Option #1: AI Use - Case Problem with Solution - Paper

Scott Miner

Colorado State University – Global Campus

Abstract

```
Flower_photos
├───evaluation
│   ├───daisy
│   ├───dandelion
│   ├───roses
│   ├───sunflowers
│   └───tulips
├───training
│   ├───daisy
│   ├───dandelion
│   ├───roses
│   ├───sunflowers
│   └───tulips
└───validation
    ├───daisy
    ├───dandelion
    ├───roses
    ├───sunflowers
    └───tulips
```

Figure 1. The folder structure of flower photos after running the p_build_dataset.py script

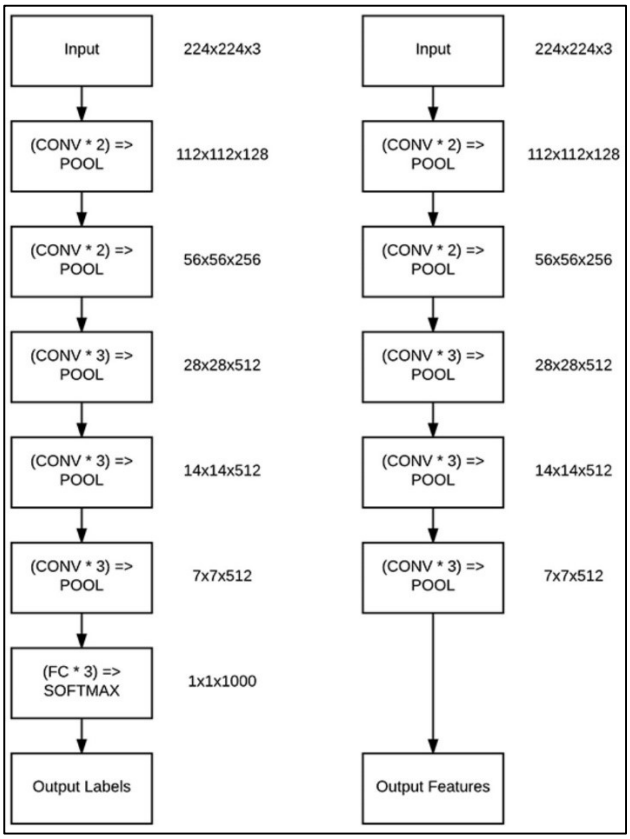

Figure 2. Transfer learning via feature extraction. On the left is the original VGG16 network architecture that outputs probabilities for each of the 1,000 ImageNet class labels. The image on the right shows the removal of the FC layer, which returns the final pool layer to the output and serves as the extracted image features (Rosebrock, 2019)

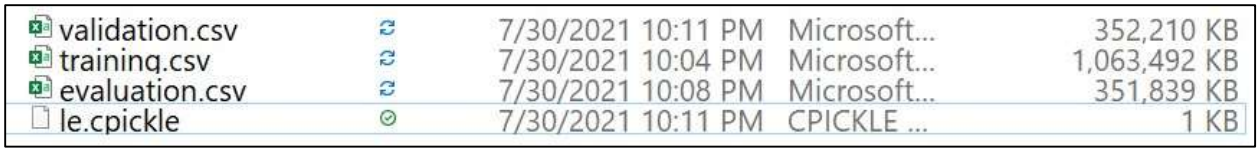

Figure 3. CSV files containing feature vectors for each data split and the label encoder for the target variable

