Multimodal Emotion Recognition for Mental Health Awareness

Swanith Ambadas – swanitha – 50560450

Manish Bikumalla – manishbi – 50560699

Project Overview:

Aim:

In this project we aim to develop a custom vision-transformer that is capable of detecting human emotions in real-time combining both facial expressions and body postures. This system will enable us to analyze emotional cues to identify signs of stress, anxiety or depression and help in early intervention for mental health awareness.

State of Art:

Although there have been extensive studies on emotion recognition like papers on Facial Emotion Recognition: State of the Art Performance on FER2013 and Human Emotion Recognition from Body Posture with Machine Learning Techniques, where they utilize only one mode of input either facial feature or body postures, we are trying to provide an approach that will be able to analyze emotions based on multimodal inputs (face and body pose)

Inputs and Outputs:

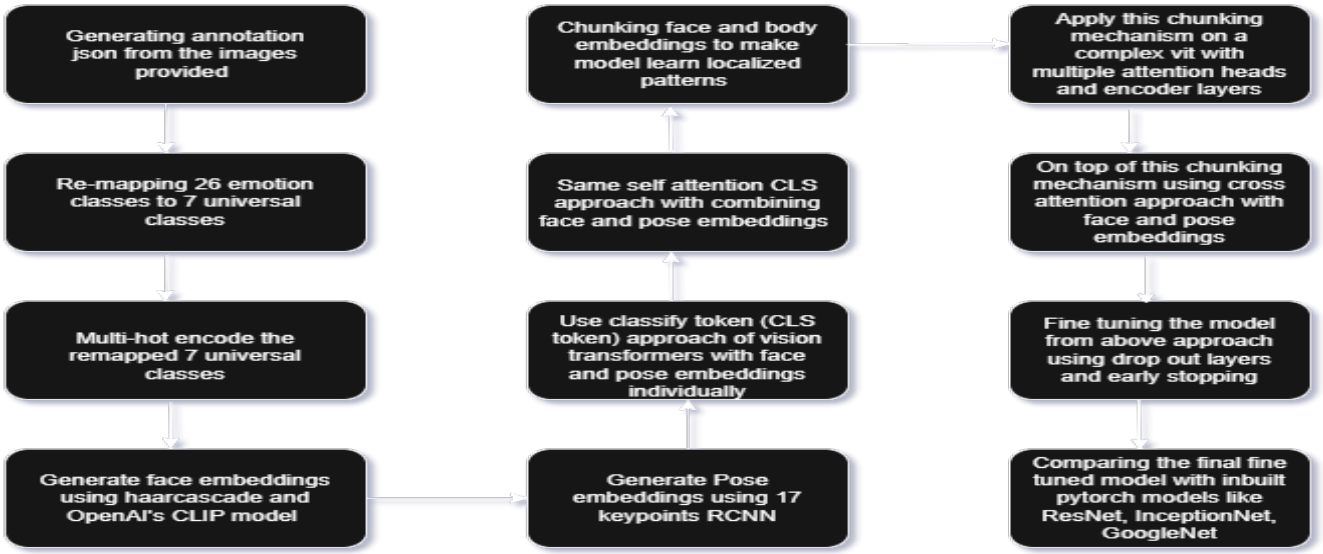
We aim to capture the emotional states of people by utilizing facial features and bodily postures from a given image of the person. We have divided the input generation into few steps as follows:

1. Initially we will use the annotations.mat file and images folder to generate the annotations.json where image_id, image_path, bounding_boxes (where each image_id containing multiple bounding boxes each for one person) are mapped with each other.
2. In our current json we have 26 discrete classes, which we are narrowing down to 7 universal classes (these classes were defined based on the coco dataset), the categorization of the classes is given below.

Universal Emotion	Emotic Categories
Anger	Anger, Annoyance, Disapproval
Disgust	Aversion
Fear	Fear
Happiness	Affection, Confidence, Engagement, Esteem, Excitement, Happiness, Pleasure, Peace
Sadness	Sadness, Suffering, Fatigue, Pain
Surprise	Surprise, Anticipation, Doubt/Confusion
Contempt	Disconnection, Disquietment, Embarrassment, Sensitivity, Sympathy, Yearning

3. Now among the 7 classes generated we multi-hot encode these classes and add it to the json to make the classification easier.
4. Now based on these bounding_boxes and image_path in the json, we generate face embeddings and body pose embedding and append them to the current json.
5. Now in the final step we send the 512-d face embeddings, 34-d pose embeddings, and 7-d multi-hod encodes into the vision transformer.

