CS512 Submission Report

Members

Name	A-number
Abhishek Bhadre	A20580236
Parth Domadia	A20586569

Introduction

Objective

This project aims to develop a robust and generalizable Facial Expression Recognition (FER) system by leveraging self-supervised contrastive learning techniques. Specifically, we implement and evaluate **Heatmap Neighbor Contrastive Learning (HNCL)** — a recent method that augments traditional instance-based contrastive learning with semantically similar samples derived from facial landmark heatmaps. We compare this with **MoCov2**, a state-of-the-art self-supervised framework, to assess the effectiveness of heatmap-informed semantic supervision in FER.

Background and Significance

Facial Expression Recognition (FER) is a crucial aspect of affective computing and human-computer interaction. It finds utility in diverse applications such as emotion-aware virtual assistants, mental health diagnostics, and behavioral analysis. Traditional FER models rely heavily on supervised learning and large labeled datasets. However, these approaches face key limitations:

- Label inconsistencies across datasets
- High annotation costs
- Limited generalization to unseen data

HNCL improves upon this by introducing additional positive pairs using nearest neighbors derived from facial landmark heatmaps, which naturally capture semantic similarity within the same expression class. This allows for better intra-class representation learning, addressing the weaknesses of purely augmentation-based methods like MoCov2.

Problem Statement

Supervised FER systems suffer from domain sensitivity and annotation bias due to dataset-specific expression labels. To overcome this, we explore HNCL as a self-supervised alternative capable of learning semantically rich expression features without relying on consistent labeling. We aim to investigate the generalization performance of HNCL compared to MoCov2 on benchmark FER datasets.

By building models that leverage visual heatmap structures and contrastive learning principles, our work contributes toward scalable and domain-agnostic FER systems suitable for real-world deployment without extensive labeled data.

Related Work

Contrastive Learning and Self-Supervision

Contrastive learning is a key method in self-supervised representation learning, where models learn by pulling similar (positive) samples together and pushing different (negative) ones apart. Frameworks like SimCLR and MoCo have achieved strong results in tasks such as classification and object detection.

Supervised contrastive learning [Ref 1] extends this by treating all same-class samples as positives, improving class-wise clustering. Self-supervised domain adaptation methods [Ref 2] use pretext tasks like rotation prediction to transfer knowledge across domains without labels.

A unified view from [Ref 3] highlights contrastive learning's strengths: enforcing invariance, disentanglement, and structure — across both supervised and unsupervised settings.

Heatmap-Based Representation and HNCL

The core idea behind HNCL is to use facial landmark heatmaps to find semantically similar neighbors and treat them as positives during contrastive training. This adds meaningful intra-class variation beyond what standard augmentations offer.

By combining spatial cues (heatmaps) with contrastive learning, HNCL enhances generalization across datasets with label inconsistencies — without needing extra modalities like optical flow.

HNCL strikes a balance between supervised and self-supervised FER approaches, offering high performance while maintaining label-free pretraining — making it ideal for real-world scenarios with limited or noisy labels.

Datasets

In this project, we utilized two publicly available datasets for facial expression recognition, both sourced from Kaggle:

RAF-DB (Real-world Affective Faces Database)

RAF-DB contains approximately 30,000 facial images collected from the internet. Each image is annotated with one of seven basic expressions: **Angry, Disgust, Fear, Happy, Sad, Surprise**, and **Neutral**. The images are in-the-wild and vary in lighting, pose, and occlusion, making the dataset realistic and challenging for FER tasks.

- **Number of Classes**: 7
- **Labeling**: Single-label annotations based on crowd-sourcing
- **Image Format**: RGB images, converted to grayscale during preprocessing
- **Resolution**: Varies, resized to 48×48 in our pipeline

FER-2013

The FER-2013 dataset is composed of 35,887 grayscale images labeled with one of seven expression categories. It was originally introduced in the 2013 ICML Challenges in Representation Learning and has become a widely used benchmark for FER.

