INM460 Computer Vision: Facial Emotion Recognition

Brenner Swenson | 190046406 | adbc288 | 9 May, 2021

***Abstract*—This work investigates the usage of three different model configurations for the task of facial emotion recognition (FER). An end-to-end classification pipeline is constructed containing a Viola-Jones face recognition algorithm; a support vector machine with Scale Invariant Feature Transform feature descriptors, a multi-layer perceptron with Histogram of Gradients feature descriptors, and a deep convolutional neural network (DCNN) are examined as the pipelines' classifiers. The DCNN is identified to greatly outperform the other methods in both accuracy and speed.**

Google Drive link: *https://drive.google.com/drive/folders/1CCoKG_V-w2BWA2IzxiDLu8C0ONURWTok?usp=sharing*

I. Introduction

A. Context

A person's mood can be conveyed physically, or verbally. Moods are conveyed verbally via conversation, tone of voice, etc. and physically via movements, gestures, and most importantly facial expressions. Facial expressions are one of the best ways that humans decipher the mood of another person, how they should approach someone, or gauge a someone's reaction. Facial expressions are widely accepted to be indicators of a universal set of emotions e.g. sadness, happiness, and disgust. [1] This work aims to use a variety of machine learning methods such as SIFT and HOG feature descriptors as well as deep convolutional neural networks (DCNN) to classify human emotions based on facial expressions. In order to classify facial expressions, a facial recognition classifier must also be implemented.

B. Related works and methods

Today, state of the art results on facial emotion recognition (FER) are almost exclusively achieved via DCNNs. The variables that materially differentiate state-of-the-art implementations from one another are the type of network: either a CNN, Restricted Boltzmann Machine (RBM), or Deep Belief Network (DBN); the size of the network in both layers and number of hyperparameters; pre-processing techniques: Viola-Jones face detectors [2], IntraFace detectors [3], data normalisation, and illumination normalisation; and the use of additional classifiers on the end of a network such as an SVM, or K-NN. It should be mentioned that this work focuses only on static-based methods, meaning that previous frames in a video are not considered during classification. There is no temporal aspect.

Many successful implementations, such as Yu et al. [4], concatenate several feature extractors and facial landmark detectors on top of one-another in a complementary manner. Other implementations, like Kim et al. [5], utilise multiple inputs in to their systems: the original image, a histogram equalized image, different face detection models, and proceed with the landmark with the highest confidence.

DCNNs require a high amount of training data in order to achieve generalisation, and most of the FER datasets available to the public do not contain more than a few thousand images. To mitigate the issue of overfitting and provide the models with varied data to train on, data augmentation methods are vital for FER. The most effective type of data augmentation is known as on-the-fly augmentation, where during the training step, the image is augmented randomly during each epoch. This contrasts with offline augmentation where the data is augmented once and saved externally as another dataset. Common augmentation steps include random cropping, random horizontal flipping, colour jittering, Gaussian blur, rotation, etc. The use of many augmentation transformations in conjunction with one-another can essentially generate never-before-seen images for a network to train on. General Adversarial Networks (GAN) can also be utilised to augment data by creating new expressions and poses. [6] Many successful implementations of FER utilise pre-trained models and build upon them (e.g., AlexNet, GoogleNet, VGG-Face). In addition to building upon pre-existing models, additional data such as datasets primarily for facial recognition (FR) can be used to further refine FER results. Kahou et al. [7] found that the use of auxiliary FR data is successful in achieving better generalisation.

As stated previously, the CNN is the building block of most highly-performant systems for FER. The differentiating factor between CNN architectures can be auxiliary blocks or layers like Meng et al.'s use of a novel island loss layer that enhanced the discriminative power of deeply learned features for FER. [8] In addition to individual network architectures, the combination of multiple networks, known as an ensemble, can outperform individual networks. [9] Of these mentioned architectures and systems, many are novel and too involved for the scope of this coursework. Consequently, the most complex architecture will be a 10-layer CNN.

