Micro gesture classification model training on images using CNN

BM40A0801 - MACHINE VISION AND DIGITAL IMAGE ANALYSIS

ABSTRACT

CNN model based on AlexNet architecture for image-level micro gesture classification, aiding in non-verbal communication analysis and human-computer interaction.

STUDENTS

Durbar Hore Partha 001473021

Content: Pages

1.Abstract:

Classifying micro gestures in the form of images level is a very significant area in Machine Vision and Digital Analysis. In this study, a deep convolutional Network technique is used to present an unique method for categorizing micro movements acquired in photos. From a large dataset of images in various class the project trains a model using the dataset and then evaluate the model performance using a test dataset.

Keywords: Image classification, Micro gestures (MG), Deep learning, AlexNet, PyTorch.

2.Introduction:

Micro gestures are minute motions that are frequently missed by people while watching, yet they can elucidate important information that is necessary to classify human interaction patterns in some extent human behavioural studies. Classifying human micro gestures from images extracted from random videos is a challenging image analysis problem in computer vision. This study aims on classifying the minute gestures from a dataset of images. In this work, we have 32 folders in datasets which represents 32 classes of micro-gestures. To solve the classification problem with the dataset with multiple classes we tackle by automatically detecting and classifying micro movements using deep learning techniques, specifically convolutional neural networks (CNNs). Convolutional Neural Network is a very frequently used technique for modelling training data and use the model in test dataset to get good accuracy in modelling. The training model was built by a deep methodology of CNN in which the architecture was established in multiple layers. By utilizing an AlexNet architecture which is a deep Convolutional Network, specifically designed to handle the complexity of micro gesture recognition, the model trains significant classification performance. For the training model of the big data-set we needed to establish a model with a large learning capacity. With potential object recognition, current approach makes essential use of Machine Learning Methods. **Alexnet** boosts GPU for training performance too[ref]. Although the used architecture requires high performance of multiple GPU's, this model training is done with success and a test with test data set was established for evaluating the model well.

3.Goal of the project:

The primary objective is to achieve good accuracy on the test subset using the training subset for the training model. This objective of the project is to create a deep learning model using CNN that can correctly classify micro hand movements and gestures at the picture level. The model works on classification of images into predetermined categories by analyzing tiny movements captured in photos using the AlexNet CNN architecture. The work aims at improving the interpretation of nonverbal cues and human-computer interaction systems by predicting human gesture on test dataset. Thus, project focuses on training a model using training dataset that provides good accuracy in predicting the human gestures from the test dataset.

4.Literature Review:

Convolutional Neural Network is a revolutionary approach in the field of image analysis and classification. The literature review focuses on key approaches and benefits of a CNN architecture. Early CNN models like LeNet , AlexNet are the foundation of complex architectures.

Training a large dataset requires optimization techniques that can be efficient in optimizing the model. Batch normalization mechanism, proposed by Ioffe and Szegedy, works by normalizing activations accelerate training by reducing internal covariate shifts. [r]

Despite being very complex, image analysis with CNN architectures with multiple layers can be regarded as efficient methodology and can bring success in image classification. Early research on micro gesture categorization different methods of conventional machine learning techniques like Random Forests and Support Vector Machines (SVMs) were used. These methods basically rely on manually created characteristics of the features extracted that are taken from pictures, like colour histograms and texture descriptors. Although somewhat successful, these techniques may not be able to fully capture the intricate patterns, complexity of gestures and variances found in micro movements.

Convolutional neural networks (CNNs) have been more and more popular among researchers as a means of classifying micro gestures since deep learning became available. For this purpose, architectures such as AlexNet, VGG, and ResNet have been modified and optimized. Deep learning has an advantage over handcrafted features since it can automatically learn hierarchical features from raw data, perhaps capturing complex patterns in micro movements more efficiently.
AlexNet is very popular deep CNN technique because of its easy implementation process and functions. **AlexNet** was Presented by Krizhevsky et al. and it was the first deep neural network (NNN) to win the 2012 ImageNet Large Scale Visual Recognition Challenge, which marked a major revolutionary advancement in computer vision.
Its design, which includes several convolutional and max-pooling layers before fully linked layers, makes it ideal for a range of image classification applications of images of large dataset.

