# CSE 256_FA24: Multimodal Language processing for Sentiment and Emotion prediction

**Author - Nikita Johny Kachappilly**
nkachappilly@ucsd.edu

## 1 Introduction

Analyzing sentiment and emotions of an individual in real-world scenarios is a complex task that cannot be properly addressed by relying solely on textual data. Text alone lacks the details that are necessary to fully capture a speaker's intent, mood or emotional state. For example, the same sentence "I can't believe you're eating that" can convey drastically different meanings - either disgust or happiness - depending on the speaker's tone of voice, facial expressions and body language. Multimodal language analysis allow for a more holistic understanding and accurate prediction of sentiment and emotions as it includes the textual, acoustic and visual context of the speaker. By incorporating multimodality, we can overcome the limitations of text-based approaches and create systems that better understand and respond to human communication in real-world.

**List of things done**:

- Collected and preprocessed dataset: DONE
- Build and train baseline model on collected dataset and examine its performance: DONE
- Implemented a model that can perform slightly well when compared to the baseline models: DONE

## 2 Related work

The MOSI-dataset for multimodality described in (Zadeh et al., 2016) focuses on analyzing the interaction between verbal and visual cues to enhance sentiment analysis. The authors proposed several baselines such as using the 'Verbal' features only from MOSI-dataset by creating a bag of words feature set from monograms and bigrams. Another model that showed an improved MAE rate was the Multimodal Dictionary that captures the co-occurrence of verbal and visual features. The dictionary was built by combining verbal words with gestures. For each word $W_i$ and gesture $G_j$, two sets were created to capture instances where both the word and gesture are present, and when the word is present without the gesture. This allowed the model to learn the relationship between words and gestures effectively, improving sentiment predictions.

The paper (Zadeh et al., 2017) proposes a Tensor Fusion Network (TFN) for multimodal sentiment analysis on the CMU-MOSI dataset. The model consist of three components : 1. Modality Embedding Subnetworks (spoken language embedding subnetwork utilizing 300-d Glove vectors, the visual embedding subnetwork utilizing OpenFace and FACET frameworks,the acoustic embedding using COVAREP acoustic analysis framework) that take as input unimodal features and output a modality embedding. 2. TFN layer then models the unimodal, bimodal and trimodal interactions using a 3-fold Cartesian product from modality embeddings. 3. The output of TFN Layer is then conditioned by a Sentiment Inference Subnetwork to perform sentiment inference.

The authors of the paper (Zadeh et al., 2018) introduced the concept of n-modal dynamics, which refers to the various combinations of modalities that can be utilized for effective sentiment/emotion analysis on the CMU-MOSEI dataset. The proposed approach is a highly complex Dynamic Fusion Graph (DFG), that considers unimodal, bimodal and trimodal (n-modalities) so as to dynamically alter its own structure by assigning dynamic efficacy scores based on the importance of each n-modal during inference. This DFG is replaced in the position

of the original fusion component in an existing Memory Fusion Network (MFN) architecture.

The paper (Amir et al., 2018) proposes a Memory Fusion Network (MFN) for multi-view sequential learning that consist of three components: 1. System of LSTMs consisting of multiple LSTM networks, one for each view. 2. Delta-memory Attention Network, an attention mechanism designed to discover both cross view and temporal interactions across different dimensions of memories in the LSTMs. 3) Multi-view Gated Memory, a unifying memory that stores the cross-view interactions over time. The MFN has been experimented for multimodal sentiment analysis using CMU-MOSI dataset and for multimodal emotion recognition using IEMOCAP dataset.

The approach implemented in the paper (Dutta and Ganapathy, 2024) for multimodality involves a Hierarchical Cross Attention Model (HCAM) that effectively integrates audio and text modalities to enhance emotion recognition. The HCAM model is trained on various datasets including CMU-MOSI by initially focusing on individual modalities(audio and text), and later combining these modalities. The model also uses a bidirectional-GRU (Gated Recurrent Unit) architecture along with self-attention mechanisms. This allows the model to capture contextual information from the input sequences, improving the representation of both audio and text data. In the last stage, the model fuses the learned representations from both modalities that is achieved through a co-attention mechanism which allows the model to focus on important parts of both audio and text inputs simultaneously.

