German University in Cairo GUC
Faculty of Media Engineering and Technology
Prof. Dr. Mohammed Abdel-Megeed Salem
Eng. Dinah Waref

DMET1067—Deep Learning for Computer Vision
Spring Semester 2024

DLCV Project

Milestone 1 Due: Friday 04.04.2025 at 11:59 pm.
Milestone 2 Due: Friday 25.04.2025 at 11:59 pm.
Milestone 3 Due: Friday 02.05.2025 at 11:59 pm.

This project aims to build an emotion classification system from scratch, using movie frame images. You will create our own dataset by extracting frames from films, both color and grayscale formats are acceptable. The core task is to classify the emotions within each frame based on the facial expression. You will begin by collecting the dataset from different movies, then using preprocessing techniques to ensure the images are optimized for the model. Following this, you will design and construct a deep learning model from scratch for emotion classification. This involves defining the network architecture and implementing the training process. Finally, you will test the model's performance on unseen data to evaluate its accuracy in classifying emotions.

You will be working in teams of four. You should submit your team details in the following form: https://forms.gle/iPGBYR9hcdqvVhzTA The deadline for team submission is *Friday 14/03/2025*.

1. Project Milestone 1: (10%)

Emotion recognition is identifying and categorizing human emotions from facial expressions, voice recordings, or text inputs. It has applications in human-computer interaction, market research, and mental health monitoring. Common emotions include happy, sad, angry, surprised, and neutral.

Classes: Happy, Sad, Angry, Surprised, Neutral

Milestone 1 Objectives:

The main objective of this milestone is to create a training and testing dataset.

1- Each team will choose a classification task and create the dataset for it.

2- The dataset created should be extracted from Egyptian movies.

3- Each team will address as many classes as the number of members.

4- The team should collect 100 frames per class. That is for a team with 4 members, they should collect 400 frames for 4 classes. For a team with 3 members, they should collect 300 frames for 3 classes.

5- Challenges such as occluded/rotated faces need to be addressed. This will be considered during grading. Each picture should contain only one clear, centered face. Please ensure the face is easily visible and avoid submitting duplicate or highly similar images. Ensure clear, unobstructed faces for accurate classification.

Technical Notes:

To have a consistent dataset that we can all use, let's fix some technical features.

1- File names and type: Frames selected to be added to the dataset should be saved as PNG images and should be named based on the following template:

C_T_N.png

> C is the class name as appears in the column "class short name". Use capital letters only.
>
> T is 2 digits representing the team. Please be sure that it is fixed in length and unique.
>
> N is 4 digits serial number. It starts with 0 and is fixed to be 4 digits.

2- Sampling rate: To avoid redundant frames, you are asked to save key-frames only in the dataset. There are serval ways to extract key frames from a video, use any of these. Any algorithm, code, or tool could be shared freely across teams.

An alternative solution is to reduce the frame rate to 10 frames per second at most, then apply a manual inspection step to delete blurred and very similar frames.

3- Image dimensions: The frame dimension should be fixed to 640x480 (landscape).

4- Image colors: It is better to consider colored movies, but gray-level movies could be included too.

Delivery:

Submit the created data using the following form: https://forms.gle/LSnrysSFazG2aes37 Put the files of each class in a (*.zip) file. Thus, if a team of 4 members has completed the data for 4 classes, then a team member should use this form to upload 4 (*.zip) files.

1- The deadline is on 04.04.2025 at 11:59 PM.

3- A deduction of 5% for the late submissions till 5:59 AM of the next morning.

4- A deduction of 50% for the late submissions after 6:00 AM on 10.04.2025.

5- The deliverables are (in one zip file named the M1-TeamName-[ID1_ID2_ID3_ID4], rar files are not accepted):

a. The zip file should be updated on Google Drive and a link to the zip submitted to the form. Make sure you have sharing on.
b. The class folders inside should be zip files

c. The notepad file (.txt) includes your names, IDs, and emails. Name the file "names.txt".

Grading:

The total point for this milestone is 100. The points will be calculated as min(number of uploaded frames/(100*number of team members), 10). Any delayed delivery will be subject to the above deduction.

Evaluation:

1. Manual and subjective assessment based on image quality, quantity, and adherence to guidelines.
2. Milestone 1 contributes 10% to your overall project grade.

Milestone 2: (70%)

In this milestone, you are tasked to build models to classify the images into one of the defined classes in Milestone 1. You will learn to create a Convolutional Network from scratch *(without the use of any external libraries)* in the first model. In the 2nd model, you can use ML libraries to create and train CNN for image classification.

