for complex, multi-step tasks." GPT-4o-mini is described as an "affordable and intelligent small model for fast, lightweight tasks." Both models have a 128,000-token context window, and 16,384 maximum output tokens. Training data is dated October 2023.

5 Methods

Our research progressed through several stages: establishing baselines, evaluating ELECTRA Base/Large capabilities alone, evaluating GPT-4o/4o-mini capabilities alone, and a variety of collaborative scenarios that shared output from the fine-tuned ELECTRA models with the GPT models (default and fine-tuned). Code and datasets are available at: `https://github.com/jbeno/sentiment`.

5.1 ELECTRA Baseline & Fine-tuning

We first developed a training pipeline that extended Stanford CS224U's PyTorch classes to support interactivity and distributed training across multiple GPUs. Training progress was tracked through Weights and Biases so we could monitor train/validation metrics (loss, macro F1, accuracy) across epochs. The final models were selected from checkpoints at convergence, or just before train/validation metrics started to diverge.

Two baseline models were established by training only classifier heads for ELECTRA Base and Large. Hyper-parameters were consistent with the fully fine-tuned versions. The fine-tuning process involved a number of trials on Lambda Labs multi-GPU instances to identify the best hyper-parameters, optimizer, and learning rate schedule.

Table 7: Minimal Template for Fine-tuning

Role	Content
System	"You are a model that classifies the sentiment of a review as either 'positive', 'neutral', or 'negative'."
User	"Those 2 drinks are part of the HK culture and has years of history. It is so bad."
Assistant	"negative"

Table 8: Prompt Template for Fine-tuning

Role	Content
System	"You are a sentiment analysis assistant."
User	"**Classify the sentiment of a review as either 'negative', 'neutral', or 'positive'.\n—\n\nFollow the following format.\n\nReview: The review text to classify.\nClassification: One word representing the sentiment classification: 'negative', 'neutral', or 'positive' (do not repeat the field name, do not use 'mixed')\n\n—\n\nReview:** Those 2 drinks are part of the HK culture and has years of history. It is so bad.\n**Classification:**"
Assistant	"negative"

Table 9: Prompt & Prediction Template for Fine-tuning

Role	Content
System	"You are a sentiment analysis assistant."
User	"Classify the sentiment of a review as either 'negative', 'neutral', or 'positive'.\n—\n\nFollow the following format.\n\nReview: The review text to classify.\n**Classifier Decision: The sentiment classification proposed by a model fine-tuned on sentiment.**\nClassification: One word representing the sentiment classification: 'negative', 'neutral', or 'positive' (do not repeat the field name, do not use 'mixed')\n\n—\n\nReview: Those 2 drinks are part of the HK culture and has years of history. It is so bad.\n**Classifier Decision: negative**\nClassification:"
Assistant	"negative"

See Appendix A for the final configuration.

We also explored alternative approaches including an ensemble of binary classifiers and staged fine-tuning (progressively training on DynaSent R2 and SST-3 after initial merged dataset training), though these did not outperform our initial approach.

5.2 GPT Data Preparation & Fine-tuning

Table 10: Fine-Tuning Job Details

Model	Code	**Format**	**Tokens**
GPT-4o-mini	FT	Prompt	11.0M
GPT-4o-mini	FT-L	Prompt w/Base Label	13.2M
GPT-4o-mini	FT-M	Minimal	5.5M
GPT-4o	FT	Prompt	11.0M
GPT-4o	FT-L	Prompt w/Large Label	13.2M
GPT-4o	FT-M	Minimal	5.5M

To use OpenAI's fine-tuning API, we converted the Merged training data to JSONL format that defined the System, User, and Assistant roles. We noticed that if the context at inference time varied even slightly from the fine-tuning context, performance would suffer. So we created three templates to enable better comparisons between fine-tuned and default models using the same DSPy signatures:

- **Minimal (FT-M):** No prompt other than System role (see Table 7)
- **Prompt (FT):** Default fine-tuning. User role included full DSPy prompt (see Table 8)
- **Prompt with Label (FT-L:** User role included DSPy prompt with ELECTRA predicted label (see Table 9)

We included the ELECTRA predictions in the third template to align the fine-tuning context with the inference time context, but also to provide an opportunity for the GPT models to learn from the ELECTRA predictions. In total there were 6 fine-tuning jobs (see Table 10).