## 3 CMU-MOSEI Dataset

The CMU-MOSEI (CMU Multimodal Opinion-level Sentiment Intensity) dataset is widely used for multimodal sentiment analysis and emotion recognition, comprising approximately 23,000 sentence utterance videos from nearly 1,000 diverse YouTube speakers. The dataset is gender-balanced (57% male, 43% female) and spans 250 distinct topics. Each video captures three modalities:

- Language: Spoken text (with tokenized words represented by 300-dimensional GloVe features)
- Visual: Facial expressions, gestures, and facial action units (extracted using FACET and OpenFace)
- Acoustic: Intonation and tone features (pitch and MFCCs via COVAREP)

The dataset underwent rigorous manual filtering (by 14 judges over a span of 3 months) and automatic filtering to ensure high quality, resulting in 3,228 videos which were then segmented and its corresponding transcripts punctuated. The dataset is available for download through CMU Multimodal Data SDK GitHub (CMU, 2018).

In this Github repository, there exist links to the .csd files (Computational Sequential Data - hierarchical data structure stored in the HDF5 format) that stores the data and metadata for each modality as well as for the target labels. These were the .csd files that I had downloaded (you can refer the cmu_mosei.py in my code for the download links):

- **CMU_MOSEI_TimestampedWordVector.csd** (1.4GB file): This file contains tokenized text from the videos, with each word represented using 300-dimensional GloVe word embeddings. The text is aligned with audio at the phoneme level using the P2FA model, ensuring multimodal synchronization without any data loss. This file stores these word embeddings as "features" and their corresponding time intervals as "intervals".
- **CMU_MOSEI_COVAREP.csd** (11GB file): This file includes acoustic features extracted using COVAREP software, such as 12 Mel-frequency cepstral coefficients (MFCCs), pitch, voiced/unvoiced segmentation, and other speech parameters like glottal source and peak slope. These features, which are needed for capturing speech emotion and tone, are stored as "features" with their corresponding time intervals.
- **CMU_MOSEI_VisualFacet42.csd** (1.5GB file): This file contains visual features extracted from video frames captured at 30Hz. The frames are processed to detect faces using MTCNN. Facial action units (FACS) were extracted through Facial Action

Coding System (FACS) and basic emotions from static facial expressions extracted using Emotient FACET. These features are stored as “features” with their corresponding time intervals.

- **CMU_MOSEI_OpenFace2.csd** (16GB file) : This file includes facial landmark data and other facial features extracted using OpenFace. It stores information on 68 facial landmarks, 20 facial shape parameters, head pose, eye gaze, and HoG features. These features are stored as “feature” vectors with corresponding time intervals.

- **CMU_MOSEI_Labels.csd** (22.7MB file): The target labels for sentiment and emotion prediction for each time “interval” is stored into “features” vector of this file.

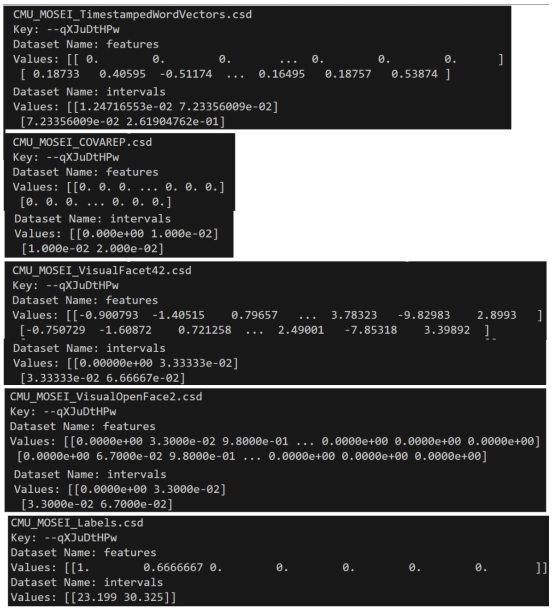


Figure 1: Dataset as csd files

**Dataset Statistics**

1. **Size**: 3,228 videos and 22,860 sentences (by segmenting the videos)

2. **Diversity**: Covers 250 topics and includes both male and female speakers

3. **Modalities**: Text, audio, and visual features.

