GTSR-Net: A Deep Neural Network for German Traffic Sign Recognition

Abstract

A deep neural network (DNN) architecture was implemented for a traffic sign classification task using the German Traffic Sign Recognition (GTSRB) dataset. Using a network that implemented features of a U-Net architecture (1), the best performing model achieved an impressive accuracy of 98.16% on the test dataset. A class confusion matrix was used to evaluate the model and show the relative distribution of classifications for each class. The project also explored the effects of class imbalance and limited data. Using oversampling and undersampling methods, experiments were designed and run to examine how test accuracy changes when class imbalance exists. Data augmentation was used to address low shot problem. The effect of class imbalance and limited data was studied by comparing overall test accuracy and comparing class confusion matrices. The results demonstrated that oversampling and undersampling methods can mitigate class imbalance and improve model performance.

1 Introduction

Traffic sign recognition is crucial for enhancing road safety and facilitating the efficient movement of traffic. Traffic sign recognition plays a vital role in enabling the development and deployment of autonomous vehicles, which rely on precise detection and interpretation of traffic signs to navigate safely on roads. The recognition of traffic signs constitutes identifying the targets and classifying into known categories. However, traffic signs visibility can vary depending on the weather conditions, distance, occlusions, illumination, etc. Hence, this poses a challenge for the field of machine learning to correctly identify and classify the targets with high accuracy.

Sign recognition is a multicategory classification problem with unbalanced class frequencies that requires well designed deep neural networks to address the challenge. The German Traffic Sign Recognition Benchmark (GTSRB) is a large dataset collected of more than 50,000 traffic sign images in 43 classes (2). The dataset has been used for training models to study their performace on the multicategory classification problem. The dataset captures the strong variations in visual appearance of signs due to illumination, weather conditions and other factors. Hence, the dataset can be used to study the performance of machine learning models on a multicategory classification task where humans achieve close to 100% correctness (2). Over the recent years, Convolutional Neural Networks (CNNs) (3) have revolutionized the field of deep learning for image recognition and classification tasks. Furthermore, autoencoder CNNs (1) have played a crucial role in image segmentation tasks. The availability of a large complex dataset and advanced neural network architectures can be used to address the traffic sign recognition problem.

This project aims to train a DNN for a multicategory traffic sign classification problem. The main contributions of this work are: (a) implementing a DNN based on a Convolutional Neural Network (CNN) that includes features of the encoder part of U-Net resulting in a high test accuracy (b) and using oversampling and undersampling methods to study the effect of class imbalance and limited data on classification performance.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

2 Related work

Previous work on traffic sign recognition systems use different approaches for the problem. Colour-based approaches are the most common and use different colour spaces for segmentation of the road image, such as RGB (4), HSV (5) and HIS (6). Other methods include shape-based approaches (7; 8) for traffic sign recognition and detection. These approaches thus rely on feature engineering that extracts features of colour nor shapes and does not depend on any prior knowledge of traffic signs. It is vital for the different approaches to work well to have a diverse dataset that contains variability in images resulting from weather and lighting conditions. The GTSRB dataset thus captures the challenge of classifying traffic sign samples of multiple categories.

A number of recent advances have improved the classification accuracy for traffic sign recognition task. Various approaches used include feature extraction, dimensionality reduction, and classification. While Support Vector Machines and Random Forests have been widely used, Convolutional Neural Networks have shown to achieve higher classification accuracies. For example, Cireşan et al. won the GTSRB contest with 99.46% accuracy using a committee of 25 CNNs with data augmentation and jittering (9). Sermanet and LeCun achieved an accuracy of 98.31% using a multi-scale CNN (10), while Jin et al. proposed a method that trained an ensemble of 20 CNNs with a hinge loss stochastic gradient descent method to achieve 99.65% accuracy (11). Yet, there is still room for improvement in reducing the memory, computation costs and inference speed. This can be achieved by modifying augmentation techniques and changing model architecture to include the essential features required for the task.

3 Part I

3.1 Part I - Methods

In this work, a DNN is used for classification of traffic sign images through a CNN. The network achieves high accuracy on test data by combining the two features of deep learning. In addition, a number of data pre-processing steps are used to reduce computational cost by reducing the number of parameters of the model.