II. Data

A. *Dataset*

The Real-world Affective Faces (RAF) dataset consists of 15,399 images taken from the internet. Each image is labelled with one of seven emotions: surprise, fear, disgust, happiness, sadness, anger, or neutral. The dataset is distributed with a pre-defined training/testing split: 12,271 images for training, and 3,068 for testing. For validation purposes, an additional validation subset of 2,455 images (20% of training data) was created. Each image is 100x100 pixels; they are also already aligned to the subject's face as seen in Figure 1, saving an additional pipeline task.

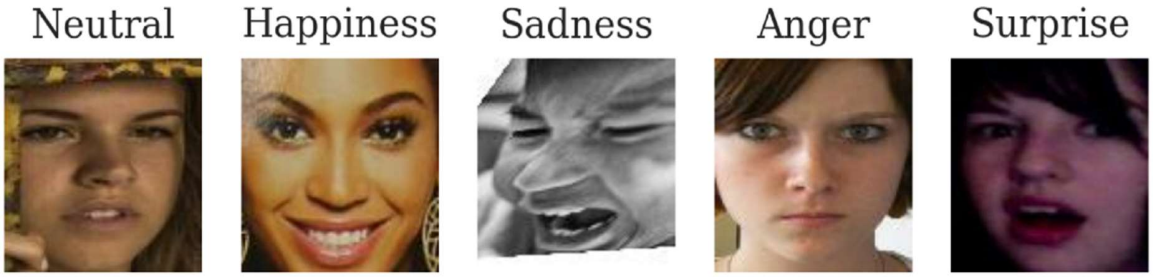

Figure 1. Selected images from training dataset with emotion labels.

B. *Augmentation*

Since the training dataset is limited to 9,816 images, image augmentation is a necessity. With the use of PyTorch's large selection of transforms [10], with each epoch (when training neural networks), effectively a new dataset is introduced, highly reducing overfitting. In Figure 1, 5 random examples are shown before and after augmentation. An image can have random colour jittering, Gaussian blur, grayscale conversion, horizontal flip, affine transform up to 15 degrees, and random erasing (the black bars). Qualitatively throughout the training process, the affine transform and random erasing have the largest impact on generalisation when tuning the augmentation pipeline.

Figure 2. Random selection of images before and after random image augmentation.

C. *Class imbalance*

As provided, the RAF dataset does not contain an equal distribution of emotions to train models with. As seen in Figure 3, the "happiness" emotion contributes to 38.89% of the training data. Without measures to counteract this class imbalance, the models' ability to generalise to other types of emotions will be hindered significantly, and as seen with the SVM model later, if this is not accounted for, the results suffer greatly.

Luckily, for the models that use PyTorch in this coursework, a weighted random sampling technique is readily available. For both the HOG-MLP and CNN implementations, during each batch, samples are randomly chosen without replacement to achieve an equal class distribution.

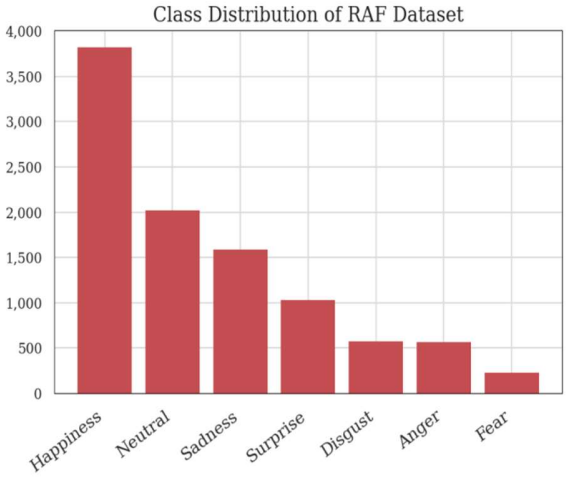

Figure 3. Large class imbalance of RAF dataset.

III. Emotion Recognition

The task of FER requires several steps: locating the faces in an image, pre-processing the image for compatibility with chosen feature extractors, feature extraction itself (HOG, SIFT, CNN), and classification. Fortunately, when training the model, the facial recognition portion is not needed, but when testing on unseen images/footage, it is. The following section will define and provide an overview of the feature extractors used, the model architectures, as well as the full implementation and achieved results.

