

**School of Computer Sciences**

**Semester 2, Academic Year 2022/2023**

**CDS521:** **Multimodal Information Retrieval**

**Assignment 2**

**G6: Roundabout, intersection, crosswalk, flyover (overpass)**

|  |  |  |
| --- | --- | --- |
| **No** | **Name** | **Matric Number** |
| **1** | Al-Shammari Sura Abdulateef Hadi | P-COM0150/22 |
| **2** | Chasith Somsak | P-COM0102/21 |
| **3** | Deenesha Murugun | P-COM0017/23 |
| **4** | Gan Zhong Li | P-COM0156/21 |
| **5** | Looi Kah Fung | P-COM0049/22 |
| **6** | Wendy Tan Hway Shin | P-COM0296/21 |
| **7** | Ubaid Mohamed Dahir | P-COM0036/22 |
| **8** | Usman Salisu Nguru | P-COM0122/23 |

**Date of Submission: July 7th, 2023**

Classification of Road Features Using Transfer Learning and CNN

Gan Zhong Li1, Wendy Tan Hway Shin1, Al-Shammari Sura Abdulateef Hadi1, Chasith Somsak1, Looi Kah Fung1, Deenesha Murugun1, Ubaid Mohamed Dahir1, Usman Salisu Nguru1

1 *School of Computer Sciences, Universiti Sains Malaysia*

# Abstract

Efficient and accurate classification of road features, such as crosswalks, intersections, overpasses, and roundabouts, is crucial for enhancing road safety and optimizing traffic management. In this study, we propose a classification approach that utilizes the power of transfer learning and convolutional neural networks (CNNs) to address the road feature classification problem. By leveraging advancements in deep learning and employing state-of-the-art CNN architectures, the proposed system aims to achieve robust and real-time classification of road features. The dataset contained 7616 images of roundabout, crosswalk, overpass, and intersection from MLRSNet dataset and manually extracted satellite images from Malaysia using Google Earth Pro. We designed a CNN architecture that consists of 24 convolution layers and eight fully connected layers. Transfer learning models such as ResNet50, MobileNetV2, VGG19 and InceptionV3 were also explored for road feature classification. From the evaluation, the best performing model for road feature classification are ResNet50 and VGG-19 with accuracy of 98.7132%.

*Keyword: Road feature classification, Transfer learning, Convolutional Neural Networks (CNN)*

# Introduction

Effective classification of road features is essential for applications in transportation systems, such as autonomous driving, traffic management, and road infrastructure maintenance. For safe and effective navigation, it is crucial to correctly identify and classify road elements such as roundabouts, crosswalks, overpasses, and crossroads.

Road feature classification traditionally depends on labor-intensive, subjective, and error-prone manual examination and interpretation by human experts. Automated categorization systems, however, have emerged as a potential approach to handle this work effectively and precisely thanks to improvements in computer vision, machine learning, and deep learning approaches.

Various strategies have been investigated in recent years to automate the classification of road features. To analyze and categorize photographs of road features, these methods utilize computer vision techniques and machine learning models, particularly convolutional neural networks (CNNs). With the ability to extract pertinent information from input photos and produce precise predictions, CNNs have shown to be exceptionally effective at image recognition tasks. In this paper, we address the classification issue of these unique road features by developing a convolutional neural networks (CNNs) architecture and using transfer learning technique to search for the best performing model.

The successful implementation of this research will have significant practical implications. It can assist traffic management authorities in automating the monitoring and control of transportation facilities, enhancing road safety, and optimizing traffic flow. Additionally, the proposed system can serve as a foundation for developing intelligent transportation systems and smart city initiatives, contributing to the overall improvement of urban mobility.

# Literature Review

In recent years, there has been a lot of interest in the classification of road characteristics, including as roundabouts, crosswalks, overpasses, and intersections, utilizing transfer learning and convolutional neural networks (CNNs). Researchers have investigated a number of methods to improve the reliability and accuracy of classifying road features in the context of intelligent transportation systems.