4. **Target Labels**: Sentiment scores (-3 to +3) and 6 emotion categories (Happiness, Sadness, Anger, Surprise, Disgust, Fear).

**Example of how emotion and sentiment should work in real-world:**

1. Input: Suppose there is a video where the speaker says ”This is amazing!”. The three input modalities would contain mostly :
   (a) Text: ”This is amazing!”
   (b) Visual: Positive facial expressions, smiling
   (c) Acoustic: Rising pitch, joyful tone.

2. Output: The target output would contain 7 values like (3, 3, 0, 0, 2, 0, 0) which indicate strongly positive Sentiment (+3) and Emotion: Joy, Surprise (3 and 2 at the second and fifth index).

**Challenges**:

- **Multimodal Alignment**: Synchronizing text, audio, and visual features accurately as per the time intervals

- **Data Complexity**: High-dimensional features like GloVe embeddings and facial landmarks contain large amount of data even for small time intervals.

### 3.1 Data preprocessing

**Why preprocessing is done?**

- Synchronization: CMU-MOSEI dataset includes multiple modalities that are recorded at different time intervals and may not be synchronized. Aligning them ensures that each time interval from each modality corresponds to the same point in time.

- Handling Missing Data: Imputation is performed to handle missing data points. The dataset may have missing values in some of the modalities, and the imputation process fills these gaps to make the dataset usable.

**Preprocessing steps:**

- For the task of preprocessing, I have used the mmsdk library (either from github repo or pip install) for aligning the CMU-MOSEI dataset features.

- Initially, every modality is word-aligned, meaning that to align modalities to ”GloVe_vectors” and handle missing data. The align function in mmsdk aligns

all sequences to a reference computational sequence. The impute function from mmsdk fills in missing data using a specified imputation function. This process of word-aligning for the "CMU_MOSEI_TimestampedWordVectors.csd" file alone took 3 hours. It took about 6 hours for the "CMU_MOSEI_COVAREP.csd". The process of word-aligning alone should have then taken about 1 day for my modern laptop. These word-aligned computational sequences for each csd file are then saved into a "word_aligned" folder.

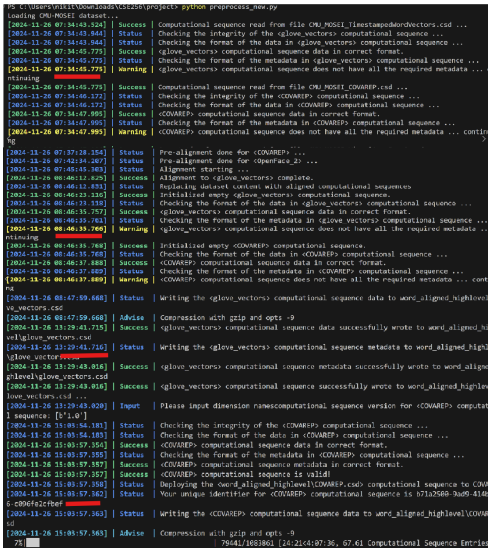

Figure 2: Pre-processing time constraint

- Next, we add the labels to the dataset and align all the data to these labels by calling the align function. We then call the 'hard_unify' function that performs a strict version of unification, removing sequences that do not match in all sequences (those sequences that contain missing modality data). These aligned csd files are then saved into a "final_aligned" folder. This particular final_align process would have taken approximately 2to 3 days in a modern laptop. Fortunately, I was able to get this "final_aligned" csd data files from the ACL challenge website (ACL, 2020). This final_aligned dataset contained 22,860 data examples and have been split into 16327 for the train set and 6533 for the test set.

