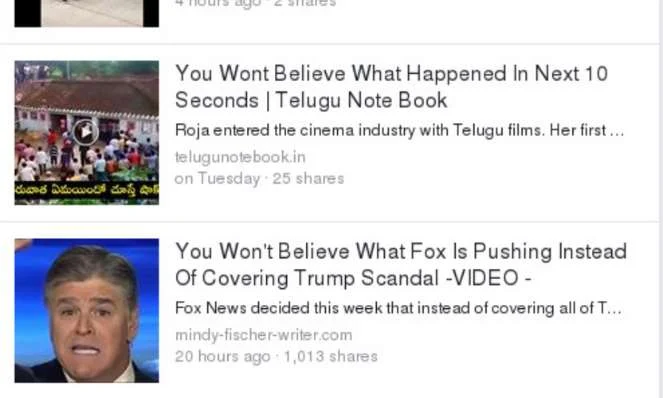
**N.L.P - Final project- "Bait Buster"**

Background

Clickbait is a style of writing that is specifically designed to grab the reader’s attention and make them click on a link or headline. It often uses dramatic, intriguing, or exaggerated language—such as open-ended questions, outlandish promises, or surprising-sounding information—even if the actual content is disappointing or different from what was promised. The goal is to increase clicks, not necessarily to provide quality information.

The distinction between fake news and clickbait is not always clear. Fake news can also be identified based on the overall context and coordination with reliable news sources. In contrast, clickbait can be identified by its unique writing style, which is designed to arouse immediate curiosity. However, there are cases where clickbait is integrated into news content. Our task is to identify clickbait based on the writing style – with some clickbait also containing false content.

Meaning of the project name: The name Bait Buster is a combination of two words: Bait (as in clickbait) and Buster (meaning breaker, fighter, or eliminator). The name reflects the project's main goal, identifying and countering headlines that serve as "bait" to lure readers into clicking, regardless of the factual content. The name is especially fitting because this project focuses not only on recognizing fake news, but more specifically on detecting stylistic patterns that are characteristic of clickbait — such as exaggerated language, curiosity-inducing questions, and emotionally charged phrasing. *Bait Buster* conveys precisely this idea: a tool designed to "bust" or expose manipulative headline styles that aim to capture user attention through clickbait techniques.



Dataset

**Dataset Generation**: we create the dataset using a python script we wrote. First, real news headlines are loaded from a CSV file called news\_data.csv, which contains original headlines that are not clickbait. Then, ten common clickbait methods are defined: Curiosity Gap, Exaggeration, Emotional Triggers, Sensationalism, Lists/Superlatives, Ambiguous References, Direct Appeals, Unfinished Narratives, Unexpected Associations, and Provocative Questions. For each real headline, several methods are randomly selected, and GPT is used to create a clickbait version while maintaining the factual content and changing only the style. This builds a rich dataset that includes source headlines, their clickbait versions, and information about which methods were used to create each one. The data is divided into two tasks: one – identifying whether a headline is clickbait or not, and the second – identifying which stylistic methods appear in each clickbait headline.

Examples of ten common clickbait methods:

1. Curiosity Gap

Description: Tease the reader with incomplete information, compelling them to click to find out more.

Example: "You Won’t Believe What This AI Just Discovered!"

1. Listicles

Description: Use numbered lists to promise a quick, digestible set of insights or tips.

Example: "10 Shocking Ways AI Is Changing Your Life"

1. Fear of Missing Out (FOMO)

Description: Create urgency or suggest that the reader might miss out on important information.

Example: "Are You Missing Out on the Latest AI Breakthrough?"

1. Shock and Awe

Description: Use dramatic language or surprising facts to grab attention.

Example: "The One AI Mistake That Cost a Company $1 Million!"

1. How-To Promises

Description: Promise easy solutions or valuable knowledge using a how-to format.

Example: "How to Master AI-Powered Marketing in Just 7 Days"

1. Controversial Statements

Description: Make a bold, potentially polarizing claim to spark curiosity or debate.

Example: "Why AI Might Be the Worst Thing to Happen to Humanity"

1. Celebrity or Authority Leverage

Description: Mention a famous person or authoritative figure to boost credibility and interest.