5.3 DSPy Signatures & Modules

Using DSPy, we explored a variety of approaches to integrating ELECTRA's output into GPT's decision-making process. Each approach was implemented as a custom DSPy signature and module. See Appendix B for full examples of each signature.

- **Classification Prompt:** Only a prompt to "Classify the sentiment of a review as either 'negative', 'neutral', or 'positive'." It had one input field 'review' described as "The review text to classify." and one output field 'classification' described as "One word representing the sentiment classification: 'negative', 'neutral', or 'positive' (do not repeat the field name, do not use 'mixed')" (See Figure 1 in Appendix B).
- **Predicted Label:** The classification prompt with an additional input field 'classifier_decision' described as "The sentiment classification proposed by a model fine-tuned on sentiment." During evaluation, the DSPy module first sends the review through the ELECTRA model to obtain it's classification decision. Then this output is inserted here (See Figure 2 in Appendix B).
- **Probabilities:** The classification prompt, but instead of 'classifier_decision' it featured three input fields for the probabilities of each class as obtained from the ELECTRA model. For example: 'negative_probability' was described as "Probability the review is negative from a model fine-tuned on sentiment". The float is converted to a percent to make it easier for the model to interpret (See Figure 3 in Appendix B).
- **Prediction & Probabilities:** This was the same as Probabilities but it also included the 'classifier_decision' to emphasize the final decision made by ELECTRA (See Figure 4 in Appendix B).
- **Top Examples:** A custom retriever was created to do cosine similarity matches of the input text against 300 example reviews from the Validation split. At run time, the input text was run through the fine-tuned ELECTRA Large model to extract the output representations (prior to the classifier head). The top five matches were retrieved with their true class labels and shown as few-shot examples in this template. This signature had an 'examples' field described as "A list of examples that demonstrate different sentiment classes." (See Figure 5 in Appendix B)
- **Balanced Examples:** Similar to Top Examples, except a different retriever was used that retrieved a total of six examples (two from each class) to ensure the few-shot examples with true labels did not bias the answer toward a particular class—although that might be desirable (See Figure 6 in Appendix B).
- **All of the Above:** And lastly, a final DSPy signature had all of the above context from ELECTRA included: classification prompt, predicted label, probabilities, and top five examples—not balanced. (See Figure 7 in Appendix B).

5.4 Experimental Runs

Our experiments are organized in six stages that established baselines, assessed fine-tuned model performance, and evaluated collaboration between ELECTRA and GPT using the DSPy signatures/modules. Details are in Table 11:

1. **Baseline Establishment** (4 runs)
   - **B1-B2:** ELECTRA Base/Large with classifier heads only
   - **B3-B4:** GPT-4o/4o-mini with Prompt only (Zero shot)
2. **ELECTRA Fine-tuned** (2 runs)
   - **E1-E2:** ELECTRA Base FT/Large FT: Fine-tuned all layers
3. **GPT-4o-mini Collaboration** (7 runs)
   - **E3:** GPT-4o-mini with ELECTRA Base FT: Prompt w/Label
   - **E4-E9:** GPT-4o-mini with ELECTRA Large FT: Prompt w/Label, Examples, Balanced Examples, Probabilities, Label & Probabilities, All Combined
4. **Fine-tuned GPT-4o-mini Evaluation** (4 runs)
   - **E10:** GPT-4o-mini FT (Prompt fine-tuning)
   - **E11:** GPT-4o-mini FT-M (Minimal format fine-tuning)
   - **E12:** GPT-4o-mini FT (Prompt fine-tuning) with ELECTRA Base FT predicted labels
   - **E13:** GPT-4o-mini FT-L (Prompt w/Label fine-tuning) with ELECTRA Base FT predicted labels