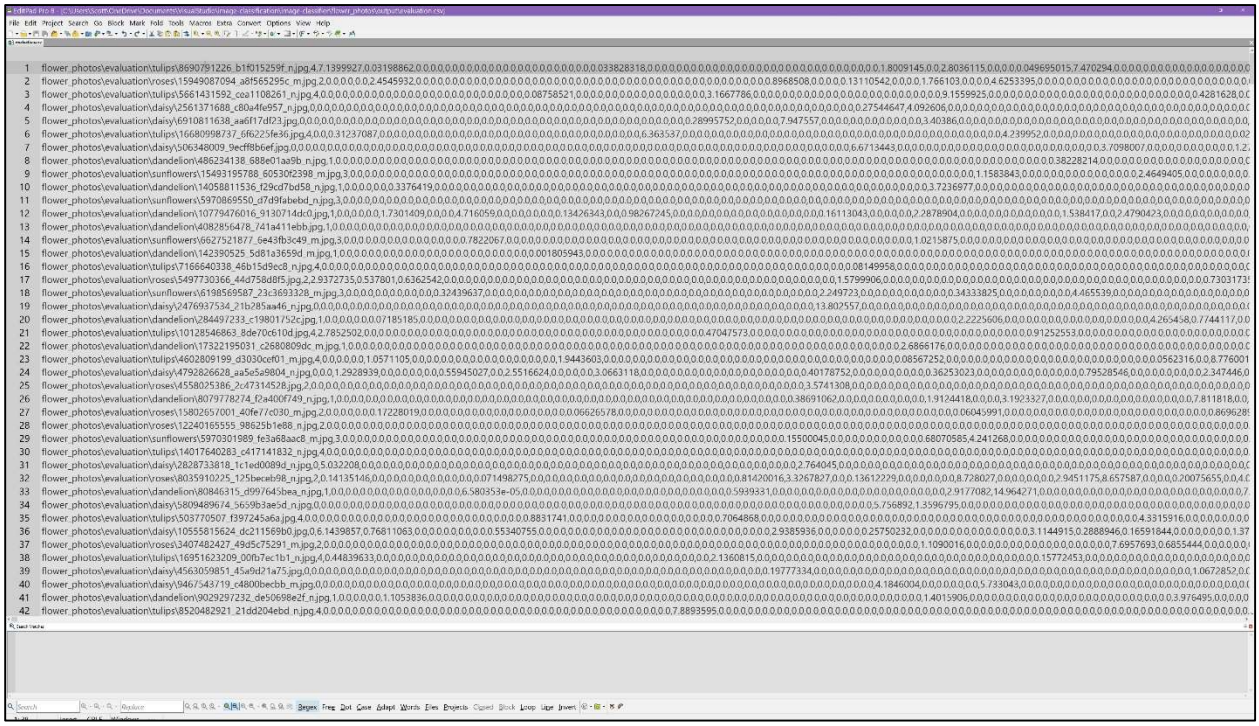

Figure 4. A CSV file showing the extracted feature vectors for the evaluation dataset

