ASL American Sign Language Detection

Bayesian Network, Naïve Bayes structure and Convolutional Neural Networks

Lappeenranta–Lahti University of Technology LUT

Project 2 of Course “Advanced Data Analysis and Machine Learning”

2022

Authors: Thanh Tran (000285359)

Nghia Nguyen (000275466)

Swapnil Baad (000323486)

Examiner(s): Professor Satu-Pia Reinikainen

Professor Heikki Haario

Professor Lasse Lensu

Professor Assistant Sabrina Duma

Contents

Figures

1. Introduction

In this project, our objective is the development of two image recognition models which are used to perform the recognition of American sign language alphabet from images. Our solutions apply the theory of Bayesian Networks and Convolutional Neural Networks to the image data.

The limitation is that our project concentrates on only 10 classes namely “K”, “L”, “M”, “N”, “O”, “P”, “Q”, “R”, “S” and “T” according to the project requirements.

The dataset can be accessed from https://www.kaggle.com/datasets/grassknoted/asl-alphabet?datasetId=23079

The source code to the project is version-controlled and made public to allow accessibility from the link below:

https://github.com/Thanhtrannd/ASL_American_Sign_Language_To_Speech.git

It should be noted that due to size of the dataset itself, the git repository is not designed to store the dataset. It is expected to be separately downloaded, unzip, rename and put into the required directory in order to be recognized by the main program. One can either do the setup in their own way or follow the instruction written in the README file of the repository. (There is a plan to develop a script to perform the auto data downloading process if our schedule allows)

2. Data Exploration

The training dataset consists of 3000 images of size 200 * 200 pixels for each class. The dataset can be used for training and validation. Additionally, there is 1 image for each class available only for testing purpose and those images shall never be included during the development process.

3. Bayesian Network and Naïve Bayesian Models

3.1. Data Pre-treatment and Feature Extraction

One of the main issues in recognition systems is pre-processing and feature extraction.

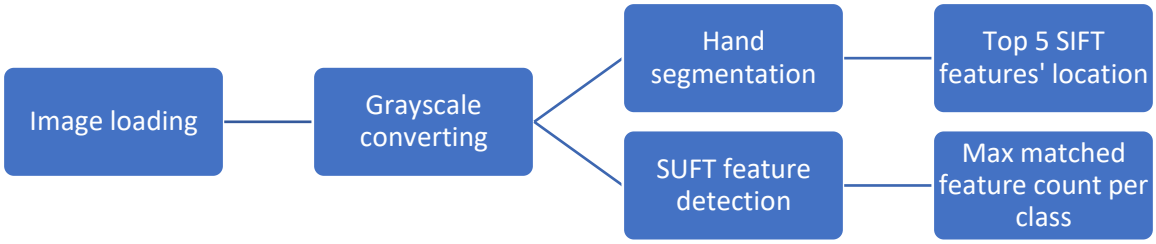


Figure 1 Feature extraction procedure for Bayesian Network and Naive Bayes structure training

3.1.1. Feature Extraction Trials

We have run some trials with totally 13 features extracted from the images as follows:

- Mean of R Channel
- Mean of G Channel
- Mean of B Channel
- Number of edge pixels (Using Canny Edge Detector)
- Contrast of Image
- Skewness
- Kurtosis
- Using Regionprops: Area
- Using Regionprops: Centroid
- Using Regionprops: Bounding Box
- SIFT Features
- SURF Features

- ORB Features

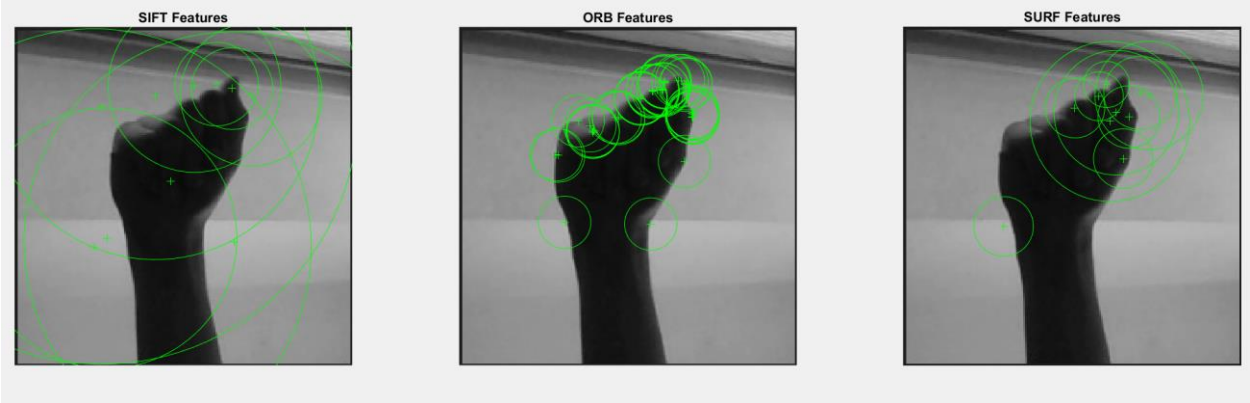


Figure 2 SIFT, ORB, SURF Features

After consideration of the relevance of features, we have decided to exclude at this moment all colour-band-derived metrics and contrast colour-related metric due to its variance highly correlated to the room lighting. Instead, we extracted features by combining the effects of multiple methods.

