Maciej Legas 2031545

CSCM45 Coursework: Object Recognition

Introduction

The problem given is to experiment with solving a classification problem of a small subset of the CIFAR-10 dataset. The given subset consists of 10 class categories, 10000 training and 1000 test uniformly distributed samples, both of the shape of 32x32x3. The proposed solution is to implement and evaluate the most suitable machine learning methods for classification, which in this case are Support Vector Machines (SVM), Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN), as well as taking into consideration the effect of using a feature descriptor, the Histogram of Oriented Gradients (HOG), and dimensionality reduction using Linear Discriminant Analysis (LDA). The experimental results found the CNN model to be the most accurate at 73.7%, as well as having the lowest validation loss of 1.44, while the models with the worst prediction accuracies were the ANN (24.5%) and SVM (25.2%) models, which used LDA components as input. The fastest training times, 1.89 and 3.97 seconds, were achieved by the models using the output of LDA as input in a SVM (24.5%), and the combination of HOG + LDA + SVM (50.1%). The slowest training time, 476 seconds, was done by inputting the data directly into a SVM model (48.4%).

Method

Data manipulation

Pre-processing the data

Since the data has been given in a 32x32x3x10000 matrix for training and 32x32x3x1000 matrix for testing, it has been loaded and reshaped to 10000x32x32x3 and 1000x32x32x3 NumPy arrays for later use in a CNN, while also providing 10000x3072 and 1000x3072 arrays for the ANN and SVM use. As the provided data is uniformly distributed and sorted in an ascending order of the class labels, the arrays have been shuffled to ensure valid training.

Histogram of Oriented Gradients

To extract features for some of the tested methods of the experiments, HOG has been used, which due to its ability to show the gradients and orientations of edges of an image [1], is commonly used for object detection in the field of computer vision [2, 3]. Default values of the scikit-image implementation have been used, which are 8x8 pixels per cell and 3x3 cells per block.

Dimensionality Reduction

As the image input results in a large number of features, growing the cost of any algorithm due to the curse of dimensionality [4], a decision was made to test the use of a dimensionality reduction technique. LDA has been chosen, due to the experimentation performing supervised learning.

Classifiers

After fitting, each of the models will have their prediction score, and loss if applicable, evaluated on a selected type of pre-processed test data, depending on how the training data was processed to be used as input beforehand. A confusion matrix and classification report will follow as well for further investigation.

Support Vector Machine

A SVM classifier model will be fit in three different ways for further evaluation:

1. Using the training data directly as input in a SVM model
2. Using the extracted features from HOG as input in a SVM model
3. Using the dimension reduced, via LDA, extracted features, from HOG, as input in a SVM model

During the implementation, the SVM models will be tested with a number of kernel functions, focusing only on the non-linear types due to the amount of features. Different values for the C regularization parameter and gamma kernel coefficient will be as well tested.

Artificial Neural Network

As in the case of using SVM models, the implementation of ANN models will be fit in three ways as well:

1. Using the training data directly as input in an ANN model
2. Using the extracted features from HOG as input in an ANN model
3. Using the dimensionality reduced, via LDA, extracted features, from HOG, as input in an ANN model

In each of these ANN models, a decision was made to use a total of two hidden, dense layers, with the optional use of a dropout layer to reduce overfitting [5] where applicable. Following the convention of deep neural networks [6], a Softmax function will be used for classification, and the Rectified Linear Unit function (ReLU) will be used for activation.

Convolutional Neural Network

Due to the sophistication of the layers used in a CNN model, which use filters to generate feature maps of the image input to extract the high-level features, such as edges [7], only one test of a CNN model will be made, which is to use the input data directly without any data manipulation such as the use of HOG or LDA. Although the output from HOG could be still used, it is found to be usually underperform in CNN models [8], being more often used with SVM models instead [9, 10].

The model architecture used has been inspired by the VGGNet architecture model [11], after evaluating its performance score on the full CIFAR-10 dataset [12]. The architecture in this project mostly consists of a repeating pattern of having a 2D convolution layer, a batch normalization layer, ReLU activation layer, up until a dropout layer, and finishing after the repetitions of the pattern with a flatten, dense, dropout and a softmax classification layer.

The convolution layers focus on generating feature maps, using same padding to ensure no feature loss during the convolutions, with the later use of max pooling for feature reduction. Before the activation of each 2D convolution layer, a batch normalization function is applied, which helps with the possible issue of overfitting [13]. The same purpose is provided by the dropout layer. As in the ANN models, the CNN model will use the ReLU function for activation and Softmax function for activation as well.