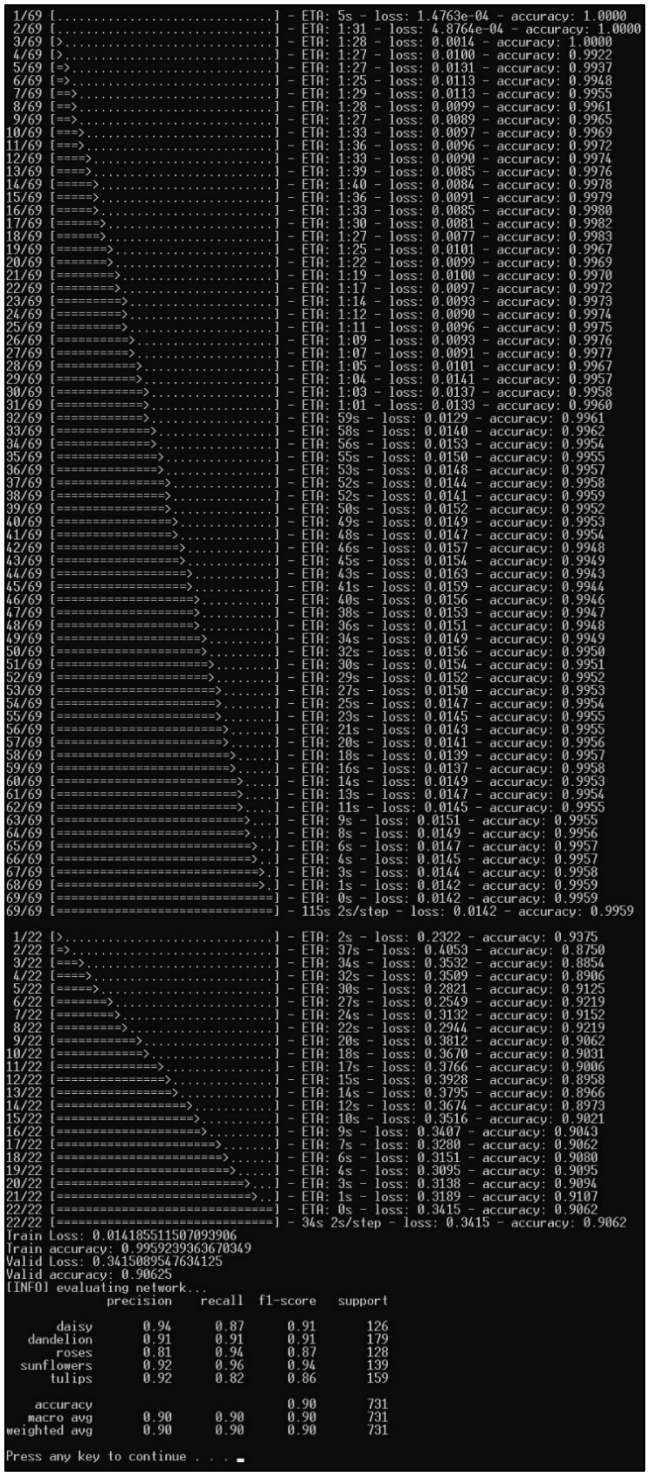

Figure 5. Accuracies for the training and validation datasets after training the NN for 5 epochs along with the classification report

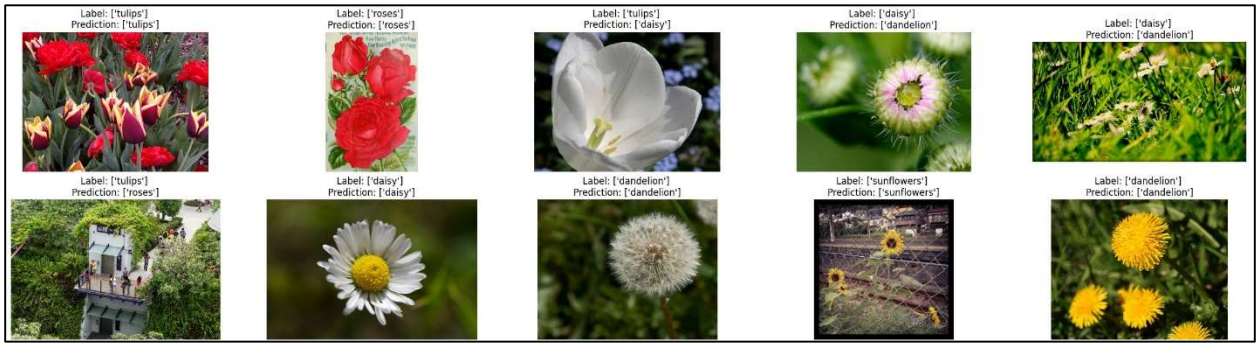

Figure 6. A random sample of 25 images, labels, and model predictions from the evaluation dataset (part 1)

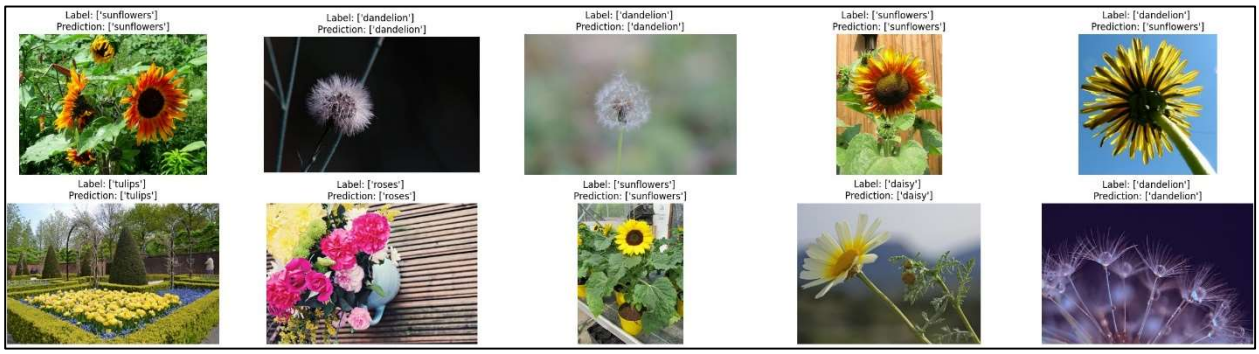

Figure 7. A random sample of 25 images, labels, and model predictions from the evaluation dataset (part 2)