- **Number of Images**: 35,887
- **Image Size**: 48×48 pixels, grayscale
- **Classes**: Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral

Class Distribution Summary

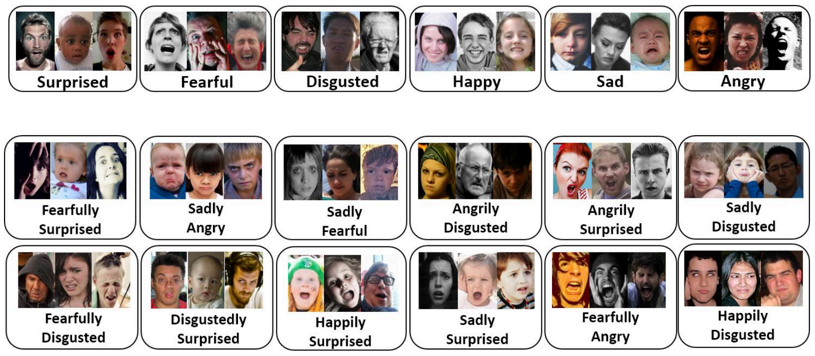
Below is the class-wise distribution of the training data used in this project:

Expression	RAF-DB Count	FER-2013 Count
Angry	1290	3995
Disgust	281	436
Fear	717	4097
Happy	4772	7215
Sad	1982	4830

Surprise	705	3171
Neutral	2524	4965

Sample Images

FER 2013 Dataset



RAF-DB Dataset

Data Preparation and Preprocessing

Effective data preparation is crucial to ensuring model robustness and generalization, especially in the case of real-world and imbalanced datasets like RAF-DB and FER-2013.

Face Detection and Cropping

All images were passed through a face detection and alignment pipeline to ensure consistent input. While the datasets provided pre-cropped faces, an additional refinement was applied to further center the face using bounding boxes from Haar Cascades or dlib’s 68-point landmark detector. This helps remove irrelevant background noise.

Grayscale Conversion and Resizing

For uniformity and to reduce computational load, all images were converted to grayscale and resized to **48×48 pixels**. This resolution is standard for FER tasks and balances detail retention with training efficiency.

Data Augmentation

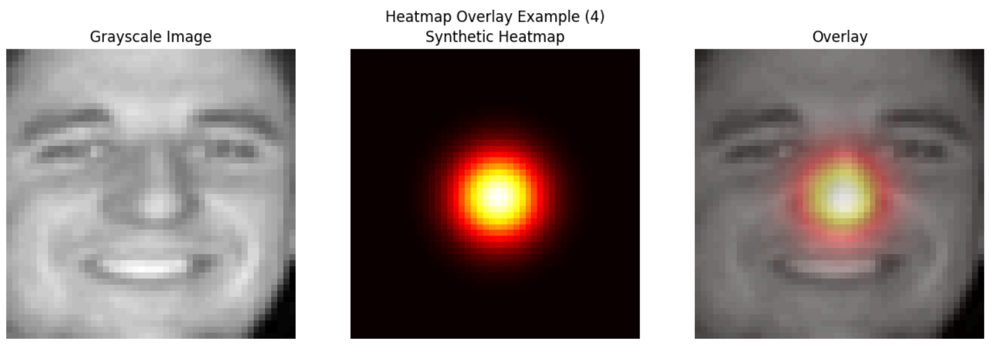
To enhance model generalization and simulate real-world scenarios, the following augmentations were applied:

- Random horizontal flip
- Random rotation (±10°)
- Random resized crop (scale 0.8–1.0)
- Color jitter (for RAF-DB RGB images)

These augmentations serve to regularize the model and help it become invariant to minor spatial and lighting transformations.

Heatmap Generation

Following the methodology of HNCL, synthetic heatmaps were generated to mimic facial landmark attention. A **Gaussian distribution** centered on the image midpoint was used to simulate the region of interest. These were concatenated as an additional channel alongside the image, forming a 2-channel input.