Tümen et al. proposed deep learning and image processing techniques to detect intersection and crosswalk [1]. They designed a multi-scale CNN architecture called the RoIC-CNN that incorporated both convolutional and pooling layers to capture spatial information at different scales. RoIC-CNN consist of ten convolution layers and eight fully connected layers. In this study other CNN models such as VggNet-5, LeNet and AlexNet are also tested to compare the performance. From the evaluation, the best performing model in detecting crosswalks, and intersections is the RoIC-CNN.

Li et al. uses the target detection model (Faster-RCNN) as a foundation to establish the overpass labelling geodatabase (OLGDB) for the OpenStreetMap (OSM) road network data of six typical Chinese cities [2]. This technique applies convolutional neural networks (CNNs) to learn task-adaptive features from raster data, then a region proposal network (RPN) to pinpoint the site of a flyover. Three different CNNs (ZF-net, VGG-16, Inception-ResNet V2) are integrated into Faster-RCNN. Five geometric metrics (perimeter, area, squareness, circularity, and W/L) are synthesized into image bands to enhance the training data, and their contribution to the flyover identification task is determined. This step involves fine-tuning to find the best learning rate and batch size combination. The results of the experiments indicate that the proposed approach performs with good accuracy (about 90%).

Another active area of research aimed at leveraging artificial intelligence techniques to improve transportation infrastructure and operations, is the analysis of the Deep learning-based classification of transportation facilities for enhanced road safety and traffic management. In the study conducted by Jilani et al (2022), a five-layered convolutional neural network (CNN) deep learning model is proposed for the traffic congestion classification [3]. The traffic congestion dataset is enhanced through GAN-based augmentation. The study used pretrained RsNet50 and DenseNet-121 as the benchmark to compare with the 5-layer CNN. The study found that the proposed CNN emerged as the best model with accuracy of 98.63% compared to ResNet50 (90.59%) and DenseNet-121 (93.15%) respectively.

# Methodology

## Dataset and Data Pre-Processing

The dataset used in this study is primarily sourced from the MLRSNet dataset which is available from Mendeley Data [4]. The dataset contains 109,621 high spatial resolution optical images of 46 different categories captured from satellites. In this study, images of roundabout, Intersection and overpass were obtained from the MLRSNet dataset. 2,040, 2,498 and 2500 images of roundabout, intersection and overpass were collected from the MLRSNet dataset respectively. The images from MLRSNet dataset have a fixed size of 256x256 pixels.

In addition to that, additional satellite images of roundabout, Intersection, crosswalk, and overpass in Malaysia were obtained by our team by taking screenshots of those features using Google Earth Pro software. The raw images collected from Google Earth Pro were not of the same size and aspect ratio with the MLRSNet dataset. Thus, the images obtained from Google Earth Pro were cropped into perfect squares. This was realized through checking the aspect ratio of the images. If the image width and height are equal, no cropping action will be performed. If the width is greater than the height, the left and right sides of the image will be cropped, and a perfect square image is returned. If the width is less than the height, the top and bottom part of the image will be cropped and return a square image. The cropped images will then be resized to a resolution of 224×224. Similarly, images obtained from MLRSNet dataset were resized to 224x224 pixels as well. Figure 1 shows the sample images for roundabout, intersection, crosswalk, and overpass.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| Roundabout | Intersection | Crosswalk | Overpass |

Figure 1: Sample images for roundabout, intersection, crosswalk, and overpass.

Table 1: Number of images for roundabout, intersection, crosswalk and overpass

|  |  |  |  |
| --- | --- | --- | --- |
| **Images** | **MLRSNet Dataset** | **Google Earth Pro** | **Total** |
| Roundabout | 2040 | 100 | 2140 |
| Intersection | 2498 | 101 | 2599 |
| Crosswalk | Not available | 242 | 242 |
| Overpass | 2500 | 135 | 2635 |

Table 1 summarizes the images collected from MLRS Dataset and Google Earth Pro software and the total number of images for all the four separate categories. As could be observed from the table, the data is highly imbalanced, where the number of images of crosswalks are significantly lesser than the images of other categories, since the images of crosswalks are not available explicitly to be downloaded from the MLRSNet Dataset. Class imbalance is detrimental towards training on classifiers, and it affects the convergence of the deep learning model during training phase and generalization of the model during test phase [5], [6]. Good model results can be attained if all the classes in the classifier are properly represented [7]. Oversampling method was found to among the best method in alleviating class imbalance problem for CNN related model training [5].