Approach:



Step by Step approach:

Step 1: Map & Split Annotations

- As our first step, we use the annotations.mat file along with actual images that are present in our emotic dataset and generate an output_annotations.json and then split the whole dataset into train, validation and test splits with 70%, 15%, 15% sizes respectively. The output_annotations.json has the following structure. It has image_id, image_path, bounding_boxes, discrete and continuous emotions.

Step 2: Multi-Hot Encode Emotions

- We utilized discrete emotions to make our classification. We first re-mapped 26 classes of discrete emotions into 7 universal emotion classes based on COCO dataset. Later we multi-hot encoded these 7 emotions and generated an intermediate json. It has image_id, image_path, bounding_boxes, discrete and continuous emotions, multi-hot.

Step 3: Extract Face and Pose Embeddings

- Now we extracted the facial feature vector of 512 dimension. We achieved this by first utilizing “Haar Cascade” to generate the bounding boxes around the faces of individuals within the limits of the original bounding boxes (it should be noted that the initial bounding boxes are for whole person rather than just for face).
- Based on these newly generated bounding boxes, using the OpenAI’s inbuilt model called CLIP ViT model we generated a 512-dimensional face feature vector and added it to our json file.
- Similarly, we generated a 34-dimension body pose vector. Body pose vector is generated by using inbuilt pytorch model “Keypoint-RCNN”. This model generated 17 key points in the body of the person (these 17 key points are: nose, eyes, ears, shoulders, hips, elbows, knees, wrists, and ankles). This added a 34-d vector which contains pixel locations of these 17 key points in both x and y direction. This generated a final json including both the body and face embeddings.

Step 4: Prototype Individual-Modality ViT

- We built a ViT model by first prepending a learnable [CLS] token to a sequence that contains the face-embedding and body-pose–embedding vectors. During each self-attention layer, the query from that [CLS] token compares against the keys (and mixes the values) of both the face and pose tokens—letting the model

decide how much facial versus body information to pull in. We then trained this whole transformer using our multi-hot emotion labels (so each sample can have multiple "true" emotions).

- The novelty in this approach is that we don't need to generate the patch embeddings of the image itself, but rather we just use the face embeddings and pose embeddings of the image that we created priorly. This significantly reduced the training time of vision transformer, making our approach more efficient comparatively.
- However, we first wanted to see how the VIT performs when it only has face embeddings or only body pose embeddings. Hence, we created 2 VITs in such a way that each one initially takes either face or body pose embeddings and evaluated the performance individually.

Step 5: Generating custom ViT for processing both Face and Pose vectors

- Now that the individual performances have been evaluated, we combined both the face and body embeddings based on the approach explained above and evaluated the performance of the model.

Step 6: Chunking the Face and Pose vectors to learn localized patterns

- In this step we divide the higher dimensional face embeddings and pose embeddings into vectors of smaller size. For example, face embeddings are divided into smaller dimensions by dividing it with face chunks of size 8. Chunking here is a way to effectively create a smaller subsequence where each sequence represents a localized region in the vector space.
- This chunking makes the sequences into manageable sizes and allows the model to learn these localized representations even more effectively. Further the model's self-attention mechanism can learn various kinds of complex interactions between these local patterns making the model even more robust.

Step 7: Improving the ViT architecture by increasing number of multi attention heads and encoder layers plus chunking

- Upon running the previous model, we wanted to improve the model architecture by increasing the complexity of the model by adding more attention heads which will help us capture better data relations in parallel (ex: one head captures mouth-signals, other might capture eye-brow movements, limb posture) and encoder layers for step wise analysis and compounding information at each layer and evaluated the model. Upon looking at the performance metrics at each stage, we realized that the model was overfitting.
- We fine-tuned the model by adding more drop out layers and introduced early stopping. This reduced overfitting and the model performed better.