3.1.2. First Feature Set: Hand Segmentation using ORB feature detection and SIFT Feature Extraction

As the extraction trials demonstrated, ORB feature detection seems to be a robust method to detect the edges of the object. Therefore, we decided to use the method together with extracted region properties for hand segmentation.

After ORB features were detected. Bounding boxes are computed by firstly detecting edges of the images using ‘Canny’ operator and secondly obtaining region properties. Bounding boxed are then ranked based on the number of ORB features that fell into its inside area and also based on the area of the box itself. For each image sample, the smallest box containing the maximum expected number of ORB features was selected. Hand object was segmented inside the selected bounding box. The location of at most 5 strongest SIFT features detected were then recorded. They were then normalized so that we can exclude the effects of size of the cropped images and the location of hand object in the cropped images and consider only the trajectory of all extracted points.

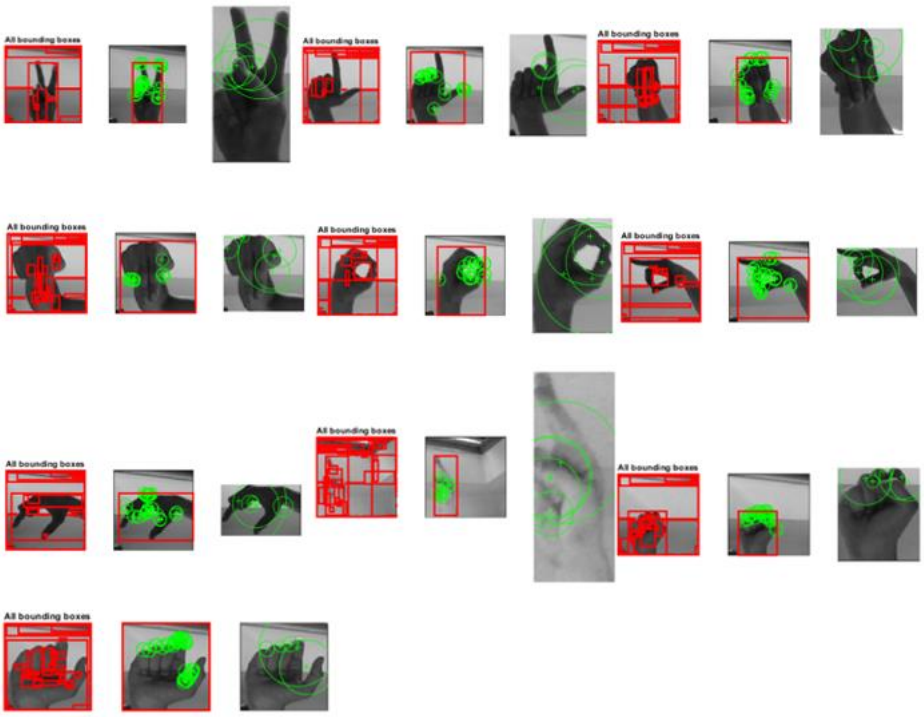


Figure 3 Feature extraction with hand detection and SIFT feature coordicates (example)

3.1.3. Second Feature Set: The maximum number of matched SURF features

Geometrically extracting and matching features can sometimes encounter difficulties due to room lighting or partly existence of the main object. Therefore, a set of data template can be used to generalize the opportunity of the detection and to ensure the robustness.

Firstly, we selected one image every 100 sample images belonging to each class as class templates. As a result, we gathered 30 templates for each class and that made 300 templates. Secondly, SURF features were detected and matched between each training sample to the 30 templates of each class. The maximum number of matched features for each class were recorded as a feature value.