3.1.1 Model architecture

The network is composed of a contracting CNN path like an encoder in U-Net (1). The first part consists of a typical architecture of a convolutional network. It consists of the repeated application of two 3x3 convolutions, each followed by a rectified linear unit (ReLU) and a 2×2 max pooling operation with stride 2 for downsampling. The Max-pooling layers (12) reduce the spatial size of the feature maps, by directly decreasing the amount of parameters along with computation costs. At each downsampling step the number of feature channels are increased. A fully connected layer connected to all activations in the previous layer is used to combine the outputs of the previous layer into a 1-dimensional feature vector. Finally, the last fully-connected layer of the network performs the classification task since it has one output neuron per class, followed by a logarithmic softmax activation function (Table 1).

The model described in Table 1 consists of a modified encoder from the U-Net architecture (1). The number of output channels go upto a maximum of 200 instead of 1024. Furthermore, the model only uses the encoder modules after which linear layers are attached to perform classification for a task consisting of 43 categories. The output is passed through a soft-max function and returned for computing accuracy of the model.

3.1.2 Training and dataset

The GTSRB dataset consists of samples that each belong to one of the 43 existing classes. The dataset was split into a training set that has $35,288$ images and a validation set that consists of $3,921$ images, which were used to measure the performance of the model during training (S1). Traffic sign samples are raw RGB images whose size varies from 15×15 to 250×250 pixels. During the pre-processing step, all the samples are down-sampled or up-sampled to 128×128 pixels, and each image is converted to grayscale.

Table 1: Model architecture for traffic sign recognition multicategory classification task

Layers	Channels_in	Channels_out	W x H	Activation	BatchNorm
Input	1	16	128 x 128	-	-
Conv_1	16	16	64 x 64	-	-
Conv_2	16	32	64 x 64	ReLU	True
Conv_3	32	32	32 x 32	ReLU	True
Conv_4	32	64	32 x 32	ReLU	True
Conv_5	64	64	16 x 16	ReLU	True
Conv_6	64	128	16 x 16	ReLU	True
Conv_7	128	128	8 x 8	ReLU	True
Conv_8	128	200	8 x 8	ReLU	True
Conv_9	200	200	8 x 8	ReLU	True
Dropout2d	-	-	-	-	-
Linear_1	12800	350	-	-	True
Dropout	-	-	-	-	-
Linear_2	350	43	-	Softmax	-

The model was implemented and trained in PyTorch for a fixed number of epochs (50) using a batch size of 100. For hyper-parameter tuning, several networks were trained for to find an adequate initial learning rate value that reaches model convergence. An initial learning rate of 0.001 was subsequently used for all models using Adam optimizer and a learning rate scheduler that reduces the learning rate on plateau by a factor of 0.5 after 5 consecutive epochs with minimum loss.

3.2 Part I - Results

The model was trained and evaluated during the 50 epochs using a loss and accuracy curve for training and validation set. Figure 1 shows that the validation loss drops immediately after 10 epochs and the accuracy reaches above 90% after a few ($\sim$ 5) epochs. Hence, the model converges soon and continues training at lower learning rates. After completion of training, the final model at the last epoch was used to perform inference on the test set, which demonstrated an accuracy of 98.16%.

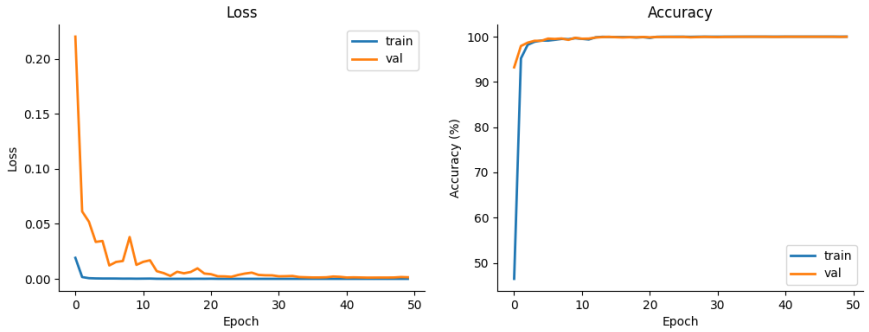


Figure 1: Loss and accuracy of GTSR-Net model during training for training and validation set.

