EVALUATING LLMS ON IRONY IDENTIFICATION

Savva Petrov*
*Stevens Institute of Technology
spetrov@stevens.edu
Spring 2025

ABSTRACT

This project aims to compare two approaches for classifying irony in a twitter message: A pre-trained RoBERTa model and custom-designed prompting through multiple LLMs. Detecting irony is important because it directly relates to the reliability of sentiment analysis systems. In addition, we can further refine content monitoring systems by distinguishing ironic and genuinely harmful content. The models will be compared using standard metrics such as accuracy, recall, and f1 score. Our findings reveal that while some models like deepseek exceed conventional irony detection, other like google's gemma2 fall drastically behind.

1 Introduction

Irony detection represents a specialized case of sentiment analysis where the literal meaning of text contradicts the intended meaning. Particularly, it is important to distinguish irony in sentiment analysis and content moderation systems. Systems should be able to reliably distinguish between genuine and disingenuous hateful, sorrowful, or exited content in order to correctly establish sentiment and user intent behind posts. To this end, I wish to compare a fine-tuned pre-trained RoBERTa model and custom prompting of three LLMs to establish which approach provides the best results when detecting irony, and if the LLMs are up to speed with conventional models for irony detection. Identifying irony in messages is particularly difficult due to a lack of context, which is heavily relied on by most models.

2 Related Work

Early on, tools such as lexical queues were used to detect irony in text. More recently, machine learning has been utilized to discern irony in text, such as using emotional cues [4], demonstrating the importance of emotional content and context when discerning irony. Further, transformers [2] have also been used for detecting irony, leading to an improved performance in irony detecting tasks. Lastly, a work by Peiling Yi and Yuhan Xia (2025) [3] explores irony prediction with no explicit training data. The pretrained ROBERT model used to measure the performance of the LLMs in this paper itself has shown an F1 score of 62.5 for its classification of irony in its relevant paper[1], which was used as a measure to compare LLMs to.

3 Data Sets

The data set used will be tweet_eval from huggingface, which consists of seven heterogeneous tasks. The task used will be irony, as described in the introduction, with only two columns: text and label. The dataset consists of three further subsets: test, train, and validation, with the test dataset used to evaluate the pretrained model(to make sure it compares to the relevant paper [1]). The dataset was preporcessed with numpy, allowing it to be fed to the machine learning algorithm, using a pre-trained tokenizer for the same model. The dataframe of the dataset can be seen in Figure 1.

	text	label
0	seeing ppl walking w/ crutches makes me really...	1
1	look for the girl with the broken smile, ask h...	0
2	Now I remember why I buy books online @user #s...	1
3	@user @user So is he banded from wearing the c...	1
4	Just found out there are Etch A Sketch apps. ...	1
5	Hey what do you know, one of the witnesses sup...	1
6	@user on stage at #flzjingleball at the @user ...	0
7	You know it's going to be a great day when you...	1
8	Halfway thorough my workday ... Woooo	1
9	Would like to thank my nephew for giving me hi...	1

Figure 1: tweet_eval irony data

4 Method

4.1 Download data

Data from tweet_eval was downloaded from huggingface, and the test set selected for model evaluation. The dataset will be preprocessed using the provided tokenizer for the pretrained model, and the text column fed on its own to LLMs.

4.2 Evaluate pretrained model

Evaluate the pretrained model using the test set to make sure its in line withe the findings from the relevant paper [1]. Additionaly contruct a confusion matrix to see how the model is placing its predictions.

4.3 Evaluate LLMs

Evaluate the LLMs one by one, collecting metrics such as accuracy, F1, precision, and recall, as well as a confusion matrix, comparing the results to the baseline pretrained model. Lastly, we will evaluate the confusion matrix to see if the model truly performs better and what may be going wrong.

5 Experimental Setup

5.1 Evaluation Metrics

Since this is a classification problem, we can use Recall, Accuracy, Precision, and F1 for evaluating the accuracy of the model. A confusion matrix was also constructed for each model.

