GROUP 37

Kilian Provost – 12235595,

Oguzhan Atmaca- 12231244,

Ylli Parduzi 12014533

Deep Image Processing

Image classification using Convolutional neural network

Introduction

Image classification is a fundamental task in computer vision, enabling machines to automatically categorize and interpret visual data. Convolutional Neural Networks (CNNs) have emerged as a powerful deep learning technique for image classification, offering state-of-the-art performance in a wide range of applications. By leveraging the hierarchical and spatial structure of images, CNNs are capable of learning complex patterns and features directly from raw pixel data, without the need for handcrafted feature extraction. In this report, we present a comprehensive study on image classification using CNNs, investigating their architecture, training methodologies, and evaluation techniques. We explore the potential of CNNs in accurately recognizing and classifying objects within images, showcasing their effectiveness and robustness across various datasets. Additionally, we delve into the challenges associated with CNNs, such as overfitting and computational complexity, and discuss strategies for addressing these issues. Through our analysis and experimental results, we aim to provide insights into the capabilities, limitations, and best practices of CNNs for image classification, paving the way for further advancements in this exciting field.

Dataset: 1 : Pubfig83

Dataset description

The Pubfig83 dataset is a widely recognized benchmark dataset in the field of computer vision and face recognition. It was introduced as a collection of face images for the purpose of evaluating and advancing the development of face recognition algorithms. The dataset consists of 83 individuals from various backgrounds, including celebrities, politicians, and public figures. Each individual is represented by a diverse set of images, capturing different facial expressions, lighting conditions, and angles. The Pubfig83 dataset has been extensively used by researchers and practitioners to develop and evaluate face recognition models, enabling advancements in facial recognition technology and contributing to the broader field of computer vision research.

Here are the names and number of instances in each class:

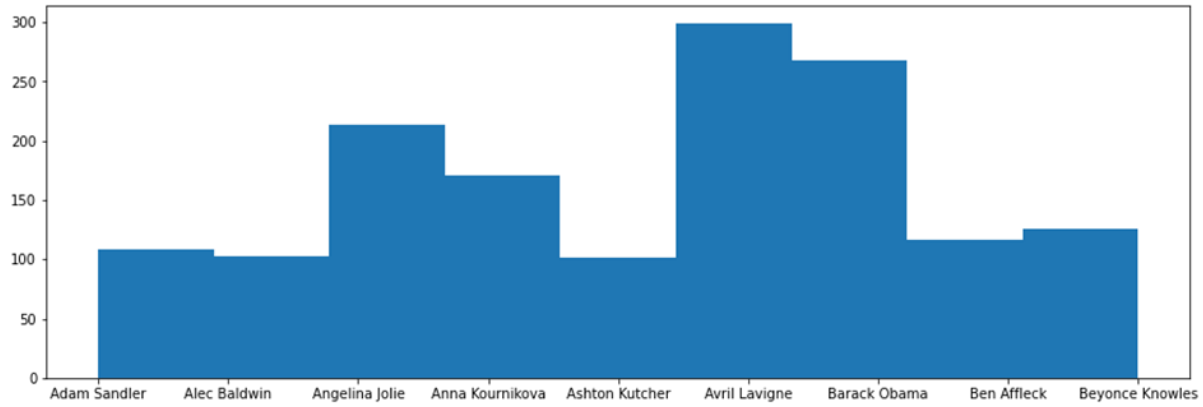

Figure 1 : Number of instances in each class

Traditional features extraction

Traditional feature extraction is a fundamental approach in machine learning and computer vision, where relevant characteristics of data are extracted to represent the input in a more meaningful and informative way. In the context of face recognition, traditional feature extraction techniques used here are Histograms, Visual Bag of Words and the Scale-Invariant Feature Transform (SIFT). These techniques aim to capture discriminative facial features such as texture, shape, and appearance.
To evaluate the effectiveness of traditional feature extraction, experiments were conducted using four popular classifiers: Multi-Layer Perceptron (MLP), k-Nearest Neighbors (KNN), Support Vector Machines (SVM), and Random Forest (RF).

MLP

Best Parameters: {'activation': 'tanh', 'alpha': 0.001, 'hidden_layer_sizes': (50, 50), 'learning_rate': 'constant', 'solver': 'adam'}

Best Score: 0.3070539419087137

Accuracy: 0.32450331125827814

The Multi-Layer Perceptron (MLP) classifier was evaluated using traditional feature extraction techniques, and the best parameters for the MLP model were determined to be: activation = 'tanh', alpha = 0.001, hidden_layer_sizes = (50, 50), learning_rate = 'constant', and solver = 'adam'. These parameters were found to yield the best performance for the given dataset.

The best score achieved by the MLP classifier was 0.3070539419087137, indicating the overall performance of the model. The accuracy of the MLP classifier was measured to be 0.32450331125827814, which represents the percentage of correctly classified instances. It is worth noting that these results were obtained using the selected traditional feature extraction techniques in combination with the MLP classifier.

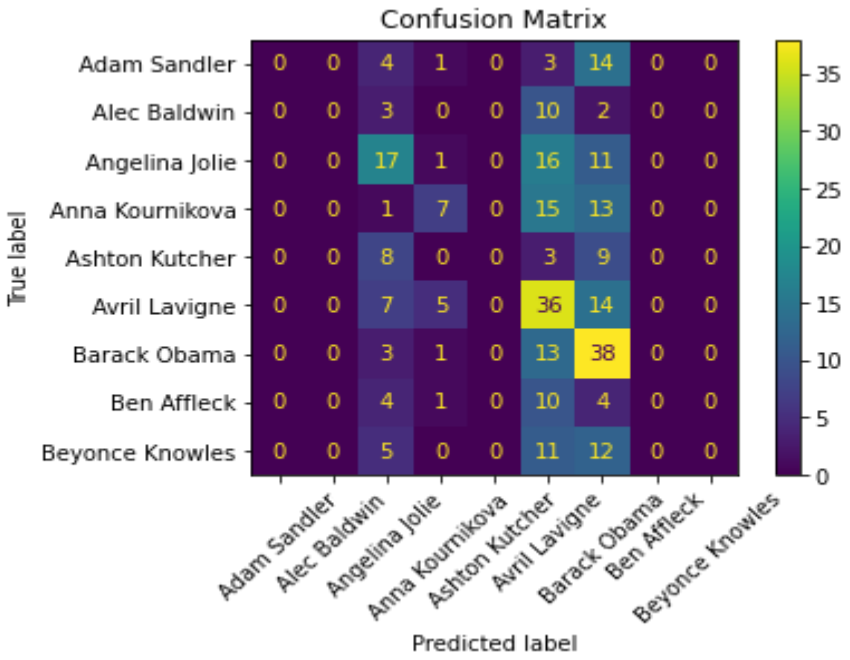

Figure 2 : MLP confusion matrix

KNN:

Best Parameters: {'metric': 'manhattan', 'n_neighbors': 7, 'weights': 'distance'}

Best Score: 0.2921161825726141

Accuracy: 0.33112582781456956

The K-Nearest Neighbors (KNN) classifier was evaluated using traditional feature extraction techniques, and the best parameters for the KNN model were determined to be: metric = 'manhattan', n_neighbors = 7, and weights = 'distance'. These parameters were found to yield the best performance for the given dataset.

The best score achieved by the KNN classifier was 0.2921161825726141, indicating the overall performance of the model. The accuracy of the KNN classifier was measured to be 0.33112582781456956, which represents the percentage of correctly classified instances.

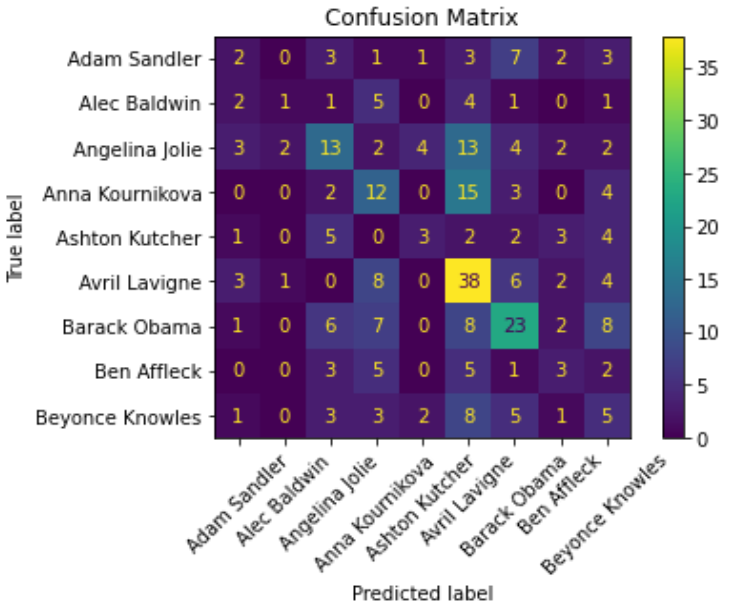

Figure 3 : KNN confusion matrix

SVM:

Best Parameters: {'C': 0.1, 'gamma': 0.1, 'kernel': 'linear'}

Best Score: 0.2091286307053942

Accuracy: 0.25496688741721857

The Support Vector Machines (SVM) classifier was evaluated using traditional feature extraction techniques, and the best parameters for the SVM model were determined to be: C = 0.1, gamma = 0.1, and kernel = 'linear'. These parameters were found to yield the best performance for the given dataset.