The trained model was evaluated on a test set, which was obtained from the database and not used during training. The test set consisted of 12,630 images from all 43 classes. The final output of the trained model evaluated on test set was saved and used to obtain test accuracy and a confusion matrix shown in Fig. 2. The figure shows the relative distribution (in percentages) of classifications for each class in each cell. The diagonal of the class confusion matrix shows highest percentage of overlap between predicted and ground truch labels indicating correct classification of targets. Most of the classes had a high overlap of over 90% whereas a few classes were below 80%.

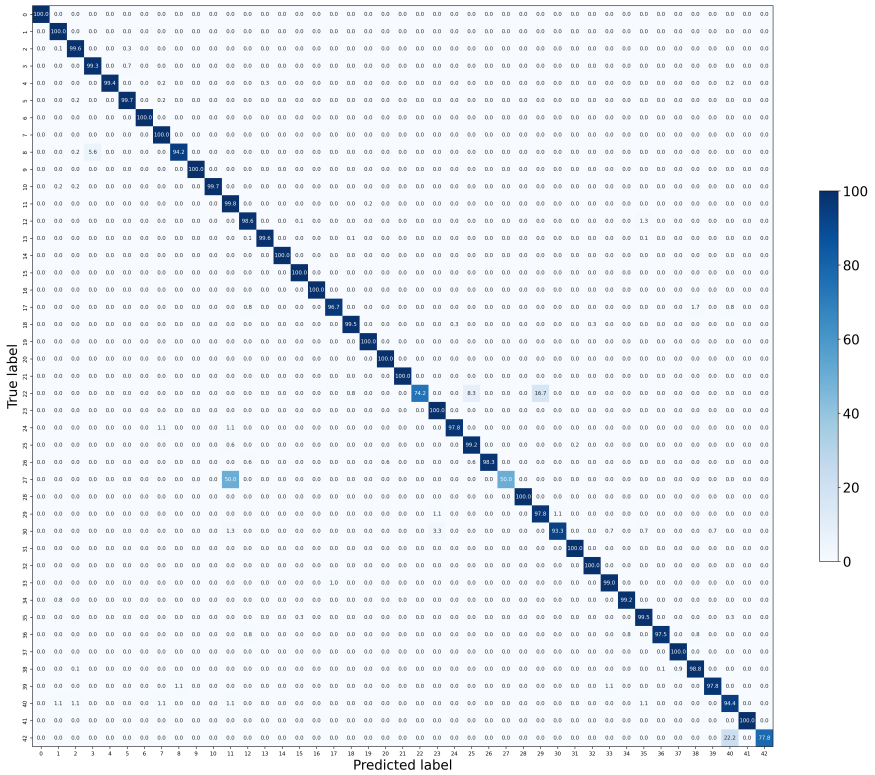


Figure 2: Class confusion matrix showing the relative distribution (in percentages) of classifications for each class in each cell. x-axis represents predicted label and y-axis represents ground truth label. Colorbar indicates the the relative percentage of overlap between ground truth and prediction.

3.3 Part I - Discussion

Most of the classes in the test set were correctly identified as shown in fig. 2. However, a small number of classes were predicted with lower accuracy. This could be due to a class imbalance problem known for convolutional neural networks (13). This can result from some classes being a minority compared to other classes that are overepresented in the training set as seen in S1. It has been established that class imbalance can have significant detrimental effect on training traditional classifiers (14). Class imbalance during training can affect both convergence during training phase and generalization of a model on the test set. Another limitation could be the lack of training data or insufficient data for a particular class resulting in lower performance. Nonetheless, the two limitations discussed can be addressed by modifying approaches during training to overcome limited data or class imbalance problem as discussed in the following section.

4 Part II

This section focuses on addressing: (a) class imbalance as seen by the number of examples in the training set for a class relative to other classes (b) low shot problem resulting from lack of sufficient training examples. The two main methods used to address the class imbalance problem include oversampling and undersampling. These methods have been shown to mitigate class imbalance (15). In order to address the low shot problem, i.e. lack of sufficient training examples, data augmentation techniques were used.