Example: "Elon Musk’s Shocking Prediction About AI’s Future"

1. Exclusive Insider Knowledge

Description: Offer hidden or insider information that readers feel privileged to access.

Example: "Inside Secrets of AI Startups That Investors Won’t Tell You"

1. Emotional Trigger

Description: Evoke strong emotions such as anger, happiness, or sadness to encourage clicks.

Example: "This Heartwarming AI Story Will Restore Your Faith in Technology"

1. Unbelievable Comparisons

Description: Use extreme or unusual comparisons to draw attention.

Example: "This AI Is Smarter Than a 5-Year-Old—And It Just Got Smarter!

Evaluation

**One-step pipeline**

In this one-step pipeline, a single large language model prompt handles both **clickbait detection** (binary: clickbait vs. non-clickbait) and **tactic attribution** (identifying which of the ten clickbait tactics appear, or labeling “non-clickbait” when none apply). Rather than separating detection and attribution into two distinct models, we prompt the LLM to produce a joint multi-label output: if a headline is clickbait, the model lists all applicable tactics; if not, it outputs “non-clickbait.” Four LLM configurations were evaluated—GPT and Gemini, each in zero-shot and few-shot prompt modes—using a balanced dataset of 1,740 headlines (870 clickbait, 870 non-clickbait).

Joint LLM Model

**Data Preparation**

* Total headlines: 1,740 (870 clickbait, 870 non-clickbait).
* Each clickbait headline is annotated with one or more of ten possible tactics (e.g., Curiosity Gap, Emotional Triggers, Sensationalism, etc.).
* Non-clickbait headlines carry a single “non-clickbait” label.

**Model & Prompt Details**  
Using the models GPT-4o mini by OpenAI and Gemini 2.5 Flash by Google DeepMind, we evaluated four configurations:

1. **GPT Zero-Shot**  
   Each prompt followed the structure bellow:
   1. A system prompt:  
      “You are an expert in detecting and analyzing clickbait. You have access to the following list of clickbait creation methods and their descriptions: {methods and descriptions in JSON form} “
   2. The task:  
      “Given the headline: ‘{HEADLINE}’  
      1. Is this clickbait? Answer "Yes" or "No".  
      2. If "Yes", list the names of the clickbait tactics used, comma separated.  
      3. If "No", list None. “
2. **GPT 2-Shot**
   1. The same instruction above, followed by 2 random examples (uniformly chosen either 2 clickbait, 2 non- clickbait or one of each).
   2. Each example was of the form:  
      “Example:   
      Headline: ‘{EXAMPLE\_HEADLINE}'   
      1. {ANSWER}   
      2. {TACTICS} “
3. **Gemini Zero-Shot & 2-Shot**  
   Identical instruction wording, run on Google’s Gemini model.

**Note**: Neither GPT nor Gemini was fine-tuned. All inference occurred on the full 1,740-headline set—no separate train/test split for the LLMs themselves.

Inference

* Each headline (counting both the source and the clickbait versions separately) was passed through each configuration’s prompt (zero/few examples).
* The LLM returns “Yes” or “No” for clickbait, plus a comma-separated list of tactics, or “None”.
* We compare its output to the ground truth multi-label annotation, after label binarization (multi-label binary vector of the methods used / predicted. The zero vector stands for “None”).

Evaluation & Results

We computed:

* **Micro-Precision / Micro-Recall / Micro-F1**: Aggregates true/false positives and negatives over *all* label decisions (across all 1,740 examples and 11 possible labels).
* **Macro-Precision / Macro-Recall / Macro-F1**: Computes precision/recall/F1 for each label individually (ten tactics + “non-clickbait”), then averages these scores.

