Erdos Institute Deep Learning Bootcamp

Multimodal Sarcasm Detection in Memes

Yiyang Liu, Eunbin S, Kim

Introduction

Effectively classifying and understanding memes—especially distinguishing between sarcastic and non-sarcastic content—has become crucial for enhancing user experience and moderating online platforms. This project explores the development of an advanced multimodal classification model that integrates both image and text data to accurately categorize memes. By leveraging techniques in deep learning and natural language processing (NLP), our approach aims to significantly improve classification performance compared to traditional unimodal models. For this project we compare a multimodal approach following a CLIP architecture with a unimodal text model implementing DistilBERT.

Problem Statement

The main task of this project is to detect the sarcasm in memes by classifying a dataset of memes into “sarcastic memes” and “regular memes”. The model should take the caption and the images of a meme and produce a score for classification. Sarcasm in memes often involves a blend of visual humor and textual irony. A multimodal approach can leverage both elements, whereas a unimodal model (like one using only DistilBERT) might miss contextual or visual cues that are crucial for accurate classification.

Key Performance Indices

1. **Accuracy**: This KPI measures the proportion of correctly classified instances out of the total instances.
2. **AUROC**: This KPI assesses a model's ability to distinguish between classes by plotting the true positive rate against the false positive rate at various threshold settings. Due to the unbalanced nature of our dataset, this is our main KPI.

Dataset Description

- Features: Images, Captions, Labels
- Total number of memes: 7000 (6830 after cleaning)
    - Sarcastic: 1884; Not Sarcastic: 4946
    - Ratio: 1 (sarcastic): 2.78 (not sarcastic)

Data Cleaning/Preprocessing

The images and their captions are cleaned respectively after we drop the pairs of data with N/A entries. For the captions, we perform the following steps:

- Remove all non alphabetical and numerical characters.
- Remove the watermarks recognized by OCR such as memecenter.com, facebook.com, etc.
- Lemmatization. That is, to break down a word into its root form or lemma.
- Remove stop words
- For the training set, we read the label and pair "a regular meme" or "a sarcastic meme" with the caption to help with the training.

For the images of the memes, we perform the following steps:

- Clean up all corrupted image files and remove extra bytes.
- Resize all images to 224 x 224. Convert all images to RGB color space.
- Normalize all pixel value tensors.

After all the data cleaning, the tokenization for the text and the patching of the images are done by DistilBERT and ViT preprocessors, respectively.

Model/Solution

Due to the multimodal nature of memes, we chose to implement and apply a deep neural network classification algorithm inspired by the OPENAI CLIP model, which can take both images and texts as the inputs and compute their interactions. The model consists of a few building blocks, the encoder block, the image/text projection heads, the CLIP-like feature interaction matrices, and finally, a classifier block. We set the projection heads and the classifiers to be trainable while fixing all other parameters. The model has **Total Params =**

162580481 and **Trainable Params = 9637121**. Next, we give detailed configurations of each block.

The Encoder Block

We use pretrained text and image encoders. We chose *distilbert-base-uncased* as the text encoder, and *dinov2* visual transformer as the image encoder. These two encoders will produce image/text embeddings of dimension 1 x 768 for each image/caption. For the sentence embedding, we use a combination of max pooling and average pooling of the word embeddings in the sentence instead of that of the [CLS] token.

The Projection Heads (Trainable)

We use the same structure for the image and text projections heads. This structure is expected to take word/image embeddings as the input and produce for each a normalized vector of size 512. Each consists of layers in the following order: A dropout layer, a fully connected layer, a GELU layer, another fully connected layer, and a LayerNorm layer. The first fully connected layer projects word/image embeddings into vectors with size 512.

The Feature Interaction Matrices

For each pair of caption and image features out of the projection heads, we construct a feature interaction matrix using their outer product. That is, for an image feature ***I*** with shape 1xm, and a text feature ***T*** of the same shape, the resulting matrix should be

$$\text{transpose}(\boldsymbol{T}) \ @ \ \boldsymbol{I}$$

with a shape of mxm. In our case, the shape is 512 x 512.

The Classifier (Trainable)

The final classifier block flattens the feature interaction matrix and produces a score for binary classification. This part of the neural network consists of two fully connected layers that gradually reduce the output dimension to 1. The final output goes through a sigmoid function.

Other Model Components and Some Choices of Parameters

For the optimizer, we choose ADAM with a learning rate of 1e-5. We use Binary Cross Entropy Loss as the loss function and evaluate the results with both prediction accuracy and the AUC-ROC score.

Results

Multimodal Model:

- Accuracy: 0.7436
- AUC-ROC: 0.7969

Uni-modal Model (DistilBERT):

- Accuracy: 0.4605
- AUC-ROC: 0.4868

Conclusions and Future Directions

- The multimodal model outperformed the unimodal model
- Benefits of the multimodal model:
    - Enhanced understanding: able to better capture the nuance of sarcasm in memes with both text and visual cues
    - Improved Accuracy: Integrating image and text data typically leads to better performance than using only text, as sarcasm often relies on both visual and linguistic features
- Text only models might miss contextual cues crucial for accurate classification provided by the image
- We modified and implemented an existing model for the task. All implementation files can be found on the project's GitHub

Some potential future directions includes:

- Re-run the algorithm with larger and more well-constructed dataset to verify the quality of the model and do detailed error analysis.
- Modify the model by concatenating text and word embedding with the flattened feature interaction matrix and see whether that increases the accuracy of the model.