An extensive evaluation on the performance between a multi-class Support Vector Machine classifier and a deep learning classifier using Convolutional Neural Network

Tien Dat Nguyen – u3234033 - Computer Vision Concepts Implementation – CVIA PG 8890

Abstract

Over the last decade, the emergence of Artificial Intelligent (AI) agents into the humanity working cycle has increased rapidly and proven considerable usefulness. AI agents have been developed to handle from simple to complicated work-related tasks with an increasing performance rate. Especially in computer vision and image analysis, AI has been widely used for multiple purposes such as surveillance, object detection and recognition, image classification, and so forth. The agents or models used in computer vision area often come with an accuracy rate or confidence rate, which represents how accurate or confident the result obtaining from the model is, which is very important especially in those tasks that require absolute precision such as medical image analysis, thus developing models with high accuracy performance are one of the main targets of the IT practitioners and researchers nowadays. This report will present some key findings of the evaluation on the performance between a machine learning (ML) multi-class Support Vector Machine (SVM) image classifier and a deep learning (DL) multi-class image classifier using Convolutional Neural Network (CNN), of which the results will be useful for those who seek to understand both the overall performance of each model, and most importantly the difference in level of performance between the two models. Toward the end, the report will also discuss what other approaches or modifications we can make to improve the outcome of the trained models.

# **Introduction**

In order to generate sufficient information for the performance evaluation, a set of procedure is followed:

* Deciding the methods and approaches for the machine learning and deep learning models, including the relevant parameters
* Preparing the dataset for training. The dataset in use is the Caltech-UCSD-Birds-200-2011 dataset which contains 11,788 images of 200 bird species. The dataset is divided into 3 subset: training, validation, and testing subsets with the ratio 60:20:20 respectively.
* Extracting the image features and image labels from the input datasets for machine learning classifier, and defining the CNN and training options for deep learning classifier
* Training the classifiers
* Test the classifiers with the test partition and calculate the accuracy as well as class-wise recognition rate for performance evaluation

There are totally 5 experiments implemented for the study. Firstly, the SVM classifier is trained and tested on the dataset with whole image as input (experiment 1), followed by the experiment 2 of the deep learning classifier being trained on the augmented whole images, validated and tested on the same validating and testing partitions. The same machine learning classifier and deep learning classifier are trained and tested in experiment 3 and experiment 4 respectively on dataset containing not whole images but bounding box image areas as input, in which the bounding box area for each image is determined in a separated text file. The forth experiment also augments the training images for training and uses the same validating and testing partitions as experiment 3 for validating and testing purposes. The fifth experiment performs fivefold cross-validation on experiment 4 for further discussion of the result compared to the others.

# **Machine Learning Classifier:**

The chosen approach for the machine learning classifier, as mentioned, is Support Vector Machine (SVM). Support Vector Machine is an algorithm that contributes the learning ability to the supervised learning models to analyse data for classification and regression analysis [1]. The algorithm works by creating a line or a hyperplane between the training inputs and mapping the training examples to points in space in order to maximize the width of the gap between the two classes. The model repeats the process by mapping new training inputs into that same space and predicting the category those new inputs belong to base on which size of the gap they fall [2]. The model using SVM works well with datasets that include data of different classes that possess highly distinctive characteristics as the clear margin of separation between classes will be much useful for the SVM model to draw the separating line between the classes, hence enhancing the classifying ability. Another advantage of using the SVM classifier is that it is highly effective in high dimensional spaces [3] therefore training a machine learning classifier for a dataset containing 200 different classes using SVM algorithm will enhance its effectiveness.