Results

Each of the models mentioned previously will be evaluated by mainly taking an insight in their prediction accuracy, validation loss (if applicable) and training time. Precision and recall for particular classes, especially if the resulting values in their classification reports are exceptionally low, will also be used for evaluation with the use of confusion matrices.

To start with the SVM models results, the use of the Radial Basis Function (RBF) kernel function and a value of 1 for C, as well as the default value for gamma from the scikit-learn's implementation has been found to be the most effective in maximizing the prediction accuracy for the SVM models.

Fitting a SVM model with the training data directly as input has proven to be quite time consuming, as it took the model 476 seconds to fit, achieving a prediction accuracy of 48.4%. Using the extracted features via the use of HOG has improved the prediction accuracy to 59.9% and drastically reduced the training time to 44.38 seconds. The usage of the LDA output has heavily reduced the training time again, down to 1.89 seconds, at the cost of the prediction accuracy being 25.2%, which is way below the experimentation's benchmark of 44.68%. Using the HOG + LDA + SVM combination found to be the best overall result for using a SVM model, with the prediction accuracy being 50.1% accuracy and the training time at just 3.97 seconds.

Interestingly, the precision and recall values, as well as the confusion matrices (Figures 1, 2, 3 and 4), show that the SVM models struggle the most with the proper classification of the bird and cat classes. It is shown the most in the LDA + SVM example, where the correct classification of the bird class is the worst amongst other SVM models, having a precision value of 0.14 and a recall value of 0.13. We can also observe an interesting occurrence of the LDA + SVM model misclassifying a lot of the vehicular classes, such as the airplane, automobile, ship and truck ones.

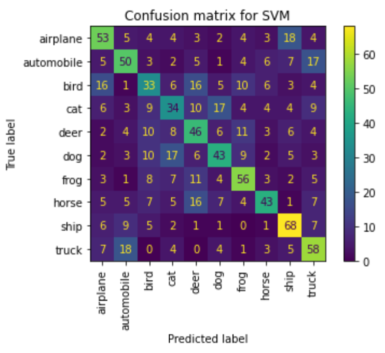

Figure 1. SVM confusion matrix.

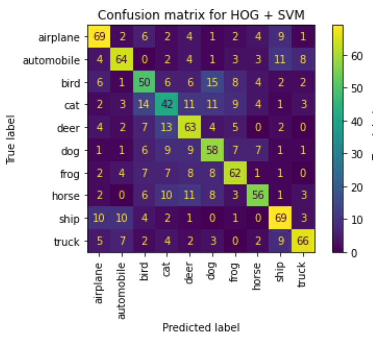

Figure 2. HOG + SVM confusion matrix.

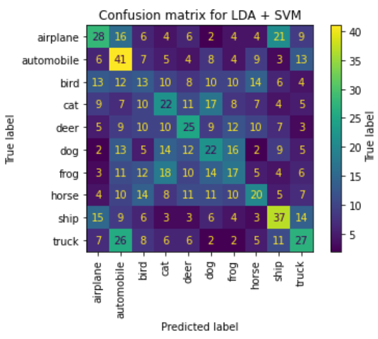

Figure 3. LDA + SVM confusion matrix.

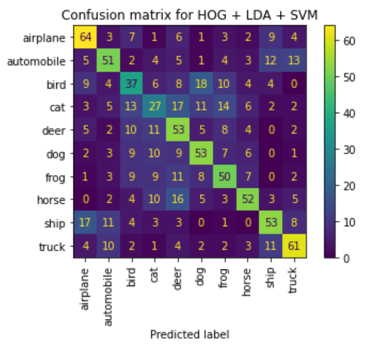

Figure 4. HOG + LDA + SVM confusion matrix.

Moving onto ANN models, when testing different optimizers, it was found that the optimizer using the Adam algorithm performed best, with a learning rate value of 0.001.

When feeding the input from the dataset directly into an ANN model, we can observe similar behaviour as before with the SVM model. The training time is again the worst among the other ANN models, being 237.73 seconds, although achieving a prediction accuracy of 41.5%. However, its validation loss is quite high at 6.19, meaning the model might be unsure in its predictions, having high prediction possibility percentages for other classes. During the testing of the implementation of dropout layers in the ANN models, it was interesting to note that using a dropout layer in this particular model made the model unable to fit, with both the validation and training accuracy never changing. It could hint at the model requiring the entire image data to converge further. The usage of extracted features via the use of HOG increased the prediction accuracy to 52.5%, dropping the validation loss to 4.92 and bringing the training time to 96.56 seconds. While the use of LDA + ANN has dropped the validation loss to 3.03 and training time to 91.48, it has reduced the prediction accuracy to 24.5%. Using the final composition of HOG + LDA + ANN has returned a prediction accuracy of 42.7% and a validation loss of 1.74, with a similar training time to LDA + ANN, taking 91.98 seconds.

