🎯 Assessment 3 NLP and Computer Vision (Multi-Modal Sentiment Analysis)

1. Introduction

This report outlines the steps and outcomes of a sentiment analysis project that required the use of Natural Language Processing (NLP) and Computer Vision (CV) methods. The aim was to discern the sentiment of social media posts through both text and image data as either positive, negative, or neutral.

2. Tasks

- **Task 1:** Dataset Preparation

The dataset I used is from Twitter Sentiment Analysis with Images. This dataset contains:

- **Text Data**: Tweets or captions associated with each post.
- **Image Data**: Corresponding images or memes attached to the posts.
- **Labels**: Sentiment of the tweet (positive, negative, or neutral).

To make the dataset useable, image files were matched with their text captions using an identifier (file name). I checked this structure and completed the verification step confirming that text and image files exist for all entries.

- **Task 2:** Text Data Preprocessing and Modeling (NLP)

2.1 PREPROCESSING STEPS

- **Tokenization**: Implemented with BERT tokenizer from HuggingFace Transformers.
  Padding & Truncation: All sequences were padded/truncated to a maximum length of 128 tokens.
- **Attention Masking**: Provided during model input to distinguish real tokens from padding.

2.2 MODEL

We used **BERT (bert-base-uncased)** for text feature extraction:
- The final hidden state (`pooler_output`) was used to represent each caption as a 768-dimensional vector.
- A classification layer was fine-tuned on top of BERT to perform sentiment prediction.

2.3 TRAINING

- Used an AdamW optimizer; train for 3 epochs.
- Achieved a validation accuracy of ~76%.
- The entire model and the tokenizer was stored in disk and was later resumed for fusion.

- **Task 3:** Image Data Preprocessing and Modeling (CV)

3.1 PREPROCESSING STEPS

- Images were resized to 224x224.
- Image normalization was done using the mean and standard deviation values of ImageNet.
- Converted to PyTorch tensors.

3.2 FEATURE EXTRACTION

ResNet50 was incorporated with a modified architecture:

- Extracted 2048-dimensional feature vectors using the final convolutional layer (via Global Average Pooling).
- Features were extracted and cached for all images.

3.3 CLASSIFICATION MODEL

- This is accomplished by a multi-layer perceptron model with a single hidden layer (256 units, ReLU) and softmax output.
- Trained using CrossEntropyLoss with class weights to balance label distribution.

3.4 TRAINING

- The model was trained for 20 epochs using Adam optimiser.
- Image-only classifier achieved a test accuracy of ~46.5% after many tuning attempts (dropout, balanced loss, more epochs).
- Prior attempts resulted in an accuracy of 40%; the use of extra feature training and longer performance improved the results.

- **Task 4:** Fusion and Final Classification

4.1 FEATURE FUSION

- Combined 768-dim text features (BERT) with 2048-dim image features (ResNet) to create a 2816-dim feature vector.
- I applied dropout before concatenation to reduce overfitting.

4.2 MODEL ARCHITECTURE

```
Input (2816) → Linear(512) → ReLU → Dropout → Linear(3) → Softmax
```

- The model was trained using same data split and CrossEntropyLoss.
- I ensured that text and image features were normalized and aligned.

4.3 TRAINING & EVALUATION

- I used a batch size of 32 and trained using the Adam optimizer with a learning rate of 1e-4.
- Trained for 10 epochs.
- Fusion model achieved a test accuracy of ~44.6%.
- Earlier versions had lower performance (~41%), I improved the model by:
  - Better dropout
  - Class-balanced loss
  - Longer training
- I evaluated model performance using accuracy, precision, recall, and F1-score to assess both overall and per-class behavior.

- The results showed: Precision = 0.4460, recall = 0.4466, and F1 score = 0.4461, which confirmed balanced performance across classes, suggesting the model is not heavily biased toward any specific sentiment.

3. Observations and Challenges

WHAT WORKED WELL

- BERT accuracy was consistently good.
- ResNet50 features were rich enough to distinguish sentiment visually when trained longer.
- Concatenating features was effective and simple.

CHALLENGES FACED

- Careful preprocessing was required for matching text and image files
- Model accuracy at the beginning was impacted by class imbalance.
- The fusion model required multiple iterations in order to tune properly, which proved to be difficult. Dropout rates, hidden layer sizes and learning rates made the models' results very sensitive.
- GPU/Colab resets sometimes required re-extracting features, which was time-consuming.

4. Conclusion

The multi-modal sentiment classifier project was successfully completed through the use of NLP and CV components.

Final Scores:
- NLP Model: Accuracy = ~76%
- Image Model: Accuracy = ~46.5%
- Fusion Model: Accuracy = ~44.6% / Precision = 0.4460 / Recall = 0.4466 / F1 = 0.4461

This proves the power of the analyzed text in sentiment classification, and the complementary role image data can serve when utilized properly.

5. Future Work

- Try attention-based or fusion-transformer models.
- Augment image data to improve generalization.
- Use larger and more diverse datasets, so that the models understand better sentiment nuances.
- Incorporate explainability strategires(e.g., GradCAM for images, attention for text).