A. *Feature Extraction*

1) *SIFT*

The Scale Invariant Feature Transform (SIFT) was developed by David Lowe in 1999. Intuitively, SIFT is invariant to translations, rotations, and scaling transformations and robust to moderate perspective transformations and illumination variations. [11]

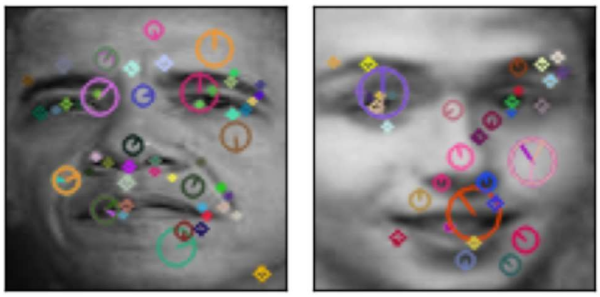

Figure 4. SIFT key points overlayed on select training images.

SIFT works by locating scale-space extrema via Difference of Gaussians (DoG) approach. Key points are then localized and assigned orientations by placing a small window around the key points and calculating the gradient magnitude. A histogram containing the orientations is then created where each pixel then votes for an orientation bin. Each vote has a weight equal to the gradient magnitude. The largest histogram bin, or the peak, ultimately decides the local dominant orientation, which is assigned to the key point. [12] In Figure 4, the orientation can be seen in the key points on subjects' faces as the line pointing toward the edge of the circles.

With key points established, feature descriptors can then be generated for each one. To do so, a small window is positioned on each key point and the orientation and gradient magnitude are computed for the entire window. The window orientations are then rotated in the same direction as the key point's dominant orientation. The same window is then divided into smaller regions. Another smaller histogram is created with 8 bins, and each pixel votes for an orientation bin of the subregion, with the weight relative to the magnitude of the gradient. [12]

2) HOG

HOG feature descriptors, or Histograms of Ordered Gradients, were originally developed by Dalal and Triggs in 2005. [13] Their original task was human detection, which it excelled at. The HOG feature extraction pipeline is very similar to SIFT; it utilises gradient magnitude and orientation, but instead of focusing on detected interest points like SIFT, HOG is defined for several grid cells in the detection window. [14]

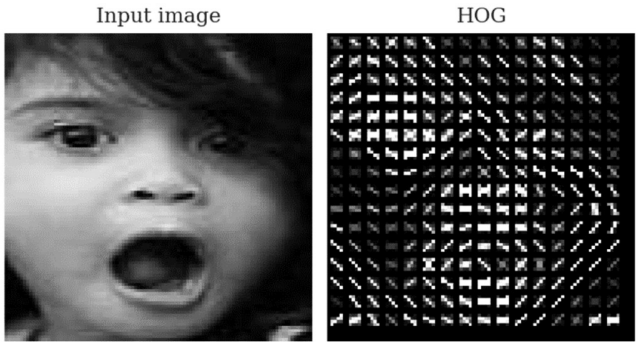

Figure 5. HOG feature descriptor on a sample training image.

Dahmane and Meunier (2011) [15] utilized HOG feature descriptors and an SVM for an FER task. They noted that HOG is primarily used for more general object detection algorithms, since HOG descriptors mainly work as edge detectors. They did not explore the use of neural networks, which can inherently handle higher dimensional data. Their method was conservative in their HOG feature descriptor hyperparameters; primarily so they did not have to use methods like PCA to reduce dimensionality for the SVM. For this reason, in this coursework, HOG was not used in conjunction with an SVM, and instead with a neural network.

B. Models

1) SIFT - Support Vector Machine

The first model and feature extractor combination investigated was a SIFT feature descriptor in conjunction with a support vector machine (SVM). The SVM algorithm is constructed by the linear function $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$, trying to generate the best line to separate classes for optimal classification performance. Figure 6 illustrates the creation of a maximum-margin plane between two classes. [14] SVMs can classify in infinite dimensions using hyperplanes constructed via the kernel trick. Because not all data are linearly separable, the SVM's learning process effectively replaces the $\boldsymbol{x}$ in the above equation with the output of a given kernel function. These functions allow the model to learn non-linearly in an n-dimensional space. [16] A radial basis function kernel was chosen for the FER task of this coursework.