Step 8: Inculcating cross-attention mechanism along with self-attention

- In this step, we utilize the previously designed chunked and complex VIT. Here, the smaller subsequences of face and body embeddings are allowed to learn the intra modally as an example how the smaller chunks of face embeddings are related to each other and how the smaller chunks of body pose embeddings are related to each other essentially capturing the internal relations between individual modalities.
- After this cross attention between body and face and vice versa are calculated. Here, the body is updated with facial context and face is updated with bodily context making the flow of information between these features even better. Then a learnable CLS token is created, and these previously learned tokens are appended to it. This has the previously fused features and is then fed into a small multi-layer perceptron for classification.
- All this ensures that the architecture learns both the intra modality and inter modality making the model robust, efficient and bringing a holistic output.

Step 9: Fine Tuning final improved ViT architecture model that utilizes chunking + cross-attention + face vectors + body vectors

- As part of the last step in building vision transformer that fits our dimensional and output requirements, we combine all the steps described in the previous step into one final ViT.

- We first utilized the chunking of the face and pose embeddings into lower dimensions to make the model learn local patterns.
- Then we trained the ViT containing multi - head attentions and multiple encoder layers with our chunked face and pose embeddings.
- We mixed both the cross-attention and self-attention mechanisms to increase the robustness of the model.
- Finally, as the model was overfitting on the validation data, we used the concept of dropout layers and early stopping to make model has robust learning of the data.

Step 10: Using inbuilt pytorch models like ResNet, InceptionNet, GoogleNet for comparison

- To have a comparison of our model's performance with state of the art, we performed the training of face and pose embeddings on ResNet, InceptionNet and GoogleNet.
- As a result of this comparison, we can confirm that our model has an on-par performance with the state-of-the-art whilst utilizing significantly less computational resources.

What aspects of the algorithms have you coded on your own? What aspects of the algorithms have you used from the online resources?

- In the first step where we map images with annotations provided into an annotations.json file was coded on our own.
- Later, the part where we encode the 26 emotic classes into 7 universal emotion classes was conceptually inspired from the paper "A Survey on Datasets for Emotion Recognition from Vision: Limitations and In-the-Wild Applicability" but was coded by us.
- We came up with a unique approach that stands out with current implementations i.e, we utilize the priorly extracted face and pose embeddings to train our ViT instead of the model generating patch embeddings at each step.
- Although the concept of utilizing face and pose embeddings is novel, the way we extract face embeddings using OpenAI's CLIP was conceptualized using NVIDIA NeMo Framework , we used the article Human Pose Estimation with PyTorch for 17 key points generation.
- Like ViT generating 16*16 patch embeddings, we chunked our face and pose embeddings utilizing the core concept of Vision Transformer models.
- Building model architetcure using both cross and selft attention mechanisms was inspired from the paper MoEmo Vision Transformer: Integrating Cross-Attention and Movement Vectors in 3D Pose Estimation for HRI Emotion Detection.
- Comparison of custom model performance with baseline models is an industry standard.

Experimental Protocol

- We used emotic dataset, which has images with multiple individuals in it. Here everyone is provided with a bounding box encapsulating the whole body of the individual along with their face.
- The emotion of the individual is quantized by the discrete and continuous emotions. The bounding boxes and the emotions of the individuals are annotated in annotations.mat file.
- The success of the model is measured by how well the model predicts the emotion of the individual based on the face and pose vectors. We estimated this quantitatively by using F1 score.

Why we used F1 score as a core evaluation metric:

We chose the F1 score because in our multi-label emotion detection setting it balances precision (did we only predict emotions that really were present?) and recall (did we catch all the emotions that actually were present?) in one robust metric. Since emotions are often imbalanced and co-occurring, F1 gives a fairer picture of overall detection performance than raw accuracy.

Results:

Below table summarizes the F1 score values of each model.