Nevertheless, issues like dataset imbalance, occlusion, and lighting variability persist in image-level micro gesture categorization while working with AlexNet CNN. Robust preprocessing procedures, data augmentation tactics, and model regularization techniques are necessary to reduce the drawbacks and overcome these issues. To guarantee the model's effectiveness and generalization ability, it is also essential to assess its performance on various datasets and real-world situations. In order to improve the implementation in image-level micro gesture categorization, current research in this area keeps investigating novel architectures, training approaches, and evaluation methods.

5. Method Description:

Our method to solve the problem of micro gesture classification at the image level is based on deep learning and makes use of the AlexNet architecture. The architecture of the network we used is summarized in Figure 1 which contains eight learned layers — five convolutional and three fully-connected.This procedure entails a number of crucial steps that are made possible by significant operations and procedures:

5.1 Preparing the dataset

The dataset comes up with 32 folders (classes) and there are images of different classes with different gestures which are to be trained in our model. We used **CustomDataset** Class for getting the dataset ready to train the CNN model. It navigates images in the directory using the 'glob' library which is used to classify photos into groups that correspond to distinct micro gestures in those classes. In addition, the class manages data loading and also applies modification of the image pixels if needed.
transforms.Compose: To compose a series of transformations to be applied to the input images present in the classes the model uses the **torchvision.transforms()** function. Scriptable transformation is only possible in this case[2]. The own transformation of the images is executed.

Data split: Then the dataset is split into two groups as the requirement of training and testing. For training 80% of the data are taken into consideration. Based on the training of training dataset the rest of the data are taken for testing. After successful training the test dataset is used for testing our model. Splitting the dataset hence was very important to assess the model after training.

Normalization is used in this instance to standardize the pixel values in the image. Dataset was onboarded in google collab by mounting from google drive.

5.2. Model Architecture:

The AlexNet architecture is implemented using PyTorch. The model consists of five convolutional layers followed by max-pooling layers, which capture hierarchical features from input images. Subsequently, adaptive average pooling is applied to generate fixed-size feature maps. The fully connected layers further process these features to make predictions.

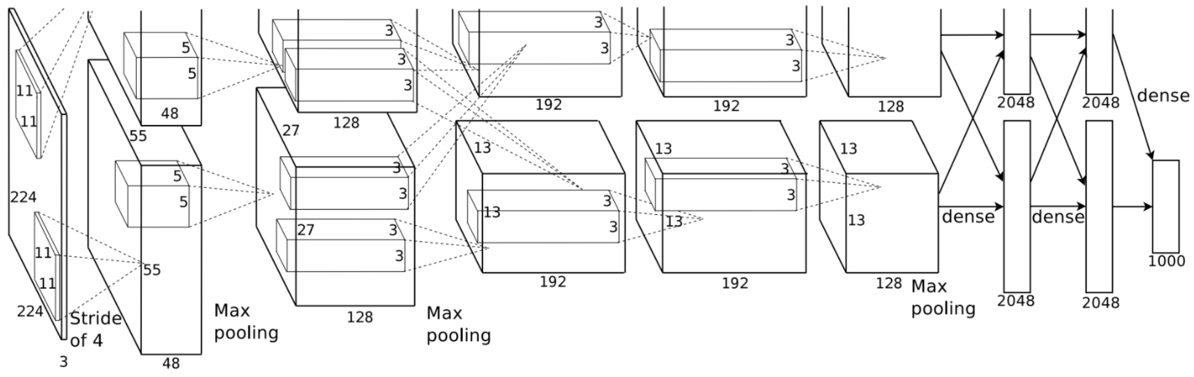


Figure1: An illustration of architecture of CNN, explicitly showing the delineation of responsibilities of GPUs. [1]