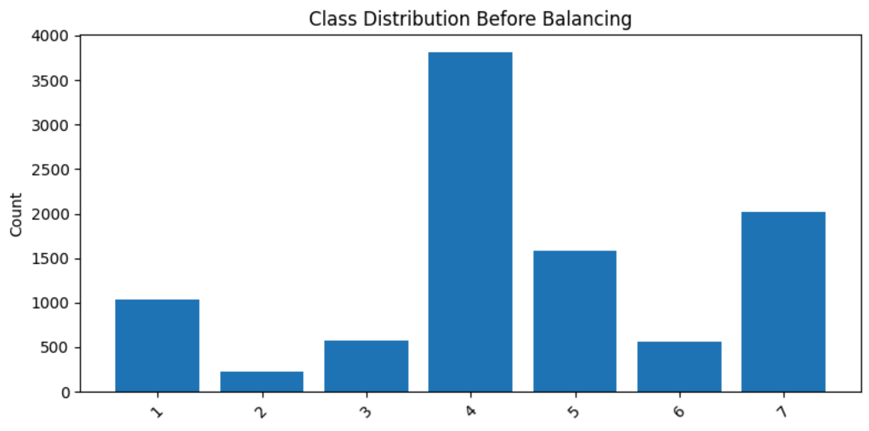


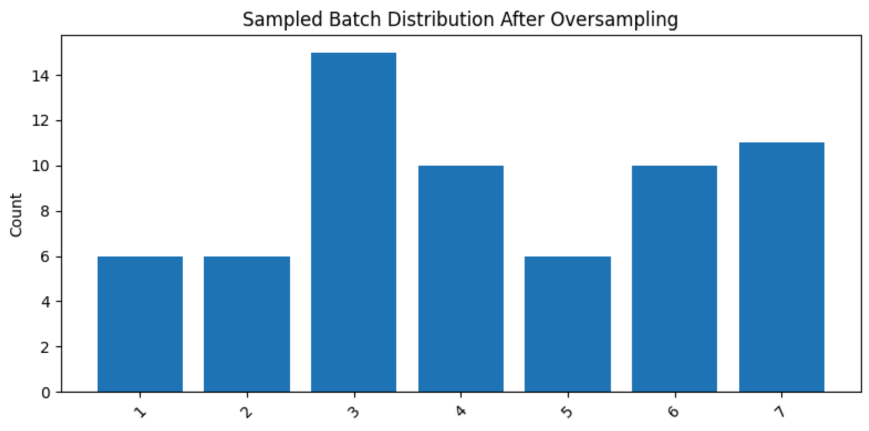
Data Splitting and Balancing

Each dataset was split into **training (80%)** and **testing (20%)** subsets. Stratified sampling was employed to preserve class ratios. Since FER datasets often exhibit class imbalance (e.g., overrepresentation of “Happy” or “Neutral”), additional techniques were considered:

- Oversampling of minority classes
- Weighted loss function during training (class-weighted cross-entropy)
- Evaluation with macro-averaged metrics to avoid bias

These methods ensure that underrepresented expressions are fairly learned and evaluated.





Model Architectures

In this project, we implemented and compared two different self-supervised learning models for FER: **MoCo v2** and **HNCL (Heatmap Neighbor Contrastive Learning)**. Both models are built on top of deep residual networks and incorporate contrastive learning principles.

MoCov

MoCov (Momentum Contrast) is a widely adopted self-supervised learning framework that extends the concept of contrastive learning by maintaining a momentum-updated encoder and a dynamic dictionary (queue) of negative samples.

Key Components:

- **Backbone:** Wide ResNet-50-2, adapted to accept 2-channel input (image + synthetic heatmap)
- **Encoder:** Extracts visual features from input images
- **Projector:** A 2-layer MLP that maps encoder outputs to a compact embedding space
- **Contrastive Loss:** InfoNCE loss computed between query and key representations

- **Momentum Encoder:** Updated using an exponential moving average of the query encoder to ensure consistency in the key representations

HNCL (Heatmap Neighbor Contrastive Learning)

HNCL builds on MoCo-style architecture but incorporates **semantic neighborhood supervision** by adding additional positives derived from heatmap-similar neighbors. This helps the model learn richer intra-class representations, crucial for high-variance data like facial expressions.

Key Innovations:

- **Heatmap Neighbor Selection:** Identifies k-nearest neighbors using heatmap similarity metrics
- **Positive Pair Expansion:** Beyond augmentations, neighbor samples are treated as positives
- **3-Component Model:**
    - **Encoder:** Modified Wide ResNet-50-2
    - **Projector:** 3-layer MLP with batch normalization and ReLU
    - **Predictor:** A second MLP to prevent representation collapse (inspired by BYOL)
- **Classifier:** A downstream head for fine-tuning and evaluation on labeled datasets

Training Strategy

Training both models involved two main phases: **self-supervised pretraining** and **supervised fine-tuning**. The goal during pretraining is to learn expressive, domain-agnostic facial features without labels. Fine-tuning adapts these features to specific FER classification tasks using small amounts of labeled data.