The following is a summary of each model’s performance in both the multi-label task (table) and the binary task (bullets):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Configuration** | **Micro-Precision** | **Micro-Recall** | **Micro-F1** | **Macro-Precision** | **Macro-Recall** | **Macro-F1** |
| GPT Zero-Shot | 0.3133 | 0.5547 | 0.4004 | 0.4089 | 0.5545 | 0.3698 |
| GPT 2-Shot | 0.3645 | 0.5351 | 0.4337 | 0.4610 | 0.5354 | 0.3984 |
| Gemini Zero-Shot | 0.3883 | 0.6092 | 0.4743 | 0.4470 | 0.6069 | 0.4606 |
| Gemini 2-Shot | 0.4002 | 0.5578 | 0.4660 | 0.4504 | 0.5571 | 0.4512 |

* **GPT Zero-Shot**:
  + True Positive (clickbait): 870 / 870
  + False Negative (clickbait → non-clickbait): 0
  + False Positive (non-clickbait → clickbait): 454
  + True Negative (non-clickbait): 416
* **GPT 2-Shot**:
  + TP: 867 / 870 · FN: 3
  + FP: 264 · TN: 606
* **Gemini Zero-Shot**:
  + TP: 870 / 870 · FN: 0
  + FP: 289 · TN: 581
* **Gemini 2-Shot**:
  + TP: 869 / 870 · FN: 1
  + FP: 324 · TN: 546

These results show that with no prior (zero-shot), Gemini reduces false positives (relative to GPT), while maintaining the same result in true positives as GPT.  
However, with prior (few-shot) Gemini seems to falter in all rubrics, while GPT falls in clickbait detection, but minimizes false positives further than both Gemini configurations.

Analysis & Discussion

**Zero-Shot vs. Few-Shot**

1. **GPT:**
   1. **Few-Shot Gains**:
      1. Micro-Precision ↑ from 0.3133 → 0.3645
      2. Micro-F1 ↑ from 0.4004 → 0.4337
      3. Macro-Precision ↑ from 0.4089 → 0.4610
      4. Macro-F1 ↑ from 0.3698 → 0.3984
   2. **Trade-Off**: Micro-Recall ↓ slightly (0.5547 → 0.5351), indicating a few more missed tactics but fewer incorrect ones.
2. **Gemini**
   1. **Zero-Shot Strength**:
      1. Highest Micro-Recall (0.6092) and Macro-Recall (0.6069), meaning it catches the most true tactics—especially on rarer categories.
      2. Highest Micro-F1 (0.4743) and Macro-F1 (0.4606) overall.
   2. **Few-Shot Effect**:
      1. Precision ↑ (0.3883 → 0.4002 micro), but Recall ↓ (0.6092 → 0.5578 micro).
      2. Net micro-F1 dips slightly (0.4743 → 0.4660), and macro-F1 falls from 0.4606 → 0.4512.

**Precision vs. Recall Trade-Off**

* All configurations show **higher recall** than precision on rare tactics (macro-recall > macro-precision for each) - the LLMs tend to output more tactics than present, rather than missing them.
* **Gemini** offers a better balance in performance with and without prior (balanced macro F1 ≈ 0.46) compared to **GPT** (macro F1 ≈ 0.40 few-shot, ≈ 0.37 zero-shot).
* **GPT** benefits further from prior on this task.
* **Both** models err on the side of inclusion, though Gemini yielded fewer erroneous tactics than GPT (higher macro-recall).

5. Qualitative Observations

Although the LLMs can identify most relevant tactics, they still struggle to be conservative, especially on rare categories (e.g., “Unfinished Narratives,” “Ambiguous References”). For practical deployment, a downstream thresholding or post-filtering step could prune unlikely tactics, mimicking the per-label threshold approach used in the improved multi-label BERT model (two-step pipeline).

Conclusion

For a one-step approach—where a single prompt both detects clickbait and attributes tactics—**Gemini Zero-Shot** emerges as the strongest overall, achieving:

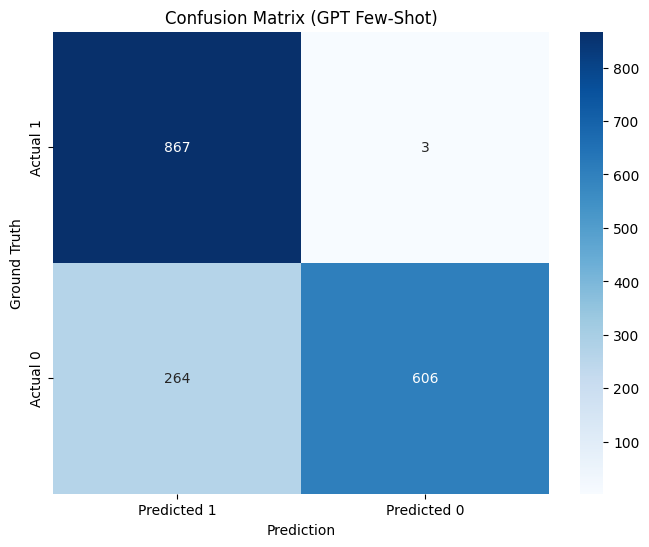
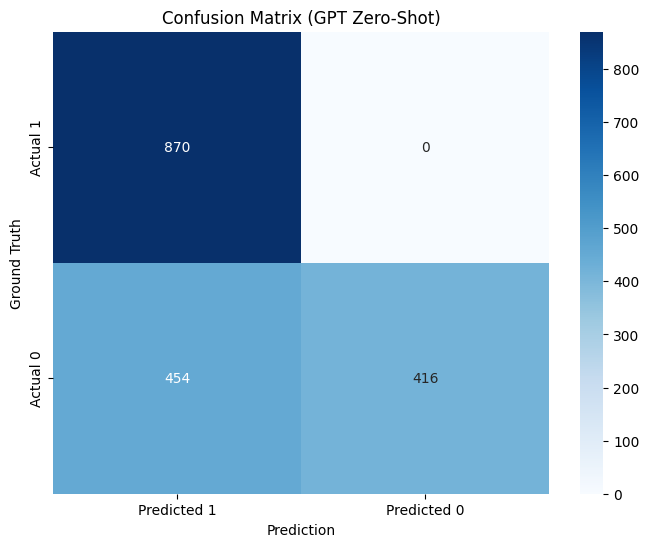
* **Micro-F1 = 0.4743** (highest among all)
* **Macro-F1 = 0.4606** (best balance across all tactics)

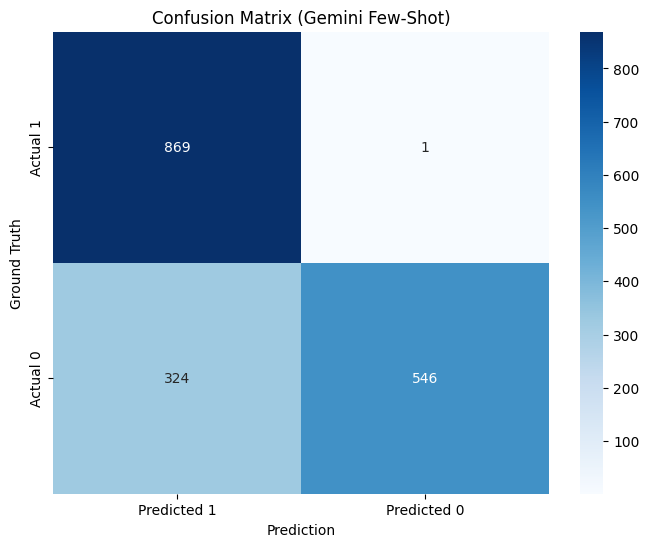
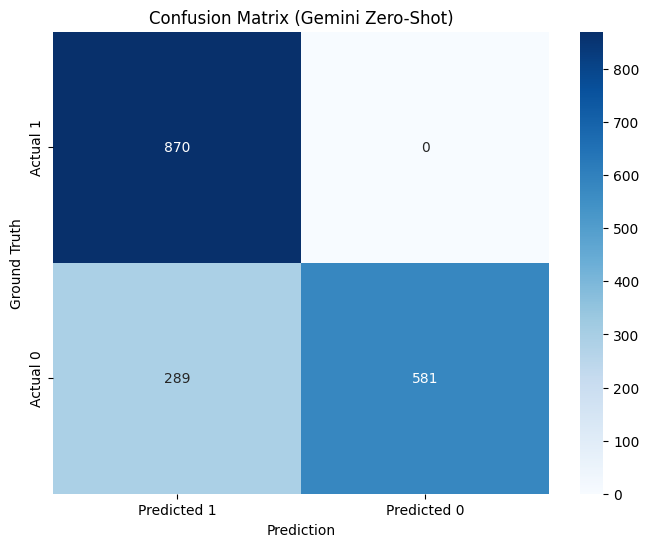
**Gemini Few-Shot** trades a small drop in F1 (0.4660 micro, 0.4512 macro) for marginally higher precision, while trending more towards “clickbait” predictions. **GPT Few-Shot** outperforms GPT Zero-Shot and both Gemini configurations in the binary detection task, but both GPT variants lag behind Gemini in the multi-label tactic attribution.