- The tensors for each aligned modality are then extracted using the get_tensors function. The tensors of "Visual_openface.csd" and "Facet4.2.csd" were concatenated into the vision tensor. The textual, vision, acoustic and label tensors were then stored into npy descriptor files as train_language.npy, train_vision.npy, train_acoustic.npy and train_labels.npy. Similarly, npy files were created for the test set.

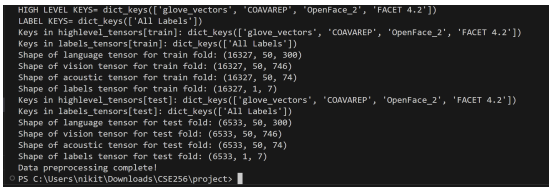


Figure 3: Train - test split on dataset

### 3.2 Data annotation

The data annotation process for the CMU-MOSEI dataset includes annotations for both sentiment and emotions, with the annotations carried out by crowdsourced workers from the Amazon Mechanical Turk platform. Each sentence is labelled in the form of a tensor [x, a, b, c, d, e, f] with x(sentiment) indicating the sentiment annotation and a(happiness), b(sadness), c(anger), d(surprise), e(disgust), f(fear) indicating the emotion annotations. These annotations are present in the npy descriptor files for "Labels" as a continuous range of values (eg, 1.6667 or 1.3333) instead of integer values (like 1 or 2).

- **Sentiment Annotation**: Each sentence in the dataset is annotated with a sentiment score on a Likert scale from [-3, 3], where [3: Highly negative, 2: Negative, 1: Weakly negative, 0: Neutral, +1: Weakly positive, +2: Positive, +3: Highly positive] are the scale meanings.

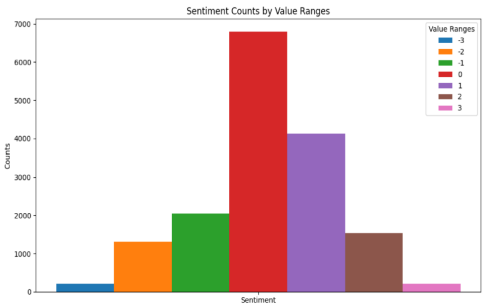


Figure 4: Sentiment distribution for train set

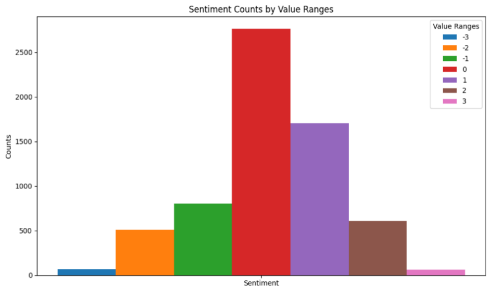


Figure 5: Sentiment distribution for test set

- **Emotion Annotation**: Each sentence also includes annotations for Ekman emotions annotated as: [”Happiness”, ”Sadness”, ”Anger, ”Surprise”, ”Disgust”, ”Fear”]. These emotions are annotated on a [0, 3] Likert scale where [0: No evidence of the emotion, 1: Weak presence of the emotion, 2: Moderate presence of the emotion, 3: Highly present emotion].