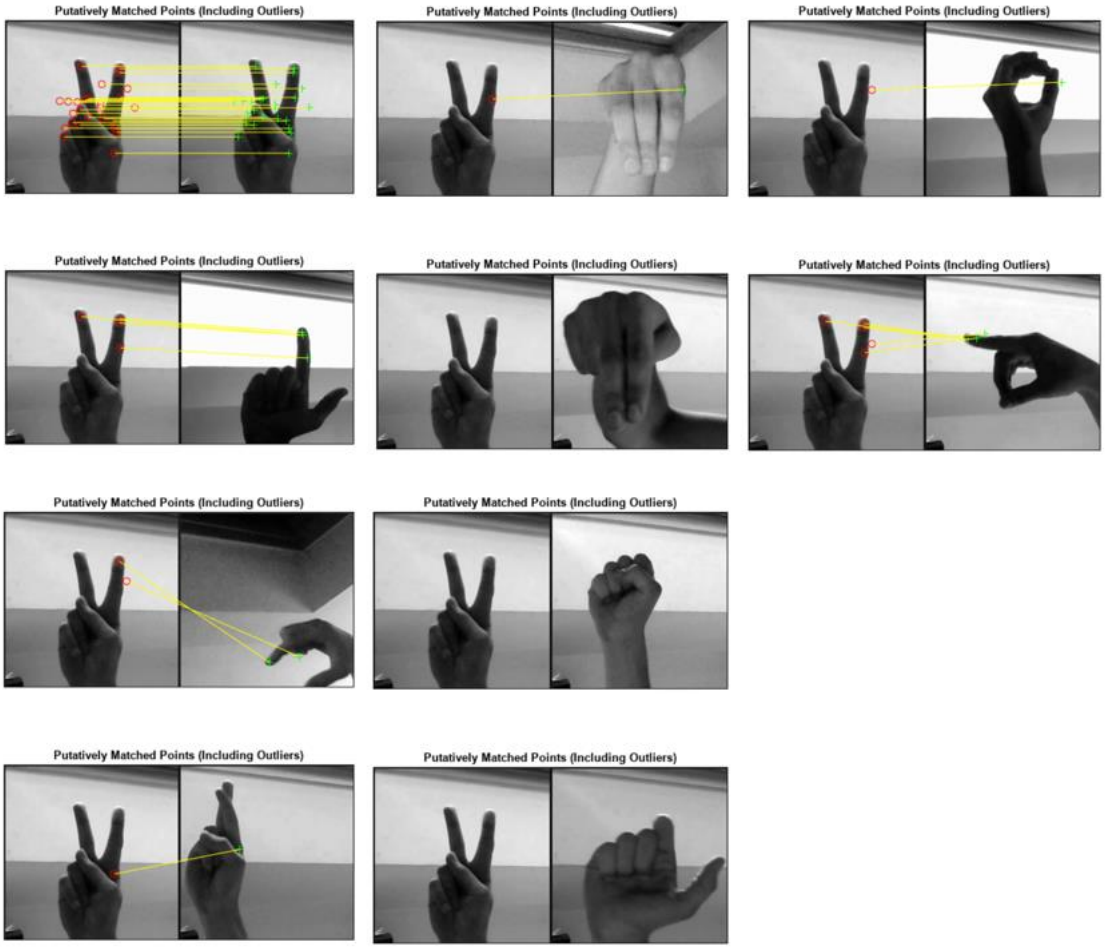


Figure 4 Feature extraction with number of matched SUFT features (example)

3.2. Bayesian Network construction and validation

We validated the model with multiple partitioning rate ranging from 0.1 to 0.3 for test partition. As a results, the accuracy improved as the rate decreased.

Besides, the partition rate, we calibrated the model with multiple values for the number of segments as well ranging from 5 to 10. A different pattern of the evolution of results was captured for this parameter, the accuracy improved significantly as the value increased.