Scale-Invariant Feature Transform (SIFT) and Histogram of Oriented Gradient (HOG) are the two classic handcrafted feature extraction methods to be used for training in the experiments. SIFT method extracts image features that are invariant to scale as well as orientation of the images, robust to illumination fluctuations, noise, partial occlusion, and minor viewpoint changes in the images [4]. The images of the birds in the dataset in use for the experiments display different angles of each bird species and are taken in different time and distances, therefore the scale-and-orientation-invariant and robust to illumination characteristics of the SIFT features will eliminate or at least minimize the image difference elements mentioned and aid extracting merely the important pixels. Moreover, the SIFT features are also affine invariant, meaning its feature descriptor could tolerate out-of-plane rotations up to 60 degrees, which will be much helpful for forming description of the birds from different angles displayed in the image dataset.

A common problem with many of the handcrafted local image feature descriptors including SIFT is that they do not capture spatial relationships inherent to known object classes. The experiment 1 and 3 will look into a possible approach that is simultaneously using SIFT feature extraction method with another image feature method like HOG method, which captures such spatial relationships of an object or class, to see if the result will be better than using merely one feature extraction method alone. HOG method focuses on the structure or the shape of an object by breaking down the input image into smaller regions and extracting the gradient and orientation (magnitude and direction) of the edges for each region, and then generates a histogram for each of these regions separately using the gradients and orientations of the pixel values [5]. The images in the dataset have a clear separation between the background and foreground therefore when extracting features with HOG descriptors, the background regions will have the same orientation and the specific regions that contain shifted orientations next to the background regions will be the ones to form the shape of the object, hence indicating the important pixel values are in the inner shaped area and combining with the location of the SIFT points could result in better combined output features for the SVM model to train on. Figure 1 shows an example of SIFT and HOG features on a random cropped image by the bounding box area in the dataset.

# **Deep Learning Classifier:**

The deep learning classifier is built using Convolutional Neural Network. Neural Networks (or known as Artificial Neural Networks (ANNs)) are a subset of machine learning and are at the heart of deep learning algorithms [6]. The structure and name of ANNs are inspired by the human brain, and by the way that biological neurons signal to one another. Convolutional Neural Network is used for the experiments because of its superior performance with image particularly compared to other types of Neural Networks [7]. CNNs are specifically designed to process pixel data and are widely used in image classification, recognition and processing. CNNs make use of the hierarchical structure of the data they are processing, that is, they break the whole input image down into smaller features, which are represented by filters. These filters are applied to different regions of the image to obtain relevant information. The features then are combined and assembled into more complex patterns when the network progresses through the layers, which allows the model using the network learn increasingly abstract representations of the input image [8].

In terms of training options to train the network, first of all, the initial learn rate is set to 0.001. The learning rate is set to be quite low in order to allow the training to reach its optimality, and the training time will certainly be slower than a higher learning rate as a trade-off for optimizing the training result. Secondly, the maximum epoch is set to 20. An epoch is the full pass of the training algorithm over the entire training set. Low number of epoch could lead to unfitted model whereas high number of epoch could lead to overfitting. Another option that can control the training progress like maximum number of epoch is the mini batch size. Tuning the mini batch size will result in different number of iterations in the training progress, and an iteration is one step taken in the gradient descent algorithm towards minimizing the loss function using a mini-batch [8], therefore we want a mini batch size that returns a good number of iterations. Results from a research have shown that using small batch sizes achieve best training stability and generalization performance and the best results were associated with the mini batch size of 32 [9], hence the chosen mini batch size for these experiments is 32. The training and validation data is shuffled before each epoch by setting shuffle option to every-epoch to enhance the quality of training and validating. Another noticeable addition to the programs for deep learning classifier is that the images in the training datastore are applied with pre-processing (augmentation) before being used to train the model. The difference in the results and inference on the possibilities of the changes will be discussed further in the latter part of the report.