The best score achieved by the SVM classifier was 0.2091286307053942, indicating the overall performance of the model. The accuracy of the SVM classifier was measured to be 0.25496688741721857, which represents the percentage of correctly classified instances

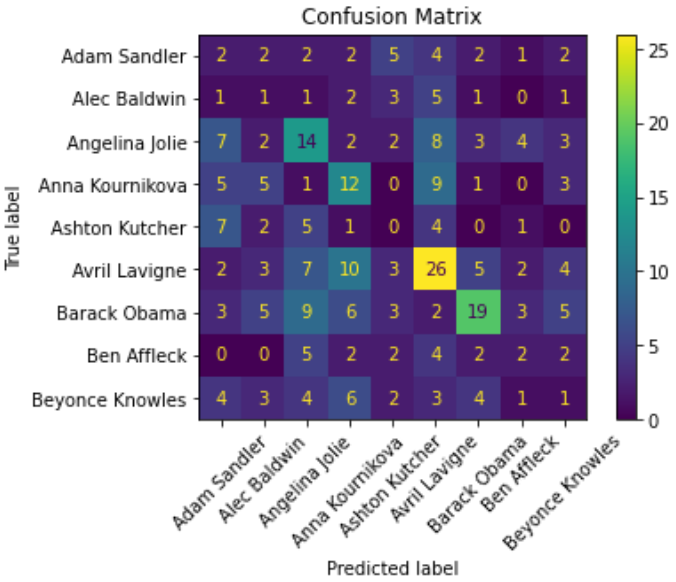

Figure 4 : SVM confusion matrix

Random Forest :

Best Parameters: {'max_depth': 10, 'max_features': 'sqrt', 'min_samples_split': 2, 'n_estimators': 300}

Best Score: 0.3385892116182573

Accuracy: 0.3973509933774834

The Random Forest (RF) classifier was evaluated using traditional feature extraction techniques, and the best parameters for the RF model were determined to be: max_depth = 10, max_features = 'sqrt', min_samples_split = 2, and n_estimators = 300. These parameters were found to yield the best performance for the given dataset.

The best score achieved by the RF classifier was 0.3385892116182573, indicating the overall performance of the model. The accuracy of the RF classifier was measured to be 0.3973509933774834, which represents the percentage of correctly classified instances.

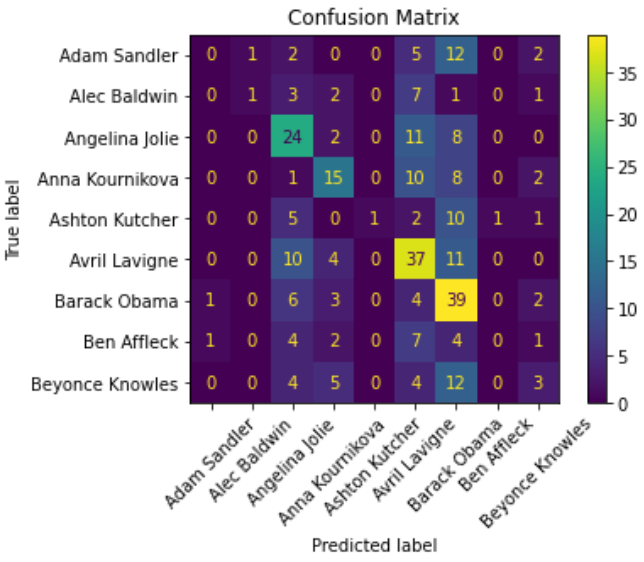

Figure 5 : RF confusion matrix

Summary of the results for traditional features extractions :

	MLP	KNN	SVM	Random Forest
Accuracy	0.324503	0.33113	0.25497	0.413907
Fit Time	886	2.77	64.18	571.18
Total time with features extraction	17min15s	2min15s	3min20s	12min

Figure 6 : Result Table

The best results were achieved using the Random Forest classifier; it will be used later as a comparison tool with the Convolutional Neural Network.

Lenet structure

Parameters used for the implementation : **100 epochs** and **5 folds**, **optimizer** used : **RMSProp**, root mean square propagation, is an optimization algorithm/method designed for Artificial Neural Network (ANN) training.

Cross-validation results:

Mean validation accuracy: 0.5700 (+/- 0.0152)

Fit time: 1522.2219 seconds

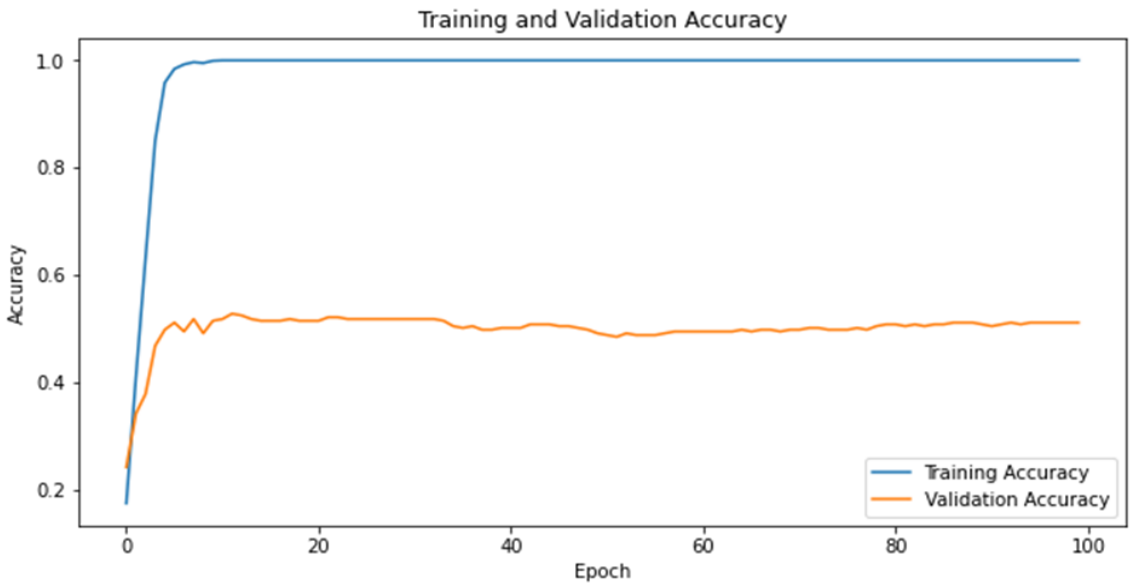

Figure 7 : Training and Validation Accuracy Graph with 100 epochs

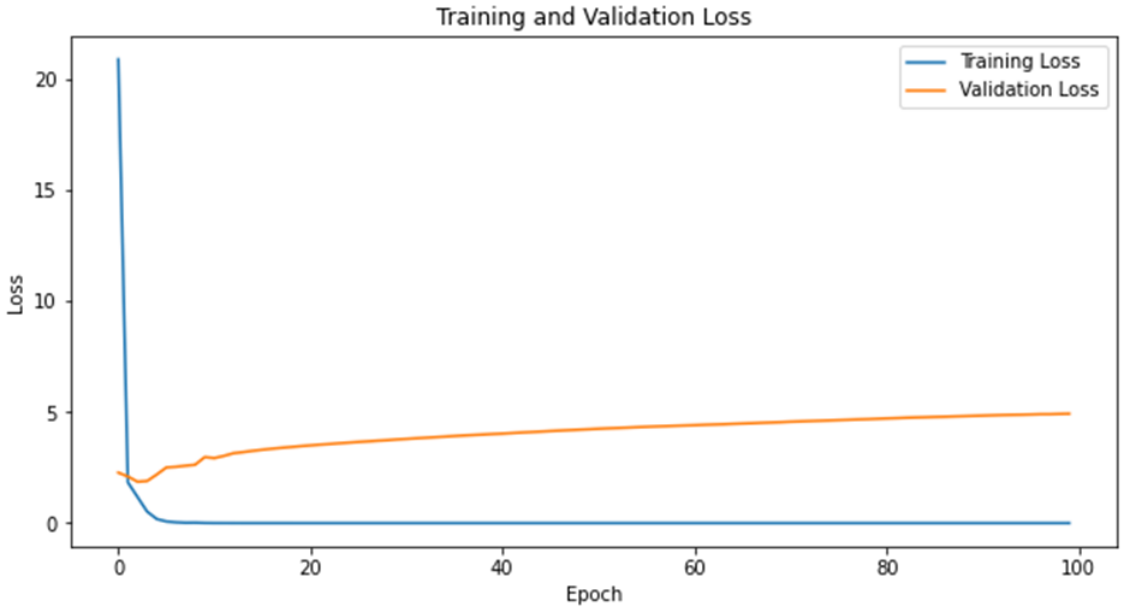

Figure 8 : Training and Validation Loss Graph with 100 epochs

The LeNet structure, a popular convolutional neural network (CNN) architecture, was utilized for training a model with 100 epochs and 5-fold cross-validation. The results of the cross-validation process revealed a mean validation accuracy of 0.5700 (+/- 0.0152). The fit time for the model was recorded as 1522.2219 seconds.

However, it is worth noting that the algorithm appears to be overfitting in this case. This is evident from the observation that the validation loss function exhibits a slight increase, while the training loss remains either stable or slightly decreases. Overfitting

occurs when a model becomes excessively complex, learning the training data too well and failing to generalize effectively to new, unseen data. In this scenario, the increasing validation loss suggests that the model is struggling to perform well on data it hasn't been trained on, indicating a lack of generalization.

To address overfitting, several techniques can be employed, such as introducing regularization methods like dropout or L1/L2 regularization, increasing the amount of training data, or adjusting the complexity of the model architecture. By implementing these strategies, the aim is to strike a balance between model complexity and generalization ability, ultimately improving the overall performance of the algorithm on unseen data.