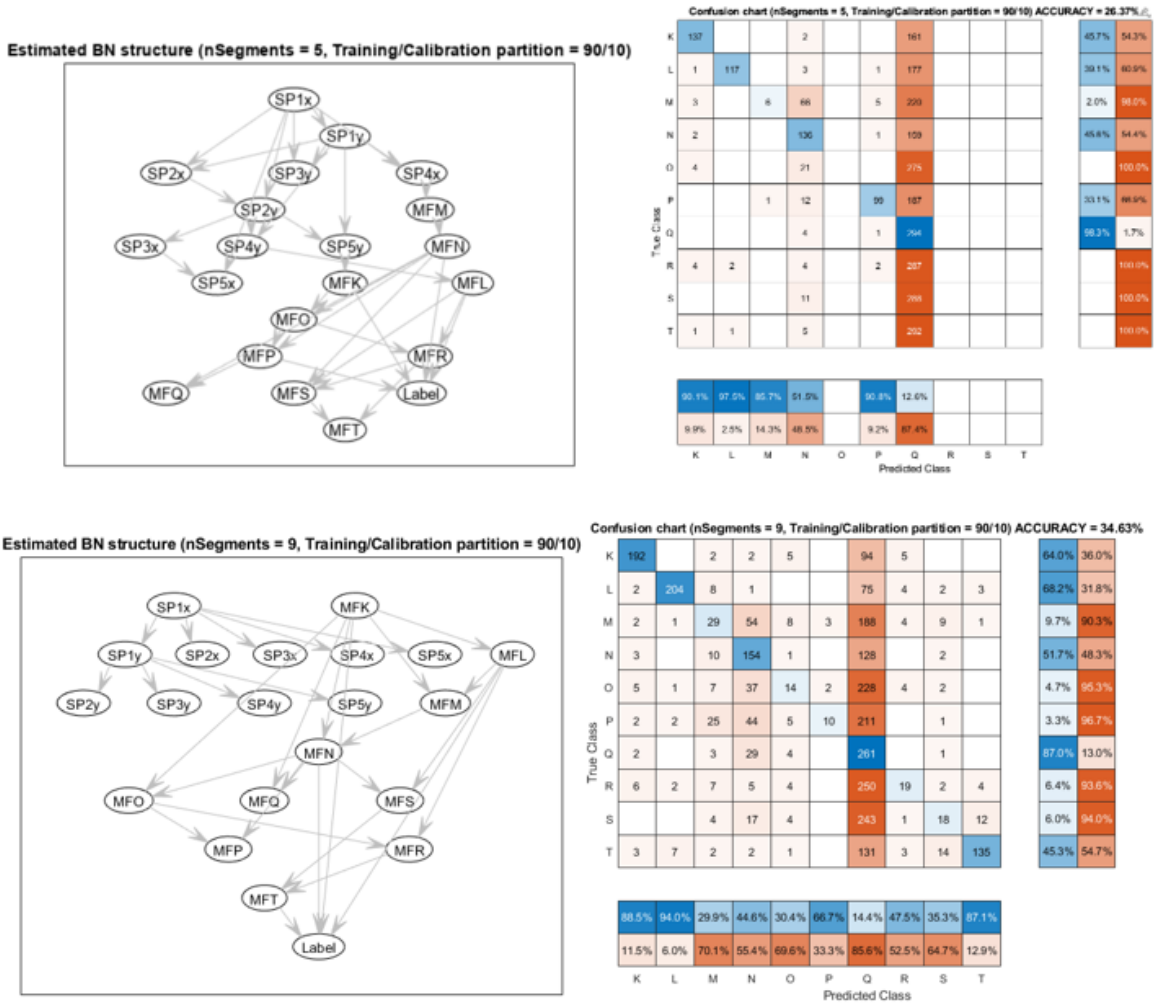


Figure 5 Bayesian Network graph and Confusion matrix on validation data set

The best accuracy during the calibration process is 34.63% which is over 3 times higher than random guess. The class “K”, “L” and “T” have high precisions while that the model tends to predict samples as class “Q” too often makes its recall measure high.

The accuracy of the model with 2 sets of features on the testing data containing 1 test sample for each class was approximately 40%.

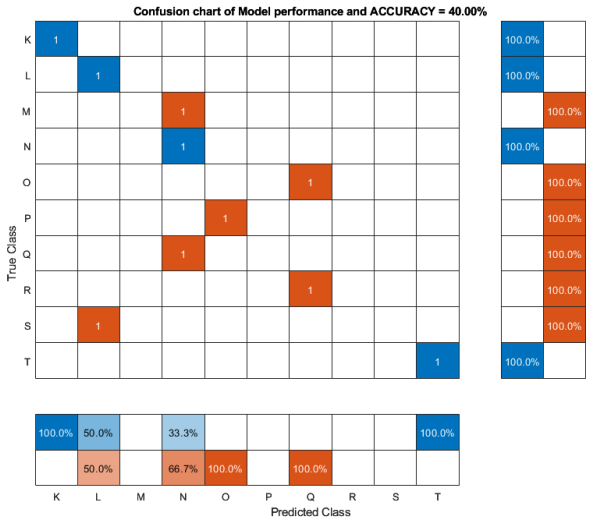


Figure 6 Testing accuracy with all features

It is undeniable that the second set of features as mentioned in 3.3 is highly directly connected to the label. Even though there is no disconnection after calibration with different parameters, it might be worth trying training model with only the second set of features to see whether there would be any improvement. As for this stage, we decided to go with the full set of features with the model trained with 9 segments and 90% of the training data.

After trying with only the second set of features, there were a slight improvement in the validation accuracy and test accuracy increased by up to 20% and the degree of complexity has been reduced significantly. We now have only 10 columns of features. Therefore, we decided to continue with the second set of features only.

3.3. Naïve Bayesian model construction and validation

We also tried building Naïve Bayesian classification model and the results were considerably improved with the validation accuracy ranging from 51% to 53% for any validation partition proportion from 0.2 to 0.8. That means that the model does not require much data to reach its best possible performance on the validation data set.

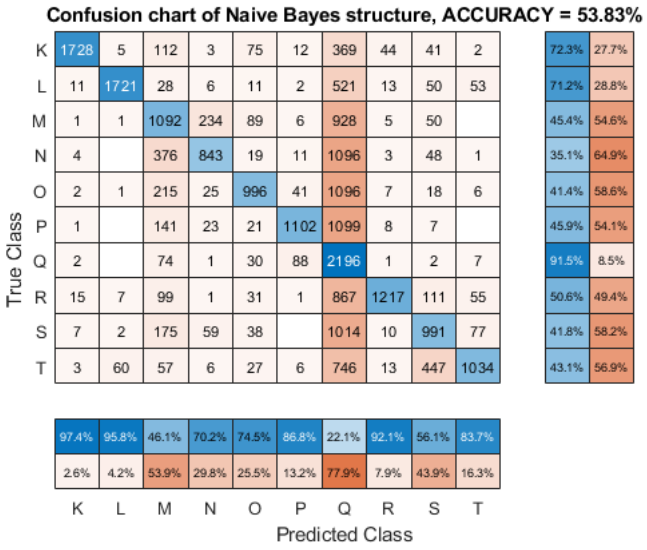


Figure 7 Confusion matrix on validation data with trained Naive Bayes structure

Class “K”, “L”, “P”, “R” and “T” had over-80% precision. Class “Q” had very low precision while its recall measure got 91.5%, that implied that this model classified objects as class “Q” too often and that results high False-Positive value. Class “K” and “L” not only had high precision, but also acceptably high recall, and that suggested that the model performed well in classifying these two classes and distinguishing them relative well from others.

The correlation between extracted features have been examined to verify how our data set aligned with the Naïve Bayesian structure’s implication of total uncorrelation among variables.

