MARMARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GELENEKSEL ÖZNİTELİKLERİ CNN MODELLERİNE ENJEKTE EDEREK UYDU GÖRÜNTÜLERİNDE ARAZİ SINIFLANFIRMASI

MEHMET ÇAĞRI AKSOY

YÜKSEK LİSANS TEZİ
Elektrik & Elektronik Mühendisliği

DANIŞMAN
Prof. Dr. Cem ÜNSALAN

İSTANBUL, 2022

MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES
IN PURE AND APPLIED SCIENCES

LAND CLASSIFICATION IN SATELLITE IMAGES BY INJECTING TRADITIONAL FEATURES TO CNN MODELS

MEHMET ÇAĞRI AKSOY

MASTER THESIS
Department of Electrical & Electronics Engineering

THESIS SUPERVISOR
Prof. Dr. Cem ÜNSALAN

ISTANBUL, 2022

ACKNOWLEDGEMENT

I would like to thank my thesis advisor Prof. Dr. Cem Ünsalan. And Dr. Beril Sırmaçek who did not spare his knowledge and support during the period. I would also like to thank my dear family and friends for their support and motivation.

July 2022 **Mehmet Çağrı Aksoy**

TABLE OF CONTENTS

ACKNOWLEDGEMENT III

ÖZET VI

ABSTRACT VII

SYMBOLS VIII

ABBREVIATIONS IX

LIST OF FIGURES X

LIST OF TABLES XII

1. INTRODUCTION 1

1.1. LITERATURE SURVEY 2

2. METHODOLOGY 5

2.1. WIDE AND DEEP LEARNING 5

2.2. OUR PROPOSED INJECTING TRADITIONAL FEATURES AS INPUTS TO THE CNN 6

3. DESIGN DETAILS 6

3.1. CNN MODELS 8

3.1.1. SqueezeNet *9*

3.1.2. MobileNetV2 ..... *10*

3.1.3. ShuffleNetV2 ..... *10*

3.1.4. VGG16 ..... *11*

3.1.5. ResNet50V2 ..... *12*

3.2. TRADITIONAL FEATURE EXTRACTION METHODS USED IN THIS STUDY 13

3.2.1. Sample Mean *13*

3.2.2. Gray Level Co-occurrence Matrix Features *13*

3.2.3. Hu Moments *15*

3.2.4. Histogram of Oriented Gradients *16*

3.2.5. Local Binary Patterns *17*

3.2.6. Color Invariants *18*

3.3. PROPOSED FEATURE INJECTION METHOD FOR CLASSIFICATION 19

4. EXPERIMENTAL STUDY 21

4.1. DATASET USED IN EXPERIMENTS 21

4.1.1. EuroSAT Dataset 21

4.1.2. Ships in Satellite Imagery 22

4.1.3. Planes in Satellite Imagery 22

4.1.4. Trees in Satellite Imagery 23

4.2. HARDWARE AND SOFTWARE REQUIREMENTS 23

4.3. EUROSAT TESTS 24

4.3.1. The Effect on using traditional features without CNN Models 24

4.3.2. The Effect on the SqueezeNet Model 24

4.3.3. The Effect on the MobileNetV2 Model 25

4.3.4. The Effect on the ShuffleNetV2 Model 26

4.3.5. The Effect on the VGG16 Model 27

4.3.6. The Effect on the ResNet50V2 Model 28

4.4. SHIP-PLANE DATASET RESULTS 28

4.4.1. The Effect on the MobileNetV2 Model 28

4.4.2. The Effect on the VGG16 Model 29

4.5. TREES DATASET RESULTS 29

4.5.1. The Effect on the SqueezeNet Model 30

4.5.2. The Effect on the MobileNetV2 Model 30

4.5.3. The Effect on the ShuffleNetV2 Model 30

4.5.4. The Effect on the ResNet50V2 Model 31

4.5.5. The Effect on the VGG16 Model 31

4.6. TREE CLASSIFICATION RESULTS 31

4.7. OVERVIEW OF THE INJECTION PERFORMANCE 32

5. CONCLUSION 34

REFERENCES 36

CURRICULUM VITAE 39

ÖZET

GELENEKSEL ÖZNİTELİKLERİ CNN MODELLERİNE ENJEKTE EDEREK UYDU GÖRÜNTÜLERİNDE ARAZİ SINIFLANFIRMASI

Bu tezde, CNN modellerinin ve özellikle küçük boyutlu modellerin doğruluğunu artırmak için yeni bir yöntem önerdim. Önerilen yöntem, geleneksel bilgisayarlı görme özelliklerini CNN modeline birleştirmeye dayanmaktadır. Önerilen yöntemi SqueezeNet, MobileNetV2, ShuffleNetV2, VGG16 ve ResNet50V2 üzerinde test ettim. Bu modellerin boyutu 0,5 MB ile 528 MB arasındadır. Geleneksel bilgisayarlı görme özelliklerine gelince, örnek ortalama, gri seviye birlikte oluşum matrisi özellikleri, Hu momentleri, yerel ikili desenler, yönlendirilmiş gradyanların histogramı ve renk değişmezlerini kullandım. Arazi sınıflandırması yapmak için önerilen yöntemi EuroSAT, Plane, Ship ve Trees veri setinde test ettim. Deneysel sonuçlarım, önerilen yöntemin, seçilen CNN modellerine kıyasla arazi verilerinin sınıflandırılmasını önemli ölçüde geliştirdiğini göstermektedir. Böylece önerilen yöntemin uç (düşük güç ve düşük bellek) cihazlarda arazi sınıflandırma problemlerini performans kaybı olmadan çözmede yardımcı olabileceğini kanıtladım.