The model will be trained on four classes of your choice from the shared dataset you collectively created.

Data Preparation: (5%)

1- Use the dataset you created in Milestone 1.

2- Download the images and store the images of each class in separate folders.

3- Split the data for the selected classes into training/validation and testing with ratios of 70/20/10. (2%)

4- Remove any redundant images. (1%)

5- Resize the images to be 512x512x3. (2%)

First Model: (35%)

Create a convolution network from scratch for feature extraction. Then use the extracted feature vector with a simple unsupervised algorithm for data clustering. To avoid the need for the backpropagation algorithm, we will use predefined filters for the convolution layers.

1- Create a single convolution layer using 5 predefined 3x3x3 filters. Input an image of 512x512x3. (10%)

a. Create a class ConvLayer which has at least two initialization methods

i. Given the number and the size of the filters. This method should create random filter weights.

ii. Given the number and the size of filters and a matrix in 3D that has the filter weights.

b. Use the following filters to initialize an object of the class ConvLayer:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

(a) (b) (c) (d) (e)

2- Create a class PoolingLayer which is initialized by the size of the pooling filter (default 2x2) and the pooling type is either (default) MAX or AVERAGE. (5%)

3- For both the ConvLayer and PoolingLayer create methods to (8%)

i. Iterate the filters on the entire input image/ matrix.

ii. Run a forward pass methods

4- Pass the output of the pooling layer to a simple activation function defined as (2%)

$$\sigma(z) = \begin{cases} z & z \geq 0 \\ 0 & otherwise \end{cases}$$

5- Use the above-created Convolution block to create a network as: (8%)

i. 3 Convolution blocks

ii. Flattening layer

iii. Downsample the output of the flattening layer to the size 1x128.

6- Use the k-means algorithm to classify the images into different classes. (2%)

Second Model: (30%)

Use your favorite DL libraries to create a classification CNN with the following architecture conditions:

1- Build a CNN with 3 convolutional layers, each of which uses ReLU as an activation function, followed by a 2x2 max-pooling filter. Use the default stride for the max-pooling filters. (suggested) (8%)

2- The filter sizes are 3x3, 3x3, 3x3, 5x5, and 7x7, and use 32, 64, 64, 32, and 16 filters for the successive convolution layers, respectively. Use valid spatial convolution only. (suggested) (6%)

3- The convolution blocks are followed by a flattening operation. (3%)

4- Use the flat vector as an input to fully connected layers with only one hidden layer. Use the Sigmoid as the activation function for this layer. (6%)

5- Use the Softmax function for the output layer. (4%)

6- All other hyper-parameters than what has been mentioned here are left to be determined by the developing team. (3%)

Milestone 2 Delivery:

All deliverables must be compressed together in one (*.zip) file. Name the file as T_M2.zip, where T is the team name. Upload the zip file to Google Drive and submit the link to the following form by Friday, April 25, 2025, at 11:59 PM: https://forms.gle/gUB5feYosz4YitMn9

1- Submit a Jupiter notebook file or .py for your well-commented code. Specify in clear text the selected hyper-parameters and any used libraries.

2- A PowerPoint slide or report for the visualization of the created model architecture annotated with the dimensions of the layers and in/out data. Present the selected classes with some samples of the inputs and the obtained results in the same file. Add one slide for the model hyper-parameters, the number of training iterations, and any additional notes.

3- A textfile (*.txt) with the team information including the names and contacts of the team members.

4- Please note that plagiarism or AI-generated codes will result in a zero grade

5- Evaluations will be held and part of your grade depends on your performance in the set evaluations.

German University in Cairo GUC
Faculty of Media Engineering and Technology
Prof. Dr. Mohammed Abdel-Megeed Salem
Eng. Dinah Waref

DMET1067—Deep Learning for Computer Vision
Spring Semester 2024

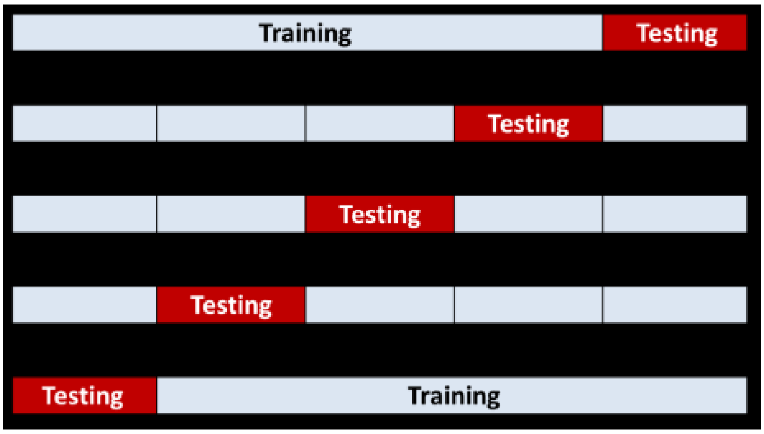
Milestone 3 Objectives: (20%)

This last milestone of the project aims to finalize and report the project outcomes.