# **Results and Discussion:**

Table 1 in the appendix shows a table of results for all experiments including classification accuracy, total time to train the classifier, and the range of class-wise recognition rate of each classifier. Initially, the programs developed for experiment 1 and 3 crashed at the stage of training the classifier, sending an out-of-memory error. The potential explanation for the event is that the target size of the input images and the feature size of the output feature were quite high to process, as well as the size of the whole dataset is quite large, which put an extensive workload on the CPU of the local machine and used up the available memory before the training is completed. Multiple attempts were made in order to ensure the programs run completely and the option of reducing the size of the input image down to [100 100] and output feature size to 194 ( SIFT feature size 50 + HOG feature size 144) resulted in the programs running completely with a trade-off of losing image information hence decreasing the classification accuracy of the models.

**Experiment 1: SVM classifier using whole images**

The total time to run the program developed for experiment 1 is 7 hours 13 minutes, which is quite long compared to the accuracy of the trained model from the program which is only 0.68%, and the range of class-wise recognition rate from 0% to 33.3%, meaning most of the images from the test set is classified into the wrong class and the class with the highest recognition rate is just 33.3%. As discussed, the extremely low accuracy of the classifier is the result of extracting merely a small amount of features from the input image, and resizing the images from the dataset resulted in losing information per image hence impacting the overall accuracy. Another attempt has been made using only SIFT feature extraction method with a the feature size of 500 resulted in a longer training time (more than 9 hours) but higher accuracy (4.17%). Comparing the time spent on each program and the noticeable difference in the accuracy could potentially mean that a model using a combination feature extraction method underperform one that uses only one feature extraction method. However, there are many differences between the two programs such as different feature sizes, different results every time the programs are run again. Moreover, the methods could possibly return different perspective if they are used for different datasets, therefore the statement above remains potential until further related studies are implemented.

**Experiment 2: CNN classifier using whole images**

There were two programs developed for the CNN model in experiment 2 in which the difference lies in the training image datastore. The first program used a training datastore which contains images that were resized only, and produced a classifier with accuracy of 10.93% after running for 47 minutes. The class-wise recognition rate of the first program is 0 – 45.5%. Figure 4 shows the training progress of the experiment 2 with the resized training images. The second program used a training datastore which had resized input images augmented using an image augmenter. The augmenter randomly reflects the input images horizontally or vertically, as well as randomly rotates the image in the range of -20 degrees to 20 degrees, and applies random affine translation from -3 degrees to 3 degrees to the image. There were a considerable difference between the results of the two programs as the latter one ran for only 40 minutes and produced a classifier that performed nearly 1.5 times better than the first one (14.19% accurate vs 10.93% of the first one). The range of the class-wise recognition rate of the second one is also much better as the highest recognition rate is 83.3% compared to 45.5% of the first one. Figure 3 shows the training progress of the experiment 2 with the augmented image datastore. The potential reason behind the better performance of the classifier when it is trained on an augmented image datastore is the different amount of important information that the classifier is trained on. The whole images used for training contain background areas of which the information is irrelevant and redundant. Augmenting the images, especially by rotating and affine translating, led to the disappearance of some of the background areas and replaced by black areas that contain no information, hence cancelling out some of the unimportant pixel values to improve the outcome of the model training progress. Figure 2 shows a sample of an augmented training image by the image augmenter where it can be seen that the background at the corners has been replaced by black colour due to rotation. However, image augmentation with random arguments in the image augmenter is not a solid solution for improving the performance of a ML or DL classifier and does not effectively work in every case when it is used. Experiment 4 will introduce a negative side effect of augmenting training images that reduce the performance of the classifier.

**Experiment 3: SVM classifier using bounding-box-image areas**

The program used in experiment 3 developed a similar SVM classifier that is trained on bounding box image areas instead of whole training images as in experiment 1. The combined feature size is also slightly bigger (244 compared to 194 in experiment 1) but the total running time of the program in experiment 3 is much less as it only took 4 hours 32 minutes to complete. However, the performance of the model developed is also very bad as its accuracy is only 0.88% and the range of class-wise recognition rate is 0-11.1%. Again the potential reason behind the poor performance of the classifier is that the number of features extracted from the training set is low resulting in sufficient information for the classifier to learn. The attempt of increasing the combined feature size has come to a failure because of the out-of-memory error again. Another attempt of using a different program which uses only one method of extracting feature which is SIFT with a feature size of 500 resulted in a longer time to complete training progress (6 hours 17 minutes) but higher accuracy (7.56%). A discussion on the difference between the results has been made in above in the experiment 1 section since what happened with the two programs in experiment 3 is similar to which in experiment 1.