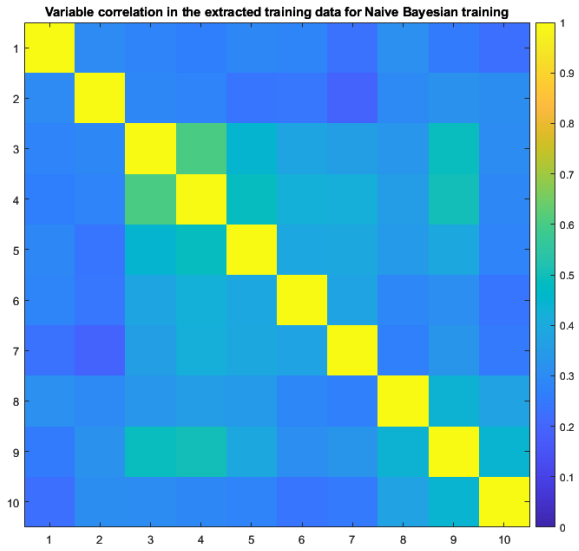


Figure 8 Correlation chart of variables used in training Naive Bayesian structure

The correlation plot demonstrated that most of the extracted features were weakly correlated with each other. Some features such as the third and the fourth ones, or those two and the nineth ones had correlation coefficient higher than the average 0.5. Even if the assumption did not strictly hold, great results can still be achieved with the Naïve Bayesian approach for our data set. The reason for this fact can be that Naïve Bayesian model can still make predictions based on the individual features, as long as the correlations are not too strong.

The Naïve Bayesian classification model was used to test with the test data to observe the overall performance.

3.4. Testing and Results

The final model trained with Naïve Bayesian structure was selected. The performance of this model on the test data set was impressive. Probably due to the small size of the test set, our model was able to obtain a perfect classification performance with 100% accuracy.

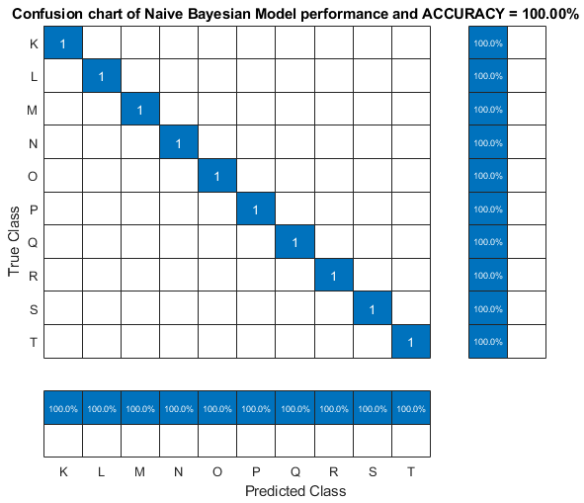


Figure 9 Confusion matrix of performance of trained Naive Bayes structure on test data

4. Convolutional Neural Network (CNN)

4.1. Data Pre-treatment and Augmentation

The data set is perfectly balanced, the sizes are the same for all images and the backgrounds do not include many noises that would adversely affect the training process. Thus, there were

no needs to perform pre-processing operations for object-centering and transformation prior training.

However, some data augmentation could be performed to diversify the training data set even further to hopefully improve the generalization of the retrained model. We could try with translation transformation that shifts the input image horizontally and vertically by a distance selected randomly from the range [-30, 30] pixels. Additionally, each image is allowed to be reflected horizontally with 50% probability.

Some other noise effects can be randomly added to the training set such as ‘gaussian’ noise and motion blur in later retraining process to improve model’s generalization.