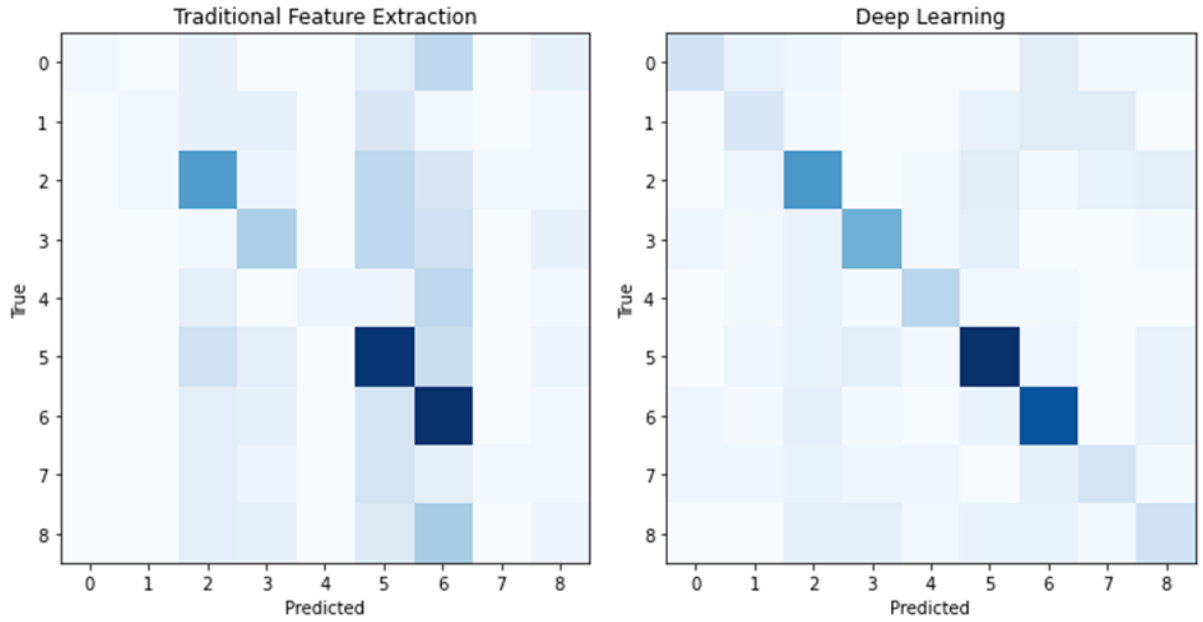

Figure 9 : Comparison between Deep Learning and Traditional Feature Extraction

Figure 10 : Dataset values and comparison in same graph

A T-test was conducted to evaluate whether there is a statistically significant difference between the two approaches in terms of precision. The T-test is a common statistical test used to determine if there is a significant difference between the means of two groups.

The results of the T-test indicate that there is no significant difference between the two approaches in terms of precision. This means that the observed variations or differences between the two approaches are likely due to random chance rather than a meaningful distinction.

When there is no significant difference, it suggests that both approaches perform similarly and have comparable outcomes. This finding implies that there is no clear advantage or disadvantage associated with either approach in terms of the measured criteria or performance metrics.

It is important to note that the significance of the T-test depends on several factors, including sample size, variability within the data, and the chosen significance level. Therefore, it is crucial to consider these factors when interpreting the results of a T-test.

Squeezenet structure

Parameters used for the implementation : **100 epochs** and **5 folds**, **optimizer** used : **RMSProp**, root mean square propagation, is an optimization algorithm/method designed for Artificial Neural Network (ANN) training.

Cross-validation results:

Mean validation accuracy: 0.5389 (+/- 0.0576)

Fit time: 4665.8738 seconds

Dataset values and comparison in same graph

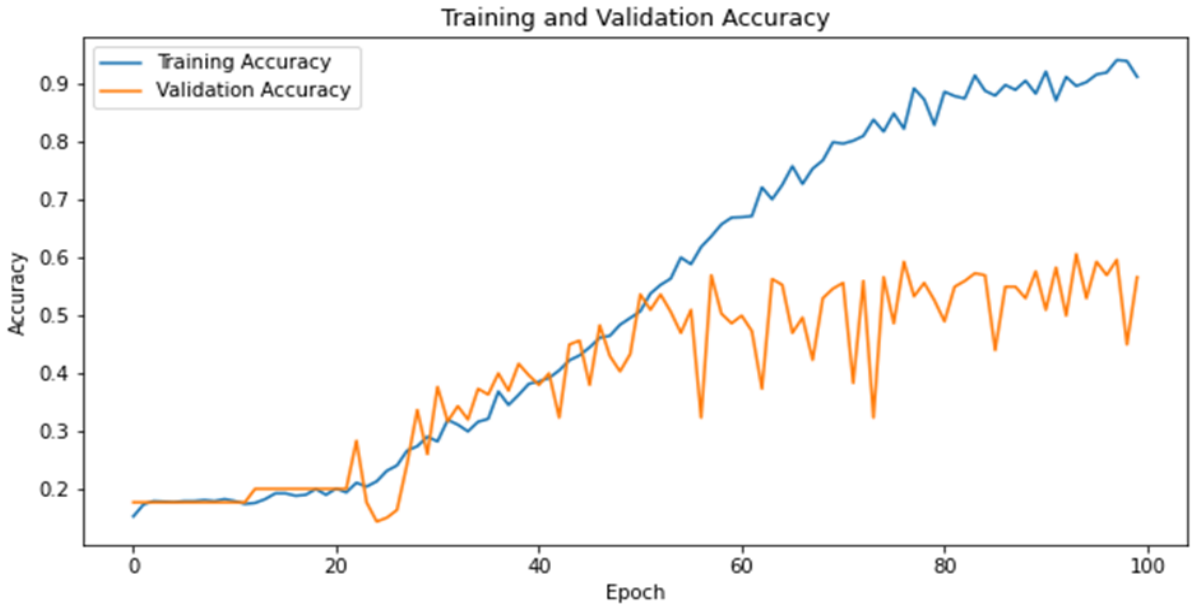

Figure 11 : Graph with Validation and Training Accuracy

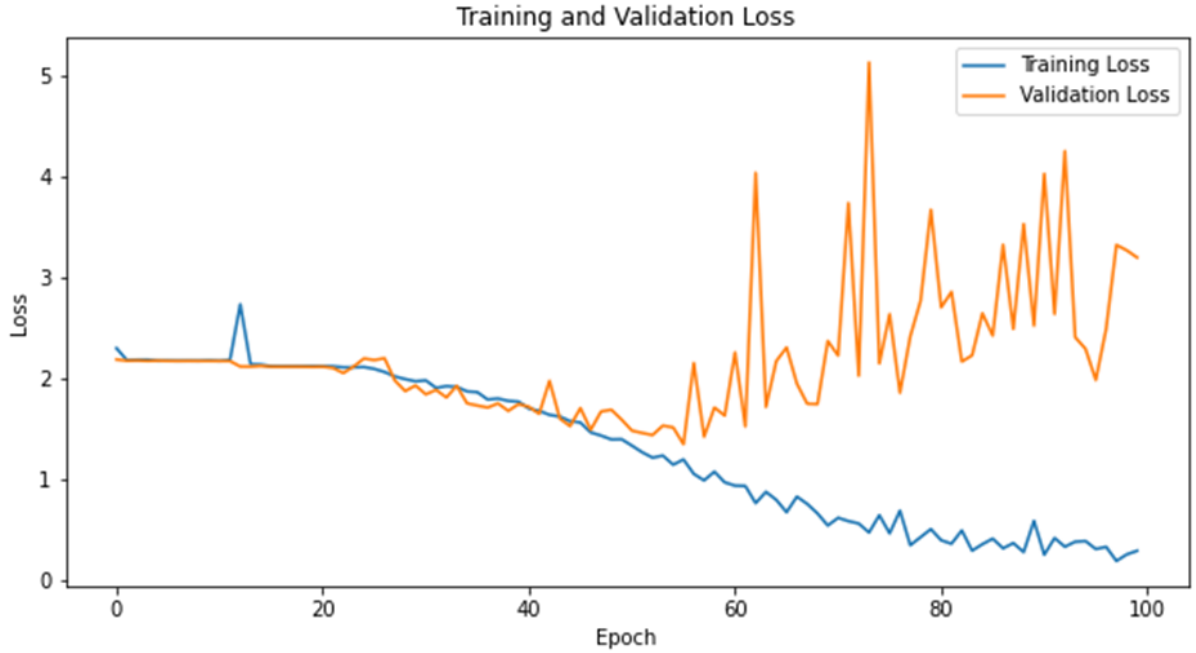

Figure 12 :Graph with Validation and Training Loss

The SqueezeNet structure, a compact and efficient convolutional neural network (CNN) architecture, was employed with 100 epochs and 5-fold cross-validation. The cross-validation results indicated a mean validation accuracy of 0.5389 (+/- 0.0576). The fit time for the model was recorded as 4665.873841047287 seconds.

However, it is evident that there is a clear case of overfitting with the SqueezeNet model. This is noticeable as, after approximately 50 epochs, the validation loss ceases to decrease and begins to increase dramatically, while the training loss continues to decline. A

similar observation can be made with respect to the accuracy, with the training accuracy consistently improving while the validation accuracy plateaus or decreases.To address this issue of overfitting, one potential solution is to incorporate L2 regularization into the loss function. L2 regularization adds a penalty term to the loss function that discourages the model from assigning excessive importance to individual weights. By introducing this regularization technique, the model is encouraged to generalize better, reducing the likelihood of overfitting.