Bunun nedenlerini sıralayacak olursak, kara alanlarının sınıflandırması ile, gerek tarım arazisi tespitinin, gerek sanayi ve şehir alanı gibi alanların tespitlerinin ve sınırlarının belirlenmesi için kullanılmaktadır. Ayrıca, sulak ve kurak alanların tespiti ile küresel ısınmanın etkileri gözlemlenmektedir. Ormanlık alanların takibi ve sınıflandırması ile, ormanlık alan durumu gözlemlenmektedir. Gemi ve uçak gibi araçların uydu görüntülerinden tespiti yapılıp takip ve sayısı hesaplanmaktadır. Burada bahsettiğim uygulamalar, kara sınıflandırması uygulamalarının sadece bir kısmını oluşturmaktadır.

Uzaktan algılama alanında derin öğrenme uygulamaları yıllardır başarıyla kullanılmaktadır. CNN tabanlı modeller, uydu görüntülerinde arazi sınıflandırma problemlerinin çözümünde güvenilir sonuçlar vermektedir. Bu modeller genellikle büyük boyutlara sahiptir. Ancak insansız hava araçlarında yapılan uygulamalar gibi uygulamalar için daha düşük güç ve bellek gereksinimleri nedeniyle küçük boyutlu modellere ihtiyaç duyulmaktadır. CNN modellerinin doğruluğu, boyutları (parametre sayısı) da azaldığında azalır.

Sonuç olarak, önerilen geleneksel yöntemleri derin öğrenme modellerine entegre etme metodu ile seçilen veri kümelerinin kara alanları tespit sonuçlarını oldukça iyi oranda arttırdığını gösterdim.

ABSTRACT
LAND CLASSIFICATION IN SATELLITE IMAGES BY INJECTING TRADITIONAL FEATURES TO CNN MODELS

In this thesis, I propose a new method to improve the accuracy of CNN models, and especially the ones with small sizes. The proposed method is based on concatenating traditional computer vision features into the CNN model. I tested the proposed method on SqueezeNet, MobileNetV2, ShuffleNetV2, VGG16, and ResNet50V2. These models have the size from 0.5 MB to 528 MB. As for traditional computer vision features, I used sample mean, gray level co-occurrence matrix features, Hu moments, local binary patterns, histogram of oriented gradients, and color invariants. I tested the proposed method on the EuroSAT, Plane, Ship and Trees dataset to perform land classification. My experimental results show that the proposed method notably improves the classification of land data compared to the chosen CNN models. Thus, I proved that the proposed method could be helpful in solving land classification problems on edge (lower power and low memory) devices without losing performance.

If we list the reasons for this, it is used for the classification of land areas, the determination of agricultural land, the determination of areas such as industry and city areas and the determination of their borders. In addition, the effects of global warming are observed with the determination of wet and dry areas. With the monitoring and classification of forest areas, the forest area status is observed. Vehicles such as ships and planes are detected from satellite images and their tracking and number is calculated. The applications I mentioned here are only a part of the land classification applications.