5.2 Pretrained Model

The pretrained model used was cardiffnlp/twitter-roberta-base-irony, which was pretrained on the irony task in the relevant paper [1]. We evaluated the pretrained model on the test dataset, in order to make sure our control is consistent with the paper. The results turned our positive, with our model showing an F1 score of 0.627, similar to the paper's 61-62.5 range. The confusion matrix shows that the model is good at discerning non ironic content, but struggles with half of the ironic cases. Additionally, compared to the llm queries, the model was much faster, performing predictions for the entire dataset in the span of 5 seconds, where the fastest llm took 27 minutes to make predictions for the entire set. A tokenizer was provided with the model for input preprossessing.

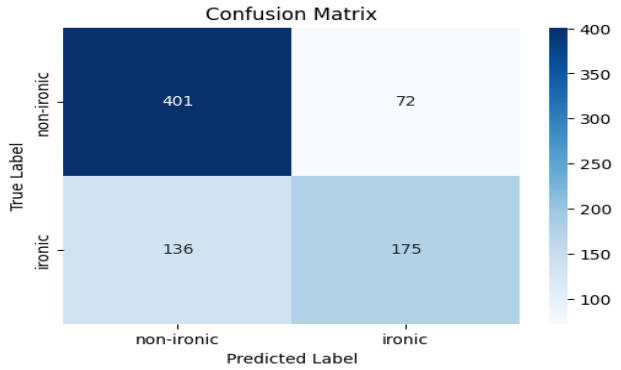

Figure 2: Resulting confusion matrix from the pretrained model

5.3 LLMs Used

All LLM queries were made through Groq, due to funding limitations. The models used were:

- llama-3.1-8b-instant
- deepseek-r1-distill-llama-70b
- gemma2-9b-it

5.4 Prompts for LLMs

The pretrained model was then compared to llm perfomance usning the pompt:

"Given the following post, classify it as "ironic" or "non-ironic". Irony involves saying something that contrasts with the intended meaning, often for the purpose of humour. Provide only the label ("ironic" or "non-ironic") as your response."

Notably, the deepseek-r1-distill-llama-70b model had no option to not output the <think><think>process tags and as such its prompt had to be modified to the following:

"Given the following post, classify it as "ironic" or "non-ironic". Irony involves saying something that contrasts with the intended meaning, often for the purpose of humour. Provide only <answer>"ironic" or "non-ironic"<answer>as your response. Tweet: tweet RESPOND WITH <answer>ironic ||non-ironic<answer>ONLY" Tweet: tweet"

With response tokens expanded from the default 10 to 1000 to ensure the true model output is not cut off, and the output preprocessed to look for answer tags. All other models were provided with max response tokens of 10.

6 Findings

6.1 llama-3.1-8b-instant

For my first model, I used facebook's llama-3.1-8b-instant model, which performed amicably with an F1 score of .68, suprassing the pretrained ROBERTa model. That being said, it boasts both a poorer precision and accuracy, meaning it is not particularly well fitted for the task:

```
Metrics Summary:
Accuracy: 0.6913
Precision: 0.5772
Recall: 0.8296
F1 Score: 0.6807
```

Figure 3: Resuls from llama

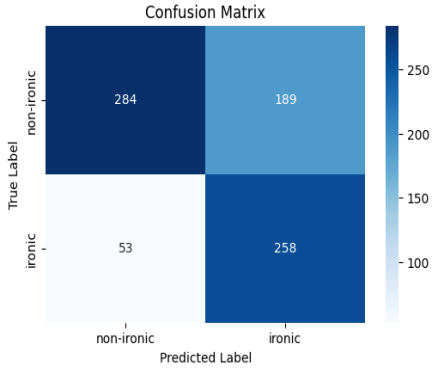

Figure 4: Resulting confusion matrix from llama

Showing the opposite problem for this model, where it misidentifies half of the non-ironic instances as ironic instead. The model ran the fastest at 27:56 minutes.

6.2 deepseek-r1-distill-llama-70b

Secondly, I used deepseek-r1-distill-llama-70b, which had to have its max_output tokens expanded to accomodate thinking. This model showed the best results, boasting an F1 score of .78, accuracy of .789 and a high recall score of .9775, showing that the model was discerning genuine trends:

```
Metrics Summary:
Accuracy: 0.7895
Precision: 0.6580
Recall: 0.9775
F1 Score: 0.7865
```

Figure 5: Results from deepseek

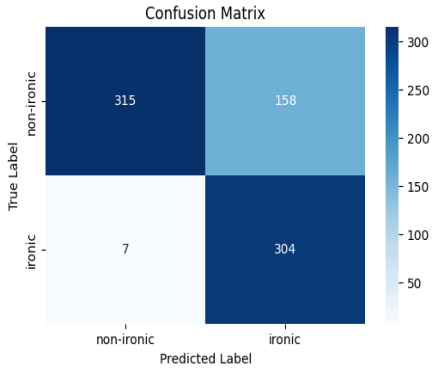

Figure 6: Resulting confusion matrix from deepseek

The model ran for 44:46 minutes.