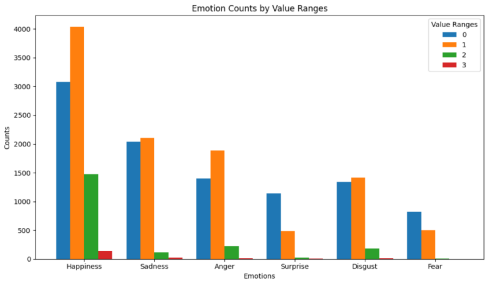


Figure 6: Emotion distribution for train set

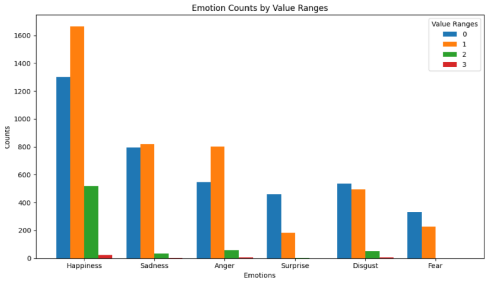


Figure 7: Emotion distribution for test set

**Distribution of Sentiment and Emotion**:
The distribution of sentiment in the dataset shows a slight shift toward positive sentiment, with a higher number of positive examples compared to negative or neutral ones. Happiness is the most common emotion, with over 12,000 positive sample points. On the other hand, fear is the least prevalent emotion, with approximately 1,900 positive sample points.

## 4 Baselines

1. **Baseline 1: Textual Modality only**
   This model consist of a feedforward neural network for sentiment and emotion prediction. The network consists of two fully connected layers with a ReLU activation between them. The input to the network is the mean-pooled embedding sequence, where the mean is taken across time steps. The input into the baseline model is the 300-d GloVe embedding features of each sentence (textual modality) stored in *.npy* files. The model's output dimension is set to 7, which corresponds to the sentiment and emotion targets in the dataset. The network is trained using mean squared error (MSE) loss, as we need to predict continuous scores for sentiment (-3,3) and emotions(0,3). Adam optimizer is used for optimization. During training, accuracy is computed by rounding the predicted continuous values to the nearest integer and comparing them with the rounded ground truth labels.
   **Hyperparameters**:
   - Input dimension: 300 (size of GloVe word embeddings)
   - Hidden dimension: 128
   - Output dimension: 7 (1 sentiment and 6 emotion values)
   - Learning rate: 0.001
   - Batch size: 32
   - Number of epochs: 30
2. **Baseline 2: Vision-Language Modality**
   This baseline is a vision-language model which consist of a feedforward neural network that processes text and vision embeddings independently before fusing them. The text and vision embeddings are mean-pooled across time steps and passed through separate fully connected layers with ReLU activation and layer normalization. These outputs are concatenated(fused) to form a fused representation, which is passed through a final fully connected layer to produce predictions.
   **Hyperparameters:**
   - Text input size: 300-d GloVe feature
   - Vision embedding dimension: 746
   - Hidden dimension: 128
   - Output dimension: 7

- Learning rate: 0.001
- Batch size: 32
- Number of epochs: 30
- Loss function: MSE Loss
- Optimizer: Adam Optimizer

# 5 Approach

## 5.1 CONCEPTUAL APPROACH

The approach implemented is a graph memory fusion network that considers each modality individually as input and then fuses them to form bimodal and trimodal representations. All these representations together help properly predict the sentiment and emotion of the datapoint.

Initially, I have loaded the *.npy* descriptor files into the train and test loader of the Pytorch framework. The input feature dimensions are 300, 746, 74 for the textual GloVe feature, visual features and the acoustic features respectively. The multimodal inputs are processed independently using LSTMs, which are initialized with orthogonal weights for stability, and their biases are adjusted to enhance gradient flow during training. The LSTM outputs are normalized using LayerNorm, and dropout is applied to reduce overfitting. The processed outputs from the LSTMs are then fed into a Dynamic Fusion Graph (DFG), where memory updates are computed for each modality using separate transformation networks (Dl, Dv, and Da).

These updates are further fused to form a combined representation through the DFG, which captures the interactions between the modalities. The fused representation is then passed through a Multi-view Gated Memory Network, which employs gated mechanisms to retain and update memory states dynamically. These gates (retain and update-gates) control how much of the fused information is kept or modified, resulting in a final memory representation that capture the most relevant multimodal information. This final memory is used to predict sentiment and emotions.

**DFG implementation:** Inside the DFG, the unimodal fusion processes each modality individually, while bimodal fusion combines pairs of modalities (language-vision, language-acoustic, vision-acoustic), and trimodal fusion combines all three modalities together. The graph dynamically weights these interactions using an efficacy network that computes scores for unimodal, bimodal, and trimodal efficacy. These scores determine the contribution of each type of fusion in the final representation, allowing the model to adaptively give importance to the most informative interactions.