AlexNet Class: This class constructs the architecture of the AlexNet model and implemented using PyTorch. It is constructed by the combination of several convolutional layers (5 in this case), followed by layers for max-pooling, batch normalization, and ReLU activation. To read and extract photos read by Open-CV it is architected for modelling.

torch.nn.Sequential: The layers of our Convolutional Network are stacked sequentially using sequential containers which helps to make it simple for implementation.
torch.nn.Conv2d: This function defines the convolutional layers with values for padding, stride, and kernel size. The values could have been assigned busing matrices but we used this function.
torch.nn.BatchNorm2d: To stabilize the data onboarding and speed up the training process, batch normalization is used after convolutional layers.
torch.nn.Rectified Linear Unit (ReLU): We use this as activation function which adds non-linearity to the model so that it can pick up intricate patterns.
torch.nn.MaxPool2d: Downsampling the feature that maps through max-pooling layers and reduces spatial dimensions and extracts dominant features.

5.3. Training:

The Model is trained using the prepared dataset. Training is done across a number of epochs, iterating over the whole dataset throughout each epoch. The model uses momentum-assisted stochastic gradient descent (SGD) to optimize its parameters during training. To reduce prediction errors, cross-entropy loss is used as the optimization criterion. By computing the loss and accuracy at each iteration and epoch, training progress is recorded and stored with execution time.

torch.optim.SGD: Frequently used for parameter optimization, a stochastic gradient descent (SGD) optimizer with momentum is used. It adjusts the model parameters based on the learning epochs in an effort to minimize the loss function using computed gradients.
torch.nn.CrossEntropyLoss: This optimization criterion computes the loss between the actual and anticipated class labels using the cross-entropy loss function.
DataLoader: It is used to effectively load training data in batches which enables functions like sampling and shuffling to increase the efficacy of training.
Model Training Loop: Several epochs of the dataset are iterated through in the training loop. Forward pass, loss computation, backward pass (gradient computation), and parameter update are carried out at each iteration.
Calculating Accuracy: To determine the model's accuracy, the percentage of correctly classified samples is calculated by comparing the predicted labels with the ground truth labels.

5.4. Evaluation:

Using a different test set, the model's performance is assessed following training. Test accuracy is calculated to evaluate how well the model generalizes to new data. Plots are also used to illustrate training and validation loss and accuracy and examine training progress over epochs.

6.Model Evaluation:

Using a different test set, the model's performance is assessed following training. Using the test data, the trained model generates predictions, and the accuracy is calculated to see how well it can generalize. The model is evaluated on the basis of the training model to the test data set. As the data set was splitted into training and test data. After training the train set the model is used in test data set and the evaluation is assessed based on accuracy of the predictions on test data set.

Visualization:

Here in the project the training loss, accuracy in each epoch and test accuracy is visulaized. Matplotlib tool is used to display accuracy as well as training and validation loss. These graphics offer perceptions into the training process and aid in spotting possible problems like under- or overfitting.