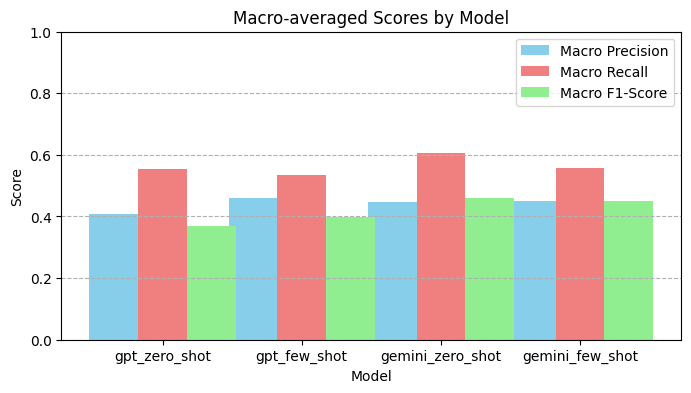
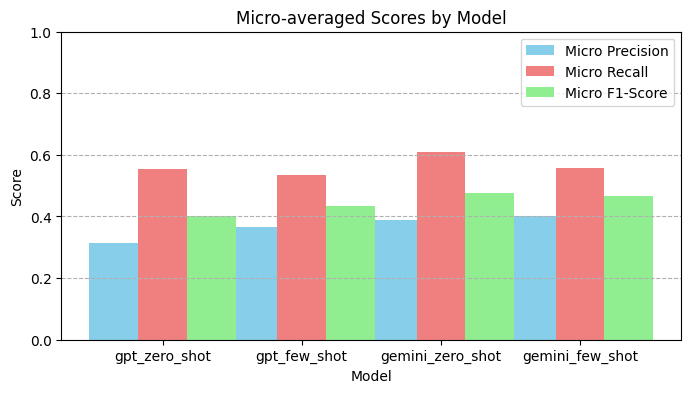
In essence, a one-step LLM pipeline can successfully handle joint detection and attribution, albeit at the cost of somewhat inflated false positives, and more so on rare tactics. Models may vary in their benefit from examples and their tendency to err on the side of inclusion, the latter taking different forms in binary detection verses multi-label.

[link to Google Colab where we ran the code for clickbait classification](https://colab.research.google.com/drive/1kdBH0gfOZDiholNAUkUYTZVEk_HdRga2?usp=sharing)

Model results in visual form







**Two-step pipeline**

At this stage, we will divide the tactics of attribution and clickbait detection models into two separate parts, which do not affect each other.

The term prompt describes the input we give to a language model (like GPT) to perform a particular task. It can be a sentence, a question, an instruction, or even an entire dialogue. Complex / few-shot prompt includes a few examples of input and output, to "show" the model how to perform the task. During the project, we will use different types of prompts to test our dataset and model.

Clickbait detection model: Fine-Tuned Bert

The code performs a complete process of training a BERT model to classify headlines as clickbait or not, using TensorFlow and HuggingFace. First, it installs all the necessary libraries, imports the relevant files from the user, and prepares the data by merging the `source` (regular headlines) and `clickbait` (clickbait headlines) columns into a single text column with a label (0 or 1). Then, it tokenizes the texts using `BertTokenizer`, divides the data into training and test sets, and builds a suitable BERT model for binary classification (`TFBertForSequenceClassification`). After compiling with a loss function and optimizer, the model is trained for three epochs. Finally, the code evaluates the model on the test set using predictions and prints a performance report including accuracy, precision, recall, and F1.