Deep learning applications has been used successfully in the field of remote sensing. CNN-based models give reliable results in solving land classification problems in satellite images. These models usually have huge size. However, small-sized models are needed for applications such as the ones implemented on unmanned aerial vehicles because of lower power and memory requirements. The accuracy of the CNN models reduces when their size (the number of parameters) is also reduced.

As a result, I have shown that the proposed injecting traditional inputs and selected datasets and models increase land classification results very well.

SYMBOLS

ms **:** Millisecond

ABBREVIATIONS

ASM **:** Angular second Moment

CNN **:** Convolutional Neural Network

NN **:** Neural Network

UAV **:** Unmanned Aerial Vehicle

HOG **:** Histogram of Oriented Gradients

LBP **:** Local Binary Pattern

GLCM **:** Gray Level Co-occurrence Matrix Features

RGB **:** Red, Green, Blue

KB **:** Kilobyte

MB **:** Megabyte

GB **:** Gigabyte

GPU **:** Graphics Processing Unit

GPU **:** Tensor Processing Unit

TFLOPS **:** Floating Point Operations Per Second

GDDR6 **:** Graphics Double Data Rate 6 Synchronous Dynamic Random-Access Memory

LIST OF FIGURES

Figure 1.1 DeepForest [7] research tree detection example 2
Figure 1.2 Individual Tree-Crown Detection in RGB Imagery Using Semi-Supervised Deep Learning Neural Networks" [8] research tree detection example 2
Figure 1.3 "Deep Learning in Forestry Using UAV-Acquired RGB Data: A Practical Review" [9] research tree detection example 3
Figure 1.4 "Coconut trees, detection and segmentation in aerial imagery using mask region-based convolution neural network" [10] research tree detection example 4
Figure 1.5 A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification Example 4
Figure 3.1 CNN architecture model blocks overview [15] 7
Figure 3.2 CNN convolution operation example [15] with kernel size 3x3. 7
Figure 3.3 ReLU activation function graph 8
Figure 3.4 SqueezeNet model explanation chart [16] 9
Figure 3.5 MobileNetV2 model explanation chart [17] 10
Figure 3.6 (a) ShuffleNet V2 with residual. (b) ShuffleNet V2 with SE. (c) ShuffleNet V2 with SE and residual. [18] 11
Figure 3.7 VGG16 Model Explanation [19] 12
Figure 3.8 ResNet50V2 Model Explanation [20] 12
Figure 3.9 GLCM Vector Samples and directions 14
Figure 3.10 Hu Moments class-based result table for EuroSAT images 16
Figure 3.11 Histogram of oriented gradients result matrix(left) the original image(right) . 17
Figure 3.12 Original image on left and histogram projection of local binary pattern results of forest image on right. 18
Figure 3.13 Original image on left and histogram projection of local binary pattern results of pasture image on right. 18
Figure 3.14 Color invariant results for each band of RGB 19
Figure 3.15 The Injecting Traditional Inputs to the CNN Models flowchart 20
Figure 4.1 EuroSAT Data Samples 22
Figure 4.2 Ships in Satellite Imagery Data Samples 22
Figure 4.3 Planes in Satellite Imagery Data Samples 23
Figure 4.4 The accuracy graph of traditional inputs using with fully connected NN. 24

Figure 4.5 The model summary of using SqueezeNet solution 24
Figure 4.6 Accuracy graph comparison of SqueezeNet 25
Figure 4.7 Accuracy graph comparison of MobileNetV2 26
Figure 4.8 Accuracy graph comparison of ShuffleNet 27
Figure 4.9 Ship & Plane Dataset 8 Accuracy graph comparison of MobileNetV2 29
Figure 4.10 Ship & Plane Dataset Accuracy graph comparison of VGG16 29
Figure 4.11 Tree Dataset Accuracy graph comparison of SqueezeNet 30
Figure 4.12 Tree Dataset Accuracy graph comparison of MobileNetV2 30
Figure 4.13 Tree Dataset Accuracy graph comparison of ResNet50V2 31
Figure 4.14 Tree Dataset Accuracy graph comparison of VGG16 31
Figure 4.15 Model size versus accuracy comparison for the selected CNN models traditional feature injection method. 33

LIST OF TABLES

Table 3.1 The parameter and model size of the selected CNN models. 9
Table 4.1 The effect of adding wide features to the SqueezeNet model. 25
Table 4.2 The effect of adding wide features to the MobileNetV2 model. 26
Table 4.3 The effect of adding wide features to the ShuffleNetV2 model. 27
Table 4.4 The effect of adding wide features to the VGG16 model. 27
Table 4.5 The effect of adding wide features to the ResNet50V2 model. 28
Table 4.6 SqueezeNet Precision, Recall, F1 Score comparison with and without injecting the wide inputs. 32
Table 4.7 SqueezeNet Precision, Recall, F1 Score comparison difference with and without injecting wide inputs 33