Table 11: Summary of Model Configurations, Performance, and Cost

				Merged[2]		DynaSent R1		DynaSent R2		SST-3		Cost ($)[3]	
ID	GPT[1]	ELECTRA	Description	F1	Acc	F1	Acc	F1	Acc	F1	Acc	FT	/F1
B1	—	Base	Baseline, Classifier head	69.65	69.83	70.96	71.28	61.59	61.67	60.85	70.14	0.65	0.01
B2	—	Large	Baseline, Classifier head	67.88	68.06	69.72	70.06	59.78	59.72	57.68	67.51	2.51	0.04
B3	4o-mini	—	Baseline (Prompt only, Zero shot)	†79.52	80.34	81.12	81.00	77.35	77.92	70.67	80.05	—	—
B4	4o	—	Baseline (Prompt only, Zero shot)	80.14	80.74	81.12	80.94	80.22	80.56	72.20	80.45	—	—
E1	—	Base FT	Fine-tune all layers	†79.29	79.69	82.10	82.14	71.83	71.94	69.95	78.24	**9.73**	**0.12**
E2	—	Large FT	Fine-tune all layers	82.36	82.96	85.91	85.83	76.29	76.53	70.90	80.36	53.26	0.65
E3	4o-mini	Base FT	Prompt, Label	†82.74	83.35	86.50	86.44	76.19	76.53	71.72	80.54	**9.73**	**0.12**
E4	4o-mini	Large FT	Prompt, Label	83.49	84.21	87.52	87.47	77.94	78.47	70.99	80.77	53.26	0.64
E5	4o-mini	Large FT	Prompt, Examples (Few shot)	83.20	83.80	86.74	86.64	78.71	79.03	71.98	80.72	53.26	0.64
E6	4o-mini	Large FT	Prompt, Balanced Ex. (Few shot)	82.68	83.28	85.84	85.69	79.61	80.00	71.45	80.41	53.26	0.64
E7	4o-mini	Large FT	Prompt, Probabilities	83.01	83.60	86.44	86.36	78.92	79.17	71.53	80.54	53.26	0.64
E8	4o-mini	Large FT	Prompt, Label, Probabilities	83.43	84.12	87.02	86.94	79.72	80.14	71.03	80.81	53.26	0.64
E9	4o-mini	Large FT	Prompt, Label, Probs, Examples	82.91	83.54	86.15	86.06	78.69	79.03	71.49	80.90	53.26	0.64
E10	4o-mini FT	—	Fine-tune w/prompt	86.77	87.26	89.86	89.75	86.90	87.08	**75.68**	**83.26**	33.15	0.38
E11	4o-mini FT-M	—	Minimal fine-tune	86.55	87.00	89.70	89.58	86.97	87.08	75.62	82.76	16.60	0.19
E12	4o-mini FT	Base FT	Prompt, Label, FT w/prompt	81.42	81.90	84.77	84.78	74.49	74.72	70.95	79.55	42.88	0.53
E13	4o-mini FT-L	Base FT	Prompt, Label, FT w/prompt, label	82.02	82.53	85.24	85.11	78.15	78.47	71.68	79.64	49.31	0.60
E14	4o	Large FT	Prompt, Label	81.57	81.98	84.03	83.86	77.69	77.92	72.94	80.23	53.26	0.65
E15	4o	Large FT	Prompt, Examples (Few shot)	83.09	83.66	86.01	85.86	81.53	81.81	72.06	80.68	53.26	0.64
E16	4o	Large FT	Prompt, Balanced Ex. (Few shot)	82.99	83.55	85.87	85.69	80.89	81.11	72.15	80.86	53.26	0.64
E17	4o	Large FT	Prompt, Probabilities	82.65	83.25	86.05	85.97	77.67	77.92	71.16	80.54	53.26	0.64
E18	4o	Large FT	Prompt, Label, Probabilities	83.08	83.71	86.44	86.33	79.23	79.58	71.48	80.77	53.26	0.64
E19	4o	Large FT	Prompt, Label, Probs, Examples	82.74	83.35	86.32	86.22	77.76	78.06	70.98	80.41	53.26	0.64
E20	4o FT	—	Fine-tune w/prompt	86.83	87.43	90.44	90.36	88.34	88.47	73.08	82.31	276.24	3.18
E21	4o FT-M	—	Minimal fine-tune	**86.99**	**87.57**	**90.57**	**90.50**	**89.00**	**89.17**	73.99	82.26	138.37	1.59
E22	4o FT	Large FT	Prompt, Label, FT w/prompt	83.82	84.47	87.80	87.72	79.49	79.86	71.18	80.68	329.50	3.93
E23	4o FT-L	Large FT	Prompt, Label, FT w/prompt, label	84.23	84.82	87.74	87.64	80.55	80.83	72.63	81.54	383.10	4.55

Bold = best overall, highlighted = best in section

†Scores relevant to Hypothesis 1 (ELECTRA prediction improving non-fine-tuned GPT performance)
[1] GPT fine-tuning types: FT = fine-tune all layers with prompt, FT-M = minimal format without prompt, FT-L = with prompt including ELECTRA label
[2] Merged dataset: Combination of test splits from DynaSent R1/R2 and SST-3
[3] Cost: FT = Fine-tuning cost, no inference-time API charges. Ratio is FT cost divided by F1 score.