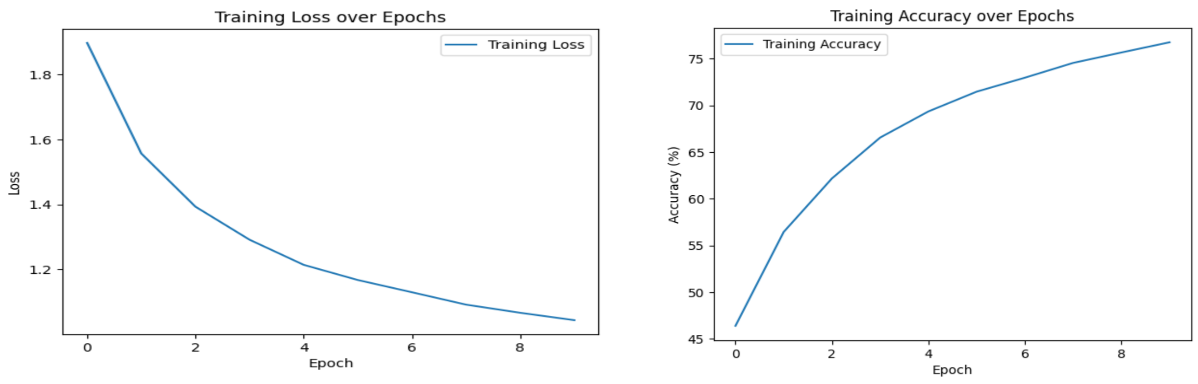
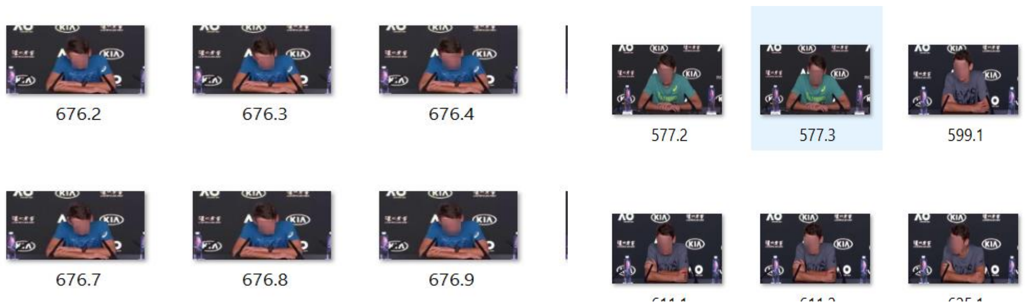


Figure: Training loss and Training accuracy

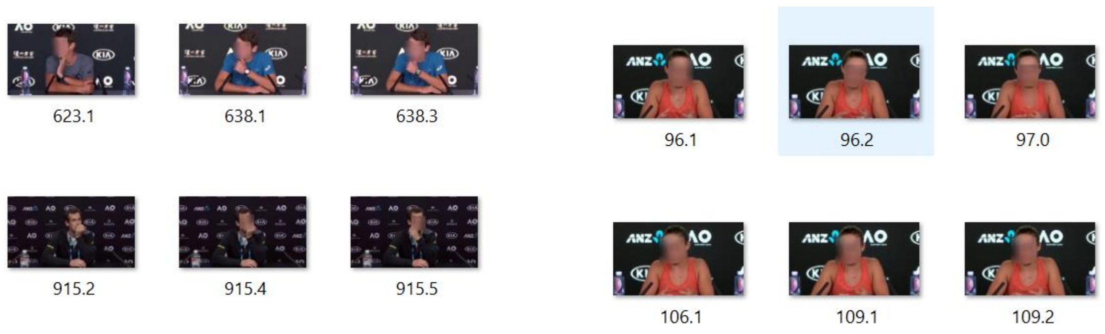
7.Experiments and Results:

For experimenting the model different batch sizes, epochs and number of classes are taken into consideration. First of all for the experimenting process first five classes are taken into consideration for testing and training. The results are very satisfactory as it increases with every iteration and epoch. Around 90% accuracy is achieved by taking first 5 classes into consideration.



Images: Similar gesture in first few classes

After successful implementation of first 5 classes and getting good accuracy the full data set is taken into consideration. There are total 32 classes in the data set of 60514 number of images.
In first 5 classes the images and the gestures are almost similar.



Images: Gesture and position differences

Thus, it is understandable to get very high accuracy from the given data set. But when all the 32 classes (folders) are taken into consideration the debugging time takes a long period of time and accuracy varies from 10% in initial iteration and it achieves around 70% after successful compilation of batch size 30 over 50 epochs and 32 number of classes.When all the 32 classes (folder) are taken into consideration, it is seen that there are big differences in the data of each class. When the training model is used in test data set the accuracy varies from 34% to 76%. For this huge amount of data this accuracy can be regarded as satisfactory.