In order to alleviate the class imbalance issue in this study, image augmentation, which is a form of oversampling method, was performed such that class balanced is achieved. Albumentations library from Python was used to perform image augmentation. In this data augmentation process, for each image class, a random image was selected, and random augmentation operations were performed. The following augmentation techniques with a 0.5 probability chance of execution, were performed.

* Random rotate- 90°
* Vertical flip
* Horizontal flip
* Random brightness contrast
* Random gamma

The augmented image would then be saved locally in storage, and this operation was repeated until a pre-defined goal of number of images were attained for each image category. Through data augmentation, the number of images for each category is increased to 2700 images respectively.

The images are then split to train, validation, and test set with a ratio of 7:2:1. Train set is the data used during the model training process, validation set is the data used to validate the model after epochs, while test set is the data used to evaluate the model after the training process. Stratification was performed during the split to ensure that the number of images for every class were the same for each batch of dataset.

## Proposed CNN Model

CNN is well-known in image-based classification tasks. CNNs process the input data using a number of interconnected layers. The first hidden layer of a CNN is usually a convolutional layer, which applies a set of filters to the input data to detect specific patterns [8]. These networks are built to automatically extract significant features at various degrees of abstraction from raw pixel input and learn hierarchical representations. Convolutional layers in CNNs allow for the extraction of regional patterns and structures, while pooling layers make it easier to down sample spatial data, which improves the model's capacity to identify important features. Based on the retrieved features, fully connected layers provide the final categorization.

Table 2: Parameter Values for CNN Model

|  |  |  |  |
| --- | --- | --- | --- |
| **Layer** | **Layer Name** | **Output Shape** | **Param#** |
| 1 | Conv2D | 224 x 224 x 32 | 896 |
| 2 | MaxPooling2D | 112 x 112 x 32 | 0 |
| 3 | Conv2D | 112 x 112 x 32 | 9248 |
| 4 | BatchNormalization | 112 x 112 x 32 | 128 |
| 5 | MaxPooling2D | 56 x 56 x 32 | 0 |
| 6 | Dropout | 56 x 56 x 32 | 0 |
| 7 | Conv2D | 56 x 56 x 64 | 18496 |
| 8 | BatchNormalization | 56 x 56 x 64 | 256 |
| 9 | Conv2D | 56 x 56 x 64 | 36928 |
| 10 | BatchNormalization | 56 x 56 x 64 | 256 |
| 11 | MaxPooling2D | 28 x 28 x 64 | 0 |
| 12 | Dropout | 28 x 28 x 64 | 0 |
| 13 | Conv2D | 28 x 28 x 128 | 73856 |
| 14 | BatchNormalization | 29 x 28 x 128 | 512 |
| 15 | Conv2D | 28 x 28 x 128 | 147584 |
| 16 | BatchNormalization | 29 x 28 x 128 | 512 |
| 17 | MaxPooling2D | 14 x 14 x 128 | 0 |
| 18 | Dropout | 14 x 14 x 128 | 0 |
| 19 | Conv2D | 14 x 14 x 256 | 295168 |
| 20 | BatchNormalization | 14 x 14 x 256 | 1024 |
| 21 | Conv2D | 14 x 14 x 256 | 590080 |
| 22 | BatchNormalization | 14 x 14 x 256 | 1024 |
| 23 | MaxPooling2D | 7 x 7 x 256 | 0 |
| 24 | Dropout | 7 x 7 x 256 | 0 |
| 25 | Flatten | 12544 | 0 |
| 26 | Dense | 256 | 3211520 |
| 27 | BatchNormalization | 256 | 1024 |
| 28 | Dropout | 256 | 0 |
| 29 | Dense | 32 | 8224 |
| 30 | BatchNormalization | 32 | 128 |
| 31 | Dropout | 32 | 0 |
| 32 | Dense | 4 | 132 |

A custom CNN model is created in this study to perform classification of roundabout, crosswalk, intersection, and overpass. The CNN model is created using TensorFlow. The CNN architecture that consists of 24 convolution layers and eight fully connected layers as listed in Table 2.