5. **GPT-4o Collaboration** (6 runs)

    - **E14-E19:** GPT-4o with ELECTRA Large FT: Prompt w/Label, Examples, Balanced Examples, Probabilities, Label & Probabilities, All Combined

6. **Fine-tuned GPT-4o Evaluation** (4 runs)

    - **E20:** GPT-4o FT (Prompt fine-tuning)
    - **E21:** GPT-4o FT-M (Minimal format fine-tuning)
    - **E22:** GPT-4o FT (Prompt fine-tuning) with ELECTRA Large FT predicted labels
    - **E23:** GPT-4o FT-L (Prompt w/Label fine-tuning) with ELECTRA Large FT predicted labels

5.5 Metrics

For each run, we recorded predictions, computed performance metrics, and conducted statistical significance tests if necessary for our hypotheses.

For this type of sentiment analysis, we are trying to distinguish between positive, negative and neutral reviews. Depending on the business application, it may be more important to detect the negative or the positive reviews.

Absent this knowledge, we will treat each class as equally important. Therefore we used the Macro-averaged F1 score (the mean of the per-class F1 scores) as the ultimate measure of classification performance.

In addition to the Macro-averaged F1, we also recorded Accuracy to provide an additional perspective on overall classifier performance. However, the Macro F1 was the metric used to evaluate our hypotheses.

6 Results

Our experiments revealed significant differences in performance across baseline, fine-tuning, and collaborative scenarios. See Table 11 for the details.

6.1 Baselines

Regarding the baselines, both GPT models outperformed the basic ELECTRA classifiers, with GPT-4o achieving 80.14 macro F1 and GPT-4o-mini scoring 79.52, compared to ELECTRA Base

(69.65) and Large (67.88). This gap demonstrates the strong zero-shot capabilities of the GPT models on sentiment tasks.

6.2 Fine-tuning

Fine-tuning improved performance across all models. ELECTRA Base's macro F1 increased from 69.65 to 79.29, while ELECTRA Large showed even greater gains, improving from 67.88 to 82.36. This difference is the result of fine-tuning all layers—the baselines had the same classifier head. The GPT models saw the largest improvements, with GPT-4o-mini rising from 79.52 to 86.77 using prompt-based fine-tuning, and GPT-4o achieving 86.99 with minimal format fine-tuning.

6.3 Collaboration

The impact of adding ELECTRA predictions to GPT prompts varied based on whether the GPT model was fine-tuned. For the non-fine-tuned GPT-4o-mini, including ELECTRA Base predictions significantly improved the macro F1 from 79.52 to 82.74 ($p < 0.0001$, McNemar's test). Similarly, non-fine-tuned GPT-4o improved from 80.14 to 81.57 with ELECTRA Large predictions ($p = 0.0106$).

We explored different formats for sharing ELECTRA's predictions with GPT-4o-mini. Using just the predicted label achieved 83.49 macro F1, while showing probabilities for each class reached 83.01. Combining both approaches (label and probabilities) yielded 83.43. For GPT-4o, the effects were similar: label-only achieved 81.57, probabilities 82.65, and the combination 83.08.

The addition of retrieved examples showed varying effects. For GPT-4o-mini with ELECTRA Large predictions, including the top five most similar examples yielded 83.20 macro F1, while using balanced retrieval (two examples from each sentiment class) achieved 82.68. GPT-4o showed slightly different patterns with example inclusion: top examples improved performance to 83.09, while balanced examples reached 82.99.

However, when applied to fine-tuned GPT models, these collaborative approaches actually decreased performance. GPT-4o-mini's performance dropped from 86.77 to 81.42 when given ELECTRA predictions, and to 82.02 when fine-tuned with the predictions included in training. Similarly, GPT-4o fell from 86.99 to 83.82 with predictions, and to 84.23 when trained with them.