6.3 gemma2-9b-it

Lastly, Google's gemma2-9b-it was tested to evaluate its ability to discern irony. While the model showed a similar F1 score to the pretrained baseline, the perfect recall (1.00) and the confusion matrix show it is overwhelmingly guessing irony, nearly equating this approach to random guessing. This is further supported by the abysmal accuracy and precision scores, showing the models inability to adapt to the task:

```
Metrics Summary:
Accuracy: 0.5191
Precision: 0.4520
Recall: 1.0000
F1 Score: 0.6226
```

Figure 7: Results from gemma2

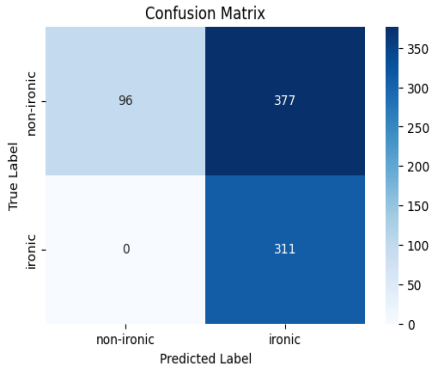

Figure 8: Resulting confusion matrix from gemma2

Overall, this model showed an inability to correctly discern irony in user posts, and was the only LLM to be essentially equivalent to guesswork in doing so. This clearly demonstrates that not all LLMs are the same when they are detecting user sentiment. The model ran for 28:19 minutes.

7 Discussion

7.1 Overall performance

Two of the three LLMs demonstrated either inconclusive or worse results when compared to the baseline pretrained model, with google being almost equivalent to guessing, and llama struggling with predicting non-ironic content, showing an inability to fully grasp user patterns. Overall, the models seem too 'trigger-happy' with calling any content irony. One exception to this, however is deepseek, which showed a genuine degree of ability to predict both labels, while still struggling to correctly identify non-irony.

7.2 Limitations

While discerning irony on its own is a difficult task due to the lack of context present in social media posts, it is possible that narrowing the potential sentiment to simply irony and non-irony for these LLMs can mess with the way they classified. Additionally, the length of processing time when using LLMs is significantly higher than the pretrained model, with all models exceeding processing time, and while deepseek demonstrates a noticeable improvement, its processing time of 47 minutes is also significantly longer than RoBERTa. As they are LLMs are too time consuming to use practically for irony detection.

8 Conclusion

Concluding, while deepseek demonstrates a better ability to discern irony than the baseline, the other LLMs fall behind, and although llama demonstrates a better F1 score, it falls behind in accuracy and precision. Additionally, the LLMs take significantly longer to make a single prediction due to the querying method, posing a question of whether they can truly be used for real world irony detection applications. That being said, if the time problem is overcome, the noticeably improved performance of deepseek can likely be leveraged for better irony detection.

Better irony detection systems can help us build systems that are better tailed to user needs, and respond less to false positives such as irony. As it stands, it seems most LLMs struggle with discerning posts that do not show irony, as opposing to the baseline struggling with mislabeling ironic content, and are too time-expensive to run practically. However, as we reach better computation and information exchange capabilities, this gap will likely narrow, allowing us to use better models like deepseek for context-poor irony detection in the future.

References

[1] Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1644–1650, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.148. URL `https://aclanthology.org/2020.findings-emnlp.148`.

[2] Hiram Calvo, Omar J. Gambino, and Consuelo Varinia García Mendoza. Irony detection using emotion cues. *ISSN 2007-9737*, 2020.

[3] Irony Detection Using Transformers. A. agrawal and a. k. jha and a. jaiswal and v. kumar. *2020 International Conference on Computing and Data Science (CDS)*, 2020.

[4] Peiling Yi and Yuhan Xia. Irony detection, reasoning and understanding in zero-shot learning. In *https://arxiv.org/html/2501.16884v1*, January 2025.