4.1 Part II - Methods

Oversampling. A number of methods are used in literature to address class imbalance though oversampling. Random minority oversampling is a basic method that replicates randomly selected

samples from minority classes. However, it can lead to overfitting. To address this, a more advanced method called SMOTE is used which creates artificial examples by interpolating neighboring data points to overcome overfitting (16). This project uses random minority oversampling as a starting point to study the effect of class imbalance on training and test accuracy.

Undersampling. Another method of overcoming class imbalance during training is through undersampling. This approach balances the number of examples in each class by randomly removing examples from majority classes until all classes have an equal number of examples. Undersampling can be preferred over oversampling in some situations. This project also investigtaed undersampling to study the effect of class imbalance. Finally, both oversampling and undersampling were used during training to investigate its effect.

Data augmentation. To address the problem of lack of sufficient training examples, data augmentation was used to expand the training set. This involved including methods such as random horizontal flip, random vertical flip, rotation and jitter to images. This expanded the training set by providing more diverse examples of traffic signs in different conditions.

4.2 Part II - Results

The results from the experiments investigating effects of class imbalance are shown in Table 2. The results show that both methods oversampling and undersampling improve the overall test accuracy by a small fraction. Oversampling alone led to most improvement which was comparable to combining oversampling and undersampling. Supplementary figures (S2, S3) show the effect of oversampling and undersampling respectively on the performance of model for the test classes. The class confusion matrices show an improvement in model training and generalization to test data as most (if not all) of the values along the diagonal of the matrix are above 80%. The low shot problem was investigated using data augmentation methods for the baseline model. The model trained with augmentation achieved an accuracy of 98.18%. The class confusion matrix and training and loss curve for the model is shown in S4 and S5.

Table 2: Addressing class imbalance during training through oversampling, undersampling and both (combined) approaches in comparison to baseline model which contains class imbalance. A baseline model trained with data augmentation was also investigated. Test accuracy is shown for the different approaches.

Method	Accuracy
Baseline	98.16 %
Oversampling	98.89 %
Undersampling	98.52 %
Both	98.74 %
Baseline + data augmentation	98.18 %

4.3 Part II - Discussion

The results show that the class imbalance can have a detrimental effect on classification performance, which can be mitigated by oversampling and undersampling. Oversampling should be applied to the level that eliminates the imbalance, whereas undersampling can perform better when the imbalance is only removed to some extent. Hence, further improvements during training could be adjusting the factor by which the samples are over- or under-sampled such that the imbalance is eliminated. In addition, other extensions of oversampling can be used which involve focusing only on examples near the class boundary and DataBoost-IM (17; 18). The aim of these approaches is to perform class-aware oversampling to ensure uniform class distribution of each mini-batch and control the selection of examples from each class. Undersampling did not provide much improvement which could be because it discards data resulting in fewer training examples. This can be addressed by carefully selecting examples for removal, such as one-sided selection, which identifies redundant examples near the class boundary. Nonetheless, the two methods mitigated class imbalance and led to some improvement in classification performance.

Data augmentation techniques to address the low shot problem did not improve the classification performance despite expanding the dataset almost five times. This could be due to methods used that

were already captured by the dataset, such as flipping and rotation. Some of the improvement in the model could be achieved by using color (RGB) images instead of grayscale for future work. This may help image augmentation techqniques, such as hue, contrast and saturation to further expand the training set. In addition, adding blurring techniques, such as motion and gaussian blur, could be further improvements for future work.

In order to address the low-shot problem, data augmentation techniques could be added during training to increase the number of training examples. Far away traffic signs can have low resolution while closer ones can be prone to motion blur. Hence, changes in traffic sign image due to illumination, motion and occlusions can be addressed by adding data augmentation. Other pre-processing or data augmentation techniques could be global normalisation and local contrast normalisation.