**Experiment 4: CNN classifier using bounding-box-image areas**

Experiment 4 was implemented with two programs developed to train a CNN model using bounding box image areas as training images. The first program transformed the training set into an augmented image datastore using the same image augmenter used in experiment 2, which created a classifier with an accuracy of 23.77% after 41 minutes of training. The class-wise recognition rate of the classifier ranges from 0% to 91% which is one of the best ranges of all experiments. Figure 5 shows the classifier’s training progress of the experiment 4 using augmented image datastore. The second program used an image datastore which contains resized bounding box image areas only. The accuracy of the classifier from the second program is 28.65%, with a class-wise recognition range of 0-83.3%% after 45 minutes of training progress, which is shown in figure 6. The difference in the results of these two programs is contrast to which in the experiment 2 as in this experiment the program that trained the classifier on an augmented image datastore produced a classifier that underperforms the one from the other program (23.77% compared to 28.65%). The rational explanation for this contrary is that the bounding box labels narrowed the input images down to the areas in which the object of interest (the birds) cover most of the areas and only few surrounding areas are covered by the background, therefore most of the pixel values in the input images are important, and augmenting the images will cancel out some of those important bits which will lead to lost of important information of the image hence negatively affecting the image understanding capability of the model.

**Experiment 5: CNN classifier using bounding-box-image areas and fivefold cross-validation**

In this experiment, the number of layers as well as the options to train the CNN model are the same as the ones in experiment 2 and 4, however the model is not trained on pre-defined training, validating and testing datasets like which in the other experiments, instead the whole image dataset is randomly divided into 5 partitions and the program trained the model five times with different parts used for training, validating and testing. The input images of the whole dataset is resized to [224 224] and no further pre-processing is applied. The results of the five runs are 27.43%, 26.13%, 27.07%, 25.87% and 23.55% which results in the average accuracy of the CNN model to be 26.01%. The total time to run the whole program for this experiment is 4 hours 10 minutes, of which each run to train the model takes around 45 – 50 minutes. The results from running the program indicate that the accuracy of the trained model could be different every time it is trained again, however even though the training, validating and testing datasets are different in each run, the performance of the trained model is still consistent as the difference in the performance is relatively small. The use of fivefold cross-validation did not enhance the performance of the model after training, because in each run, the program produces a different model that is trained and tested on different data partitions; however there are still many advantages with the use of cross-validation. Firstly, training five different models on different subset of the input data will give us a broad visualization of the model performance in order to evaluate the overall effectiveness of the model as well as the chosen parameters. Moreover, using fivefold cross-validation can also aid detecting overfitting since running multiple simulations can increase the chance of finding overfitting if it exists [10]. In addition, pre-defined subsets for training, validating, and testing used in the other experiments reduced the amount of data that we can use for training, and also could result in biased outcome because the input images are pre-defined, not randomized. The use of cross-validation minimized this drawback by randomizing the training, validating and testing partitions in each run and producing different results as shown above, and from which we can determine the range of possible outcomes of the model for further evaluation. Comparing these results to which in experiment 4, we can see that at one run the accuracy of the CNN model using resized-only image still performed slightly worse than the one using augmented image datastore (23.55% compared to 23.77%), which gives us another perspective for further studies on evaluating the outcome of the CNN models using different pre-processed data.