**HyperParameters**

1. Input Dimensions: 300 for Language, 746 for Vision, 74 for Acoustic
2. Size of hidden state in LSTM layers, Delta networks, and Dynamic Fusion Graph: 128
3. Each modality-specific LSTM is a single-layer LSTM.
4. Dropout applied after LSTM layers: 0.3
5. Activation Functions: Tanh is used for LSTM outputs, modality updates and Sigmoid is applied in gates (retain, update, unimodal, bimodal, and trimodal).
6. Output Dimensions: 1 for sentiment, 6 for emotions
7. Learning Rate: 0.001
8. Optimizer: AdamW optimizer
9. Loss function : SmoothL1Loss
10. Batch size: 32, Epochs : 40

## 5.2 Working Implementation

A working implementation of the model has been given in the code links. The model implemented here is a simpler version of the graph memory fusion network with a dynamic fusion graph as explained in (Zadeh et al., 2018). The code files gmf_model.py, dfg.py and main.py have the code necessary for the model. [ChatGPT has been used for the code implementation]

## 5.3 Compute

The experiments of model training were run on my personal laptop which is a Dell XPS 14 with Intel Core Ultra 7 155H processor. The issues faced which were unsolvable are :

- Nan/Inf values in processed npy files of acoustic data: This issue was managed by adding preprocessing steps in the code to replace Nan or Inf with zero before feeding

data into the network. However, a better approach might be possible.

- The accuracy on the sentiment prediction for the test set is pretty low (approximately 50%).

### 5.4 RUNTIME

The entire model took about 1 hour for 20 epochs of training and about 2.5 hours for 40 epochs.

### 5.5 Results

For the prediction of each data row in this model, the final value is rounded to its nearest integer so that the targets fall in the range of [-3,3] for sentiment and [0,3] for emotion.

```
Epoch 36/40
Training Loss: 0.0910, Train Accuracy: 0.8931
Training MSE: 0.0597, MAE: 0.1455
Test Loss: 0.3299, Test Accuracy: 0.8061

Epoch 37/40
Training Loss: 0.0892, Train Accuracy: 0.8938
Training MSE: 0.0583, MAE: 0.1438
Test Loss: 0.3370, Test Accuracy: 0.8061

Epoch 38/40
Training Loss: 0.0875, Train Accuracy: 0.8947
Training MSE: 0.0575, MAE: 0.1430
Test Loss: 0.3287, Test Accuracy: 0.8078

Epoch 39/40
Training Loss: 0.0869, Train Accuracy: 0.8968
Training MSE: 0.0570, MAE: 0.1416
Test Loss: 0.3278, Test Accuracy: 0.8099

Epoch 40/40
Training Loss: 0.0841, Train Accuracy: 0.8973
Training MSE: 0.0552, MAE: 0.1397
Test Loss: 0.3244, Test Accuracy: 0.8084
```

Figure 8: Epoch training

The overall precision(P), recall(R), train and test accuracy of the model at varying epochs are as shown:

| Epoch | TrainAcc | TestAcc | P | R |
|---|---|---|---|---|
| 20 | 86.5 | 80.5 | 78.6 | 80.6 |
| 30 | 88.2 | 80.8 | 77.7 | 80.8 |
| 40 | 89.7 | 81.4 | 77.7 | 80.8 |

Table 1: Metrics for varying Epochs

The train accuracy, test accuracy and F1-score on my DFG model and the 2 baseline models are:

| Model | TrainAcc | TestAcc | F1 |
|---|---|---|---|
| My DFG | 89.7 | 80.8 | 79.0 |
| Baseline 1 | 81.3 | 80.3 | 78.7 |
| Baseline 2 | 80.1 | 80.2 | 77.5 |

Table 2: Metrics for varying Epochs

The following images shows how the training and testing accuracies differ for for each emotion across the 3 models (my implementation and the 2 baselines).