Performance varied notably across datasets. GPT-4o FT achieved the highest scores on both DynaSent rounds, reaching 90.57 macro F1 on Round 1 and 89.00 on Round 2. Surprisingly, GPT-4o-mini FT performed best on SST-3 with a 75.68 macro F1, exceeding even GPT-4o FT's performance of 73.99.

The cost efficiency of different approaches showed substantial variation. The most cost-effective approach was the combination of ELECTRA Base FT with non-fine-tuned GPT-4o-mini at $0.12 per F1 point. GPT-4o-mini FT provided a good compromise at $0.38 per F1 point, while GPT-4o FT with ELECTRA predictions proved most expensive at $4.55 per F1 point.

7 Analysis

Our results provide interesting insight into the dynamics of sentiment classification with transformer models.

7.1 H1: Including ELECTRA Predictions Would Boost Performance

The significant improvement in GPT-4o-mini's performance when augmented with ELECTRA Base predictions (improving from 79.52 to 82.74 macro F1, $p < 0.0001$) strongly supports H1. The collaborative approach performed better than either model alone, and yielded the lowest cost/performance ratio ($0.12/F1 point).

7.2 H2: Improvement Would be Less for Fine-tuned GPTs

H2 was supported more strongly than anticipated. For a fine-tuned GPT model, including the ELECTRA prediction actually decreased performance. Initially, we thought this was because the fine-tuning context did not include the ELECTRA prediction in the prompt. But we saw the same decrease in performance when we fine-tuned the GPT models on prompts that included the actual ELECTRA predictions. Clearly they ended up being distractions.

7.3 H3: Format of Prediction Would Impact Performance

Regarding H3, the best overall score for GPT-4o-mini in collaboration with ELECTRA Large FT was 83.49 by just including the predicted labels. This was true for the Merged dataset and DynaSent R1. However, including the probabilities in addition to the labels yielded the best score for DynaSent R2 (79.72). For GPT-4o-mini in collabo-

ration with ELECTRA Large FT, including probabilities yielded the best score for DynaSent R2 (86.44). This suggests the probabilities were able to help the models make better decisions with the more challenging datasets.

7.4 H4: Including Examples Would Improve Performance

Regarding H4, for GPT-4o-mini in collaboration with ELECTRA Large FT, including examples without the prediction yielded the best score for SST-3 (71.98). For GPT-4o-mini in collaboration with ELECTRA Large FT, including examples without the prediction yielded the best score for DynaSent R2 (81.53). Similar to including probabilities, it appears the examples were able to help the models make better decisions with these datasets.

7.5 Cost-Performance Trade-offs

The results present clear implications for deployment scenarios with different resource constraints. While GPT-4o FT achieved the highest overall performance at 86.99 macro F1, its cost ($1.59/F1) may be prohibitive for many applications. GPT-4o-mini FT offers a compelling alternative, achieving nearly equivalent performance (86.77 F1) at a much lower cost ($0.38/F1). For scenarios where fine-tuning isn't feasible due to cost or privacy concerns, the combination of ELECTRA Base and GPT-4o-mini provides strong performance (82.74 F1) at minimal cost ($0.12/F1).

7.6 Dataset-Specific Challenges

Performance variations across datasets reveal their distinct challenges. While models achieved their highest scores on DynaSent R1 (up to 90.57 F1), suggesting more straightforward sentiment patterns, SST-3 proved consistently more challenging (maximum 75.68 F1). This likely reflects SST-3's more nuanced expressions of sentiment.

8 Conclusion

This paper investigated collaborative approaches to sentiment classification between bidirectional transformers (ELECTRA Base/Large) and autoregressive LLMs (GPT-4o/4o-mini). Our results show that such collaboration can significantly improve performance when using non-fine-tuned GPT models, achieving up to 3.22 points of gain in the macro F1 score ($p < 0.0001$). This was also the lowest cost/performance ratio—or the "best bang for the buck." However, this benefit disappeared once the GPT models were fine-tuned.