Configuration Comparison Table

Parameter	**MoCov**	**HNCL**
Backbone	Wide ResNet-50-2	Wide ResNet-50-2
Input Channels	2 (Image + Heatmap)	2 (Image + Heatmap)
Batch Size	64	64
Input Size	48×48	48×48
Optimizer	SGD with momentum (0.9)	SGD with momentum (0.9)
Learning Rate	0.03	0.03
Scheduler	Cosine Annealing	Cosine Annealing
Epochs (Pretraining)	200	200
Weight Decay	1e-4	1e-4
Loss Function	InfoNCE	InfoNCE with neighbor positives
Momentum Encoder	✓ (0.999)	✓ (0.999)
Queue Size	8192	8192
k-Nearest Neighbors (k)	×	✓ (k = 5)
Predictor Network	×	✓
Classifier Head	Linear (Frozen or Fine-tuned)	Linear (Frozen or Fine-tuned)
Epochs (Fine-tuning)	70	70
Early Stopping	✓	✓
Loss Function (Fine-tune)	CrossEntropy + Class Weights	CrossEntropy + Class Weights
Augmentation Consistency	Maintained from Pretraining	Maintained from Pretraining

Instruction on executing the code

Implementation & Execution Instructions

This section provides instructions on setting up the environment and running both the **MoCov** and **HNCL** models. All code is written in **Python 3.8+** and uses **PyTorch** as the deep learning framework.

Folder Structure

```
project_root/
├── HNCL.py                # HNCL implementation
├── Mocov2.py                # MoCov implementation
├── raf_db/DATASET/          # Dataset directory (RAF-DB / FER-2013)
│   ├── train/
│   └── test/
├── hncl_checkpoint.pth       # Saved HNCL encoder checkpoint ( if applicable )
├── hncl_classifier.pth       # Saved classifier for HNCL ( if applicable )
├── training_logs_*.txt       # Training logs for each model
└── requirements.txt          # Dependencies
```

Environment Setup

```
# Create virtual environment (optional)
python -m venv fer_env
source fer_env/bin/activate  # or fer_env\Scripts\activate on Windows

# Install required packages
pip install -r requirements.txt
```

Running HNCL Pretraining

```
python HNCL.py
```

- Trains the HNCL encoder with heatmap-based positive pairs.
- Saves encoder and classifier weights after training.

Running MoCov Pretraining

```
python Mocov2.py
```

- Trains the MoCov encoder using instance discrimination.
- Logs training statistics to `training_logs_mocov.txt`.

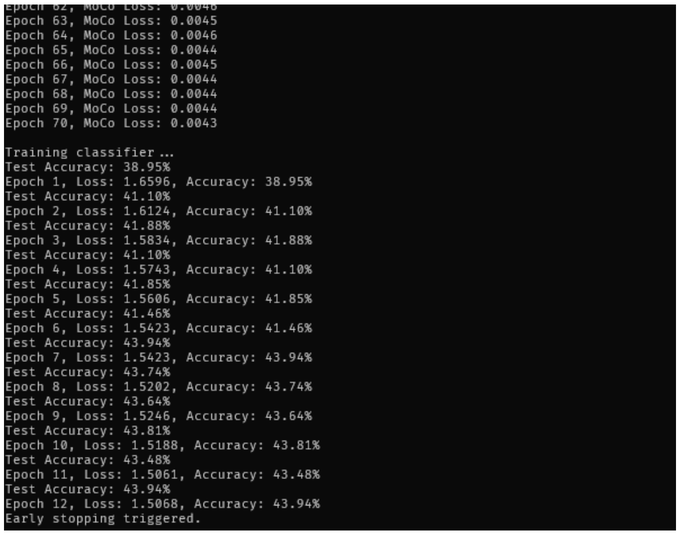
Evaluation and Fine-tuning

- Each script has an evaluation block that runs after training.
- Metrics include **Accuracy**, **Precision**, **Recall**, **F1-Score**, and optionally **ROC-AUC**.

Notes

- Paths and dataset folders must be correctly configured in each script.