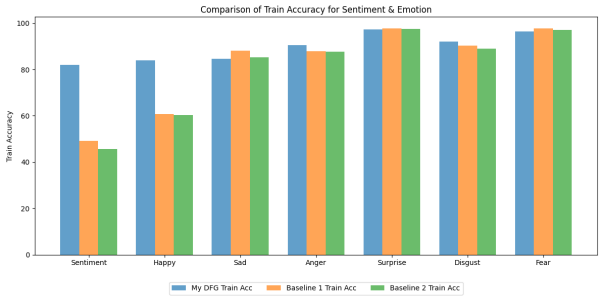


Figure 9: Training Accuracies

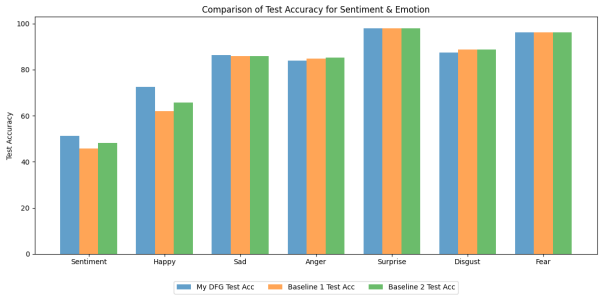


Figure 10: Testing accuracies

The following image shows how the F1 scores differ for for each emotion across the 3 models on the test sets.

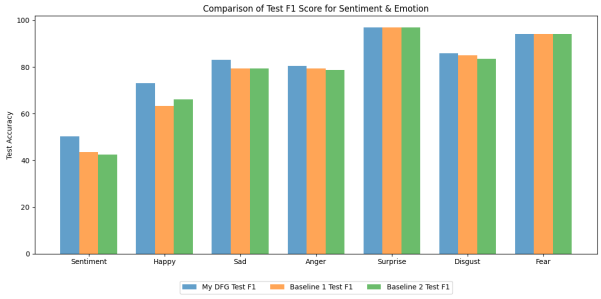


Figure 11: F1 score on test set

**Comparison between baselines and my DFG model**: The DFG model acheives a comparatively high training accuracy than the two baseline models indicative of a successful training on the

dataset. However, for the test accuracy, all these 3 models are approximately on the same level (80%). The graph plot on Fig.10 showcase how each emotion and sentiment test accuracies span over the 3 models. It is clear from the figure that the test accuracy of the sentiment is at 51% for my DFG, and 45%(Baseline 1) and 48%(Baseline 2) for the other models. This means that my model has not been able to generalize well on the test set for the Sentiment predictions. Even the F1-scores shown in Fig.11 show an overall poorer F1 score for the Sentiment prediction using my DFG model compared to other emotion predictions by my model.

To check further, I had tried to implement a pre-trained Distilbert sentiment baseline on the same test set as was used by my model. Though Distilbert predicts sentiments as 'positive' and 'negative' sentiments only (unlike our highly positive, weakly positive, positive) categories for sentiment; Distilbert achieved over 75% test set accuracy compared to my DFG (51%). All these baseline model comparisons indicate that my DFG model needs to work better on the test sets.

## 6 Error analysis

The Fig 12 shows the count of number of predictions that match the true labels of the test set. The number of matches (True Positives) for the "Sentiment" test set is really low. The "Happiness" emotion shows a better TP rate compared to the Sentiment prediction, but still is lower when analysed with the other emotions.