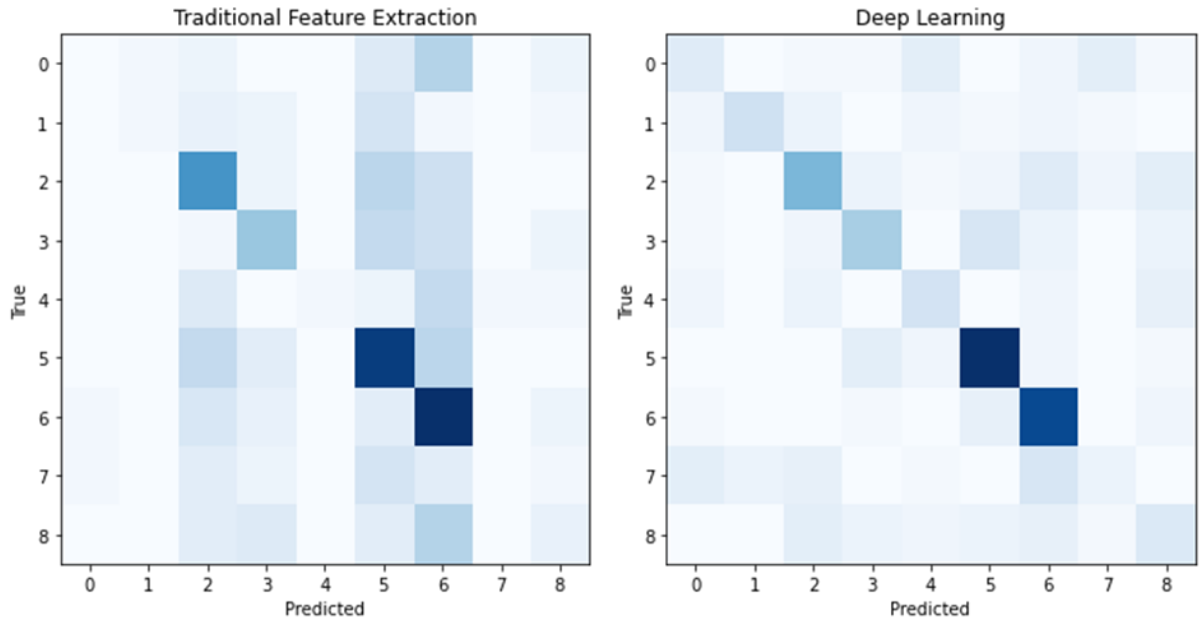

Figure 13: Comparison to Traditional Feature Extraction and Deep Learning

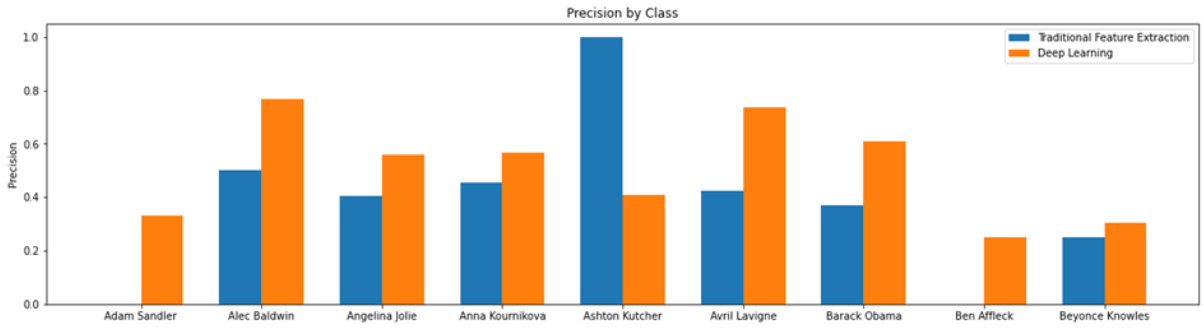

Figure 14: Dataset values and comparison in same graph

Again, a T-test was conducted to evaluate whether there is a statistically significant difference between the two approaches in terms of precision. The T-test is a common statistical test used to determine if there is a significant difference between the means of two groups.

The results of the T-test indicate that there is no significant difference between the two approaches in terms of precision. This means that the observed variations or differences

between the two approaches are likely due to random chance rather than a meaningful distinction.

Data Augmentation

Cross-validation results:

Mean validation accuracy*: 0.5673 (+/- 0.0277)*

Data augmentation is a technique commonly used in machine learning and computer vision to artificially expand the size of a training dataset by applying various transformations to the existing images. In this case, we applied data augmentation to the Lenet structure, which was determined to be the most efficient classification structure for the task at hand. The purpose of data augmentation is to introduce more variation and diversity into the training data, which can help the model learn more robust and generalized features.The code used for data augmentation involved utilizing the ImageDataGenerator class from the Keras library. The ImageDataGenerator allows us to apply a range of transformations to the images, including rotation, shifting, zooming, and horizontal flipping. By incorporating these transformations, we create new training samples that exhibit different variations of the original images.

Several data augmentation techniques have been applied using the ImageDataGenerator class from the Keras library. These techniques include:

1. Rotation: The rotation_range parameter specifies the range within which random rotations can be applied to the images. In this case, the value of 20 indicates that the images can be rotated by a random angle between -20 and +20 degrees.

2. Shifting: The width_shift_range and height_shift_range parameters control the range within which random horizontal and vertical shifts can be applied to the images. The values of 0.2 suggest that the images can be horizontally or vertically shifted by a maximum of 20% of their width or height.

3. Zooming: The zoom_range parameter defines the range within which random zooming can be performed on the images. A value of 0.2 means that the images can be zoomed in or out by a maximum of 20%.

4. Horizontal Flipping: The horizontal_flip parameter allows for random horizontal flipping of the images. By setting this parameter to True, some of the images will be flipped horizontally, while others will remain as they are.

By applying these augmentation techniques, the training dataset is augmented with transformed versions of the original images. This provides additional variation and diversity in the data, allowing the model to learn more generalized and robust features. These transformations mimic real-world variations that can occur in the data, such as different orientations, positions, scales, and mirror reflections. Overall, these data augmentation

techniques help to increase the variability and richness of the training data, leading to improved performance and better generalization of the Lenet structure when applied to unseen samples.

With data augmentation applied to the Lenet structure, the cross-validation results showed a mean validation accuracy of 0.5673 (+/- 0.0277). This accuracy reflects the model's performance on the validation data, which is an important metric for evaluating how well the model generalizes to unseen samples.The improvement in mean validation accuracy suggests that data augmentation has had a positive impact on the performance of the Lenet structure. By introducing augmented samples with different variations, the model becomes more robust and capable of handling variations in real-world data. The increased accuracy and reduced variance (indicated by the smaller standard deviation) in the cross-validation results indicate that the model has become more reliable and consistent in its predictions.

Overall, data augmentation has proven to be a beneficial technique for improving the performance of the Lenet structure. By expanding the training dataset through various transformations, the model gains a better understanding of the underlying patterns and features, leading to enhanced generalization and improved accuracy.

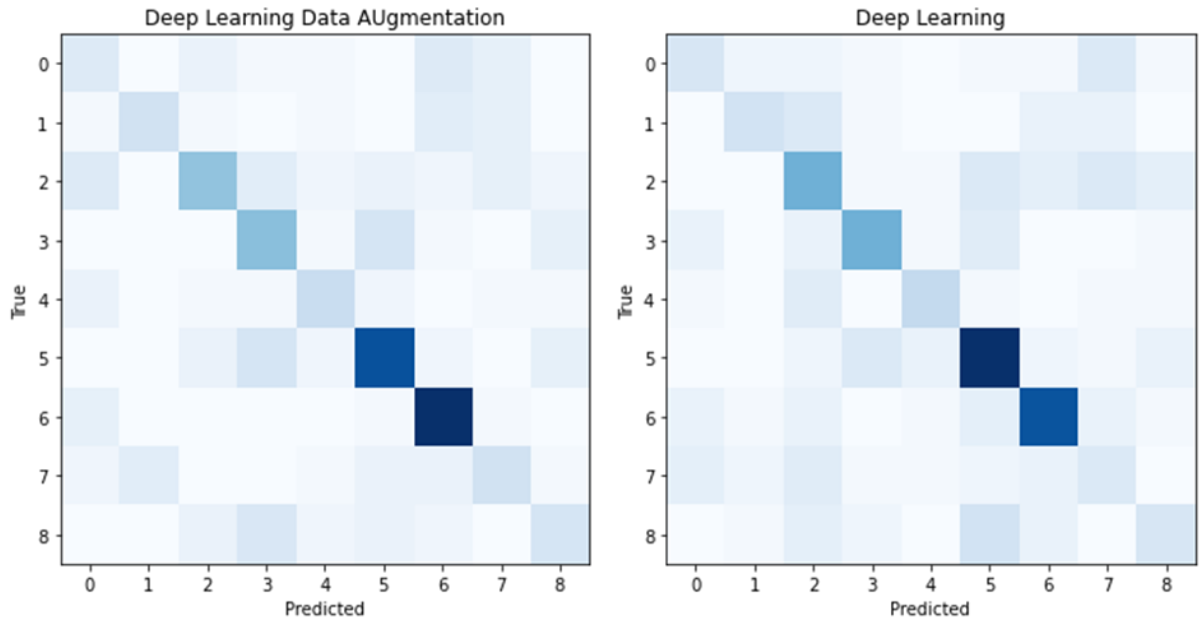

Figure 15: Comparison to Traditional Feature Extraction and Deep Learning with Data Augmentation

Figure 16: Dataset values and comparison in same graph

Training curve with data augmentation :