1. INTRODUCTION

Due to the increase in the human population and the increasing damage to our environment, many remote sensing projects have become very popular in areas such as environmental diagnosis, changes, risk analysis, and prevention of disasters. As we know, the land areas and trees, which form the main subject of this thesis, have a significant place in our environment and our future. We understand that the existence of the tree has a better effect on production-consumption. Due to research by "Nowak et al.: Oxygen Production by Urban Trees" [1], consumption from the atmosphere is very important for climate change and our existence. Also, in the research "Forest health in a changing world: effects of globalization and climate change on forest insect and pathogen impacts" [2] the health of the forest has cumulative effects on our life in terms of many fields, like socioeconomic. Additionally, the research focused on the importance of forest management and the classification of forests to understanding to understand factors that impair the health of trees and forests.

Especially Turkey and many Mediterranean countries suffer from extensive forest fires in the summer season. Each year, we lost many forests and the creatures that live in these forests. Due to the research "Forest diversity, climate change and forest fires in the Mediterranean region of Turkey" [3], shows us that only in Turkey 12 million tree is highly sensitive to fires. Additionally, these fires can be prevented by forest management scenarios.

There is many felling of trees due to unplanned settlements, natural disasters, and industrial purpose destructions, the tree population is decreasing considerably, and it is important to actively monitor the change.

Tree diagnostic projects have been used for many years and are very popular, thanks to deep learning algorithms.

Land detection applications are used for many different classes and objects apart from tree classification. Classes such as ship, airplane, river, plantation, and city used in this thesis constitute the main part of the tests. The detection and classification of ships and planes in the image is especially used in the field of logistics. Determination of fields and wetlands is very important for agricultural and economic reasons. In addition, the percentage change of the areas and the effect of global warming and environmental disasters on the selected area are observed. For such reasons, many products, studies and datasets have been introduced. In this thesis, I also talked about these studies and

showed the advantages of my proposed method using some datasets.

Tree classification and generally land classification is already a popular task for computer vision algorithms and solutions. Nowadays, much research working on this field thanks to the open access to datasets, in the remote sensing field especially.

1.1. Literature Survey

Deep learning methods have proven themselves for many years with their success in various fields. One of the areas where these methods are frequently used is remote sensing. Owing to their success in this field, many deep learning models have been used and tested in various studies such as [4][5][6]. A well-known solution a is for tree detection is DeepForest [7]. DeepForest is a python package for understanding and detecting tree images from satellite images or airborne RGB images. A result of DeepForest prediction can be seen on Figure-2.1.

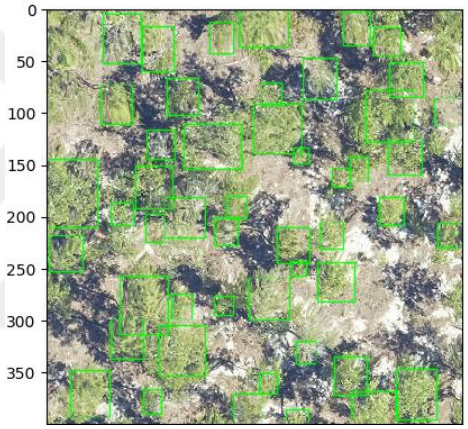


Figure 1.1 DeepForest [7] research tree detection example

Another proposed solution and the research of the work is "Individual Tree-Crown Detection in RGB Imagery Using Semi-Supervised Deep Learning Neural Networks "[8]. In this work, they have focused the semi supervised learning instead of the unsupervised one to catch some missing features in case of some noisy data. So that,

Figure 1.2 Individual Tree-Crown Detection in RGB Imagery Using Semi-Supervised Deep Learning Neural Networks" [8] research tree detection example

they have proposed that, the using their semi-supervised method usage increase the detection accuracy. Their examples can be seen on Figure 1.2.

Additional work is based on UAV RGB tree detection and segmentation, named “Deep Learning in Forestry Using UAV-Acquired RGB Data: A Practical Review” [9]. They have divided the research by 3 step that, individual Tree Detection, tree Species Classification, and forest Anomaly Detection. Due to results and conclusion of this work the using a single deep learning solution is not enough. The semantic segmentation way they proposed is the way they offer. Their examples can be seen on Figure 1.3.