We trained a BERT-based classification model on a balanced dataset of 1,740 headlines – half clickbait (870) and half regular (870). The data was split into 80% for training (1,392 examples) and 20% for testing (348 examples). The balanced split between the groups allows for an accurate assessment of the model’s performance.

תמונה שמכילה טקסט, צילום מסך, קבלה, גופן

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

As can be seen, the model learned well from the data. There are no signs of overfitting (large difference between the lines), and the Validation Accuracy is even higher throughout – a sign of a general and robust model.

Also, in a confusion matrix showing the performance of a classification model for identifying clickbait headlines. The model is almost perfect. It correctly identified 173 out of 174 non-clickbait headlines. It correctly identified all 174 clickbait headlines. Only one mistake was made: one non-clickbait headline was mistakenly classified as clickbait. In other words: very high accuracy, with only one mistake out of 348 examples.

Below is the link to Google Colab where we ran the code for clickbait classification: <https://colab.research.google.com/drive/1AdURB7l3qE6Bkz4Vs9lKpRcPc2F5uJ0w?usp=sharing>

Model results in visual form

תמונה שמכילה טקסט, קו, תרשים, עלילה

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.תמונה שמכילה טקסט, צילום מסך, מספר, תרשים

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

Figure 1

Confusion Matrix 1

Clickbait detection model: GPT-4 with few-shot prompt

In the GPT-4-based clickbait detection model using few-shot prompting, we built a Prompt that included examples from the file `clickbait\_dataset.csv` – some clickbait headlines and some regular headlines – along with sample answers (`Answer: Yes/No`). We then added several headlines from the same file, and GPT-4 was required to classify each one. It is important to note that this is the same file that we also used to train the classic BERT model, thus maintaining consistency between the methods.

Results: a total of 16 queries (divided into several separate messages) were sent to the chatGPT 4o model. It can be said that the model was able to correctly classify the queries that were sent.

Below is a link to the chat conversation:

<https://chatgpt.com/share/68320acf-024c-800c-b57b-152b387cde69>

Tactics attribution model: Fine-tuned multilabel BERT

The code trains a multi-label BERT model to identify clickbait tactics from news headlines. It first loads the data file `clickbait\_dataset.csv`, which contains columns of clickbait headlines and tactic lists as vectors of 0 and 1. After converting the tactics to a list structure, the texts are tokenized using `BertTokenizer` to convert them to a numeric structure that matches the input of the BERT model. Next, a `TFBertForSequenceClassification` model is built that is configured for a multi-label problem with 10 possible labels. The model is compiled with a binary loss function and includes accuracy, precision, and recall metrics. Next, the data is divided into a training set and a test set, and the model is trained over three epochs. At the end of training, a prediction is made on the test set and the probabilities are converted to binary values ​​according to a threshold of 0.5. Finally, a report is printed that includes Precision, Recall, and F1-score for each of the ten tactics, in order to evaluate the model's performance.

The first model (before improvements) used basic BERT (bert-base-uncased) to perform multi-label classification on clickbait headlines. The input was a single headline, and the output was a binary vector of size 10 indicating which clickbait tactics were found. The model was trained with a BinaryCrossentropy loss function, with the texts first converted to input\_ids using BertTokenizer. After training on about 3 epochs, predictions were made and probabilities were converted to 0/1 values ​​according to a fixed threshold (0.5), and performance was evaluated using Precision, Recall, and F1 for each tactic.

תמונה שמכילה טקסט, צילום מסך, גופן, מספר

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

The report in the image shows the performance results of the multi-label BERT model before the improvements. You can see that the Recall is very high (almost 1.00) for all tactics, which means that the model recognizes almost all cases where a tactic is present – ​​but the Precision is low (usually around 0.30), meaning that it also “guesses” many labels that are not actually present. As a result, the overall F1-score is relatively low (macro avg = 0.33), with a weak balance between precision and sensitivity. Conclusion: The model tends to over-recognize and returns labels even when it shouldn’t – meaning it is too sensitive but not accurate.