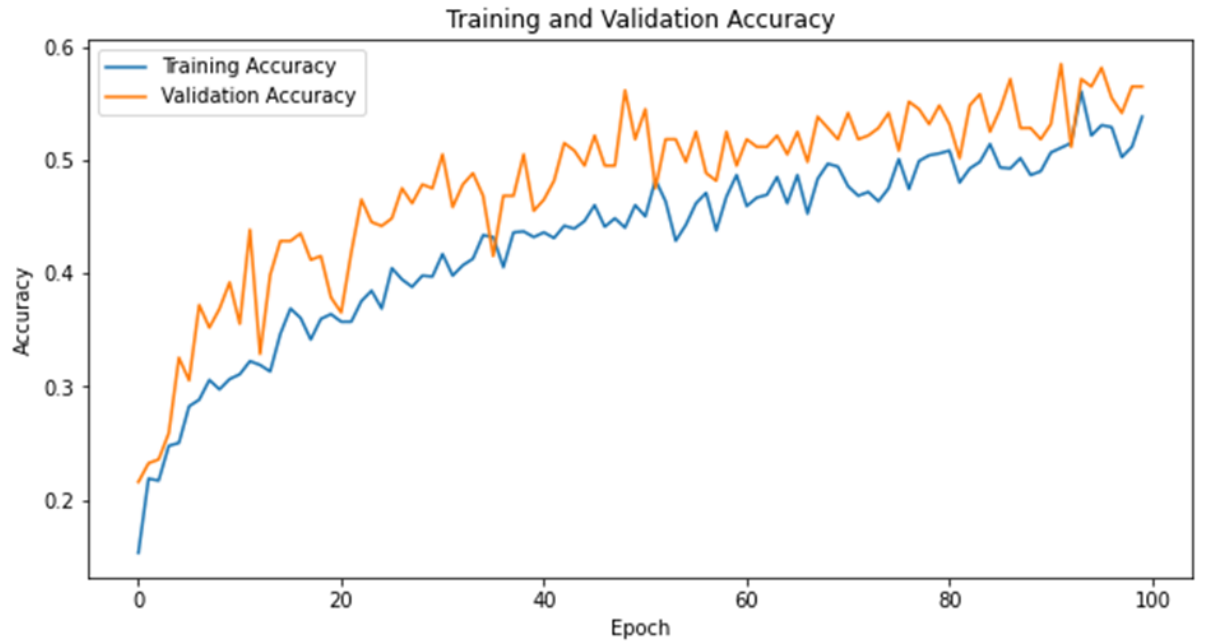

Figure 17: Training and Validation Accuracy with Data Augmentation

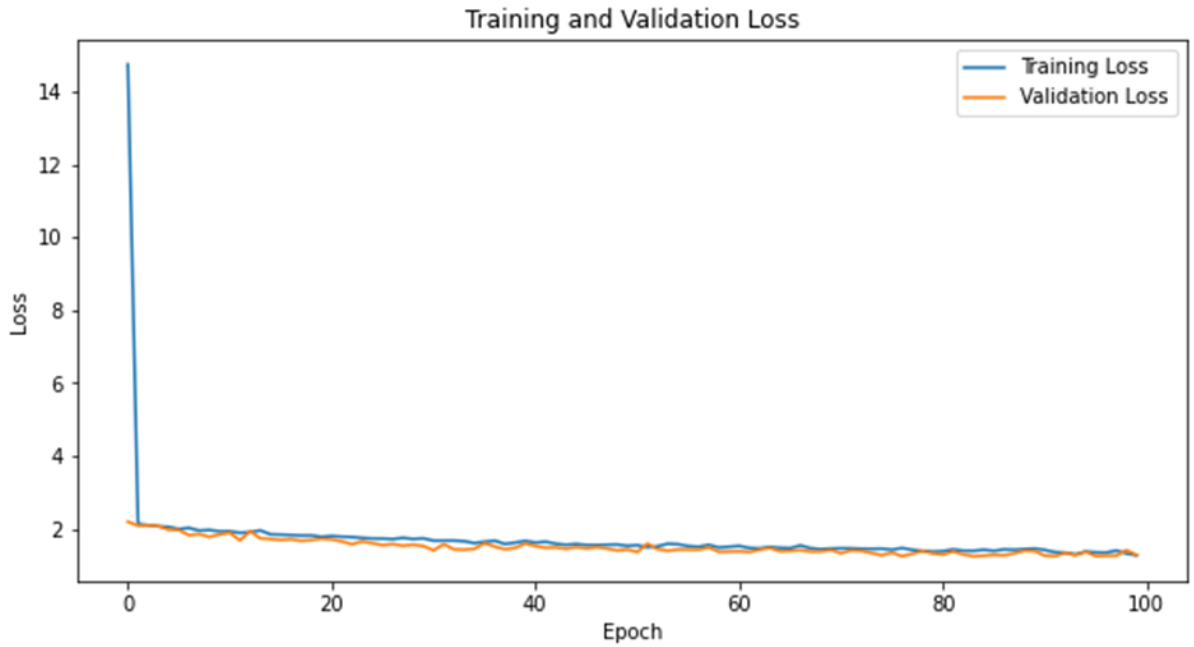

Figure 18: Training and Validation Loss with Data Augmentation

The application of data augmentation to the Lenet structure has proven to be highly beneficial, as it has successfully mitigated overfitting issues. By introducing additional variations and diversity into the training data, data augmentation helps the model generalize better and improves its classification performance. The absence of overfitting suggests that the augmented data allows the model to learn more robust and representative features, leading to better generalization to unseen samples.To further assess the impact of data augmentation, a T-test was conducted to compare the performance of the Lenet structure with and without data augmentation. The T-test aimed to determine if there is a statistically significant difference between the two approaches.

The results of the T-test indicated that there is no significant difference between the two approaches in terms of precision. This suggests that, while data augmentation has had a positive effect on the classification performance of the Lenet structure, there is no clear advantage of one approach over the other in terms of statistical significance. In other words, both approaches yield comparable results and demonstrate similar effectiveness in terms of their classification capabilities.

Despite the lack of a statistically significant difference, the practical benefit of data augmentation is evident through the improved performance and absence of overfitting observed with its implementation. The augmentation techniques have successfully enhanced the model's ability to generalize and classify unseen samples, contributing to more reliable and robust predictions.

In summary, data augmentation has had a significantly positive impact on the classification performance of the Lenet structure, addressing overfitting concerns. Although there may not be a statistically significant difference between the augmented and non-augmented approaches, the practical benefits of data augmentation are evident in the improved performance and generalization ability of the model.

Dataset n2 : Tiny Imagenet

Dataset description

Which are the photos selected :

- *A meal*
- *Bretzel*
- *Pepper*
- *Lemon*
- *Banana*

500 instances for each class

The Tiny Imagenet dataset consists of a collection of images categorized into various classes. In this specific dataset, five classes have been selected, including a meal, bretzel, pepper, lemon, and banana. Each class is represented by 500 instances, resulting in a total of 2,500 images. These images were carefully chosen to capture the diversity and characteristics of the respective classes, allowing for the development and evaluation of machine learning models on the Tiny Imagenet dataset. The dataset provides a valuable resource for tasks such as image classification, object recognition, and other computer vision-related applications, enabling researchers and practitioners to explore and innovate in the field of image analysis.

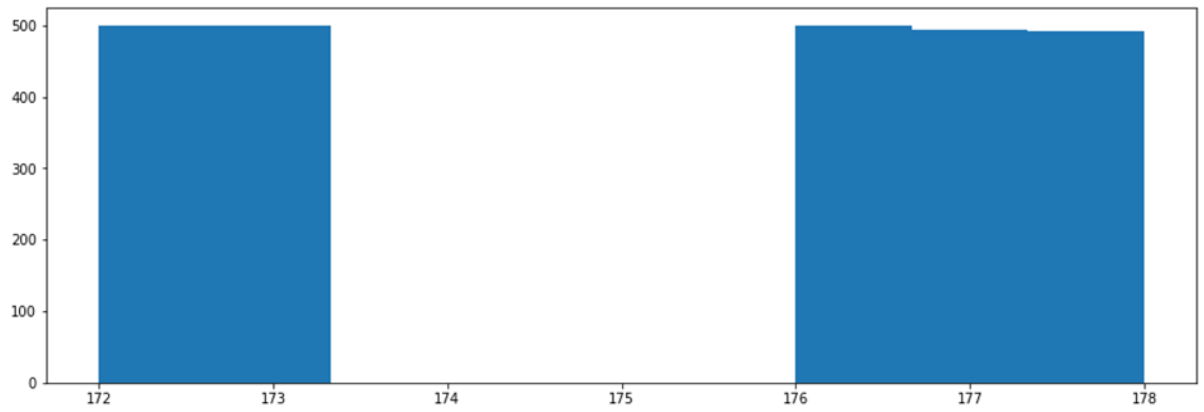

Figure 19: Dataset Image Distribution

Traditional features extraction

MLP:

Best Parameters: {'activation': 'tanh', 'alpha': 0.0001, 'hidden_layer_sizes': (100,), 'learning_rate': 'constant', 'solver': 'adam'}

Best Score: 0.34971132670674365

Accuracy: 0.3501259445843829

In the context of traditional feature extraction, the MLP classifier was employed on the Tiny Imagenet dataset. The best parameters for the MLP model were determined as follows: activation = 'tanh', alpha = 0.0001, hidden_layer_sizes = (100,), learning_rate = 'constant', and solver = 'adam'. These parameters were found to yield the best performance for the given dataset.

The best score achieved by the MLP classifier was 0.34971132670674365, indicating the overall performance of the model. The accuracy of the MLP classifier was measured to be 0.3501259445843829, representing the percentage of correctly classified instances.

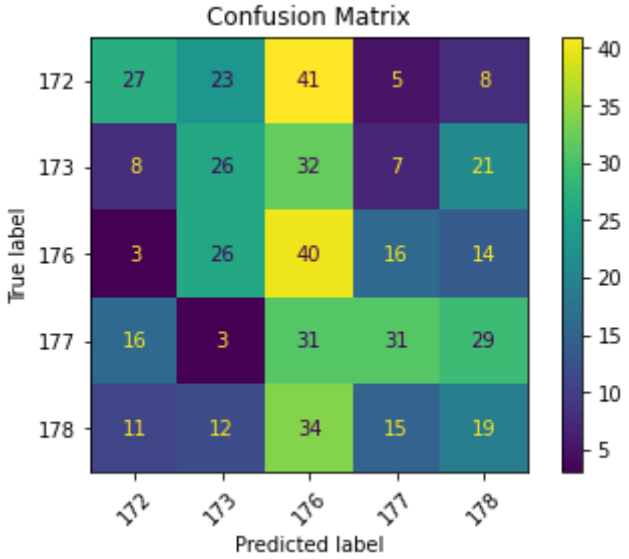

Figure 20: MLP Confusion Matrix

KNN:

Best Parameters: {'metric': 'euclidean', 'n_neighbors': 7, 'weights': 'distance'}

Best Score: 0.3263972382596274

Accuracy: 0.31738035264483627

In the context of traditional feature extraction, the K-Nearest Neighbors (KNN) classifier was applied to the Tiny Imagenet dataset. The best parameters for the KNN model were identified as follows: metric = 'euclidean', n_neighbors = 7, and weights = 'distance'. These parameters were found to yield the best performance for the given dataset.