Therefore you are requested to complete the following tasks:

1- Build a curve for the accuracy vs the number of iterations. (3%)

2- Perform K-fold cross-validation and report the resulted accuracy and the average of the recorded K-accuracies. The minimum chosen value for K is 4. (5%)



3- Compute and report the confusion matrix. (4%)

Multi Class Confusion Matrix		Predicted				*Accuracy*
		Class 1	Class 2	...	Class N	
Actual	Class 1	M_{11}	M_{12}	...	M_{1N}	
	Class 2	M_{21}	M_{22}	...	M_{22}	
	...	...	...	...	...	
	Class N	M_{N1}	M_{N2}	...	M_{NN}	
Reliability						*Overall Accuracy*

- Several important information can be observed from the confusion matrix such as:

1-	Accuracy of a class:	$ACx = (x,x) / \sum_y (x,y)$
2-	Overall Accuracy:	$AC = \sum_x (x,x) / \sum_x \sum_y (x,y)$
3-	Overall Error rate:	$ER = \sum_x \sum_{y\, y \neq x} (x,y) / \sum_x \sum_y (x,y)$

1. The entries in the main diagonal represent the number of correct classified pixels, for example (1,1) represent the number of pixels classified as being belong to class 1 while they were actually belonging to class 1.
2. The entries of the form (x,y) such that x ≠ y, represent the number of pixels which were wrongly classified to belong to class y, while they are actually belong to class x.

4- Build a block diagram for your network architecture. Add in bullet points all chosen values for the hyperparameters. (4%)

5- Report any steps of preprocessing or post-processing of the data. (2%)

6- Compile your work report, results, analysis, and discussion in one Word document or PowerPoint presentation. (2%)

Milestone 3 Delivery:

All deliverables must be compressed together in one (*.zip) file. Name the file as T_M3.zip,

where T is the team name. Upload the zip file to Google Drive and submit the link to the following form by Friday, May 10, 2024, at 11:59 PM: https://forms.gle/G5MbQjHdrCheCZQB9

1- Submit a Jupiter notebook file or .py for your well-commented code. Specify in clear text the selected hyper-parameters and any used libraries.

2- A PowerPoint slide or report for the visualization of the created model architecture annotated with the dimensions of the layers and in/out data. Present the selected classes with some samples of the inputs and the obtained results in the same file. Showing all the visualization and graphs you created as well as an explanation of them

3- A textfile (*.txt) with the team information including the names and contacts of the team members.

4- Please note that plagiarism or AI-generated codes will result in a zero grade

5- Evaluations will be held and part of your grade depends on your performance in the set evaluations.

Bonus: (3%)

We're giving a 1.5% bonus for each of the extra tasks below completed, but there's a limit of 3% in total, meaning if you do one bonus task you get 1.5% and if you do two you get 3% which is the maximum. The first four are bonus tasks for Milestone 2 and should be submitted with it, and tasks 5 to 6 are bonus tasks for Milestone 3 and should be submitted with it.

1. Data Augmentation: Implement data augmentation techniques such as rotation, flipping, and cropping to artificially increase the size of your dataset and improve the generalization of your models.
2. Model Regularization: Experiment with different regularization techniques such as L1/L2 regularization, dropout, and batch normalization to prevent overfitting and improve the generalization of your models.
3. Hyperparameter Tuning: Conduct a hyperparameter tuning process using techniques such as grid search or random search to find the optimal values for parameters such as learning rate, batch size, and number of filters in the convolutional layers.
4. Transfer Learning: Explore the use of pre-trained convolutional neural network models (e.g., VGG, ResNet, or Inception) as feature extractors and fine-tune them on your dataset to potentially improve classification performance.
5. Advanced Evaluation Metrics: In addition to accuracy, compute other evaluation metrics such as precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC) to gain a more comprehensive understanding of your model's performance.
6. Model Deployment: Investigate methods for deploying your trained models into production environments, such as converting them to optimized formats (e.g., TensorFlow Lite or ONNX) for inference on mobile or edge devices.
7. Error Analysis: Perform an in-depth analysis of the misclassified examples to identify common patterns or challenges that your models struggle with, which could provide insights for future improvements.