With the experiment with test data set we execute the trained model for our test data set. There is a separate testing code for testing the model and accuracy calculation. With each iteration over epochs it shows significance improvement in modelling. Although the accuracy that achieved in testing is pretty low comparing with the testing.

The experiment using the Alexnet convolutional Network the model is able to provides accuracy for this huge dataset. With iteration and more cpochs more accurate results can be achieved using this model.

8.Discussion:

The training model takes number of classes: 32 (folders); 10 epochs; 30 batch sizes and iteration as per the length of the data loader. Over iterations this model shows increasing accuracy and in data set the prediction accuracy increases over iterations. With visualization we can observe the increasing accuracy over epochs. Decreasing loss in training model in each epoch is also observed by visualizing the model. Apart from Alexnet CNN, clip based method was also used to train the model. But due to irregular accuracy AlexNet is chosen for modelling purpose. Also, it is important that Pytorch is used for modelling. For reading the huge set of data Open-cv python is used so that the data set is read with more efficiency. Thus, the result of ever-increasing accuracy is observed. Although the model can be more accurate taking more epochs and batch sizes into modelling. Due to time constraint of google collab GPU it was not sufficient for modelling. For better result it is advised to increase batch sizes and epochs.

For testing the testing data set is tested for accuracy and accuracy is calculated. Although the accuracy is not satisfactory but it can be said that the model training and testing is established without error.

9.Limitations

For the huge dataset the first limitation was data on boarding. In google collaborator to mount the dataset with google drive took a lot of time for rendering. Also, it took a significant amount of time for the dataset to be read. High power V1000 and A1000 was used to train the model.

Another limitation is higher loss rate. Over epochs the loss rate is very high. Even after the training is done and tested it shows very high loss rate. This can be happened for the limitations of using AlexNet CNN in this study. Although the model reads the images and path with classes very well using Open CV python it lacks lower loss over epochs.

Also the training time was a big limitation as the GPU by google collab comes with limitations. Google Collab Pro is used to use high RAM GPU in T4, A1000 and V1000.

In this project it was tried to optimize the time constraint and onboarding dataset. The limitations can be mitigated by utilizing this model for multiple GPU with higher RAM, more epochs and batch sizes for modelling and integrating other Convolutional model.

10.Conclusion

We can conclude by stating that, this study focused on a modelling that trains the training dataset to generate good accuracy. The model tries to increase accuracy of the test data set by each iteration. When the model is tested in test dataset it shows accuracy in iterations and over epochs. But both of the cases loss is high and execution time is also high. Thus this model should be improved in future for lower loss rate over epochs. Although the limitations of handling number of classes may make this model slow but it was directed to improve in more epochs and batch size. This model can be more improved taking the classes into consideration with clip based model or other CNN methods.

REFERENCES

- Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*.
- Gherardi, S., 2022. Human micro-gesture recognition by adversarial training.
- Goh, K.M., Ng, C.H., Lim, L.L. and Sheikh, U.U., 2020. Micro-expression recognition: an updated review of current trends, challenges and solutions. *The Visual Computer*, *36*, pp.445-468.
- https://github.com/linuxsino/iMiGUE/blob/main/README.md
- https://link.springer.com/article/10.1007/s11263-023-01761-6
- https://github.com/mikecheninoulu/SMG
- Yuan, Z.W. and Zhang, J., 2016, August. Feature extraction and image retrieval based on AlexNet. In *Eighth International Conference on Digital Image Processing (ICDIP 2016)* (Vol. 10033, pp. 65-69). SPIE.

- Yuan, Z.W. and Zhang, J., 2016, August. Feature extraction and image retrieval based on AlexNet. In *Eighth International Conference on Digital Image Processing (ICDIP 2016)* (Vol. 10033, pp. 65-69). SPIE.