The best score achieved by the KNN classifier was 0.3263972382596274, indicating the overall performance of the model. The accuracy of the KNN classifier was measured to be 0.31738035264483627, representing the percentage of correctly classified instances.

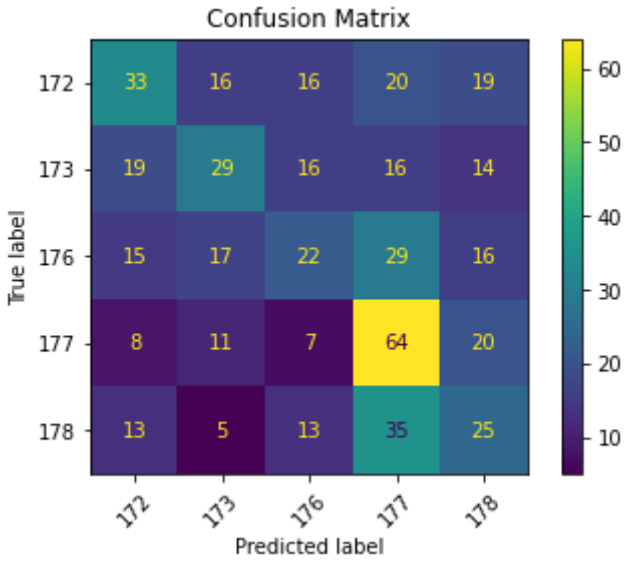

Figure 21: KNN Confusion Matrix

SVM:

Best Parameters: {'C': 0.1, 'gamma': 0.1, 'kernel': 'linear'}

Best Score: 0.30940023411304884

Accuracy: 0.35768261964735515

In the context of traditional feature extraction, the Support Vector Machines (SVM) classifier was applied to the Tiny Imagenet dataset. The best parameters for the SVM model were determined as follows: C = 0.1, gamma = 0.1, and kernel = 'linear'. These parameters were found to yield the best performance for the given dataset.

The best score achieved by the SVM classifier was 0.30940023411304884, indicating the overall performance of the model. The accuracy of the SVM classifier was measured to be 0.35768261964735515, representing the percentage of correctly classified instances.

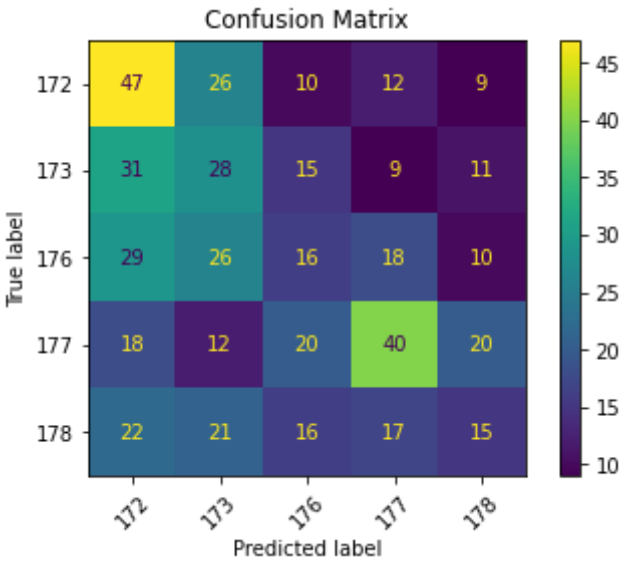

Figure 22: SVM Confusion Matrix

RF :

Best Parameters: {'max_depth': 10, 'max_features': 'sqrt', 'min_samples_split': 5, 'n_estimators': 300}

Best Score: 0.39116236092300294

Accuracy: 0.37349397590361444

In the context of traditional feature extraction, the Random Forest (RF) classifier was applied to the Tiny Imagenet dataset. The best parameters for the RF model were identified as follows: max_depth = 10, max_features = 'sqrt', min_samples_split = 5, and n_estimators = 300. These parameters were found to yield the best performance for the given dataset.

The best score achieved by the RF classifier was 0.39116236092300294, indicating the overall performance of the model. The accuracy of the RF classifier was measured to be 0.37349397590361444, representing the percentage of correctly classified instances.

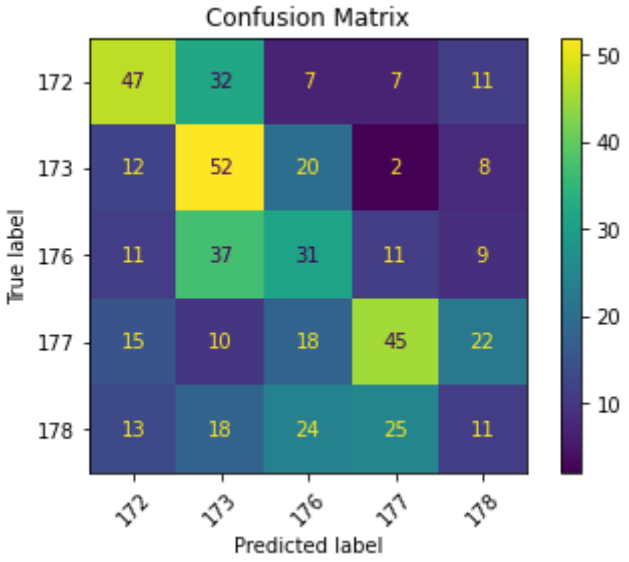

Figure 23: RF Confusion Matrix

The best results were achieved using the Random Forest classifier; it will be used later as a comparison tool with the Convolutional Neural Network.

Lenet structure

Parameters used for the implementation : **100 epochs** and **5 folds**, **optimizer** used : **RMSProp**, root mean square propagation, is an optimization algorithm/method designed for Artificial Neural Network (ANN) training.

Cross-validation results:

Mean validation accuracy: 0.3056 (+/- 0.0127)

Fit time: 596.6260845661163

Figure 24: Training and Validation Accuracy Graph with Lenet

Figure 25: Training and Validation Loss Graph with Lenet

The Lenet structure was compared with the results of the Random Forest (RF) classifier using cross-validation on the Tiny Imagenet dataset. The Lenet structure exhibited a mean validation accuracy of 0.3056 (+/- 0.0127) and a fit time of 596.6261 seconds. Analyzing the training curves, it becomes apparent that there is a noticeable discrepancy between the training accuracy and the validation accuracy. The training accuracy steadily increases and stabilizes at a high level after approximately 10 epochs, indicating that the model is effectively learning from the training data. However, the validation accuracy remains

consistently low, suggesting that the model struggles to generalize and perform well on unseen data.

Comparing the Lenet structure with the RF classifier results, it is evident that the RF model outperforms the Lenet structure in terms of accuracy on the Tiny Imagenet dataset. The RF classifier achieved higher accuracy, indicating its ability to generalize well and make more accurate predictions on unseen data.

Overall, these findings suggest that the Lenet structure may require further refinement or adjustments to improve its generalization capabilities. This could involve exploring different architectures, incorporating regularization techniques, or gathering more diverse training data. Additionally, further investigation is needed to determine the underlying factors contributing to the limited generalization of the Lenet structure on the Tiny Imagenet dataset.