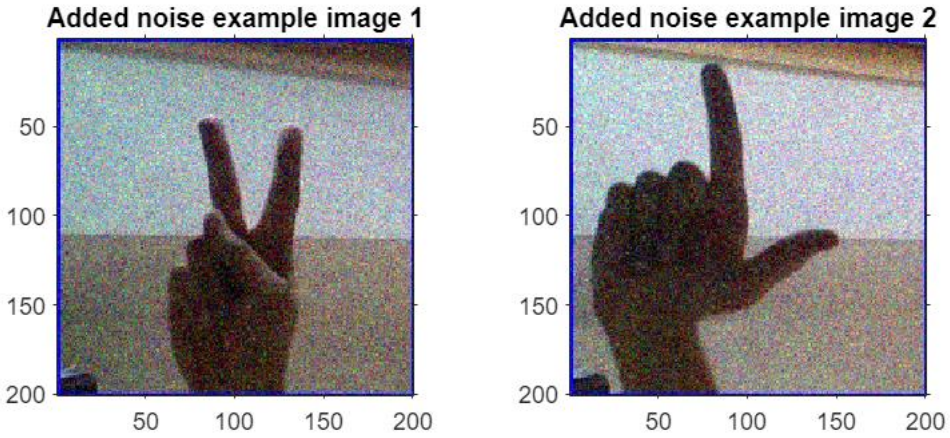


Figure 10 Examples of Gaussian noise added to training images

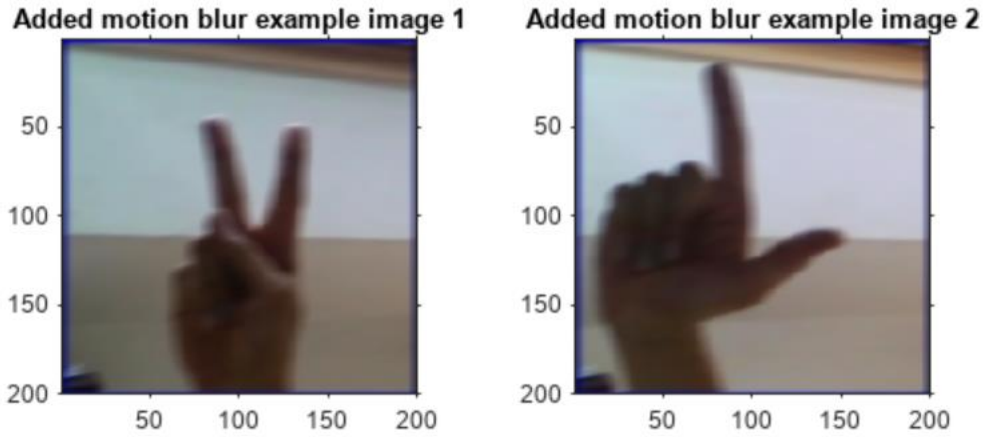


Figure 11 Examples of motion blur added to training images

4.2. Transfer Learning from “Resnet18” and “Resnet50” DNNs without Gaussian noise and motion blur effects added

4.2.1. Training and Validation

A trial retraining process has been performed with 30% of all data for training and 70% for validation and the selected pretrained DNN was “resnet18”. Even though the partition ratio was determined by mistake, the performance of the model was surprisingly perfect. The process took approximately 16 hours of fully training on a single GPU with max 6 epochs with the final validation accuracy of 99.98%.

The architecture of the retrained model was kept the same as the selected pretrained model that it was transferred learning from. A slight modification was done to the “fullyConnectedLayer” and the “classificationLayer” to adapt to the training data of this project. The trained model has 72 layers with 18 layers deep. The pretrained model was trained to classify 1000 different object categories with image size of 224x224x3. This model was selected due to the high similarity in the input and output types as our expected model and the fact that the size of images fits our dataset with only 3 pixels resizing horizontally and vertically.

Another model was retrained from the pretrained DNN “resnet50”. The architecture was similarly modified as in the previous model training. Instead of keeping to training until the max number of epochs was reached as previously, we decided to stop the training after acknowledging that the learning could not further improved, and the validation accuracy already reached maximum. It took roughly 13 hours for training.

The retrained models can be accessed from the repository with the relative paths “*ASL_American_Sign_Language_To_Speech/services/resnet18_transfernet.mat*” and “*ASL_American_Sign_Language_To_Speech/services/resnet50_transfernet.mat*”

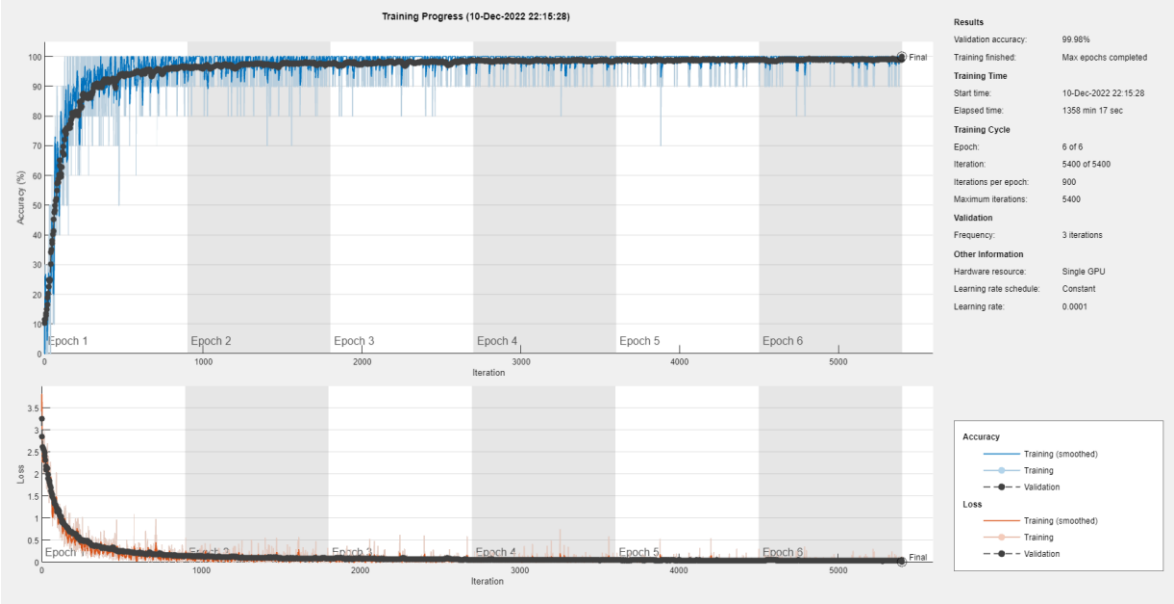


Figure 12 Training monitor of model resnet18_transferednet

Figure 13 Training monitor of model resnet50_transferednet

4.2.2. Testing and Results

The above retrained models performed very well on our test data with a perfect 100% accuracy.