For organizations that are unable to fine-tune GPT models due to data privacy concerns or resource limitations, augmenting prompts with predictions from fine-tuned local models is a viable option. When fine-tuning is possible, we found that GPT-4o-mini was surprisingly capable and cost-effective—achieving nearly the same performance as GPT-4o at roughly a quarter of the cost.

Known Project Limitations

Due to resource and time constraints, not every possible combination of ELECTRA model, GPT model, fine-tuning state, and prompt context was explored. Once we saw ELECTRA Large had the best performance, we shared the ELECTRA Large predictions with both GPT-4o and GPT-4o-mini to evaluate the different input contexts.

Many experimental runs involved prompting the GPT models, and these prompts could be further optimized with frameworks like DSPy. Some of the prompts are a bit wordy, and these additional tokens could reduce the signal of the ELECTRA context.

We tried to give the GPT models a chance to learn the ELECTRA predictions during fine-tuning (GPT-4o FT-L, GPT-4o-mini FT-L). We included the prediction in the prompt. But there might be a more direct way of training GPT so that it learns when to trust ELECTRA's prediction, and when to decide for itself.

Authorship Statement

This paper is the result of the individual work of the author, James P. Beno, for the Natural Language Understanding course (XCS224U) from July to November 2024. This course is part of the Artificial Intelligence Professional Program offered by the Stanford Engineering Center for Global & Online Education. Anthropic Claude was used for content editing and LaTeX formatting.

References

Claudio M. V. de Andrade, Washington Cunha, Davi Reis, et al. 2024. A strategy to combine 1st gen transformers and open LLMs for automatic text classification. *arXiv preprint arXiv:2408.09629*.

Mala J B, Anisha Angel S J, Alex Raj S M, and Rajeev Rajan. 2023. Efficacy of ELECTRA-based language

model in sentiment analysis. In *2023 International Conference on Intelligent Systems for Communication, IoT and Security (ICISCoIS)*, pages 682–687.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan A, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2024. DSPy: Compiling declarative language model calls into state-of-the-art pipelines. In *The Twelfth International Conference on Learning Representations*.

Kiana Kheiri and Hamid Karimi. 2023. SentimentGPT: Exploiting GPT for advanced sentiment analysis and its departure from current machine learning.

Jan Kocoń, Ireneusz Cichecki, Oliwier Kaszyca, et al. 2023. ChatGPT: Jack of all trades, master of none. *Information Fusion*, 99:101861.

Brice Valentin Kok-Shun, Johnny Chan, Gabrielle Peko, and David Sundaram. 2023. Intertwining two artificial minds: Chaining gpt and roberta for emotion detection. In *2023 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, pages 1–6.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, et al. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

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

Walaa Medhat, Ahmed Hassan, and Hoda Korashy. 2014. Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, 5(4):1093–1113.

OpenAI. 2024a. Gpt-4 technical report.

OpenAI. 2024b. Hello GPT-4o.

OpenAI. 2024c. GPT-4o mini: advancing cost-efficient intelligence.

Christopher Potts, Zhengxuan Wu, Atticus Geiger, and Douwe Kiela. 2021. DynaSent: A dynamic benchmark for sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2388–2404, Online. Association for Computational Linguistics.

Tyreal Yizhou Qian, Weizhe Li, Hua Gong, Chad Seifried, and Chenglong Xu. 2024. Experience is all you need: a large language model application of fine-tuned gpt-3.5 and roberta for aspect-based sentiment analysis of college football stadium reviews. *Sport Management Review*, 0(0):1–25.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI Blog*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*.

Noam Shazeer. 2020. Glu variants improve transformer.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

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.

Mengzhou Xia, Mikel Artetxe, Jingfei Du, Danqi Chen, and Veselin Stoyanov. 2022. Prompting ELECTRA: Few-shot learning with discriminative pre-trained models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11351–11361. Association for Computational Linguistics.

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

Boyu Zhang, Hongyang Yang, Tianyu Zhou, Muhammad Ali Babar, and Xiao-Yang Liu. 2023. Enhancing financial sentiment analysis via retrieval augmented large language models. In *Proceedings of the Fourth ACM International Conference on AI in Finance*, ICAIF '23, page 349–356, New York, NY, USA. Association for Computing Machinery.