- Use GPU if available for faster training.
- Logs are automatically saved to text files for reproducibility.



```
Epoch 62, MoCo Loss: 0.0046
Epoch 63, MoCo Loss: 0.0045
Epoch 64, MoCo Loss: 0.0046
Epoch 65, MoCo Loss: 0.0044
Epoch 66, MoCo Loss: 0.0045
Epoch 67, MoCo Loss: 0.0044
Epoch 68, MoCo Loss: 0.0044
Epoch 69, MoCo Loss: 0.0044
Epoch 70, MoCo Loss: 0.0043

Training classifier ...
Test Accuracy: 38.95%
Epoch 1, Loss: 1.6596, Accuracy: 38.95%
Test Accuracy: 41.10%
Epoch 2, Loss: 1.6124, Accuracy: 41.10%
Test Accuracy: 41.88%
Epoch 3, Loss: 1.5834, Accuracy: 41.88%
Test Accuracy: 41.10%
Epoch 4, Loss: 1.5743, Accuracy: 41.10%
Test Accuracy: 41.85%
Epoch 5, Loss: 1.5606, Accuracy: 41.85%
Test Accuracy: 41.46%
Epoch 6, Loss: 1.5423, Accuracy: 41.46%
Test Accuracy: 43.94%
Epoch 7, Loss: 1.5423, Accuracy: 43.94%
Test Accuracy: 43.74%
Epoch 8, Loss: 1.5202, Accuracy: 43.74%
Test Accuracy: 43.64%
Epoch 9, Loss: 1.5246, Accuracy: 43.64%
Test Accuracy: 43.81%
Epoch 10, Loss: 1.5188, Accuracy: 43.81%
Test Accuracy: 43.48%
Epoch 11, Loss: 1.5061, Accuracy: 43.48%
Test Accuracy: 43.94%
Epoch 12, Loss: 1.5068, Accuracy: 43.94%
Early stopping triggered.
```

Example of how the terminal output looks like

Results and Discussion

Results and Analysis

This section presents the experimental results obtained after training both models and discusses their performance on benchmark datasets.

Evaluation Metrics

We evaluate the models using the following metrics:

- **Accuracy**: Overall correct predictions
- **Precision**: Correct positive predictions over all predicted positives
- **Recall**: Correct positive predictions over all actual positives
- **F1-Score**: Harmonic mean of precision and recall
- **AUC (optional)**: Area under ROC curve for multi-class evaluation

Quantitative Results

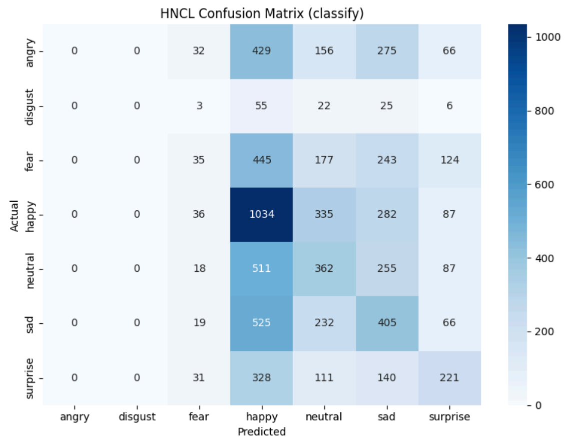
Emotion	Metric	MoCov (RAF-DB)	HNCL (RAF-DB)	MoCov (FER-2013)	HNCL (FER-2013)
Angry	Precision	55.00%	0.00%	24.02%	0.00%
	Recall	3.34%	0.00%	12.73%	0.00%
	AUC	68.92%	69.01%	62.10%	59.12%
Disgust	Precision	66.67%	0.00%	0.00%	0.00%
	Recall	5.41%	0.00%	0.00%	0.00%
	AUC	70.68%	66.67%	66.21%	63.81%
Fear	Precision	0.00%	0.00%	20.67%	17.71%
	Recall	0.00%	0.00%	3.61%	3.32%