Tactics that the model does not learn at all:

|  |  |  |  |
| --- | --- | --- | --- |
| tactics | Precision | Recall | F1 |
| Sensationalism | 0.00 | 0.00 | 0.00 |
| Ambiguous References | 0.00 | 0.00 | 0.00 |
| Unfinished Narratives | 0.00 | 0.00 | 0.00 |

-The model does not recognize these tactics at all, perhaps they are too rare or confused in the data.

The improved model upgraded the initial model in three main ways:

(1) Architecture upgrade – instead of basic BERT, RoBERTa (roberta-base) was used, a more advanced model that is often considered more accurate, especially for short texts.

(2) Augmentation – duplicate versions of titles with different wordings (e.g. with bold words, exclamation marks, marketing wording) were added, to diversify the data and improve the overall ability of the model.

(3) Customized prediction threshold (per-label threshold) – instead of determining that probability > 0.5 indicates a positive label, an optimal threshold was calculated separately for each tactic using ROC analysis on the validation set.

Thanks to these improvements, performance improved significantly: Recall and F1 increased for almost every tactic, especially those that the previous model failed to identify (e.g. Provocative Questions and Emotional Triggers).

תמונה שמכילה טקסט, צילום מסך, תפריט, מספר

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

The report shows the performance of the model after the improvements (RoBERTa + Augmentation + adjusted threshold), with a clear improvement in Recall (almost 1.00 for most tactics) and a higher F1-score than the first model. However, the Precision is still low (≈ 0.3), which indicates that the model predicts too many tactics (False Positives). Some tactics were not detected at all, which lowers the Macro F1. In summary, the model is good at detection but needs improvement in accuracy – possibly by setting a more precise threshold or better balancing the data.

תמונה שמכילה צילום מסך, ריבוע, מלבן, כחול חשמלי

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

תמונה שמכילה טקסט, צילום מסך, תרשים, עלילה

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

Below is the link to Google Colab where we ran the code for multilabel BERT attribution:

<https://colab.research.google.com/drive/1HRMGxgCZf3o5XR9fOzPBBzDm8X_Mwp0s?usp=sharing>

Tactics attribution model: GPT 4o a few-shot prompt

At this point, we used GPT-4o to attribute clickbait tactics using few-shot prompting. i.e., we presented the model with several examples of headlines along with the tactics they used and then asked it to analyze a new headline and indicate which tactics appeared in it.

Results: of the 6 titles sent to the chat GPT 4o, he classified the correct tactics for each title, but added additional tactics that did not belong ("False Positives").

We speculate that the result is this way because chat gpt tends to be less conservative in its response, meaning it prefers to “cover” as many possibilities as possible so as not to miss any possible tactics. As a result, it strives for high Recall, to identify every tactic that could be present. Even if this comes at the expense of Precision, which is actually the more important metric in the case of accurate attribution of clickbait tactics.

In addition, the model tends to interpret tactics too broadly. For example, words like “surprising” may be mistakenly considered as Curiosity Gap, or emotional expressions like “You’ll Be Furious” will also be labeled as Direct Appeals, even though their main meaning is to activate emotion.

Below is a link to the chat conversation:

<https://chatgpt.com/share/6834a5b0-7584-800c-ae3e-fb3210270197>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| pipeline | Detection model | accuracy | precision | recall | F1-Score |
| Two-steps | Fine-tuned BERT | 1.0 | 1.0 | 1.0 | 1.0 |
| Two-steps | GPT – 4o Zero-shot | 1.0 | 1.0 | 1.0 | 1.0 |
| Single-step | GPT- 4o  Zero-Shot | |  | | --- | | 0.7391 | | 0.6571 | 1.0000 | 0.7931 |
| Single-step | GPT-4o  Few-Shot | 0.8466 | 0.7666 | 0.9966 | 0.8666 |
| Single-step | Gemini 2.0 Flash  Zero-Shot | 0.8339 | 0.7506 | 1.0000 | 0.8576 |
| Single-step | Gemini 2.0 Flash  Few-Shot | 0.8132 | 0.7284 | 0.9989 | 0.8425 |