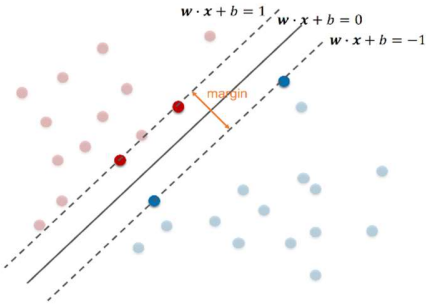

Figure 6. Example of SVM creating optimal margin between two classes.

This work's approach utilises a Bag of Visual Words (BoVW), which is very similar to an approach in natural language processing. The approach loosely follows the following steps: extraction of SIFT feature descriptors, clustering (mini KMeans in this applications) to create codewords, creation of codeword histograms for each individual image, and then the training of the classifier (SVM in this case). The implemented SVM's C parameter was 100 and a radial basis function kernel.

2) HOG - Multi-layer Perceptron

The multilayer perceptron (MLP), a supervised learning model, is one of the first widely used artificial neural networks. The MLP attempts to simulate the neurons in a brain; many perceptron algorithms are concatenated or cascaded with one another, where the output of one neuron becomes the input to another. [17] The connections between each layer are weighted; using input labels to the training data and back-propagation, the network can adjust the weights between each and every neuron to slowly approximate a function to classify or predict a target value. [16] The goal of training a neural network is to converge slowly to an optimum using a loss function suitable to the task at hand.

The second model architecture investigated in this coursework was a combination of the HOG feature descriptor and an MLP. As previously stated, HOG feature descriptors are typically used for object detection, so consequently, very high resolution is required when attempting to classify facial expressions in order to detect minor differences in the human face.

Layer	Input Size
Fully connected 1	14,112
Dropout (p=0.05)	7,056
Fully connected 2	7,056
Dropout (p=0.05)	3,528
Fully connected 3	3,528
Dropout (p=0.05)	1,764
Fully connected 4	1,764
Softmax (7 classes)	7

Table 1. HOG-MLP architecture.

Table 1 contains the neural network architecture used with the HOG feature descriptors. The input size of the network is a direct result of the options that were chosen when creating the HOG outputs. The resulting hyperparameters were 8 orientations, 6 pixels per cell, and 3 cells per block. During experimentation, as the size of the network grew, so did the accuracy, but so did the training time. Due to the large input size, several additional layers were added to maintain the relationships between neurons. To better promote generalization, dropout layers with 10% probability were added in between each layer. This dropout percentage was reduced until the validation loss and training loss were decreasing at a very similar rate. The final hyperparameter configuration included 6 pixels per cell (HOG), 8 orientations (HOG), 3 cells per block (HOG), a batch size of 32, a maximum learning rate of 0.001, and trained over 100 epochs.

3) Convolutional Neural Network

Deep Convolutional neural networks (DCNN) have largely replaced most feature extraction pipelines for image classification like SIFT, SURF, HOG, etc. due to their efficiency and large increases in accuracy. DCNNs utilize convolutional layers that move a window/kernel across an entire image and generating a scalar output for each position traversed by the window. [18]

Layer Block	Out Filters	Input Shape
Conv + ReLU + BN + Dropout	32	3 x 100 x 100
Conv + ReLU + BN + MaxPool + Dropout	64	32 x 100 x 100
Conv + ReLU + BN + Dropout	128	64 x 50 x 50
Conv + ReLU + BN + MaxPool + Dropout	128	128 x 50 x 50
Conv + ReLU + BN + Dropout	256	128 x 25 x 25
Conv + ReLU + BN + MaxPool + Dropout	256	256 x 25 x 25
Flatten	-	256 x 5 x 5
Linear + Dropout + ReLU	-	6,400
Linear + Dropout + ReLU	-	1,024
Linear + Dropout + ReLU	-	512
Softmax	-	7

Table 2. Implemented CNN architecture

Each neuron of each convolutional layer is only able to see a certain region of the previous layer's output, so each neuron and subsequently each outputted feature map are looking for certain things in the image, like edges, or eyes, for example.