## Proposed Transfer Learning Models

In addition to the proposed CNN, transfer learning model such as ResNet50, MobileNetV2, VGG19 and InceptionV3 are used to compare the accuracy with our CNN model developed.

Transfer learning is a powerful technique in computer vision tasks which enables models to leverage knowledge learned from pre-trained models trained on large-scale datasets [9]. By transferring this knowledge to a new task, transfer learning can significantly enhance classification performance. In our approach, we harness the benefits of transfer learning by utilizing pre-trained models from the literature such as ResNet50, MobileNetV2, VGG19 and InceptionV3. In this study, a classification layers of the transfer learning models were dropped and replaced with a new classification layer similar to the proposed CNN model. The weights are initialized to the models’ weight trained on ImageNet. All the transfer learning layers were allowed to be trained in this study. Table 3 summarizes the common hyperparameters configured for the transfer learning models.

Table 3: Hyper-parameters used in the convolutional base model InceptionV3, ResNet50, VGG-10 and MobileNetV2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameter** | InceptionV3 | ResNet50 | VGG-19 | MobileNetV2 |
| **Input image** | (224,224,3) | (224,224,3) | (224,224,3) | (224,224,3) |
| **Weight** | Initialized to ImageNet | Initialized to ImageNet | Initialized to ImageNet | Initialized to ImageNet |
| **Optimizer** | Adam | Adam | Adam | Adam |
| **Loss function** | Sparse categorical cross entropy | Sparse categorical cross entropy | Sparse categorical cross entropy | Sparse categorical cross entropy |
| **Classifier** | Softmax | Softmax | Softmax | Softmax |
| **Epochs** | 50 | 50 | 50 | 20 |
| **Dropout rate** | 0.2 | 0.2 | 0.2 | 0.2 |

### **ResNet50**

ResNet50 is a 50-layer deep convolutional neural network design that uses residual connections to support the training of extremely deep networks. It has demonstrated impressive performance in image classification challenges, overcoming the vanishing gradient issue and facilitating simpler deep model optimization.

### **MobileNetV2**

A compact convolutional neural network architecture called MobileNetV2 was created for effective mobile and embedded vision applications. It is appropriate for devices with limited resources because it uses depth wise separable convolutions and inverted residual blocks to simplify computations while retaining competitive accuracy.

### **VGG19**

Convolutional neural network architecture VGG19 is renowned for its efficiency and simplicity. There are 19 layers total, including several 3×3 convolutional layers followed by max-pooling layers. Although VGG19 has more parameters than other architectures, it performs image classification tasks with a high degree of accuracy.

### **InceptionV3**

The deep convolutional neural network architecture known as InceptionV3 makes use of the idea of inception modules. These modules use parallel convolutional layers with various kernel sizes to capture features at various scales. InceptionV3 reduces the number of parameters using dimensionality reduction techniques to obtain high accuracy in image recognition tasks while retaining computing efficiency.

## Performance Metric

During the testing phase, the CNN models’ performance is evaluated. One of the most used measures for evaluating the performance of a Convolutional Neural Network (CNN) is accuracy. Accuracy is the ratio of accurately predicted images to all images. The formula for accuracy is:

Where, TP/true positive is when the model correctly predicts a sample as belonging to a specific class; TN/ true negative is when the model correctly predicts a sample as not belonging to a specific class; FP/False Positive is when the model incorrectly predicts a sample as belonging to a specific class and False Negative/FN is when the model incorrectly predicts a sample as not belonging to a specific class, but it actually belongs to that class.