Figure 1.3 “Deep Learning in Forestry Using UAV-Acquired RGB Data: A Practical Review” [9] research tree detection example

Different research named “Coconut trees, detection and segmentation in aerial imagery using mask region-based convolution neural network” [10] is aimed at the palm tree zones that are affected by disasters. Examples such as tree detection, calculating the number of trees, and calculating the estimated wooded area are used in this research. With the help of these estimates, the amount that can be used for lumbering, the estimated oxygen production capacity, or the change of wooded areas from year to year and calculations can be calculated as a percentage. Their examples can be seen on Figure 1.4.

Figure 1.4 "Coconut trees, detection and segmentation in aerial imagery using mask region-based convolution neural network" [10] research tree detection example

Deep learning methods, by their very nature, acquire many features and learn on that basis. The point to be noted here is that especially convolutional neural networks (CNN) based methods usually extract features because of the filtering. This results in the inability to predict whether CNN-based methods utilize more complex feature information or not.

I explained the datasets used in land use classification in the next section. But I would like to talk about a study in here. Along with the EuroSAT research [27], a classification is made as in the example picture shown in Figure 1.5. With this classification, the density and type of land use can be easily calculated.

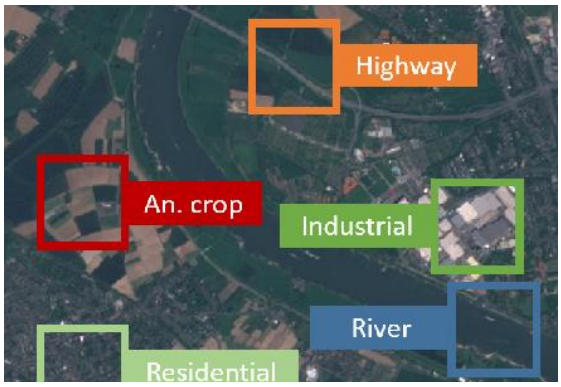


Figure 1.5 A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification Example

All the researches mentioned above constitute our source of motivation. I needed to do this study especially because of the necessity of determining agricultural lands, the determination of forest areas due to forest fires, the need to follow up urbanization, which is taking place today in Turkey especially. I shared the data I obtained during my thesis study as open source and I hope it will be useful to all researchers.

2. METHODOLOGY

In this thesis, I propose a method to further increase the object recognition performance of CNN-based methods on satellite images. This method is based on wide and deep learning that has been used for years from e-commerce sites to search engines, especially in recommendation and recommendation systems. The wide and deep learning is proposed by the HT Cheng [11].

2.1. Wide and Deep Learning

As I mentioned, the wide and deep learning is created for the recommender systems and many leads tech firms (Google, Amazon etc.) are using this kind or architecture for their systems due to the research "Modern recommender systems: from computing matrices to thinking with neurons" [12]. The general schematic of wide and deep learning can be seen on the Figure 2.1.

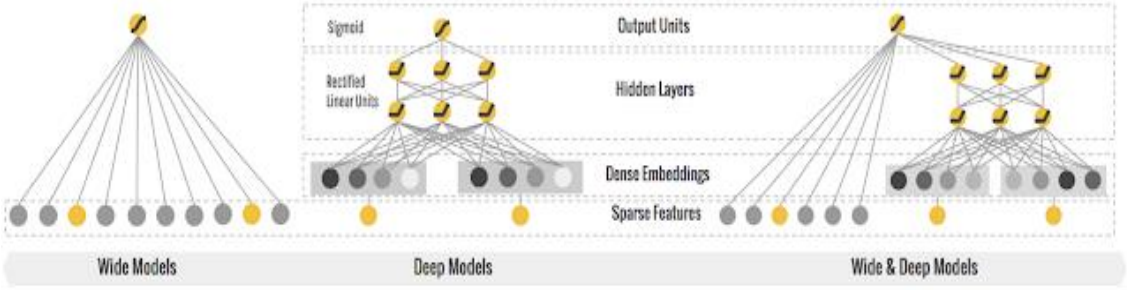


Figure 2.1 Proposed Wide and Deep Learning Explanation in Google AI blog [13]

Memorization and the generalization are the important part of our brains, and we are using these things in our daily life. Due to the proposed purpose of the wide and deep learning, "Can we teach computers to learn like humans do, by combining the power of memorization and generalization? [13]". We can concatenate these features by combining the wide and deep inputs into one model.