**Cross-evaluation**

The SVM classifier generated in experiment 1 using whole images as training input performed the worst of all models and took longest to complete the training progress. With the same dataset of whole images, the CNN classifier in experiment 2 produced a much higher accuracy while taking much less time to complete training. Further comparison has been made with experiment 3 and 4 as experiment 3 generated a SVM model which is trained on cropped images, and the accuracy of this model is still much less than the accuracy of the CNN model built in experiment 4 with the same training dataset. This has led to an insight that on a particular dataset that contains images as data, a deep learning model will perform better than a machine learning model in terms of classification. The potential reason is that deep learning models handle the unstructured data (data such as images, audio or video files) better than machine learning models which are commonly used for handling structured data (organized, predefined-formatted data such as dates, phone numbers, customer names, prices and so forth) [11]

Regarding data pre-processing and its effectiveness, it can clearly be seen from the results of all four experiments that the models trained with pre-determined bounding box image areas, regardless of the chosen approach (machine learning or deep learning), performed significantly better than the models trained with whole images. The difference is believed to be coming from the difference of the object coverage over the input image, as in the whole image the background takes up most of the areas and the birds cover only a small part whereas in the cropped images the birds take up most of the area, leaving only small parts of the images for the background. By converting these differences into ratios, we will see that the higher the percentage of the bird area in the training images is, the better the classifiers perform. A relevant discussion on how the amount of important pixel values retained in the input images affects the general performance of the classifiers has also been presented in the previous part where we discussed the different outcomes of the models using augmented image datastore and using resized-only image datastore.

Even though there are considerable differences in the result regarding the time spent for training progress of each experiment, especially training a CNN model takes only less than an hour to finish while training a SVM model takes up to and more than five hours, it still does not completely reflect the efficiency of using one of the approaches because the option of training the CNN model using GPU speeded up the training progress whereas the training of SVM model did not have the option of using the GPU therefore it was slower. If the program of generating the CNN model were to be run on a machine that does not have a GPU then the progress could take up to or more than 10 hours to finish, which means that there is a trade-off of the better performance of a CNN model that it will take longer to complete the training progress and could possibly take up more space in the memory as well.

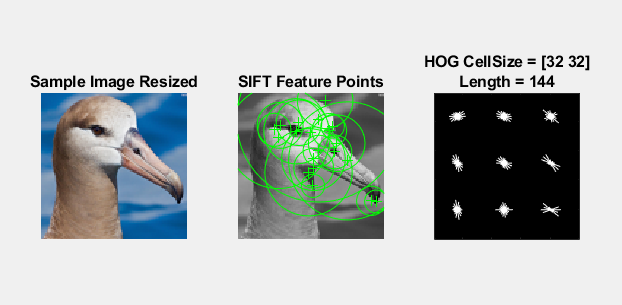
# **Conclusion and future work:**

When working with complex data that are not easy to analysed and leverages like images, using a deep learning model (e.g. model using CNN) will be a better option than a machine learning model (e.g. model using SVM) for complicated tasks like classification, and also the outcome of the model’s performance will partly depend on how well the data are pre-processed. Even though this is a preliminary study and the proposed models need to be optimized and improved on a larger dataset, there are still some future perspectives on what modifications can be made in order to improve the outcomes. Initially, understanding the impact of data pre-processing on the results, we could apply more processing to the data to improve the performance of the models, by some other methods such as using histogram equalisation to enhance the contrast of the background and foreground, or using image segmentation to segment the background and foreground and remove the background so as to ensure the model train solely on the important bits. Moreover, with the SVM models, we can try running the program on a stronger machine where the memory issue is no longer a problem preventing the program to complete. In that case, attempts to increase the size of the input images as well as increase the size of the output features could possibly result in better performance of the models since these practices could retain more information per training image for the model to learn. Furthermore, experimenting the model using different feature extraction methods or even combined methods with the available methods such as HOG, SIFT, SURF, MSER, Harris Corners, and so forth and evaluating the difference in the results will be useful to generate insights on the performance of the model with different method, as there has not been any practical knowledge on which method is better than the others. The SVM program did not make use of the validation set as well because there is no option for validating the training of the model, therefore we can also combine the validation set into training set to train the SVM models, which will certainly increase the performance of the classifier as well. With the CNN models, adding more hidden layers will possibly increase the performance of the models since the deeper the layers are, the better learning the model is capturing. Moreover, we can modify the minibatch size and the number of epochs to add up more iterations into the training progress because, as mentioned, one iteration is one step taken towards minimizing the loss function.