## Hyperparameter Tuning

In order to attain the best performing model, hyperparameter tuning was performed in order to obtain the best performing model. In this study, two hyperparameters were tested, namely learning rate and batch size. Learning rate is a hyperparameter which dictates the extent of change a deep learning model in response to the estimated error each time when the weights of model are updated [10]. Batch size meanwhile refers to the number of samples processed before the model is updated [11]. Various studies have concluded that learning rate and batch size have significant impact on a neural network’s performance [12]–[14].

Learning rates of 0.01, 0.001 and 0.0001 were tested during hyperparameter tuning. As for batch size, batch sizes of 32 and 64 were tested. A full grid search was performed during the hyperparameter tuning process and thus, 6 experiments were performed for each of the image classification model during the hyperparameter tuning process.

The hyperparameter tuning process is realized using the Keras Tuner library in Python, and the models were allowed to be trained for 50 epochs. After hyperparameter tuning was performed for each image classification model, the model with the best validation accuracy was rebuilt, and the model was then tested with the test dataset to obtain the test accuracy.

Subsequently, the performance of the best performing model for each image classification model, whether it is the custom CNN model proposed here, or the transfer learning model will be evaluated, compared and discussed in subsequent section.

# Experiment Results

Figure 2 shows the training and validation loss of model during hyperparameter tuning process. In general, the training and validation loss reduces as epoch increases. However, some of the trials failed to converge and remain stationary across epochs, which are especially true for trial 1 and trial 2 of transfer learning models, where the learning rate for both were 0.01. While trial 5 and trial 6, which uses 0.0001 learning rate shows the lowest training and validation loss during the model training of transfer learning models, both the trials had the highest loss during the training of the proposed CNN networks.

|  |  |  |
| --- | --- | --- |
| **Model** | **Training Loss** | **Validation Loss** |
| Proposed CNN |  |  |
| InceptionV3 |  |  |
| ResNet50 |  |  |
| VGG-19 |  |  |
| MobileNetV2 |  |  |
|  | | |

Figure : Training and validation loss of models plotted on log scale during hyperparameter tuning process.

Table : Best validation accuracy of image classification models for different learning rate and batch size used during the hyperparameter tuning process

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **Learning Rate** | **Batch Size** | **Validation Accuracy (%)** | | | | |
| **Proposed CNN** | **InceptionV3** | **ResNet50** | **VGG-19** | **MobileNetV2** |
| 1 | 0.01 | 32 | 92.7974 | 25.2323 | 25.4647 | 25.6506 | 25.9294 |
| 2 | 0.01 | 64 | 93.9591 | 25.4647 | 32.0167 | 26.0223 | 91.5428 |
| 3 | 0.001 | 32 | **95.1208** | 97.4907 | 95.9572 | 90.0093 | 97.3048 |
| 4 | 0.001 | 64 | 94.6097 | 97.9554 | 96.0967 | 89.0799 | 97.2583 |
| 5 | 0.0001 | 32 | 91.7286 | 98.7918 | 98.5130 | 97.7695 | 97.3048 |
| 6 | 0.0001 | 64 | 92.1468 | **98.9777** | **98.6524** | **97.9082** | **98.3271** |

Table 4 summarizes the proposed CNN and transfer learning models’ best validation accuracy for different learning rate and batch size used during the hyperparameter tuning process. In general, in this study, larger batch size of 64 would result in better validation accuracy as compared to a lower batch size of 32. As for the effects of learning rate, for transfer learning models, smaller learning rates results in better validation accuracy. However, this trend is not observed in the proposed CNN model, where the highest validation accuracy is observed when the learning rate is 0.001 instead. It is worth noting that for all the transfer learning models trained with learning rate of 0.01, except for MobileNetV2 model trained with batch size of 64, the accuracies of the models were very low at less than 40%. This was due to the models failed to converge during, as evident in the training and validation loss curves of those models shown in Figure 2 which does not decrease as the training epochs increases.