DCNN models can learn features directly from the data itself as opposed to the reliance of feature descriptors that have been historically used. This is possible in part due to the model's ability to capture the spatial dependence and structure within images using local operators

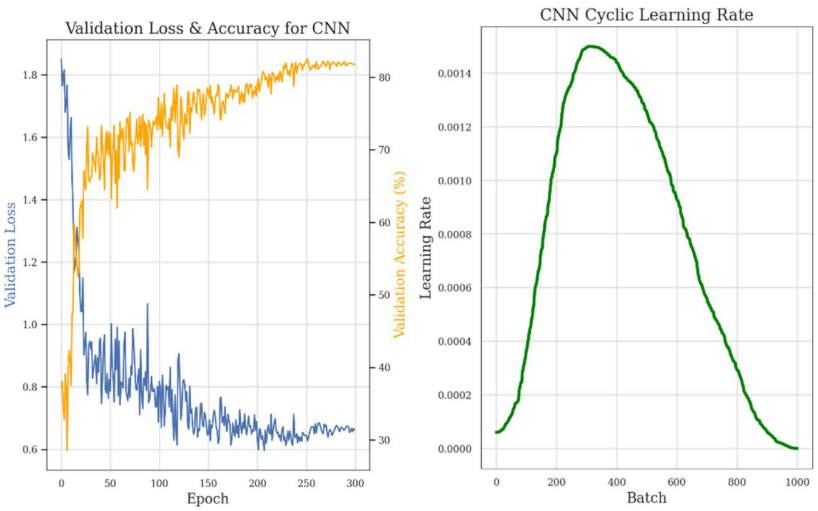

Figure 7. Validation loss and accuracy for the CNN across 300 epochs (left). The cyclic learning rate used during training (right)

and the hierarchical make-up of the extracted feature maps. For this reason, the SIFT-SVM and HOG-MLP greatly underperform the CNN in this work, because they are unable to establish spatial relationships within image.

Table 2 contains the architecture used for classifying facial emotions in this coursework. The model takes the 100 x 100 image as input; the architecture follows a pattern of one convolutional layer to increase the number of filters, then another layer of convolution to further increase filters and then max pooling. This structure allows the DCNN to learn more, because there is not a down-sampling operation in each layer. After three convolution/pooling blocks, the tensor is flattened and passed through 3 linear layers each decreasing in size.

The best DCNN was trained across 300 epochs with a cyclic learning rate, a batch size of 32, a dropout rate of 0.25 between most layers, and a weight decay constant of 0.0001. The learning rate steadily increased until epoch 150 when it reached 0.00125, then decreased again until 300. This type of learning rate schedule proved to be extremely useful in preventing overfitting. The validation accuracy and loss can be seen in Figure 7 along side its cyclic learning rate.

C. *Implementation*

1) Facial recognition

The RAF dataset contains faces that are pre-aligned, meaning that in order to train, no face recognition is required. To test the models on data other than the provided test set, a face recognition algorithm is necessary. A Viola-Jones [2] technique utilizing a large set of Haar wavelets. OpenCV provides a robust implementation of this technique via their CascadeClassifer object.

Due to the dynamic nature of face orientation in video frames and un-aligned images, the frontal-face default Haar cascade model implemented in OpenCV fails to recognize faces when they are not directly facing the camera. When classifying video frames, subjects tend to move a lot, and the bounding boxes found by OpenCV can be very sporadic. To rectify this, a second Haar cascade model trained on subjects in the profile orientation, from the side, was used in conjunction with the default model. For each frame in a video, both models look for any faces they can find, then the bounding box coordinates are compared with one another, and if they are similar enough with one another, then the default face is taken (this is to avoid duplicates), otherwise, the profile face is added to the array of found faces.

Figure 8. Only front-facing Haar cascades for face detection (Left). Ensemble of front-facing and profile Haar cascades with annotated emotion classifications. (Right)

As seen in Figure 8 on the left, the default Haar wavelet model in OpenCV does not do an adequate job at finding faces in the profile orientation. When used in conjunction with the profile-face model (right), two more faces are detected in the image that were not on the left. However, one subject was not detected; this is due to the profile-face model being trained only on left-facing people, so those facing right will not be detected.