# **References**

1. Noble, W. What is a support vector machine*?. Nat Biotechnol* 24, 1565–1567 (2006). <https://doi.org/10.1038/nbt1206-1565>
2. Wikipedia, Support Vector Machine. <https://en.wikipedia.org/wiki/Support_vector_machine>.
3. Scikit-learn.org, Support Vector Machine. <https://scikit-learn.org/stable/modules/svm.html>
4. Y. Lu, A. Liu and Y. Su, “Chapter 6 – Mitosis detection in biomedical images”, *Computer Vision for Microscpoy Image Analysis*, 2021, pp 131-157. <https://doi.org/10.1016/B978-0-12-814972-0.00006-0>
5. A. Singh, “Feature Engineering for Images: A Valuable Introduction to the HOG Feature Descriptor”, 2020. <https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/>
6. IBM, “What is neural networks?”, <https://www.ibm.com/topics/neural-networks>
7. MATLAB, “List of Deep Learning Layers”,<https://au.mathworks.com/help/deeplearning/ug/list-of-deep-learning-layers.html?searchHighlight=list%20of%20deep%20learning%20layers&s_tid=srchtitle_list%20of%20deep%20learning%20layers_1>
8. MATLAB, “trainingOption”, <https://au.mathworks.com/help/deeplearning/ref/trainingoptions.html?s_tid=doc_ta>
9. D. Masters and C. Luschi, “Revisiting Small Batch Training for Deep Neural Networks”, 2018. <https://doi.org/10.48550/arXiv.1804.07612>
10. Scikit-learn.org, Cross-validation, <https://scikit-learn.org/stable/modules/cross_validation.html>
11. M. Federchuk, The difference between AI, ML, and DL, 2022. <https://www.cengn.ca/information-centre/innovation/difference-between-ai-ml-and-dl/#structured-vs-unstructured-data>

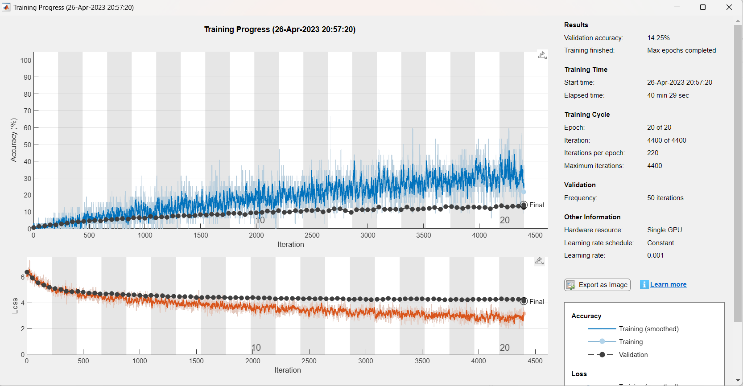
# **Apdendix:**



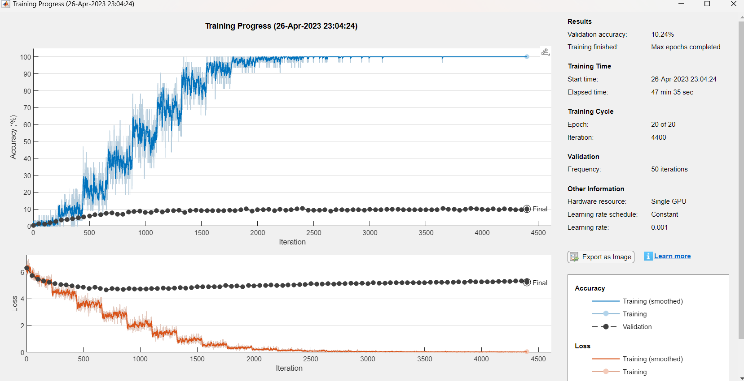
1. SIFT and HOG features in a sample image from the dataset