A ELECTRA Fine-tuning Details

Table 12: ELECTRA Fine-Tune Configuration

Setting	ELECTRA Base FT	ELECTRA Large FT
Source	Hugging Face	Hugging Face
Source Model ID	google/electra-base-discriminator	google/electra-large-discriminator
Encoder Blocks	12	24
Embedding Dimension	768	1024
Attention Heads	12	16
Feedforward Size	3072	4096
Parameters	110 Million	335 Million
Custom Pooling Layer Method	Mean	Mean
Classifier Head Hidden Layers	2	2
Classifier Head Hidden Dimension	1024	1024
Classifier Head Hidden Activation	SwishGLU	SwishGLU
Finetuned Encoder Blocks	12	24
Total Layers	104	200
Total Parameters	112,830,979	338,293,763
Trainable Parameters	100%	100%
Learning Rate	$1e^{-5}$	$1e^{-5}$
Learning Rate Decay	0.95	0.95
Batch Size	16	16
Accumulation Steps	2	2
Target Epochs	50	50
Actual Epochs	20	23
Selected Best Epoch	14	13
Dropout Rate	0.30	0.30
L2 Strength	0.01	0.01
Optimizer	AdamW	AdamW
Zero Redundancy	Yes	Yes
Scheduler	CosineAnnealingWarmRestarts	CosineAnnealingWarmRestarts
Scheduler: T_0	5	5
Scheduler: T_mult	1	1
Scheduler: eta_min	$1e^{-7}$	$1e^{-7}$
Early Stop	Validation F1 Score	Validation F1 Score
N Iterations No Change	10	10
Dataset	Merged (Dyn R1, Dyn R2, SST-3)	Merged (Dyn R1, Dyn R2, SST-3)
Train Size	102,097	102,097
Train Label Distribution	Neu: 49,148, Pos: 31,039, Neg: 21,910	Neu: 49,148, Pos: 31,039, Neg: 21,910
Validation Size	5,421	5,421
Validation Label Distribution	Neu: 1,669, Pos: 1,884, Neg: 1,868	Neu: 1,669, Pos: 1,884, Neg: 1,868
Hosting Provider	Lambda Labs	Lambda Labs
GPU Type	Tesla V100	A100
GPU Memory	16 GB	40 GB
GPU Quantity	8	8
Rate	$4.40/hour	$10.32/hour
Training Time (Up to Selected Epoch)	02:12:44	05:09:23
Training Time (Total)	03:09:40	09:23:29
Cost (Up to Selected Epoch)	$9.73	$53.26
Cost (Total)	$13.91	$96.92

B DSPy Prompt Signature Examples

Figure 1: Basic Prompt DSPy Signature

```
Classify the sentiment of a review as either 'negative', 'neutral', or 'positive'.

---

Follow the following format.

Review: The review text to classify.
Classification: One word representing the sentiment classification: 'negative', 'neutral', or 'positive'
(do not repeat the field name, do not use 'mixed').

---

Review: Those 2 drinks are part of the HK culture and has years of history. It is so bad.
Classification:
```

Figure 2: Prompt with Predicted Label DSPy Signature

```
Classify the sentiment of a review as either 'negative', 'neutral', or 'positive'.

---

Follow the following format.

Review: The review text to classify.
Classifier Decision: The sentiment classification proposed by a model fine-tuned on sentiment.
Classification: One word representing the sentiment classification: 'negative', 'neutral', or 'positive'
(do not repeat the field name, do not use 'mixed')

---

Review: I was told by the repair company that was doing the car repair that fixing the rim was
"impossible" and to replace it.
Classifier Decision: negative
Classification:
```

Figure 3: Prompt with Probabilities DSPy Signature

```
Classify the sentiment of a review as either 'negative', 'neutral', or 'positive'.

---

Follow the following format.

Review: The review text to classify.

Negative Probability: Probability the review is negative from a model fine-tuned on sentiment

Neutral Probability: Probability the review is neutral from a model fine-tuned on sentiment

Positive Probability: Probability the review is positive from a model fine-tuned on sentiment

Classification: One word representing the sentiment classification: 'negative', 'neutral', or 'positive'
(do not repeat the field name, do not use 'mixed')

---

Review: Those 2 drinks are part of the HK culture and has years of history. It is so bad.

Negative Probability: 99.85%

Neutral Probability: 0.04%

Positive Probability: 0.12%

Classification:
```