A number of other modifications could improve the overall test data classification accuracy. Transfer learning on a different task can be used to pre-train the encoder which could improve generalization of the model. In addition, spatial transformer networks (STNs) have been shown to improve performance of classification tasks (19). The localization module of the network can be added after the CNN layers and prior to the fully connected layers. The spatial network can improve recognition of local features and lead to better classification accuracy. However, this would increase the computational costs by increasing the number of parameters of the model for training.

In summary, this project implemented an encoder CNN for a multicategory traffic sign classification task. The model achieved an impressive test accuracy of 98.16%. In addition, other methods including oversampling and undersampling further improved the test accuracy by mitigating class imbalance in training dataset.

Code

The project codebase can be found on github: https://github.com/Atika-Syeda/DLCV_GTSRB.

References

[1] Ronneberger O, Fischer P, Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation; 2015.

[2] Stallkamp J, Schlipsing M, Salmen J, Igel C. The German traffic sign recognition benchmark: a multi-class classification competition. In: The 2011 international joint conference on neural networks. IEEE; 2011. p. 1453-60.

[3] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proceedings of the IEEE. 1998;86(11):2278-324.

[4] De La Escalera A, Moreno LE, Salichs MA, Armingol JM. Road traffic sign detection and classification. IEEE transactions on industrial electronics. 1997;44(6):848-59.

[5] Shadeed W, Abu-Al-Nadi DI, Mismar MJ. Road traffic sign detection in color images. In: 10th IEEE International Conference on Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003. vol. 2. IEEE; 2003. p. 890-3.

[6] Maldonado-Bascón S, Lafuente-Arroyo S, Gil-Jimenez P, Gómez-Moreno H, López-Ferreras F. Road-sign detection and recognition based on support vector machines. IEEE transactions on intelligent transportation systems. 2007;8(2):264-78.

[7] Loy G, Barnes N. Fast shape-based road sign detection for a driver assistance system. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566). vol. 1. IEEE; 2004. p. 70-5.

[8] Barnes N, Loy G, Shaw D. The regular polygon detector. Pattern Recognition. 2010;43(3):592-602.

[9] Cireşan D, Meier U, Masci J, Schmidhuber J. Multi-column deep neural network for traffic sign classification. Neural networks. 2012;32:333-8.

[10] Sermanet P, LeCun Y. Traffic sign recognition with multi-scale convolutional networks. In: The 2011 international joint conference on neural networks. IEEE; 2011. p. 2809-13.

[11] Jin J, Fu K, Zhang C. Traffic sign recognition with hinge loss trained convolutional neural networks. IEEE transactions on intelligent transportation systems. 2014;15(5):1991-2000.

[12] Scherer D, Müller A, Behnke S. Evaluation of pooling operations in convolutional architectures for object recognition. In: Artificial Neural Networks–ICANN 2010: 20th International Conference, Thessaloniki, Greece, September 15-18, 2010, Proceedings, Part III 20. Springer; 2010. p. 92-101.

[13] Buda M, Maki A, Mazurowski MA. A systematic study of the class imbalance problem in convolutional neural networks. Neural Networks. 2018 oct;106:249-59. Available from: https://doi.org/10.1016%2Fj.neunet.2018.07.011.

[14] Japkowicz N, Stephen S. The class imbalance problem: A systematic study. Intelligent data analysis. 2002;6(5):429-49.

[15] Buda M, Maki A, Mazurowski MA. A systematic study of the class imbalance problem in convolutional neural networks. Neural networks. 2018;106:249-59.

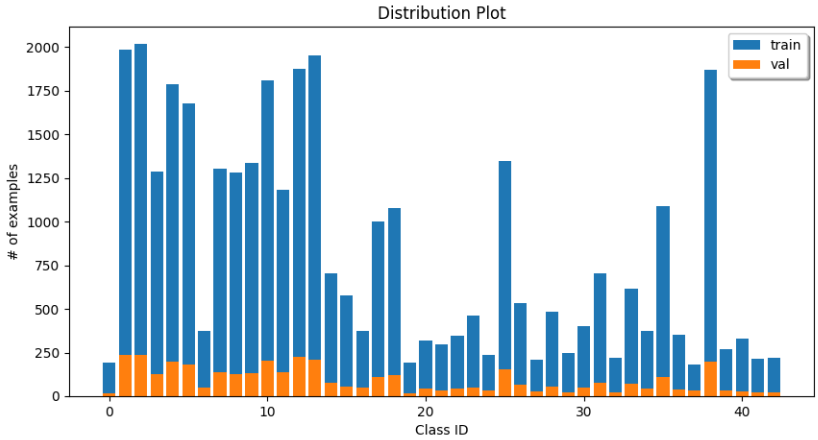
[16] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research. 2002;16:321-57.