Inspecting Figures 5, 6, 7 and 8 shows us similar issues for the ANN and HOG + ANN combinations to the ones encountered with the use of SVM models, with the bird and cat classes being misclassified the most, and the continued common misclassification of vehicular classes. However, the situation worsens in the LDA + ANN and HOG + LDA + ANN models. The LDA + ANN model has the worst recall values for the frog and horse classes, being 0.17 and 0.18. The HOG + LDA + ANN model reaches the worst correct classification values for the cat and deer classes, having only 7 and 9 correct classifications, with very low precision values of 0.3 and 0.24 for those classes. While the frog class in this model has 46 correct classifications, its precision value is just 0.28.

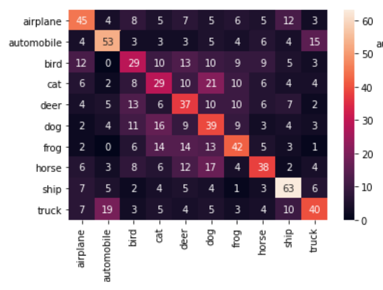
Figure 5. ANN confusion matrix.

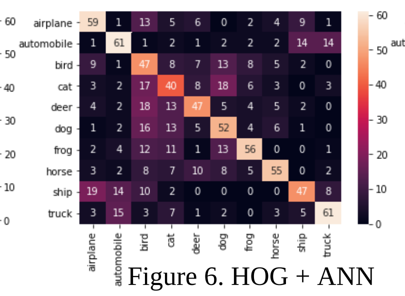
Figure 6. HOG + ANN confusion matrix.

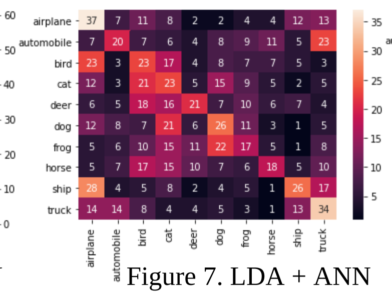
Figure 7. LDA + ANN confusion matrix.

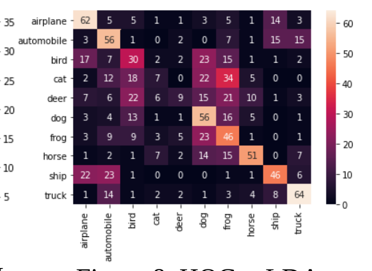
Figure 8. HOG + LDA + ANN confusion matrix.

Continuing to the CNN model, similarly to the case of the ANN models, the Adam optimizer performed best for the CNN model, with a learning rate value of 0.001 as well.

After training the CNN for 134.22 seconds, we achieve an exceptionally high prediction accuracy of 73.7%, with just 1.44 validation loss.

Investigating the confusion matrix shows us the lowest amount of misclassified classes yet, although we can see a major issue with the cat class (0.73 precision value) being heavily misclassified as the dog class label. We can also observe the reoccurrence of the misclassification of vehicular classes, with airplane and automobile class labels being wrongly classified the most.

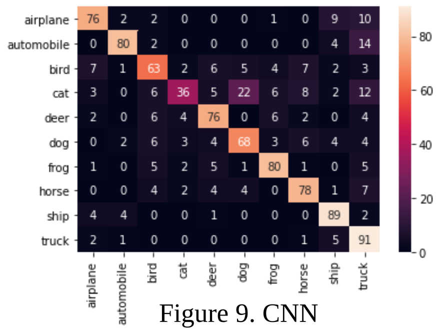

Figure 9. CNN confusion matrix.

Conclusion

After testing the following models: SVM, HOG into SVM, LDA into SVM, HOG into LDA into SVM, ANN, HOG into ANN, LDA into ANN, HOG into LDA into ANN, and lastly, CNN, the results found, as expected, the CNN model to perform best, having a prediction accuracy of 73.7%, with the HOG + SVM model following at 59.9%, and the HOG + ANN model at 52.5%. With all the three best models using feature extraction methods, either by input data pre-processing or internally, it shows how important it is to consider the representation of our data in machine learning. Considering the trade-off between training speed and prediction accuracy, the HOG + LDA + SVM model performed best, at an accuracy of 50.1% and training time of 3.97 seconds. All of the methods mentioned above in this conclusion beat the given classification accuracy benchmark of 44.68%.