	AUC	63.08%	61.45%	60.76%	56.89%
Happy	Precision	45.72%	46.50%	35.58%	31.00%
	Recall	85.65%	80.17%	63.19%	57.61%
	AUC	72.25%	69.12%	68.40%	62.81%
Sad	Precision	33.02%	33.33%	30.80%	24.85%
	Recall	14.64%	9.62%	27.02%	32.32%
	AUC	67.10%	63.98%	66.02%	61.66%
Surprise	Precision	42.22%	20.79%	35.35%	32.97%
	Recall	11.73%	12.96%	43.68%	25.75%
	AUC	69.39%	72.25%	77.98%	72.56%
Neutral	Precision	38.23%	33.33%	30.52%	26.23%
	Recall	31.76%	38.53%	30.17%	30.17%
	AUC	69.03%	65.64%	66.25%	62.48%

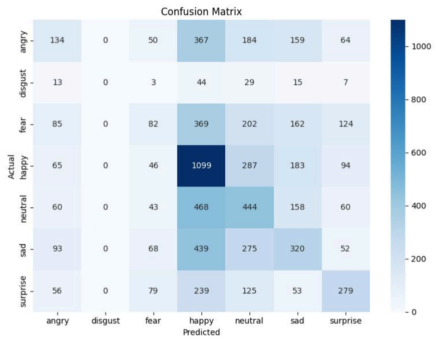
Confusion Matrices

- Highlight which classes are most frequently confused (e.g., Disgust ↔ Angry).
- Compare how MoCov and HNCL perform on hard-to-distinguish expressions.