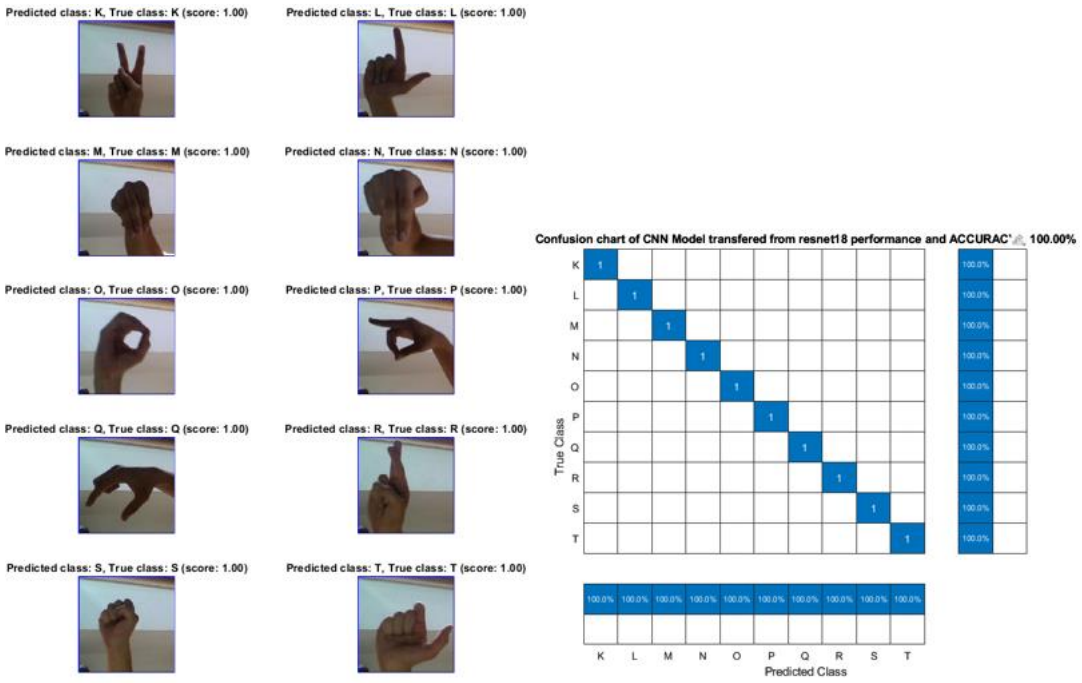


Figure 14 Classification performance of trained model resnet18_transferednet on original test set

4.3. Transfer Learning from "Resnet18" DNNs with Gaussian noise and motion blur effects added

4.3.1. Training and Validation

Due to the eagerness to generalise the model even further with noisy data, another model has been transferred learning from the "Resnet18" pretrained DNN and trained on a new augmented dataset with images having randomized Gaussian noise, motion blur effect and x-, y- translation effects added. The aim of this extra effort was to generate a model which was expected to be able to somewhat correctly perform classification with real-time snapshot with unseen noises. Therefore, the capability of generalization was highly concentrated for improvements.

The model reached perfect validation and training accuracy at the second epoch after around 6 hours. The model can be accessed from the repository with the relative path "*ASL_American_Sign_Language_To_Speech/services/resnet18_transfernet.mat*".

Figure 15 Training monitor of model resnet18_transferednet_with_noise

4.3.2. Testing and Results

The test set was also augmented with the same set of distortion effects but with high random probability to allow relatively more distortions applied. The above retrained model performed very well on the augmented test data with a perfect 100% accuracy, some classes get slightly less than 100% confidence.