Figure 8. A random sample of 25 images, labels, and model predictions from the evaluation dataset (part 3)

Table 1. Modules and corresponding topics in CSC510: Foundations of Artificial Intelligence

Module 1	AI Safety
Module 2	Foundational Theories and Their Application
Module 3	Neural Networks
Module 4	Intelligent Searches
Module 5	Cognitive Systems
Module 6	Classification
Module 7	First-Order Logic
Module 8	Symbolic Planning

Option #1: AI Use - Case Problem with Solution – Paper

This paper outlines a fully functioning AI program to solve a real-world problem, utilizing at least two modules from CSC510: Foundations of Artificial Intelligence. The problem the AI program aims to solve is to help a flower retailer automate classifying "champion images" for its website. Overgoor *et al.* (2019) describe "champion images" as akin to thumbnail images. They are the first images users encounter when viewing products online. The flower retailer sells a handful of flower types: (a) daisies, (b) dandelions, (c) roses, (d) sunflowers, and (e) tulips. The retailer is looking to automatically classify these images according to their type. That way, when the retailer takes new pictures of their product each week, they no longer have to organize their photos manually, a time-consuming, labor-intensive process. This paper addresses the tools, libraries, APIs, and deep learning (DL) models that the program uses and explains how it represents knowledge. Additionally, the paper explores how the retailer can expand this program into expert systems (ES) in the future.

The primary tools, libraries, and APIs that the AI program uses include TensorFlow, Keras, NumPy, matplotlib, and Scikit-learn. TensorFlow is a machine learning system that focuses on training deep neural networks (DNNs) (Abadi *et al.*, 2016). It is one of the most widely used low-level DL frameworks, whereas Keras is the most popular high-level DL framework, providing a second level of abstraction to DL model development (Moolayil, 2019). The NumPy package provides NumPy arrays, which are the standard representation for numerical data in Python and have found widespread usage in industries ranging from gaming to space exploration (Van Der Walt *et al.*, 2011). Finally, matplotlib is a library for making 2D plots in Python, and Scikit-learn is a Python module that provides state-of-the-art

implementations of many popular machine learning algorithms (Hunter & Dale, 2007; Pedregosa *et al.*, 2011).

To run the program, all one needs to do is run the p_build_dataset.py file. For further information, consult the READ_ME.md file. The program begins by downloading the flower_photos.tgz file from the corresponding URL and extracts the 3,670 flower images into folders corresponding to their type. The size of the dataset is 218 MB. Next, the program preprocesses the data, creating training, validation, and testing directories, moving all images into subdirectories within the training folder, and then migrating 20% of the data into the validation folder (n = 731), 20% into the testing folder (n = 731), and leaving 60% in the training folder (n = 2,208). Training datasets are used to construct models, whereas validation datasets challenge and compare models (Xu & Goodacre, 2018). Finally, evaluation datasets are used to assess models' likely future performance (Mutuvi, 2021). Figure 1 shows the newly created folder structure, where the AI program parses the flower image dataset.

The program uses the ResNet-50 convolutional neural network (CNN), pre-trained on the ImageNet dataset, to extract features from the flower images (Rosebrock, 2019). CNNs are comprised of three types of layers: (a) convolutional layers, (b) pooling layers, and (c) fully-connected (FC) layers (O'Shea & Nash, 2015). Furthermore, Morid *et al.* (2019) describe ResNet-50 as a variation of ResNet, which aims to solve the accuracy saturation and vanishing gradients problems that some CNNs experience when more layers are added to them. One of the drawbacks of CNNs is that they require enormous amounts of data and extensive computational and memory resources to fully train from scratch. An alternative approach, known as transfer learning (TL), allows parameters of well-trained CNN models to be transferred to targeted tasks. Some CNN models, including ResNet-50, have proved very accurate in classifying ImageNet

pictures. Moreover, Huh *et al.* (2016) describe TL as the de facto standard for solving a wide range of computer vision problems. One of the primary benefits of TL is that it reduces the time taken to develop and train models (Theckedath & Sedamkar, 2020).