2) Video

The full pipeline for video-annotation with emotion classifications can be seen in Figure 9. After loading all video frames via OpenCV, the faces are detected using the ensemble method described above. To fit the input size of the CNN model (the model with best performance), the images are then resized. The images are then passed to the classifier and classifications are obtained; the classes are converted to their labels (e.g., Happiness), and the annotated on each individual frame along with the bounding box surrounding each subject's face. It should be mentioned that the minimum face size during detection is dynamic to the resolution of the video itself—i.e., the resolution divided by some factor. A factor of 10 generalized well for most applications.

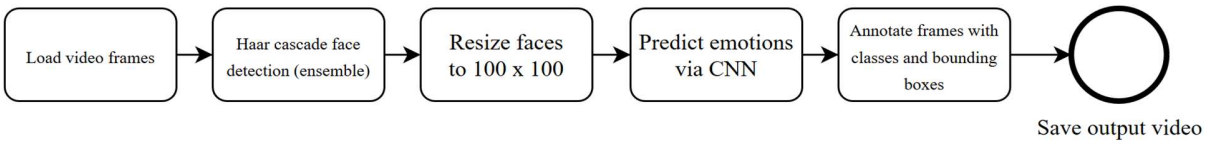

Figure 9. Process diagram detailing how a video is annotated with emotion classifications.

3) Results

As expected, the CNN greatly outperformed the other feature-model configurations when evaluated on the provided testing set. As seen in the confusion matrix for the CNN model in Figure 10, the class imbalance found in the training set is also present in the test set. If no action were taken to counteract this, the "happiness" column would contain much higher numbers, since it is most popular class.

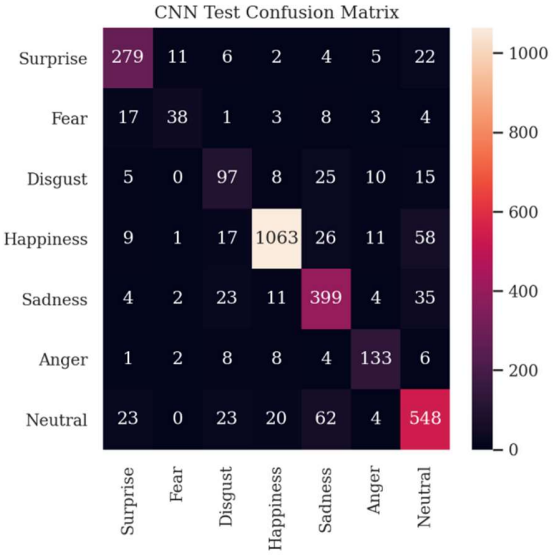

Figure 10. CNN confusion matrix for RAF test dataset

The ramification of class imbalance is seen in the results of the SVM classifier that was not trained via a batched learning approach where stratified random sampling could be implemented. Table 3 contains selected performance metrics of the three implemented models; the SIFT-SVM model performed abysmally, and that can especially be seen in the F1 score.

Model	Accuracy	Recall	Precision	F1
SIFT-SVM	40.12	40.12	36.60	37.23
HOG-MLP	68.02	68.02	72.71	69.46
CNN	83.34	83.34	83.82	83.47

Table 3. Performance metrics for selected models on RAF test dataset

The HOG-MLP configuration performed relatively well with an overall accuracy of 68.02%. However, the training time was much, much longer when compared with the CNN as the vector sizes are much larger, and it takes a significant amount of time to generate the HOG feature descriptors during each training batch.

Figure 11. Frame from video after emotion annotation.

Only the CNN was implemented in the video pipeline as it performed so much better than the other methods. Not only does it perform better, but it is able to classify images faster as a feature descriptor does not need to be generated. A still from an annotated video can be seen in Figure 11; the classifications accurately represent the emotions within the restrictions of the labels in the dataset.

D. Discussion and conclusion

This coursework attempts to build an end-to-end pipeline for facial emotion recognition, a popular and difficult task in computer vision. Three different model configurations were investigated: a combination of SIFT feature descriptors with an SVM, HOG feature descriptors with a multi-layer perceptron, and a deep convolutional neural network. As hypothesized, the DCNN generalized the best with 83.34% accuracy on the testing dataset, and qualitatively excellent results on unseen data in the form of videos.