[17] Han H, Wang WY, Mao BH. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In: Advances in Intelligent Computing: International Conference on Intelligent Computing, ICIC 2005, Hefei, China, August 23-26, 2005, Proceedings, Part I 1. Springer; 2005. p. 878-87.

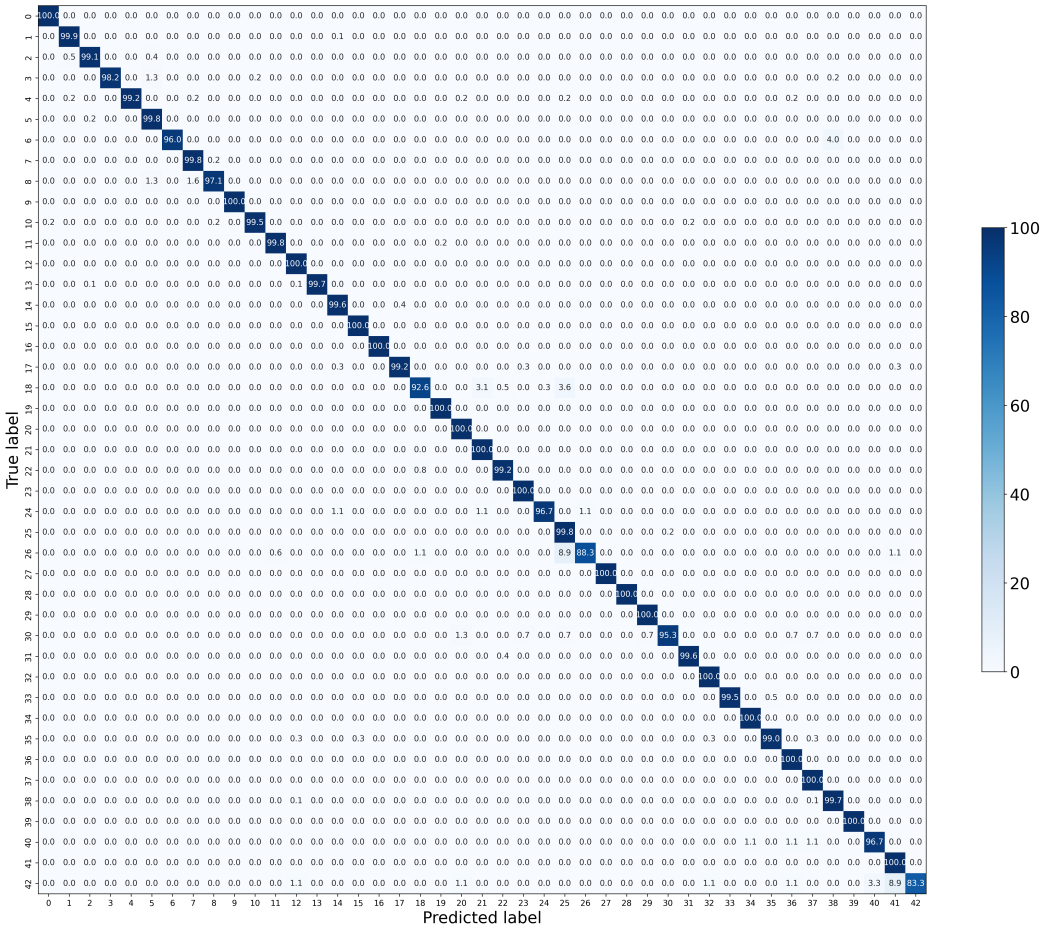
[18] Guo H, Viktor HL. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. ACM Sigkdd Explorations Newsletter. 2004;6(1):30-9.