Rosebrock (2019) writes that, in general, there are two types of TL when applied to DL for computer vision: (a) treating networks as arbitrary feature extractors and (b) using fine-tuning. This paper describes the first technique: TL via feature extraction. Figure 2 illustrates this method applied to the Visual Geometry Group (VGG16) CNN, another CNN shown to have highly accurate results on the ImageNet dataset. The image on the left shows the original VGG16 network that outputs probabilities for each of the 1,000 ImageNet class labels. In contrast, the image on the right shows what happens when the FC layer is removed: the image stops propagating at this arbitrary but pre-specified position, enabling the program to extract all values from this layer, the max-pooling layer, and flatten them into a feature vector containing 25,088 attributes.

Rosebrock (2019) writes that by setting the parameter "include_top=False" in the ResNet-50 object's constructor, the FC layer that contains the Softmax classifier is removed. Removing the FC layer allows for extracting feature vectors for each image rather than vectors representing class membership probabilities (Brownlee, 2021). Next, the program iterates over the training, validation, and evaluation datasets, extracting each image's category from the folder structure and using the label encoder found in the Scikit-learn package to convert each class label to an integer representation before writing all 25,088 features for each image to one of three CSV files, one for each data split. The program outputs these files, along with the label encoder, to the output folder. Figure 3 shows the folder structure after the feature vectors have been created, and Figure 4 shows the CSV file for the evaluation data split. Rosebrock writes that once the AI

program has obtained these feature vectors, it can train any off-the-shelf machine learning model, including random forests, decision trees (DTs), and neural networks (NNs). The AI program this paper describes trains a simple, feedforward NN implemented using Keras via incremental learning. Incremental learning allows for the training of models on datasets too large to fit into memory by training them on small subsets of data called batches.

The simple feedforward NN this paper describes contains 4 layers: (a) an input layer, (b) two hidden layers that use the rectified linear unit (ReLU) activation function, and (c) an output layer that uses the Softmax activation function. The output of a ReLU neuron is equal to zero when the input is less than zero and equal to the input otherwise, which helps avoid overfitting of the training data and decrease the computation time needed to train the model (Kessler *et al.*, 2017). Additionally, ReLU is the default activation function for most modern DNNs that achieve state-of-the-art results (Brownlee, 2019). When training a NN, the cost function is key to adjusting the weights and creating a better fitting model (Ho & Wookey, 2020). The AI program uses categorical cross-entropy (CCE) to calculate the model's error and adjust its weights accordingly. CCE is the most popular error measure to train DL structures on classification tasks (Rusiecki, 2019).

Figure 5 shows the program's output after training the model over 5 epochs, along with a classification report. After the 5th training epoch, the model achieves accuracies of 99% and 90% on the training and validation datasets, respectively. The classification report demonstrates the model's robustness and presents the precision, recall, and f1-score for each category of the evaluation dataset. The model obtained the lowest precision (.81) on the roses category, indicating that roses had the highest number of false positives. Likewise, the model achieved the highest recall for sunflowers (.96), indicating that sunflowers had the fewest false negatives.

Figures 6 – 8 show a sample of 25 images from the evaluation dataset, along with the image's correct classification and the model's prediction. Of these 25 images, the model correctly predicted 19 out of 25 categories (76%). The program randomizes these 25 images whenever the RANDOM_SEED value is updated in the p_config.py file. Finally, the program saves the model as an HDF5 file allowing the retailer to predict the categories of new images in the future.

In conclusion, this paper explained a fully functioning AI program to help a flower retailer automate the process of classifying thumbnail images for its website. Table 1 shows the topics covered in each of this course's modules. The primary modules that the program utilizes are modules 3 and 6, relating to neural networks (NNs) and classification, respectively. In effect, intelligent search methods were not used to solve this image classification problem. Similarly, no aspects of expert systems (ES) or symbolic planning were used in the program. However, if the retailer wishes to model additional factors in the future, such as the click-through rate (CTR), the data could be appended to the CSV files and modeled using decision trees (DTs). The logic from the DTs could then be programmed into ES to help streamline the process of selecting champion images that produce the highest CTR.