Deep learning algorithms and models are usually doing their job very good, thanks to their generalization power. But the generalization not always best option for the

prediction. For example, if we want to teach fly skills of animals, a deep learning algorithm by feeding many animal images, and in result the algorithm probably will learn that all birds can fly. The results come from the generalization ability. Because 99% of birds can fly but what about the rest? Ostriches and penguins cannot fly. The deep learning missing that part.

CNN are extracting the features using the linear filters such as convolutional layers with some different window size. So that the adding non-linear features might better effect, and this nonlinearity comes from the wide inputs. If we continue with the previous example, adding the inputs of "Ostriches and penguins cannot fly" as like wide input, increase the accuracy of results. Wide inputs can be count as exceptions, or the manual inputs that we are adding or manipulating the deep learning side.

2.2. Our proposed Injecting Traditional Features as Inputs to the CNN

Our proposed method replaces the input values used as text input in the proposal system with CNN and traditional computer vision features. In this way, I have developed a unique object recognition method in the field of remote sensing. As a result of our literature review, I determined that such a method has not yet been used exactly as stated in the field of object recognition. The closest study to the proposed method was developed by "M Jbene et al: Fusion of convolutional neural network and statistical features for texture classification" [14]. In this study, statistical features are focused on increasing the performance of the CNN model. As a result of the research, not much performance increase was achieved. With the innovations in the proposed method, I able to achieve a significant performance increase.

3. DESIGN DETAILS

The design details of the system can be divided into 2 groups. The proposed system is using deep learning model with famous convolutional neural network (CNN) models such as SqueezeNet, MobileNetV2, ShuffleNet, VGG16 and ResNet50V2and the wide learning model. CNN is a dominant subclass of the neural networks. It is mostly used in computer vision and detection tasks. It is designed for learning features through the backpropagation and given convolution layers due to the research [15]. The overview of the CNN model block chart can be seen in Figure 3.1.

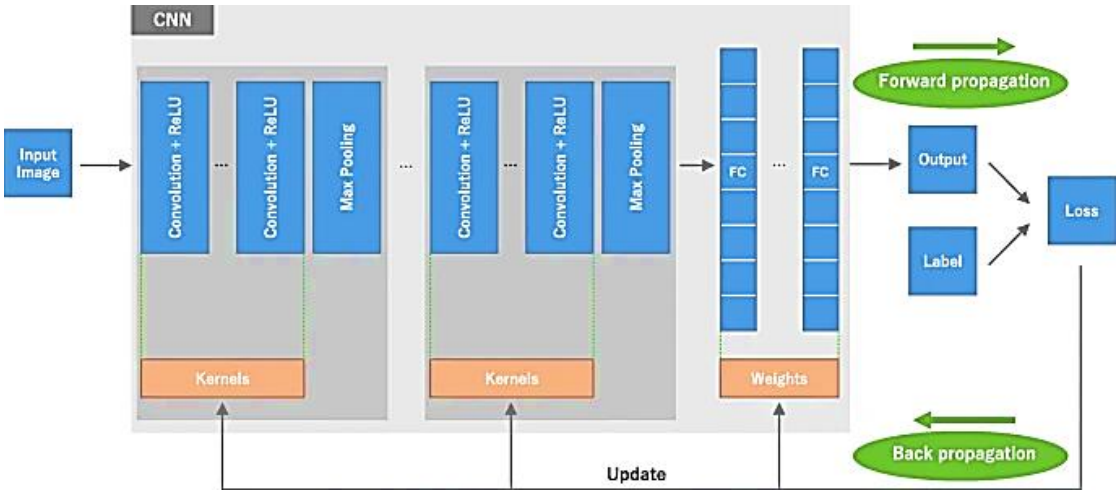


Figure 3.1 CNN architecture model blocks overview [15]

The ability that differs CNN from other feature extraction and classification method is CNN do not have to feed with the manually feed with the features, CNN can handle the feature extraction steps it on its own. Also, the building blocks of CNN is different. It has mainly, convolutional, pooling and the fully connected layers. The data is transformed and the manipulated through the layers and last, the prediction can be made. The convolutional layer is the important part of the CNN as we understand from its name. It does the convolutional operation for feature extraction, by given a kernel size.