[19] Jaderberg M, Simonyan K, Zisserman A, Kavukcuoglu K. Spatial Transformer Networks; 2016.

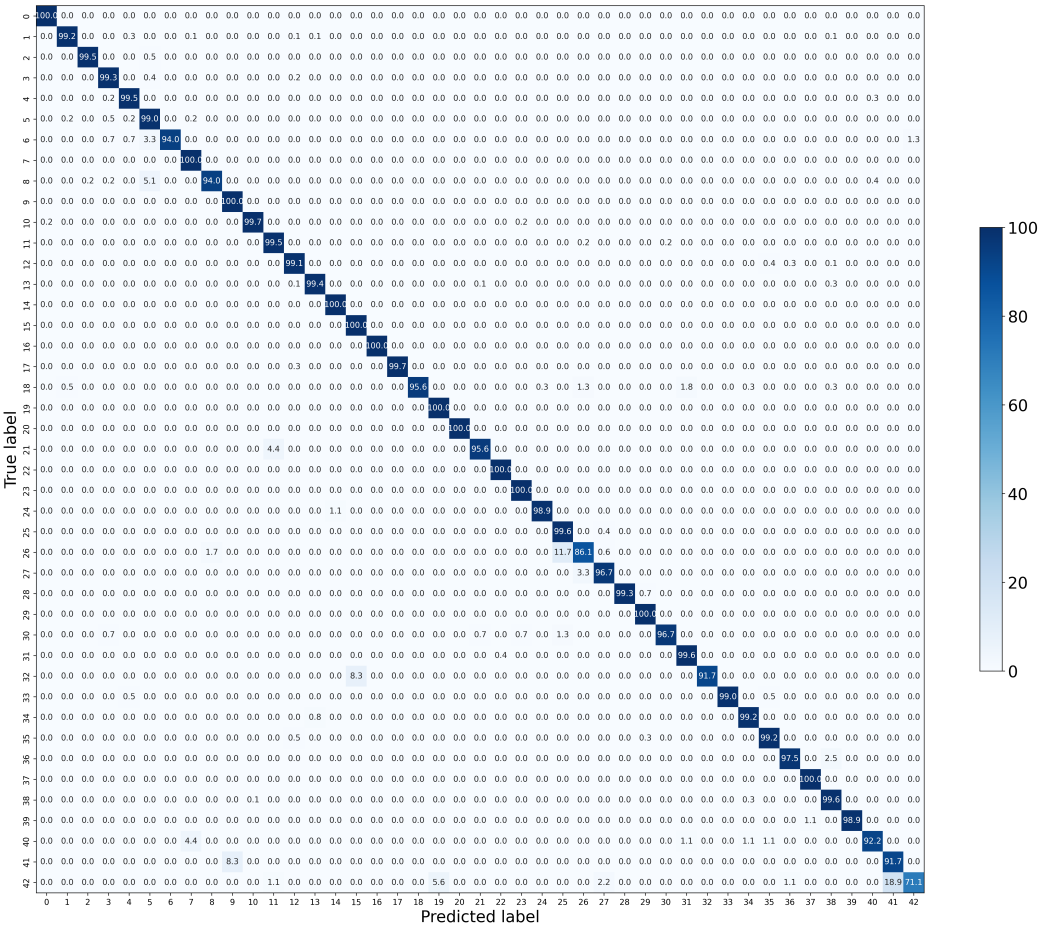
A Appendix



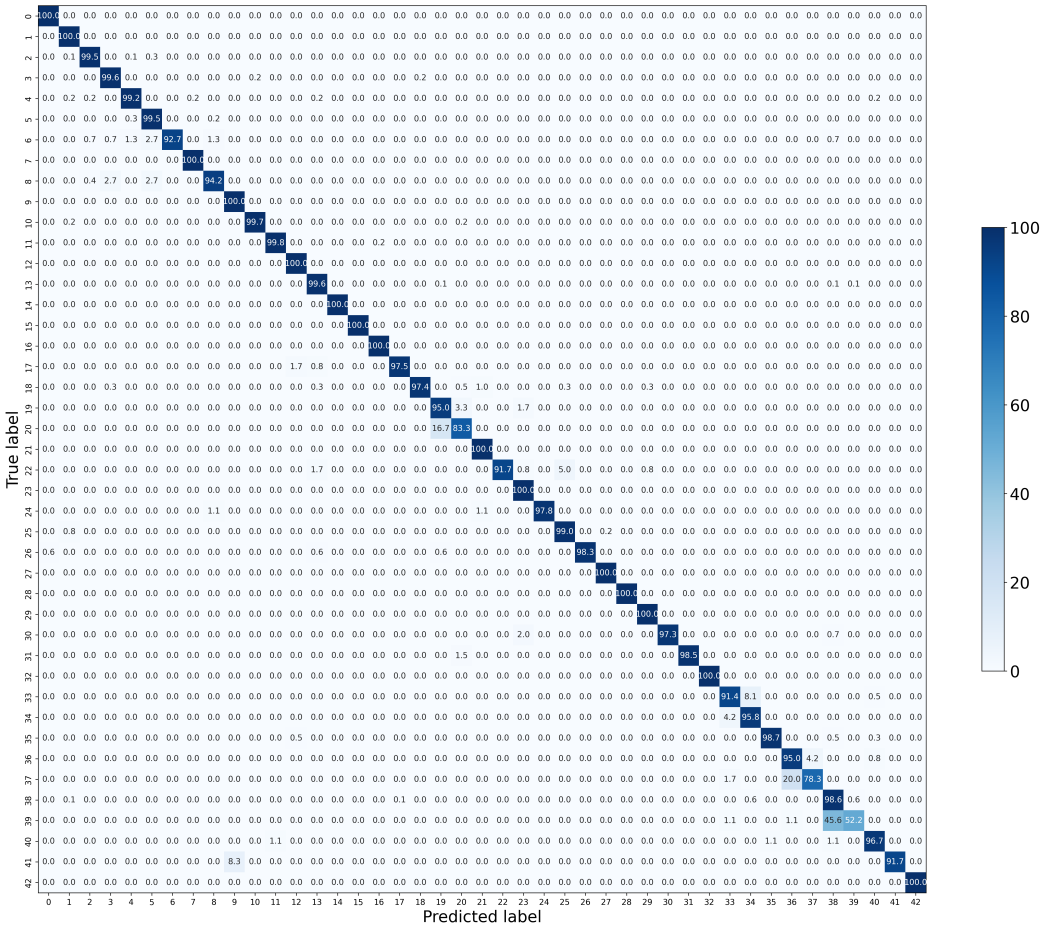
S1: Distribution of classes across training and validation dataset.



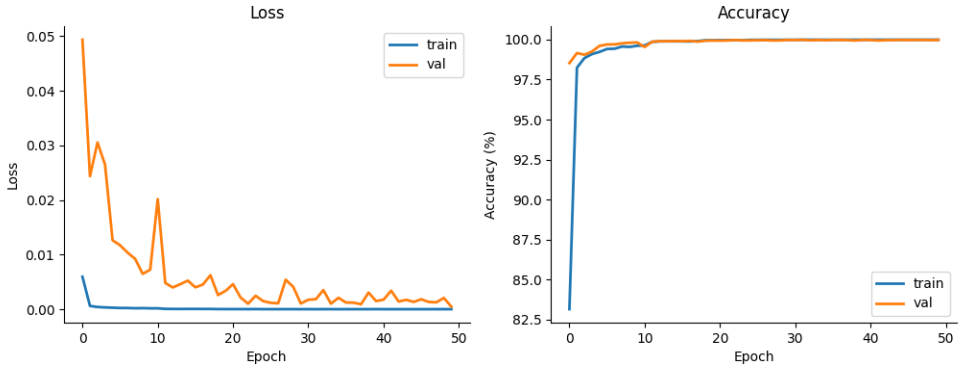
S2: Class confusion matrix for test data using oversampling during training.



S3: Class confusion matrix for test data using undersampling during training.



S4: Class confusion matrix for test data using baseline model and data augmentation during training.



S5: Loss and accuracy curve of baseline model trained with data augmentation.