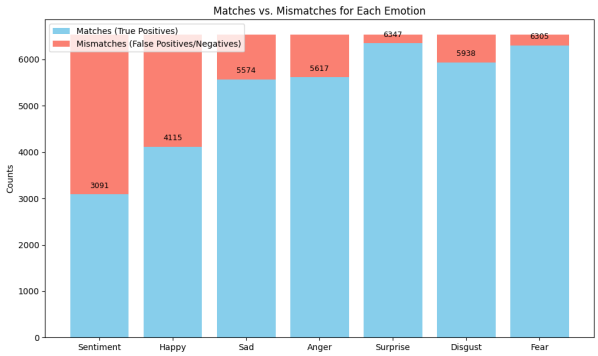


Figure 12: Analysing True Positives on my DFG

The analysis on Fig 13 (the baseline model 1) show a similar trend of TP rate (as observed in my DFG) towards the "Anger", "Surprise" and "Fear" emotions. This is mainly due to the skewed test

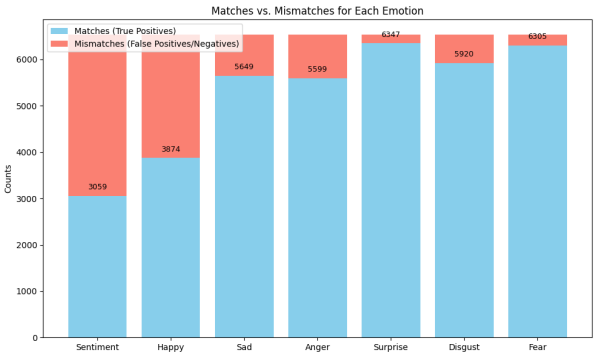


Figure 13: Analysing True Positives on Baseline 1

set count for these emotions. Fig 7 shows how for these 3 emotions, their labels occur majorly in the value range of 0(no emotion) and 1(weak presence) than the other value ranges (2 amd 3). This seems to be a valid reason as to why the test accuracy for these 3 emotions shoot up. The "happiness" test set (Fig 7) on the other hand, has a wider range of values (0,1,2,3) which makes it better balanced than the other emotions. Since there is a wider range of values to predict from, this could be a possible reason as to why the TP rate of "happiness" is lesser than the TP rate of the other emotions. From the graphs, it is obvious that the baseline model as well as my DFG model fails to properly predict multimodal "sentiment" on the dataset.

## 7 Conclusion

**Takeaways**: The DFG model performed slightly better than the baselines on the test set and surprisingly well on the training set, proving that some amount of interactions between the modalities are contributing towards the predictions.

**Difficulty**: Some emotions like surprise and fear had better accuracy suggesting that these labels might be easier to predict due to distinct patterns in the test dataset. This proved difficult for me as it is really hard to find out patterns in such large datasets. Another difficulty was the development of a graph network that can capture the weights from each modality properly.

**Future direction**: If I could continue working on the project, I would try to improve the architecture of my graph network to include attention mechanisms and maybe use other techniques that help different modalities dynamically update

the parameters of the model. Also, I would like to research on a way in which the Nan or Inf values observed in the acoustic npy file could be dealt with. This way, I hopefully would be able to improve the low accuracy scores on my test set for the Sentiment prediction.

## 8 Acknowledgements

Generative AI tools - ChatGPT has been used for proofreading this report.

- Some parts of the programming (code of this project) and Fig 4-7,9-13 have been illustrated with the help of ChatGPT.
- Major modifications have been made to the outputs generated by this AI tool.

## References

ACL (2020). Acl aligned cmu mosei dataset.

Amir, Z., Pu, L. P., Navonil, M., Soujanya, P., Erik, C., and Louis-Philippe, M. (2018). Memory fusion network for multi-view sequential learning.

CMU (2018). Cmu mosei dataset.

Dutta, S. and Ganapathy, S. (2024). Hcam – hierarchical cross attention model for multi-modal emotion recognition.

Zadeh, A., Chen, M., Poria, S., Cambria, E., and Morency, L.-P. (2017). Tensor fusion network for multimodal sentiment analysis.

Zadeh, A., Liang, P. P., Poria, S., Cambria, E., and Morency, L.-P. (2018). Multimodal language analysis in the wild: Cmu-mosei dataset and interpretable dynamic fusion graph. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2237–2242.

Zadeh, A., Zellers, R., Pincus, E., and Morency, L.-P. (2016). Mosi: Multimodal corpus of sentiment intensity and subjectivity analysis in online opinion videos.