SL NO	MODEL	F1 SCORE
1	Face VIT	0.7288
2	Body pose VIT	0.7109
3	Simple_VIT	0.7210
4	Chunked_VIT	0.7207
5	Complex_chunked_VIT	0.7427
6	Finetuned complex chunked VIT	0.7489
7	Cross Attention incorporated chunked vit	0.7279
8	Fine tuned cross attention VIT	0.7210
9	Pretrained RESNET50	0.7107
10	Pretrained INCEPTION_V3	0.7134
11	Pretrained GOOGLENET	0.7098

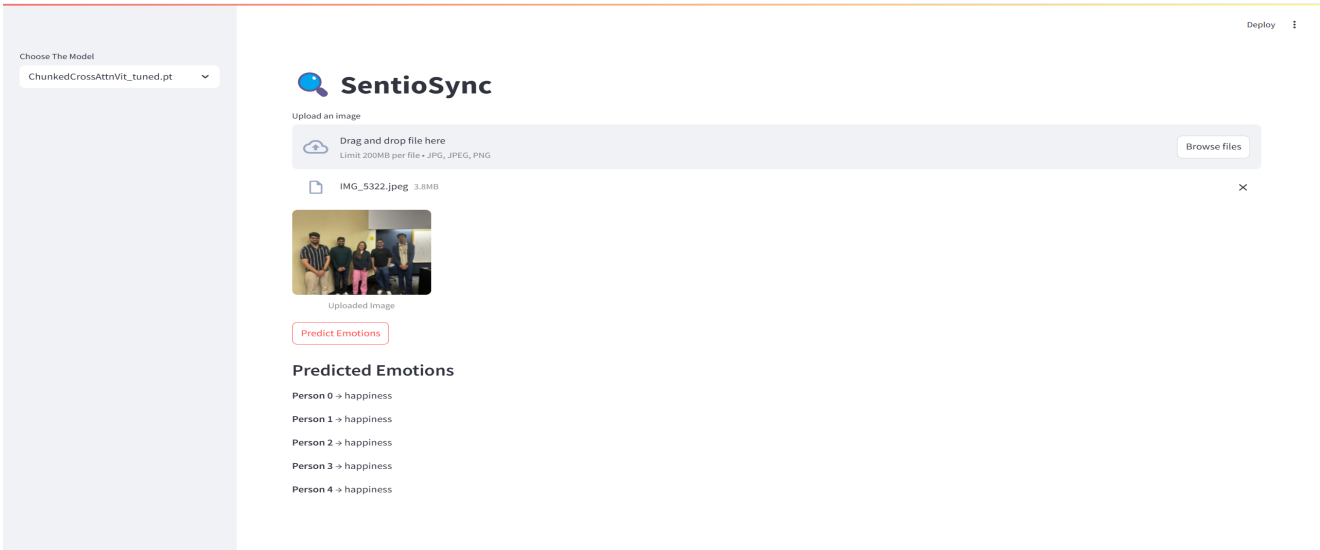
Below are certain inferences that can be drawn based on our model performances:

- One observation that can be made based on these VIT models is that multimodal beats unimodal in case of our complex chunked and cross attention models. This proves that in case of emotion detection utilizing facial expressions for detection is insufficient and adding body cues will give better results.
- Splitting each modality into smaller chunks and learning the localised patterns helped improving the model performance further fusing self-attention with cross attention yielded richer results compared to singular modality.
- On comparison with the state-of-the-art models which were pretrained on image classification tasks performed poorly when compared to the Vision transformers built exclusively for the emotion recognition tasks opening various possibilities in the field of computer vision.
- All in all fine-tuned complex chunked model outperformed every other model in terms of f1 score.

Analysis

We built streamlit web application to detect the emotions of multiple individuals in real-time using the models that are built and evaluated in the previous steps.

The results are as follows:



Advantages:

- Computational Efficiency: By only feeding compact face and pose vectors instead of full image patches, training runs noticeably faster with far lower GPU load.
- Modality-Aware Modeling: Separate face and body streams let the model learn subtle intra-modal cues (like eyebrow raises or limb posture) before fusing them.
- Robust Multi-Label Handling: Multi-hot outputs naturally capture co-occurring emotions—no more forcing a single "winner" when people feel mixed emotions.