The CNN performs very well once given aligned faces; the struggle is accurately locating faces in video frames or images. Due to the different aspect ratios of videos and the size of a potential face in a video, the Haar wavelet method of face recognition can take quite a long time to tweak/tune and obtain satisfactory results. Further improvements and investigations can be made in to increasing not only the accuracy and recall of face recognition in diverse images, but also in the speed of the face-recognition process as it is quite slow and could not be implemented on a live video feed.

Recent research has proven that the use of generative adversarial networks in data augmentation can be extremely helpful in promoting generalization. [6] Through the implementation of facial image synthesis similar to Zhang et al.'s work, the CNN's performance could improve drastically.

IV. References

[1] P. Ekman, "Facial expression and emotion," *Am. Psychol.*, vol. 48, no. 4, pp. 384–392, 1993, doi: 10.1037/0003-066X.48.4.384.

[2] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001, vol. 1, doi: 10.1109/cvpr.2001.990517.

[3] F. De La Torre, W. S. Chu, X. Xiong, F. Vicente, X. Ding, and J. Cohn, "IntraFace," Jul. 2015, doi: 10.1109/FG.2015.7163082.

[4] Z. Yu and C. Zhang, "Image based Static Facial Expression Recognition with Multiple Deep Network Learning," doi: 10.1145/2818346.2830595.

[5] B. K. Kim, H. Lee, J. Roh, and S. Y. Lee, "Hierarchical committee of deep CNNs with exponentially-weighted decision fusion for static facial expression recognition," in *ICMI 2015 - Proceedings of the 2015 ACM International Conference on Multimodal Interaction*, Nov. 2015, pp. 427–434, doi: 10.1145/2818346.2830590.

[6] F. Zhang, T. Zhang, Q. Mao, and C. Xu, "Joint Pose and Expression Modeling for Facial Expression Recognition."

[7] S. E. Kahou *et al.*, "Combining Modality Specific Deep Neural Networks for Emotion Recognition in Video," 2013, doi: 10.1145/2522848.2531745.

[8] J. Cai, Z. Meng, A. S. Khan, Z. Li, J. Oreilly, and Y. Tong, "Island loss for learning discriminative features in facial expression recognition," in *Proceedings - 13th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2018*, Jun. 2018, pp. 302–309, doi: 10.1109/FG.2018.00051.

[9] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3642–3649, doi: 10.1109/CVPR.2012.6248110.

[10] "torchvision.transforms — Torchvision master documentation." https://pytorch.org/vision/stable/transforms.html (accessed Apr. 30, 2021).

[11] T. Lindeberg, "Scale Invariant Feature Transform," *Scholarpedia*, vol. 7, no. 5, p. 10491, 2012, doi: 10.4249/scholarpedia.10491.

[12] G. Tarroni, W. Bai, and G. Slabaugh, "Computer Vision INM460/IN3060 Lecture 5 Image matching: interest point detection and feature descriptors." Accessed: May 01, 2021. [Online]. Available: http://www.city.ac.uk/feedback.

[13] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, 2005, vol. I, pp. 886–893, doi: 10.1109/CVPR.2005.177.

[14] G. Tarroni and W. Bai, "Computer Vision INM460/IN3060 Lecture 6 Image classification: linear classifiers, support vector machines and ensembles."

[15] M. Dahmane and J. Meunier, "Emotion recognition using dynamic grid-based HoG features," in *2011 IEEE International Conference on Automatic Face and Gesture Recognition and Workshops, FG 2011*, 2011, pp. 884–888, doi: 10.1109/FG.2011.5771368.

[16] B. Swenson, "Classifying Music Genres | A Comparison of Multilayer Perceptrons and Support Vector Machines Using Convolved Mel-Spectrograms," 2020.

[17] G. Tarroni and W. Bai, "Computer Vision INM460/IN3060 Lecture 7 Image classification: neural networks."

[18] G. Tarroni and W. Bai, "Computer Vision INM460/IN3060 Lecture 8 Image classification: convolutional neural networks."