Although the dimensionality reduction had a negative impact on the prediction accuracy in both the HOG + LDA + SVM and HOG + LDA + ANN models, it did help significantly reduce the fitting time in the case of using SVM, lowering it down to 1.89 seconds. Since the HOG + LDA + ANN model had a training time similar to the HOG + ANN model, it could be possible that the model had an overly deep construction, disallowing it from reducing the training time further, which is a potential drawback of this implementation, however it did have the lowest validation loss of all ANN models, being at 1.74.

This project also lacked investigating how the ANN models would perform with a reduction in hidden layers or removing them altogether, which could potentially reduce the discovered overfitting of those models. The amount of epochs has not been properly evaluated as well, being set to 100 for each ANN and CNN model after brief testing. As another shortcoming, the usage of LDA classification should have been tested as well to see how it performs on its own with the dataset. While the Principal Component Analysis (PCA) has been discarded for the use of LDA, it could as well be evaluated in the test to see whether it would outperform LDA in the combined use with other models. PCA and LDA could also be used together due to their different techniques of dimensionality reduction.

As a potential improvement for future work, due to the small amount of images in the subset of the CIFAR-10 dataset, image data augmentation could be used, where by using multiple image transformation techniques we are able to increase the amount of images for the CNN model [14]. This could essentially improve the accuracy of the model even higher.

References

[1] A. Singh, "Feature Engineering for Images: A Valuable Introduction to the HOG Feature Descriptor", *Analytics Vidhya*, 2019. [Online]. Available: https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/. [Accessed: 17 Dec 2020]

[2] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection", *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2005 [Online]. Available: https://ieeexplore.ieee.org/document/1467360. [Accessed: 17 Dec 2020]

[3] M. Aslan, A. Durdu, K. Sabanci and M. Mutluer, "CNN and HOG based comparison study for complete occlusion handling in human tracking", *Measurement*, vol. 158, p. 107704, 2020 [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0263224120302426. [Accessed: 17 Dec 2020]

[4] F. Kuo and I. Sloan, "Lifting the Curse of Dimensionality", *Notices of the AMS*, vol. 52, pp. 1320-1329, 2005 [Online]. Available: https://www.ams.org/notices/200511/fea-sloan.pdf. [Accessed: 17 Dec 2020]

[5] C. Versloot, "What is Dropout? Reduce overfitting in your neural networks – MachineCurve", *MachineCurve*, 2019. [Online]. Available: https://www.machinecurve.com/index.php/2019/12/16/what-is-dropout-reduce-overfitting-in-your-neural-networks/. [Accessed: 17 Dec 2020]

[6] A. Agarap, "Deep Learning using Rectified Linear Units (ReLU)", *arXiv.org*, 2019. [Online]. Available: https://arxiv.org/abs/1803.08375. [Accessed: 17 Dec 2020]

[7] S. Saha, "A Comprehensive Guide to Convolutional Neural Networks—the ELI5 way", *Medium*, 2018. [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. [Accessed: 17 Dec 2020]

[8] M. Sofian, "Is it possible to use Deep Learning (RCNN etc.) with HOG descriptors?", *ResearchGate.net*, 2016. [Online]. Available: https://www.researchgate.net/post/Is-it-possible-to-use-Deep-Learning-RCNN-etc-with-HOG-descriptors. [Accessed: 17 Dec 2020]

[9] S. Lee, M. Bang, K. Jung and K. Yi, "An efficient selection of HOG feature for SVM classification of vehicle", *2015 International Symposium on Consumer Electronics (ISCE)*, 2015 [Online]. Available: https://ieeexplore.ieee.org/document/7177766. [Accessed: 17 Dec 2020]

[10] Y. Pang, Y. Yuan, X. Li and J. Pan, "Efficient HOG human detection", *Signal Processing*, vol. 91, no. 4, pp. 773-781, 2011 [Online]. Available: https://doi.org/10.1016/j.sigpro.2010.08.010. [Accessed: 17 Dec 2020]

[11] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", *arXiv.org*, 2019. [Online]. Available: https://arxiv.org/abs/1409.1556. [Accessed: 17 Dec 2020]

[12] B. Ayinde, T. Inanc and J. Zurada, "On Correlation of Features Extracted by Deep Neural Networks", *arXiv.org*, 2015. [Online]. Available: https://arxiv.org/abs/1901.10900. [Accessed: 17 Dec 2020]

[13] H. Jansma, "Don't Use Dropout in Convolutional Networks - KDnuggets", *KDnuggets*, 2018. [Online]. Available: https://www.kdnuggets.com/2018/09/dropout-convolutional-networks.html. [Accessed: 17 Dec 2020]

[14] C. Shorten and T. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning", *Journal of Big Data*, vol. 6, no. 1, 2019 [Online]. Available: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0197-0. [Accessed: 17 Dec 2020]