The program this paper describes represents knowledge using transfer learning (TL) and the ResNet-50 network with weights pre-trained on the ImageNet dataset to extract 25,088 features from the flower image dataset. The features are then input into a simple, feedforward NN to model an image's category. After 5 training epochs, the model achieved an overall accuracy of 99% and 90% on the training and validation datasets, respectively. The classification report shows the model performed well on the evaluation data, as well. Finally, the model outputs predictions for 25 images, along with their correct classifications, and saves the model as an HDF5 file so the retailer can use it to classify thumbnail images in the future.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., & Isard, M. (2016). TensorFlow: A system for large-scale machine learning. *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 265–283.

Brownlee, J. (2019, January 8). A Gentle Introduction to the Rectified Linear Unit (ReLU). *Machine Learning Mastery*. https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

Brownlee, J. (2020, January 2). How to Calculate Precision, Recall, and F-Measure for Imbalanced Classification. *Machine Learning Mastery*. https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/

Brownlee, J. (2021, January 17). How to Choose an Activation Function for Deep Learning. *Machine Learning Mastery*. https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/

Ho, Y., & Wookey, S. (2020). The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling. *IEEE Access*, *8*, 4806–4813. https://doi.org/10.1109/ACCESS.2019.2962617

Huh, M., Agrawal, P., & Efros, A. A. (2016). What makes ImageNet good for transfer learning? *ArXiv:1608.08614 [Cs]*. http://arxiv.org/abs/1608.08614

Hunter, J., & Dale, D. (2007). The matplotlib user's guide. *Matplotlib 0.90. 0 User's Guide*.

Kessler, T., Dorian, G., & Mack, J. H. (2017). Application of a rectified linear unit (ReLU) based artificial neural network to cetane number predictions. *Internal Combustion Engine Division Fall Technical Conference*, *58318*, V001T02A006.

Moolayil, J. (2019). *Learn Keras for Deep Neural Networks: A Fast-Track Approach to Modern Deep Learning with Python*. Apress. https://doi.org/10.1007/978-1-4842-4240-7

Morid, M. A., Borjali, A., & Del Fiol, G. (2021). A scoping review of transfer learning research on medical image analysis using ImageNet. *Computers in Biology and Medicine*, *128*, 104115. https://doi.org/10.1016/j.compbiomed.2020.104115

Mutuvi, S. (2021, April 8). *Introduction to Machine Learning Model Evaluation*. Medium. https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f

O'Shea, K., & Nash, R. (2015). An Introduction to Convolutional Neural Networks. *ArXiv:1511.08458 [Cs]*. http://arxiv.org/abs/1511.08458

Overgoor, G., Chica, M., Rand, W., & Weishampel, A. (2019). Letting the Computers Take Over: Using AI to Solve Marketing Problems. *California Management Review*, *61*(4), 156–185. https://doi.org/10.1177/0008125619859318

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., & Dubourg, V. (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, *12*, 2825–2830.

Rosebrock, A. (2019, May 27). Keras: Feature extraction on large datasets with Deep Learning. *PyImageSearch*. https://www.pyimagesearch.com/2019/05/27/keras-feature-extraction-on-large-datasets-with-deep-learning/

Rusiecki, A. (2019). Trimmed categorical cross-entropy for deep learning with label noise. *Electronics Letters*, *55*(6), 319–320. https://doi.org/10.1049/el.2018.7980

Theckedath, D., & Sedamkar, R. R. (2020). Detecting Affect States Using VGG16, ResNet50, and SE-ResNet50 Networks. *SN Computer Science*, *1*(2), 79. https://doi.org/10.1007/s42979-020-0114-9

Van Der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). The NumPy array: A structure for efficient numerical computation. *Computing in Science & Engineering*, *13*(2), 22–30.

Xu, Y., & Goodacre, R. (2018). On splitting training and validation set: A comparative study of cross-validation, bootstrap, and systematic sampling for estimating the generalization performance of supervised learning. *Journal of Analysis and Testing*, *2*(3), 249–262.