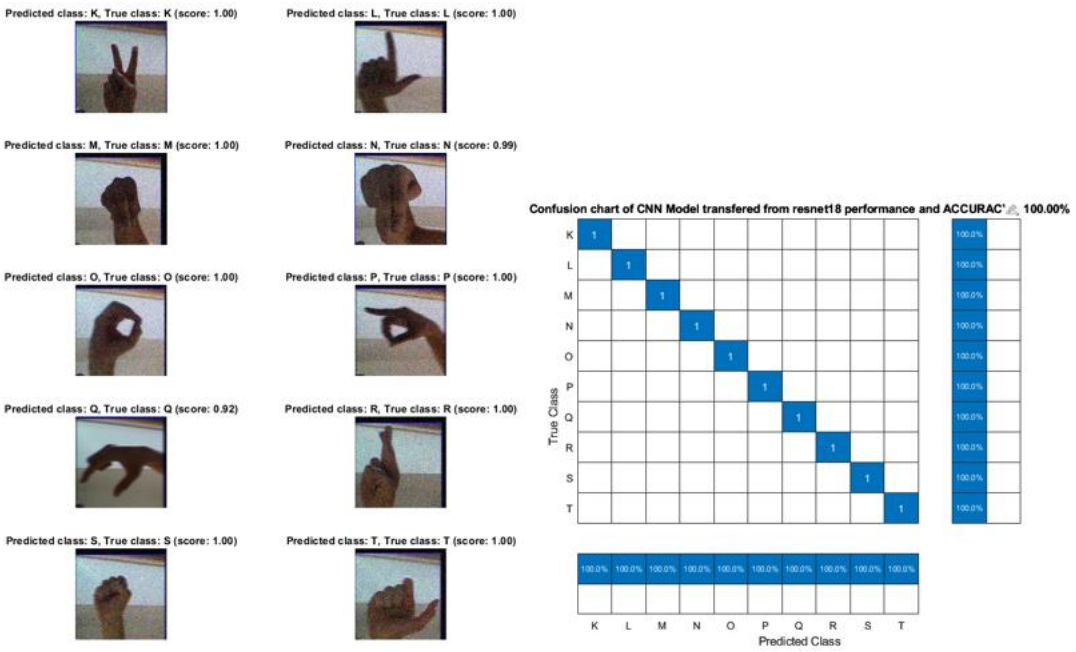


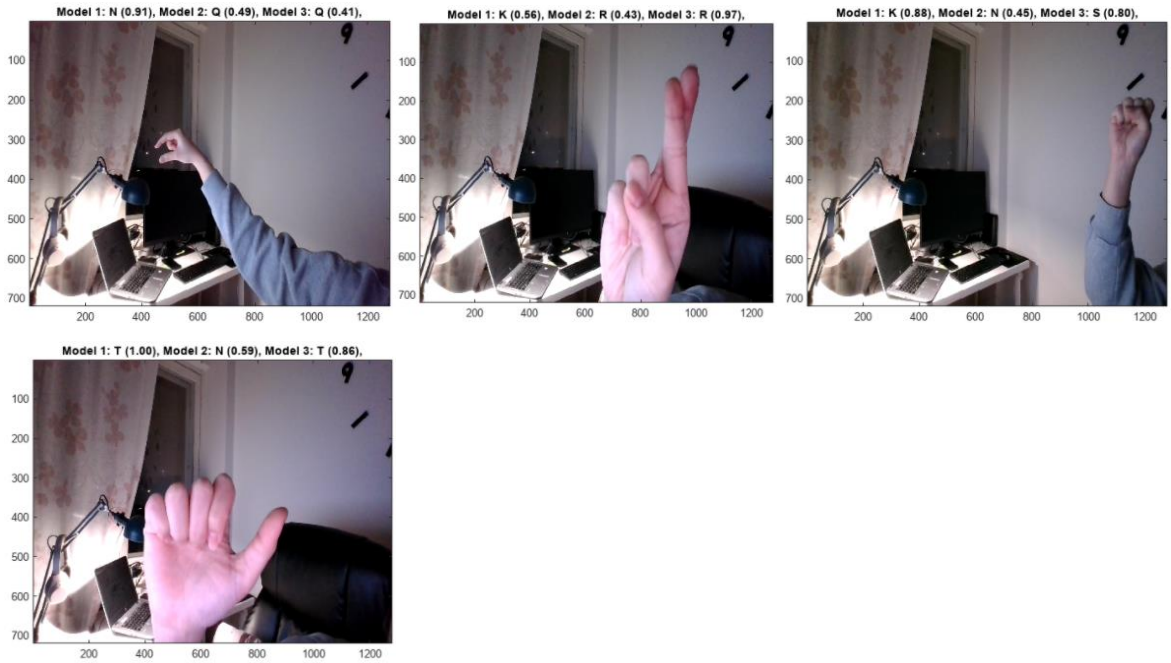
Figure 16 Classification performance of trained model resnet18_transferednet_with_noise on noisy augmented test set

4.4. Real-time experiment

In summary, we have totally 3 different models transferred learning from 2 pretrained DNNs which are “resnet18” and “resnet50”. Even though all of them performed perfectly correct on our available test data, we can test their performance on new images taken as a continuous snapshot with various unseen noises such as new shapes of hand, lighting condition, background having various objects, etc.

All audiences are welcomed to do an experiment in their own pace. The program can be found from “*ASL_American_Sign_Language_To_Speech/webcam_experience.mlx*”.

Below is one example of experiment with all 3 trained models namely resnet50_transferednet (model 1), resnet18_transferednet (model 2) and resnet18_transferednet_with_noise (model 3) respectively.



The last trained model "resnet18_transferednet_with_noise" had the highest chance to provide the correct class during the experiment among the three models. Class K was correctly detected all the time, class L was also detected correctly most of the time and sometimes, it was wrongly classified as class T. Class M was rarely distinguished from class N. Class P and class Q were similar to the models. The distance of the hand from the camera and the angle of the hand can affect output of the program.

4.5. Conclusion and Scope of Improvement

After assessing the performance of all three trained models, the model "resnet18_transferednet_with_noise" was selected as our final model for this classification problem.

Due limited computational and time resources, we decided to proceed with transfer learning method using pretrained deep neural network models. For the same reason, there was no time for the performance of hyperparameter optimization and tuning, all retrained models have the same architecture as the model that they were transferred learning from. Therefore, there are still rooms for improvements.

5. Summary

In this project, a Bayesian network, a Naïve Bayesian model and 3 transferred learning convolutional neural networks have been trained. Their performance has been examined and compared. For Bayesian-related method, the Naïve Bayesian structure produced much higher classification accuracy than the Bayesian network with only one set of extracted features. For CNN, even though all three models were able to classify the test data set with perfect 100% accuracy, the last model trained with multiple noise and blur effects added had higher capability of generalization on unseen images.