Table : Validation accuracy and test accuracy of image classification models with their most optimal hyperparameters.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Hyperparameters** | **Validation Accuracy (%)** | **Test Accuracy (%)** |
| Proposed CNN | Learning rate: 0.001, Batch size: 32 | 95.1208 | 94.4852 |
| InceptionV3 | Learning rate: 0.0001, Batch size: 64 | **98.9777** | 97.7022 |
| ResNet50 | Learning rate: 0.0001, Batch size: 64 | 98.6524 | **98.7132** |
| VGG-19 | Learning rate: 0.0001, Batch size: 64 | 97.9082 | **98.7132** |
| MobileNetV2 | Learning rate: 0.0001, Batch size: 64 | 98.3271 | 98.0698 |

The highest validation accuracies for each model in the hyperparameter tuning process were bolded in Table 4 and accuracy values, together with its corresponding hyperparameters were summarized in Table 5. From the hyperparameter tuning process, the best performing proposed CNN model which was learning rate of 0.001 and batch size of 32 had validation accuracy of 95.1208%. As for other transfer learning models, highest validation accuracies were attained with learning rate of 0.0001 and batch size of 64. The validation accuracies were 98.9777%, 98.6524%, 97.9082% and 98.3271% for InceptionV3, ResNet50, VGG-19 and MobileNetV2 respectively. Based on this validation accuracy score, InceptionV3 was found to provide the best validation accuracy, followed by ResNet50, MobileNetV2, VGG-19 and the proposed CNN network.

In addition, using the best performing models, the models are tested with the test dataset to obtain the test accuracy. In general, the test accuracy is comparable to the validation accuracy. However, the sequence in terms of the test accuracy performance is not the same. Both ResNet50 and VGG-19 had test accuracy of 98.7132%, followed by MobileNetV2 with 98.0698% test accuracy, InceptionV3 with 97.7022% test accuracy. The proposed CNN model had the lowest test accuracy compared to the other models at 94.4852%.

The reason to why the proposed CNN model is not performing as well as other transfer learning models could be attributed to insufficient model training. As could be seen from the loss curves of the CNN model, stationarity had yet to be obtained by the end of the 50 epochs. This is different from the transfer learning models, where stationarity is observed towards the end of the 50 epochs. Thus, it could be said that the proposed CNN model had yet to attain convergence, and thus resulting in the lower accuracy of the model. In addition to that, the transfer learning models used here were created by industry experts in CNN and initialized with pre-trained weights which have been optimized with the training of ImageNet dataset. Thus, convergence can be attained earlier with the transfer learning models trained with the new satellite images. The proposed CNN model can be trained with more epochs until convergence is attained, and perform hyperparameter tuning with more hyperparameters as part of the future work to attain better performance with the model.

# Conclusion

In this paper, we used transfer learning and convolutional neural networks (CNNs) to present a unique method for the classification of road feature. Transfer learning models ResNet50, MobileNetV2, VGG19 and InceptionV3 are used in this problem and the accuracies are compared against the proposed CNN model. The effects of different hyper-parameters such as batch-size and learning rate are interrogated in this work. The results proved that the proposed CNN model and the transfer learning models which had undergone hyperparameter tuning can classify roundabout, crosswalk, intersection, and overpass with relatively high accuracy. The best performing model during validation phase is InceptionV3 with accuracy of 98.9777%, whereas the best performing model during test phase are ResNet50 and VGG-19 models with accuracy of 98.7132%. The proposed CNN model got 95.1208% and 94.4852% accuracy during validation and test stage.

The CNN architecture proved to be well-suited for road feature classification tasks, capturing spatial dependencies and extracting discriminative features from road images. Our model achieved high accuracy, robustness, and efficiency, outperforming existing methods in terms of classification performance.

The implications of our research extend to various domains within intelligent transportation systems, including autonomous driving, traffic management, and road infrastructure maintenance. Accurate classification of road features enables safer navigation, improved traffic flow, and effective decision-making in transportation planning and management.

Future research can explore additional road feature categories and expand the classification system to handle real-time scenarios. Further investigations can also focus on optimizing the model architecture, refining transfer learning strategies, and incorporating contextual information for more comprehensive road analysis.