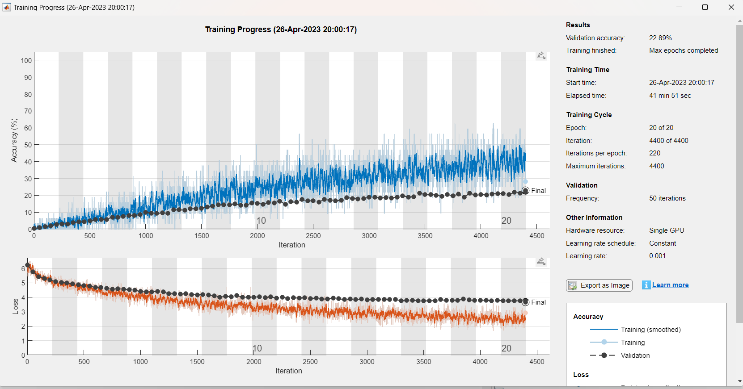
1. Augmented training image vs resized validating and testing images



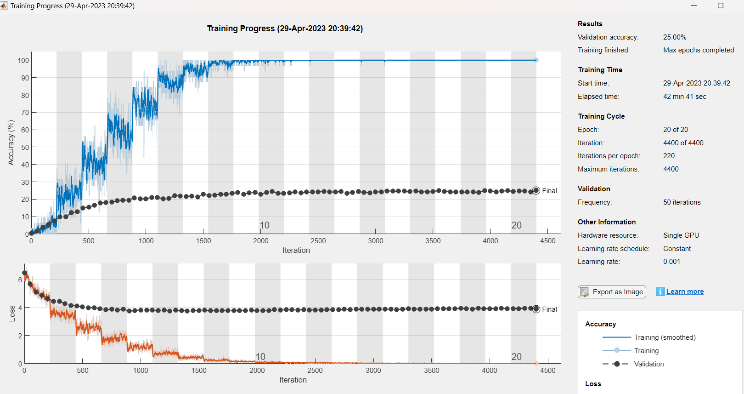
1. Experiment 2 – Training Progress with augmented image datastore



1. Experiment 2 – Training Progress with resized-only image datastore



1. Experiment 4 – Training Progress using augmented image datastore



1. Experiment 4 – Training Progress using resized-only image datastore

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Experiment 1 (SIFT + HOG SVM – whole image) | Experiment 2 (CNN – whole image) | Experiment 3 (SIFT + HOG SVM – BB area) | Experiment 4  (CNN – BB area) | Experiment 5 (CNN – Cross-validation) |
| Training accuracy | NA | 100% 60% (augmented) | NA | 100% 60%(augmented) | 100% |
| Validation accuracy | NA | 10.24% 14.25%(augmented) | NA | 25% 22.89%(augmented) | 24.67% |
| Test Accuracy | 0.68 % | 14.19 % | 0.88 % | 28.64 % | 26.01 % |
| Total running time | 7 hours 13 minutes | 40 minutes 29 sec | 4 hours 32 minutes | 41 minutes 51 sec | 4 hours 10 minutes |
| Class-wise recognition rate (CRR) | 0 – 33.3 % | 0 – 83.3 % | 0 – 45.5 % | 0 -91.7 % | 0 – 100 % |
| CRR’s Mean | 0.67 | 14.21 | 0.86 | 28.80 | 26.09 |
| CRR’s Mode | 0 | 8.3 | 0 | 11.1 | 8.3 |
| CRR’s Median | 0 | 16.7 | 0 | 25 | 27.4 |

Table 1: Experiment resuls