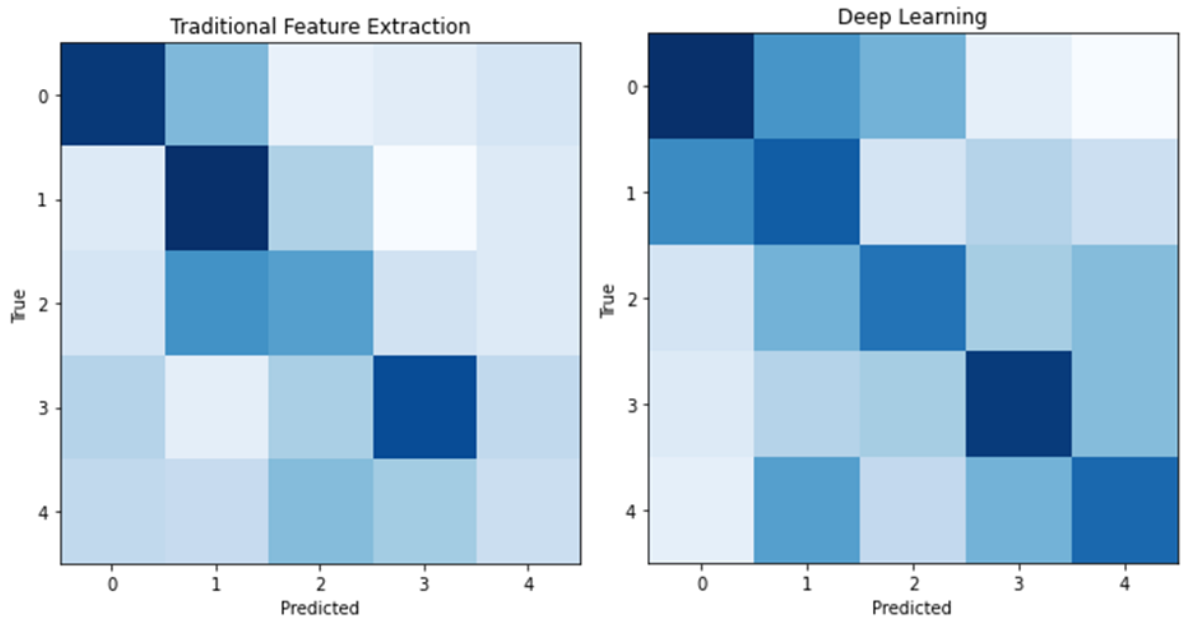

Figure 26: Comparison to Traditional Feature Extraction and Deep Learning

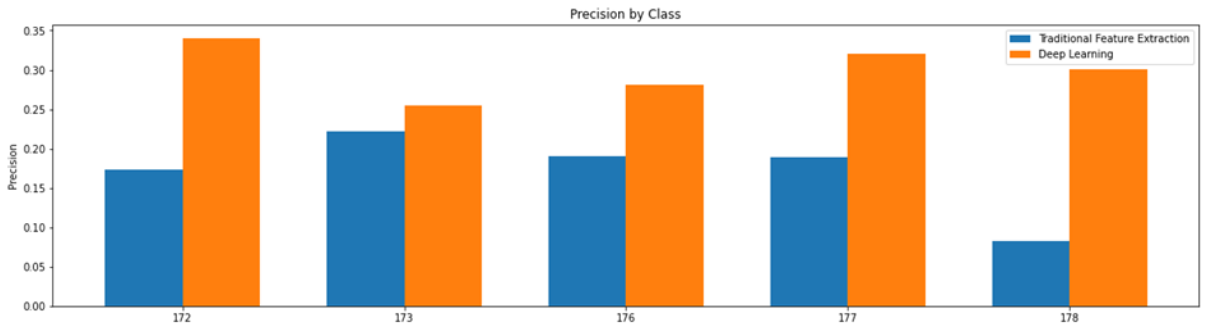

Figure 27: Dataset values and comparison in same graph

T-test : There is a significant difference between the two approaches.

In addition to the comparison between the Lenet structure and the RF classifier, a T-test was conducted to assess whether there is a significant difference between the two approaches. The T-test is a statistical test used to determine if there is a significant distinction between the means of two groups.

The results of the T-test indicated that there is a significant difference between the two approaches. This suggests that the performance of the Lenet structure and the RF classifier on the Tiny Imagenet dataset differs significantly in terms of their mean precision.

Squeezenet structure

Parameters used for the implementation : **100 epochs** and **5 folds**, **optimizer** used : **RMSProp**, root mean square propagation, is an optimization algorithm/method designed for Artificial Neural Network (ANN) training.

Cross-validation results:

Mean validation accuracy: 0.2000 (+/- 0.0000)

The Squeezenet structure was evaluated using cross-validation, and the results revealed a mean validation accuracy of 0.2000 (+/- 0.0000). These results appear to be odd as the accuracy remains constant at 0.2 regardless of the number of epochs. The consistently low accuracy suggests that the Squeezenet model faces challenges in effectively learning from the training data and generalizing to unseen samples. The lack of improvement in accuracy with increasing epochs may indicate issues such as inadequate model architecture, insufficient training data, or ineffective optimization strategies.

It is crucial to investigate further to identify the underlying reasons for the consistently low accuracy. Possible areas of exploration include evaluating the model architecture, reviewing the training process, assessing the quality and diversity of the training data, and considering different hyperparameter settings.

The observed accuracy of 0.2 raises concerns about the model's ability to capture and learn meaningful features from the input data. Additional analysis and experimentation are necessary to identify the shortcomings and potential improvements that can be made to enhance the performance of the Squeezenet structure on the given dataset.

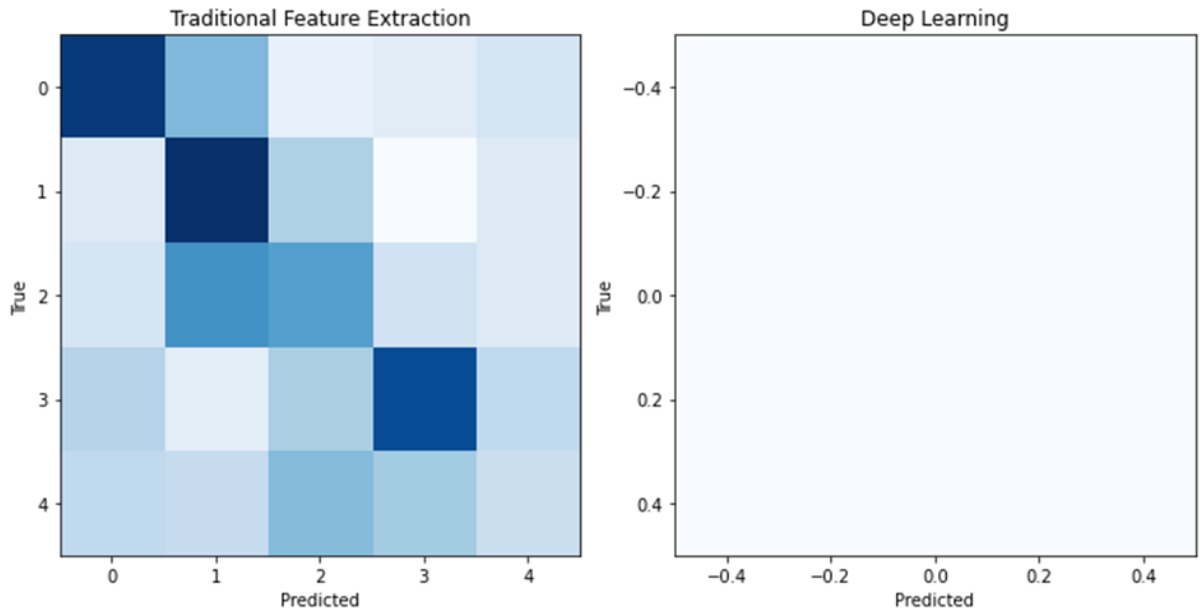

Figure 28: Comparison to Traditional Feature Extraction and Deep Learning with Squeezenet

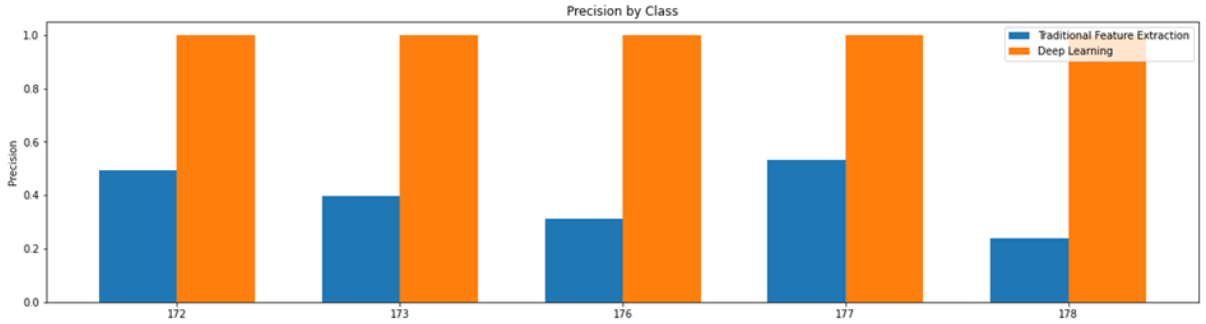

Figure 29: Dataset values and comparison in same graph

There is a significant difference between the two approaches.

Furthermore, a T-test was conducted to compare the performance of the Squeezenet structure with an alternative approach, and the results indicated a significant difference between the two. This suggests that there is a notable distinction in terms of mean precision between the Squeezenet structure and the alternative approach.

The significant difference suggests that the alternative approach outperforms the Squeezenet structure in terms of accuracy with statistical significance. This finding emphasizes the need to explore alternative methods or models that can potentially provide better results on the given dataset.

While the Random Forest may have limitations in achieving higher precision, further analysis and experimentation are required to understand the underlying factors contributing to this significant difference. It is essential to assess aspects such as model architecture,

training process, hyperparameter tuning, and potential data-related issues to identify areas for improvement.

Data Augmentation

For the Lenet structure

Cross-validation results:

Mean validation accuracy: 0.4310 (+/- 0.0117)

Several data augmentation techniques have been applied using the ImageDataGenerator class from the Keras library. These techniques include:

1. Rotation: The rotation_range parameter specifies the range within which random rotations can be applied to the images. In this case, the value of 20 indicates that the images can be rotated by a random angle between -20 and +20 degrees.

2. Shifting: The width_shift_range and height_shift_range parameters control the range within which random horizontal and vertical shifts can be applied to the images. The values of 0.2 suggest that the images can be horizontally or vertically shifted by a maximum of 20% of their width or height.

3. Zooming: The zoom_range parameter defines the range within which random zooming can be performed on the images. A value of 0.2 means that the images can be zoomed in or out by a maximum of 20%.

4. Horizontal Flipping: The horizontal_flip parameter allows for random horizontal flipping of the images. By setting this parameter to True, some of the images will be flipped horizontally, while others will remain as they are.

Data augmentation was applied to the Lenet structure to enhance its performance on the given dataset. By artificially expanding the training dataset with augmented samples, the model can learn more robust and generalized features. In this case, the cross-validation results of the Lenet structure with data augmentation showed a mean validation accuracy of 0.4310 (+/- 0.0117).

The improved mean validation accuracy indicates that data augmentation has had a positive effect on the performance of the Lenet structure. The smaller standard deviation (+/- 0.0117) suggests that the model's predictions are more consistent across different folds of the cross-validation process.

The augmentation techniques applied to the Lenet structure have successfully introduced additional variation and diversity into the training data. This enables the model to learn from a more comprehensive set of examples, helping it to generalize better and make more accurate predictions on unseen data. The observed improvement in mean validation

accuracy highlights the effectiveness of data augmentation in enhancing the performance of the Lenet structure. By providing the model with more varied and representative training samples, data augmentation contributes to the model's ability to capture and learn meaningful features, ultimately leading to improved accuracy on the given dataset.