In conclusion, our study demonstrates the effectiveness of transfer learning and CNNs for the classification of road features. The proposed approach offers a practical and efficient solution for accurate identification and categorization of road features, contributing to the advancement of intelligent transportation systems and enhancing overall road safety and efficiency.

# References

[1] V. Tümen and B. Ergen, “Intersections and crosswalk detection using deep learning and image processing techniques,” *Physica A: Statistical Mechanics and its Applications*, vol. 543, p. 123510, Apr. 2020, doi: 10.1016/j.physa.2019.123510.

[2] H. Li, M. Hu, and Y. Huang, “Automatic Identification of Overpass Structures: A Method of Deep Learning,” *ISPRS Int J Geoinf*, vol. 8, no. 9, p. 421, Sep. 2019, doi: 10.3390/ijgi8090421.

[3] U. Jilani, M. Asif, M. Rashid, A. A. Siddique, S. M. U. Talha, and M. Aamir, “Traffic Congestion Classification Using GAN-Based Synthetic Data Augmentation and a Novel 5-Layer Convolutional Neural Network Model,” *Electronics (Basel)*, vol. 11, no. 15, p. 2290, Jul. 2022, doi: 10.3390/electronics11152290.

[4] X. Qi *et al.*, “MLRSNet: A Multi-label High Spatial Resolution Remote Sensing Dataset for Semantic Scene Understanding,” *Mendeley Data*. May 19, 2023.

[5] M. Buda, A. Maki, and M. A. Mazurowski, “A systematic study of the class imbalance problem in convolutional neural networks,” *Neural Networks*, vol. 106, pp. 249–259, Oct. 2018, doi: 10.1016/j.neunet.2018.07.011.

[6] D. Dablain, K. N. Jacobson, C. Bellinger, M. Roberts, and N. Chawla, “Understanding CNN Fragility When Learning With Imbalanced Data,” Oct. 2022, [Online]. Available: http://arxiv.org/abs/2210.09465

[7] J. M. Johnson and T. M. Khoshgoftaar, “Survey on deep learning with class imbalance,” *J Big Data*, vol. 6, no. 1, p. 27, Dec. 2019, doi: 10.1186/s40537-019-0192-5.

[8] “Image Classification Using CNN: Introduction and Tutorial,” *Datagen*. https://datagen.tech/guides/image-classification/image-classification-using-cnn/ (accessed Jul. 01, 2023).

[9] P. Sharma, “Understanding Transfer Learning for Deep Learning,” *Analytics Vidhya*, Oct. 30, 2023. https://www.analyticsvidhya.com/blog/2021/10/understanding-transfer-learning-for-deep-learning/ (accessed Jul. 01, 2023).

[10] J. Brownlee, “Understand the Impact of Learning Rate on Neural Network Performance,” *Machine Learning Mastery*, Jan. 25, 2020. https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/ (accessed Jul. 01, 2023).

[11] J. Brownlee, “Difference Between a Batch and an Epoch in a Neural Network,” *Machine Learning Mastery*, Aug. 15, 2022. https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/#:~:text=The%20batch%20size%20is%20a%20number%20of%20samples%20processed%20before,samples%20in%20the%20training%20dataset. (accessed Jul. 01, 2023).

[12] G. Bruno, D. Antonelli, and D. Stadnicka, “Evaluating the effect of learning rate, batch size and assignment strategies on the production performance,” *Journal of Industrial and Production Engineering*, vol. 38, no. 2, pp. 137–147, Feb. 2021, doi: 10.1080/21681015.2021.1883133.

[13] S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le, “Don’t Decay the Learning Rate, Increase the Batch Size,” Nov. 2017.

[14] I. Kandel and M. Castelli, “The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset,” *ICT Express*, vol. 6, no. 4, pp. 312–315, Dec. 2020, doi: 10.1016/j.icte.2020.04.010.

# Appendix

The codes and log files for the paper can be found in the following GitHub repository.

<https://github.com/zl-gan/CDS521_Assignment2>