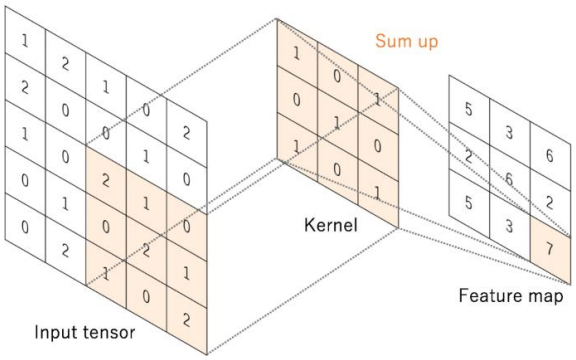


Figure 3.2 CNN convolution operation example [15] with kernel size 3x3.

The kernel size, also known as window size, is used during convolution operation by summarizing the number of this array. Then, each element of the new array will be filled with the elements gathered from convolutional operation. This operation can be seen at Figure 3.2.

The other layers are pooling layers, the pooling is doing simply down sampling operation for decreasing the number of complexity and dimension.

The last but not the least layers are the fully connected layers. In Keras we are using these layers as “Dense” layers till the end. In these layers are feeding with the end of pooling layers with “Flatten” flattening operation. This gives us a 1D array and this array is directly used by Dense layers. The layers are responsible for the eliminates the lowest possibilities on the given array. For instance, if we need 10 class of object in the end, the fully connected layer exit must be 10 class, means that the eliminate lowest possible alternatives. Each layer is used by the non-linear activation function that required for the explained behavior. The activation functions are sigmoid, SoftMax, ReLU, LeakyReLU, tanh, swift and so on. The most famous one is ReLU and the formula is below and graph can be seen on Figure 3.3.

$$\mathrm{ReLU}(x) = \max(0, x)$$

The wide learning part is mainly covered by the traditional inputs. The manually extracted features (will be introduced in following sections) will concatenate with the deep learning model just before feed the fully connected layer. After concatenating these features to the deep learning side features, the pool size will increase, and the pool passes to the fully connected layers next.

3.1. CNN Models

I picked different CNN models in this study. Therefore, I briefly summarized them in this section. I summarize the CNN models considered in the previous section here. The

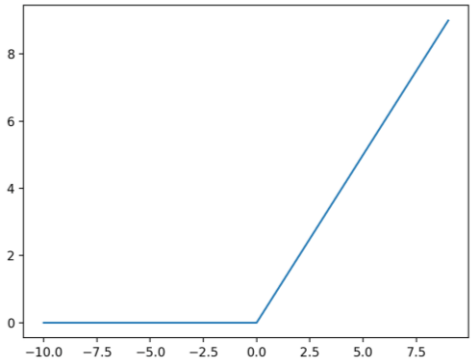


Figure 3.3 ReLU activation function graph

aim is to compare these models in terms of their number of parameters and mode size. Therefore, I tabulate these values in Table 3.1. As can be seen in this table, the selected CNN models have size between 0.5 MB and 528 MB. The number of parameters in these models are within the range of 729 K to 23.5 M. I will use these models while

injecting traditional features to them.

Table 3.1 The parameter and model size of the selected CNN models.

Model Name	# Parameters	Model Size
SqueezeNet	729K	0.5MB
MobileNetV2	2.2M	14MB
ShuffleNetV2	4M	10MB
VGG16	14.7M	528MB
ResNet50V2	23.5M	98MB

3.1.1. SqueezeNet

The first model I picked in this study is SqueezeNet proposed by [16]. This model has the low size specifically developed to be used in embedded systems. It offers an AlexNet level accuracy in “ImageNet” weights in only 0.5MB space. Also, it has 50x fewer parameters due to the authors by [16]. Therefore, it represents the low end of the CNN models available in literature. The unique part of SqueezeNet is the fire modules and squeeze operation. It uses a strategy to decrease number of parameters, uses fire modules via 1x1 convolution layers. The explanation of the SqueezeNet model can be seen on Figure 4.4. During usage of this model, I freeze the SqueezeNet top layers with pre-trained "ImageNet" weights.