Figure 4: Prompt with Predicted Label and Probabilities DSPy Signature

```
Classify the sentiment of a review as either 'negative', 'neutral', or 'positive'.

---

Follow the following format.

Review: The review text to classify.

Classifier Decision: The sentiment classification proposed by a model fine-tuned on sentiment.

Negative Probability: Probability the review is negative

Neutral Probability: Probability the review is neutral

Positive Probability: Probability the review is positive

Classification: One word representing the sentiment classification: 'negative', 'neutral', or 'positive'
(do not repeat the field name, do not use 'mixed')

---

Review: Those 2 drinks are part of the HK culture and has years of history. It is so bad.

Classifier Decision: negative

Negative Probability: 99.85%

Neutral Probability: 0.04%

Positive Probability: 0.12%

Classification:
```

Figure 5: Top Examples DSPy Signature

```
Classify the sentiment of a review as either 'negative', 'neutral', or 'positive'.

---

Follow the following format.

Examples: A list of examples that demonstrate different sentiment classes.

Review: The review text to classify.

Classifier Decision: The sentiment classification proposed by a model fine-tuned on sentiment.

Classification: One word representing the sentiment classification: 'negative', 'neutral', or 'positive'
(do not repeat the field name, do not use 'mixed')

---

Examples:
- negative: We've been to about 5 or 6 other Verizon stores in Vegas, and they all give us a hard time
about everything and never solve any issue.
- negative: Then Raj then had the balls to send me an email after my box was closed to tell me they were
ready to receive the key for my mailbox after closing it.!
- negative: Always and issue here even with take out orders.
- negative: SHOULD YOU HAVE ANY DISPUTE, THEY IMMEDIATELY WILL THREATEN YOU WITH MECHANICS LIENS.
- negative: We were waiting for them to get our order out, but the lady came out and gave the car behind
us their order first!

Review: I went back in to ask for cilantro dressing the shift leader even smile or greet me.

Classifier Decision: negative

Classification:
```

Figure 6: Balanced Examples DSPy Signature

```
Classify the sentiment of a review as either 'negative', 'neutral', or 'positive'.

---

Follow the following format.

Examples: A list of examples that demonstrate different sentiment classes.

Review: The review text to classify.

Classifier Decision: The sentiment classification proposed by a model fine-tuned on sentiment.

Classification: One word representing the sentiment classification: 'negative', 'neutral', or 'positive'
(do not repeat the field name, do not use 'mixed')

---

Examples:
- negative: Beware of all the fake 5 star reviews of this place, just take a look at these people.
- negative: 3- girls look even cheaper than the club.
- neutral: Not to mention the esso across the street also has cheaper gas.
- neutral: I wish that they would open up by 6am so that I can pick up a coffee or tea before work, but
what boba place is opened that early?
- positive: The plumbers did not give up and continued to work on the drain for two days.
- positive: This is my 6th gun to add to my collection and if I had not wanted it so bad, I would have
walked out 2 minutes after walking in.

Review: She greeted customers by holding the scanner toward them without even looking.

Classifier Decision: negative

Classification:
```

Figure 7: All Context DSPy Signature

```
Classify the sentiment of a review as either 'negative', 'neutral', or 'positive'.

---

Follow the following format.

Examples: A list of examples that demonstrate different sentiment classes.

Review: The review text to classify.

Classifier Decision: The sentiment classification proposed by a model fine-tuned on sentiment.

Negative Probability: Probability the review is negative

Neutral Probability: Probability the review is neutral

Positive Probability: Probability the review is positive

Classification: One word representing the sentiment classification: 'negative', 'neutral', or 'positive'
(do not repeat the field name, do not use 'mixed')

---

Examples:
- negative: The only negative I can think for this place is it's price-point.
- positive: This place will be the death of my waist (but not my wallet).
- negative: Expensive, if you are looking for something more affordable, don't go here; you will miss
  the best dishes.
- positive: Thank you so much for dealing with my crabby ass
- positive: I think I scarfed it down so quickly because it was that good! It was bad.

Review: The gentleman staffing the bar seemed a bit gruff, but a good caffeine fix will help me forgive
even the orneriest grump.

Classifier Decision: negative

Negative Probability: 84.37%

Neutral Probability: 0.53%

Positive Probability: 15.10%

Classification:
```