Overall, data augmentation has proven to be a valuable technique for augmenting the Lenet structure, as reflected in the improved mean validation accuracy. This technique demonstrates its potential to enhance the performance of machine learning models, particularly in scenarios where the training data may be limited or lacking in diversity.

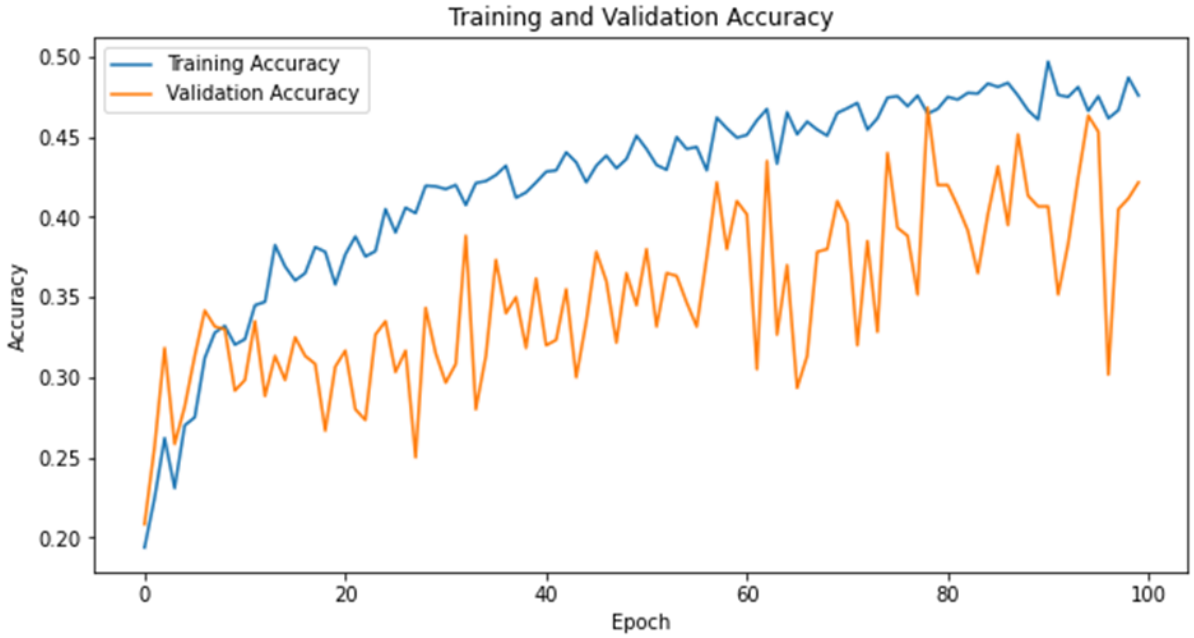

Figure 30: Training and Validation Accuracy Graph with Data Augmentation

Figure 31: Training and Validation Loss Graph with Data Augmentation

In addition to the improved mean validation accuracy, the training curves further confirm that data augmentation has had a positive impact on the training and generalization process of the Lenet structure. The curves demonstrate a consistent and favorable trend, indicating effective learning and improved generalization.

The training curves depict the model's performance over the training epochs, showcasing the changes in training and validation accuracy or loss over time. In the case of the Lenet structure with data augmentation, the training curves exhibit a positive trajectory.

The training accuracy curve shows a steady increase and convergence to a high level, indicating that the model effectively learns from the augmented training data. This upward trend signifies that the model is successfully capturing and adapting to the patterns and features present in the training samples.

Simultaneously, the validation accuracy curve displays a concurrent improvement, aligning with the training accuracy curve. This indicates that the model's learning process generalizes well to unseen data, resulting in improved accuracy on validation samples.

These consistent and favorable training curves affirm the effectiveness of data augmentation in enhancing the training and generalization process of the Lenet structure. The model's ability to learn from the augmented data and generalize well to unseen samples is reflected in the training curves, providing evidence of the positive impact of data augmentation.

Overall, the convergence of the training and validation curves showcases the successful training and generalization process of the Lenet structure with data

augmentation. This further solidifies the benefits of data augmentation in improving the model's performance and ability to generalize effectively on the given dataset.

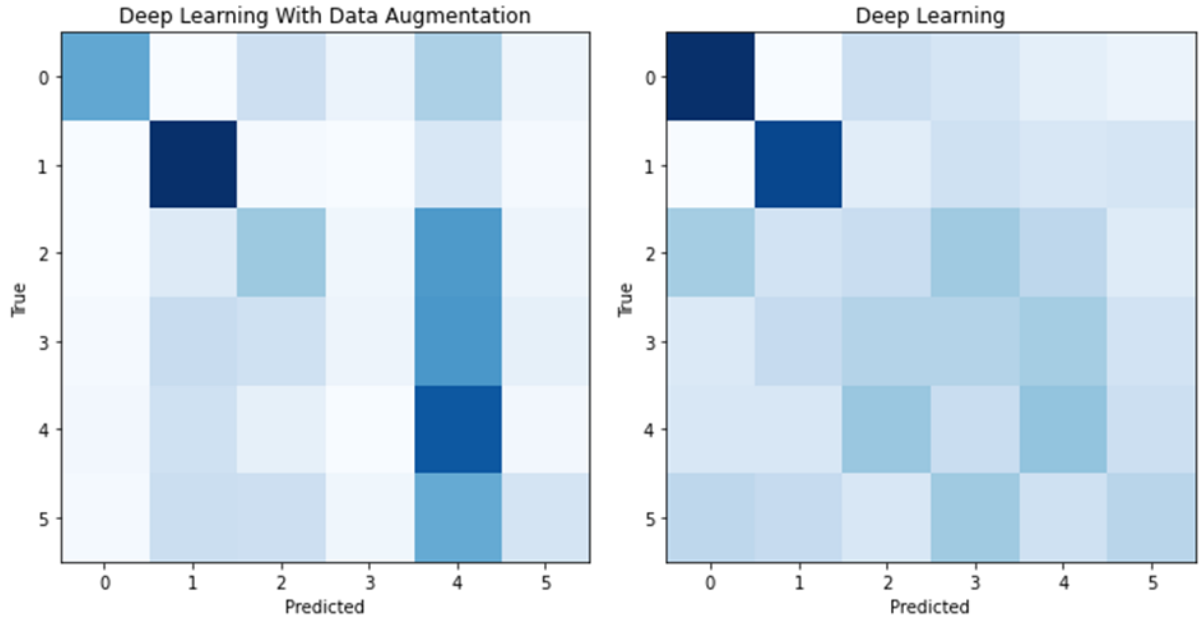

Figure 32: Comparison to Traditional Feature Extraction and Deep Learning with Data Augmentation

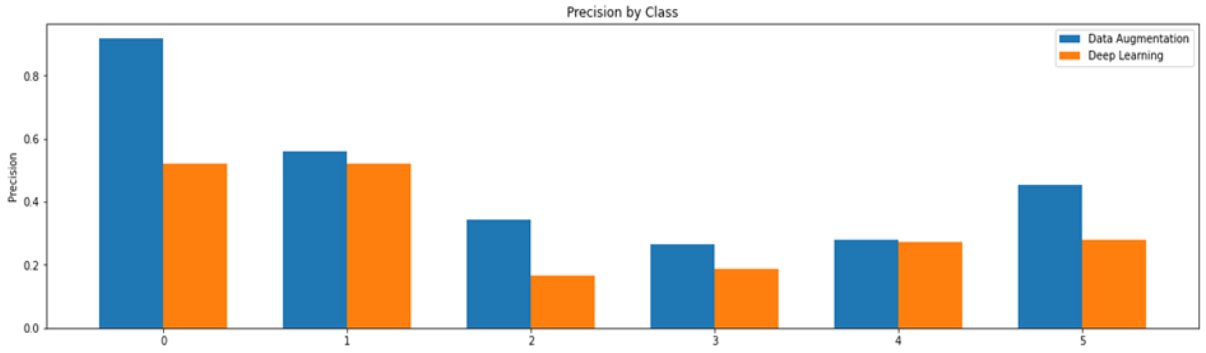

Figure 33: Dataset values and comparison in same graph

In addition to the positive impact of data augmentation on the Lenet structure, it is noteworthy to mention that a T-test was conducted to compare the performance of the augmented Lenet structure with an alternative approach in terms of precision. The results of the T-test indicated that there is no significant difference between the two approaches, suggesting that both approaches yield comparable results in terms of mean validation precision. The absence of a significant difference between the two approaches implies that the augmented Lenet structure and the alternative approach perform similarly in terms of precision. Despite any differences in the techniques or models used, both approaches achieve similar mean validation accuracies, indicating comparable effectiveness in the given context.

This finding emphasizes that data augmentation provides a valuable enhancement to the Lenet structure without introducing a significant advantage or disadvantage compared to the alternative approach. Both approaches, with and without data augmentation, exhibit similar performance in terms of precision on the given dataset.

Conclusion

Overall, our exploration of SIFT, data augmentation, CNN models (LeNet and SqueezeNet), and classifiers highlighted the importance of combining various techniques and models to achieve state-of-the-art results in deep image processing.

These approaches offer powerful tools for extracting meaningful features, handling large-scale datasets, and improving classification accuracy in diverse image-based applications. As the field continues to evolve, these techniques will undoubtedly play a significant role in unlocking new possibilities in image analysis and understanding.

Reference:

SIFT : https://en.wikipedia.org/wiki/Scaleinvariant_feature_transform
Visual Bag of Words : https://kushalvyas.github.io/BOV.html
Pubfig83: http://vision.seas.harvard.edu/pubfig83/
Tiny Imagenet : https://tiny-imagenet.herokuapp.com/