Disadvantages:

- Loss of Global Context: Skipping full-image patches means the model can miss critical scene clues (like a birthday cake or sports gear) that help explain certain emotions.
- Fixed Embedding Dimensions: Once face and pose vectors are set (512-D and 34-D), the model can't flexibly zoom into micro-expressions or fine-grained gestures beyond those fixed sizes.
- Further the model architecture assumes that both modalities are always available. Which isnt the case in real world (Ex. selfies). This fusion mechanism may struggle unless explicitly engineered for this purpose

Discussion and Lessons Learned:

Lessons Learned:

- Balancing Capacity & Overfitting: Ramp-up of heads and layers showed how easy it is to over-parameterize
- Value of Modular Design: Splitting out face and pose into dedicated pipelines taught me how clean, modular feature extraction lets each component be optimized independently.
- Power of Sparse Inputs: Using just two compact embeddings—one for face and one for pose—showed how much extra clutter sits in raw images, and that by picking only the most meaningful features you can train faster without losing accuracy.

Future Extensions

- Re-incorporate Global Context: Add a lightweight "scene token" from full-image patches so the model can leverage environmental cues (e.g. objects, other people) that inform emotions like surprise or disgust.
- Additional Modalities: Extend to audio and text (e.g. speech transcripts or posts), fusing in a truly multimodal transformer for richer emotion understanding.

Bibliography:

- Class Notes
- Link to the dataset - emotic.zip
- https://openaccess.thecvf.com/content_cvpr_2017_workshops/w41/papers/Lapedriza_EMOTIC_Emotions_in_CVPR_2017_paper.pdf#:~:text=with%20people%20in%20real%20environ%02ments%2C,with%2023%2C%20788%20annotated%20people
- https://www.mdpi.com/2076-3417/13/9/5697#:~:text=scenario,emotional%20categories%20and%20a%20neutrality
- https://discuss.huggingface.co/t/dataset-label-format-for-multi-label-text-classification/14998
- https://docs.opencv.org/4.x/d2/d99/tutorial_js_face_detection.html#:~:text=Object%20Detection%20using%20Haar%20feature,detect%20objects%20in%20other%20images
- https://docs.nvidia.com/nemo-framework/user-guide/24.09/nemotoolkit/multimodal/vlm/clip.html
- https://medium.com/@alexppppp/how-to-train-a-custom-keypoint-detection-model-with-pytorch-d9af90e111da

- https://medium.com/@kabilankb2003/human-pose-estimation-with-pytorch-and-ros2-a-complete-guide-a95f4e79a3ef
- https://huggingface.co/google/vit-base-patch16-224#:~:text=Images%20are%20presented%20to%20the,to%20the%20layers%20of%20the
- https://arxiv.org/abs/2310.09757#:~:text=transformer%20,a%20joint%20representation%20to%20derive
- https://pmc.ncbi.nlm.nih.gov/articles/PMC10458371/#:~:text=CNN%20ResNet12%2C%20ResNet18%2C%20ResNet34%2C%20ResNet50%2C,GoogleNet%2C%20GoogleLeNetv2%2C%20LeNet%2C%20YOLOv3%2C%20EfficientNet
- https://openaccess.thecvf.com/content_CVPR_2020/papers/Mittal_EmotiCon_Context-Aware_Multimodal_Emotion_Recognition_Using_Freges_Principle_CVPR_2020_paper.pdf#:~:text=through%20experiments%20on%20EMOTIC%2C%20a,an%20improvement%20over%20prior%20methods
- https://medium.com/correll-lab/building-a-vision-transformer-model-from-scratch-a3054f707cc6
- https://preciousorekha1.medium.com/facial-emotion-recognition-for-mental-health-monitoring-ab3de9545428
- https://medium.com/analytics-vidhya/haar-cascades-explained-38210e57970d
- https://medium.com/correll-lab/building-clip-from-scratch-68f6e42d35f4
- https://ieeexplore.ieee.org/document/10229334
- https://www.analyticsvidhya.com/blog/2022/02/a-comprehensive-guide-on-hyperparameter-tuning-and-its-techniques/
- https://arxiv.org/abs/1512.03385
- https://docs.pytorch.org/vision/main/models.html
- https://arxiv.org/abs/1409.4842v1
- https://arxiv.org/abs/1409.4842
- https://pallawi-ds.medium.com/which-human-pose-estimation-model-should-you-pick-to-realise-your-ideas-for-a-video-analytics-6ca754cc1f4e