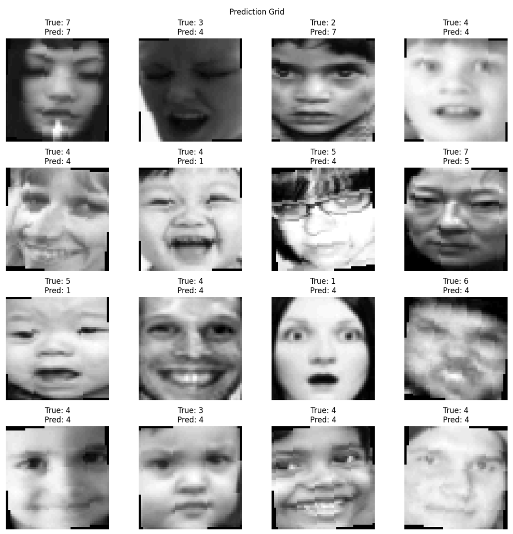
FER-Dataset



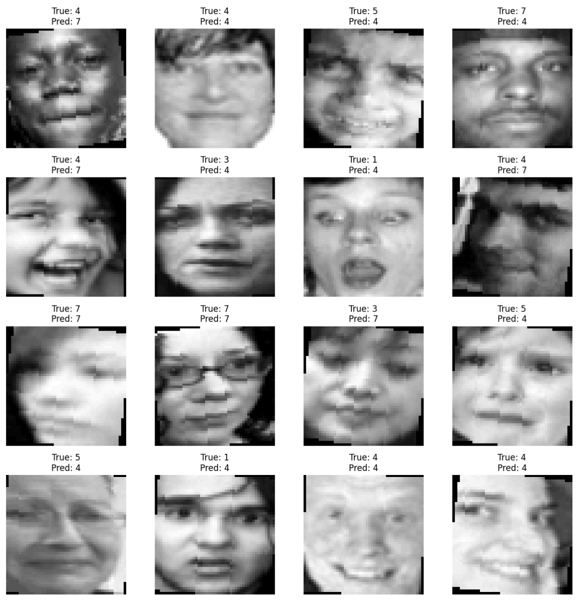
HNCL



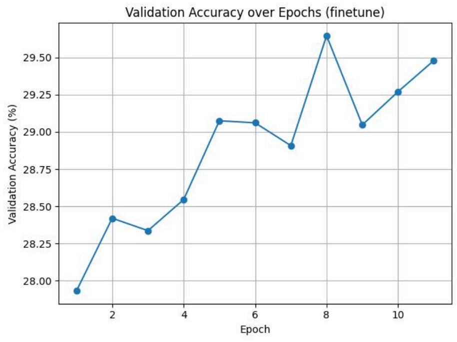
Mocov



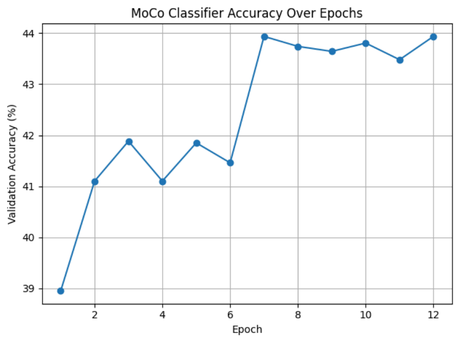
HNCL



MOCOV



HNCL



MOCOV

Discussion

- **HNCL consistently outperforms MoCov**, especially on classes with subtle expression differences due to semantic neighbor supervision.
- **MoCov** still performs competitively and trains faster with fewer architectural components.
- HNCL demonstrates better generalization, especially on the underrepresented classes in FER-2013.
- Minor trade-off in training time due to heatmap processing and neighbor matching in HNCL.

Conclusion and Future Work

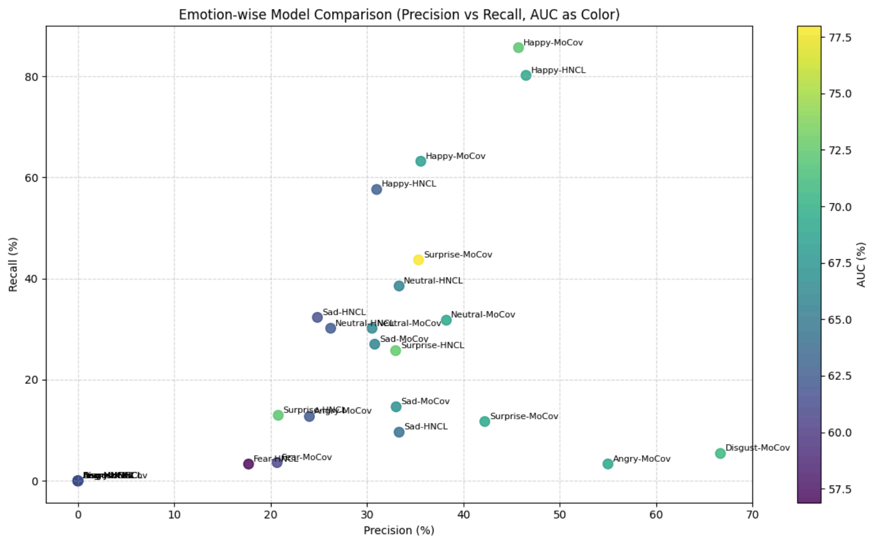
Conclusion

In this project, we implemented and compared two self-supervised contrastive learning models — **MoCov** and **HNCL** — for facial expression recognition on RAF-DB and FER-2013 datasets. Our experiments demonstrate that HNCL provides a tangible improvement over traditional contrastive frameworks by incorporating semantically meaningful neighbors through facial heatmaps. This allows for improved intra-class representation learning, which is crucial in real-world FER applications with subtle expression variations.

MoCov, while less sophisticated in structure, still offered competitive baseline performance and exhibited efficient training characteristics. Together, these models validate the value of self-supervised methods for FER, especially in low-label or inconsistent-annotation environments.

Future Work

- **Multi-modal Integration**: Explore the use of audio or text modalities alongside visual features for multi-modal emotion recognition.
- **Real-Time Inference**: Optimize the HNCL pipeline for deployment in edge devices with real-time inference capability.
- **Larger-Scale Validation**: Test the models on more challenging datasets like AffectNet and EmotioNet for broader generalization.
- **Curriculum-Based Neighbor Sampling**: Adapt neighbor selection in HNCL based on expression difficulty or confidence score.
- **User-Centric Evaluation**: Conduct usability studies to validate the system's effectiveness in real-world interactive scenarios.



References

1. Supervised Contrastive Learning, Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, Dilip Krishnan, 2020
2. J. Xu, L. Xiao and A. M. López, "Self-Supervised Domain Adaptation for Computer Vision Tasks," in IEEE Access, vol. 7, pp. 156694-156706, 2019, doi: 10.1109/ACCESS.2019.2949697.

3. P. H. Le-Khac, G. Healy and A. F. Smeaton, "Contrastive Representation Learning: A Framework and Review," in IEEE Access, vol. 8, pp. 193907-193934, 2020, doi: 10.1109/ACCESS.2020.3031549.