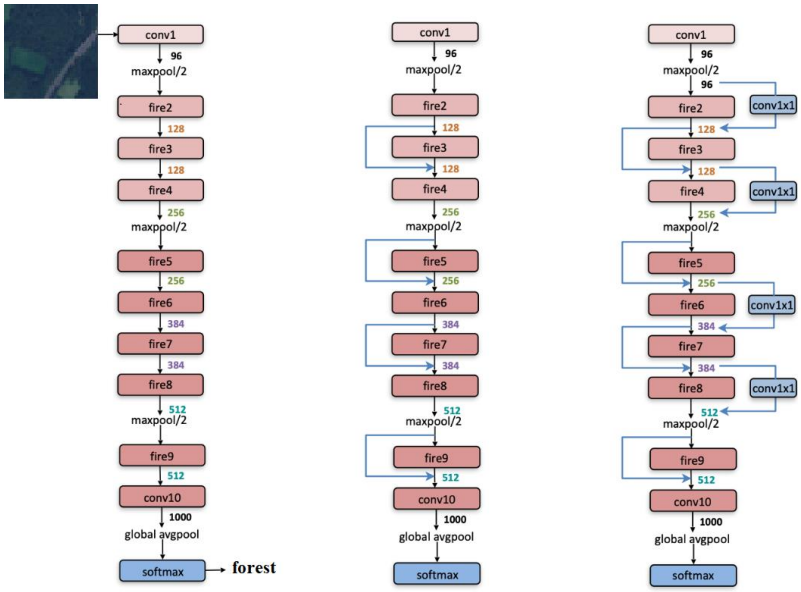


Figure 3.4 SqueezeNet model explanation chart [16]

3.1.2. MobileNetV2

MobileNetV2 is the next model considered in this study. It has been proposed by [17]. As in SqueezeNet, it has low model size such that it can be safely used on mobile and edge devices. The explanation of the MobileNetV2 model can be seen on Figure 3.5.

The unique part of MobileNetV2 is the activation function of ReLU6 and its Deepwise layers. I freeze the MobileNetV2 top layers with pre-trained "ImageNet" weights. I have used average pooling layer after extracted all deep features thanks to the TensorFlow MobileNetV2 implementation function. With this, I have 1280 parameter length array, then I will put these in Batch Normalization layer, in the last step using "SoftMax" I have predictions of 10 classes. The MobileNetV2 has inverted residual structure. In this version they have removed the nonlinear narrow layers.

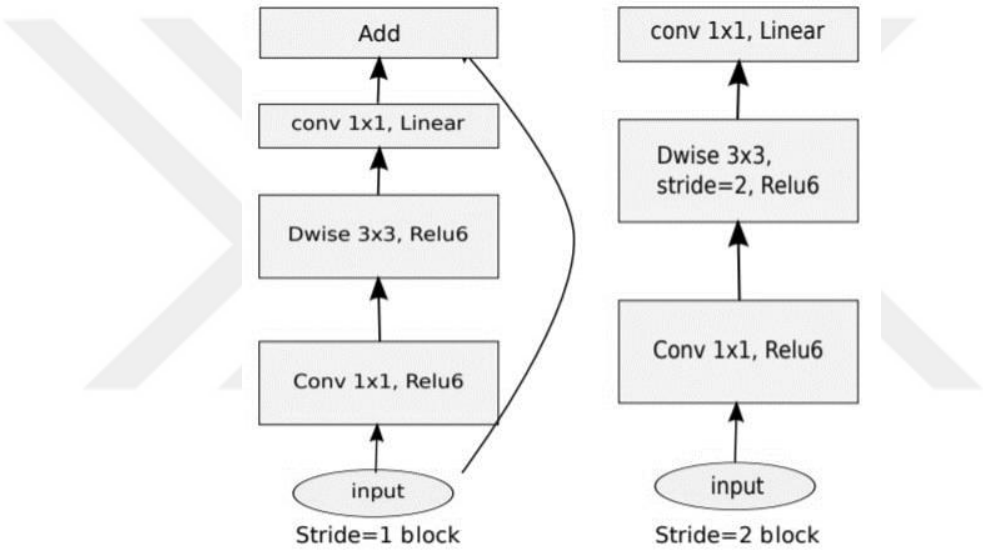


Figure 3.5 MobileNetV2 model explanation chart [17]

3.1.3. ShuffleNetV2

ShuffleNetV2 is the third model considered in this study. It has been proposed by [18]. ShuffleNetV2 has similar number of parameters and model size as in MobileNetV2. The explanation of the ShuffleNetV2 model can be seen on Figure 3.6.

The unique part of ShuffleNetV2 is channel shuffle and using MobileNet`s DeepWise or SENet`s SE modules. Hence, it also represents the low model size option. I freeze the ShuffleNetV2 top layers with pre-trained "ImageNet" weights.

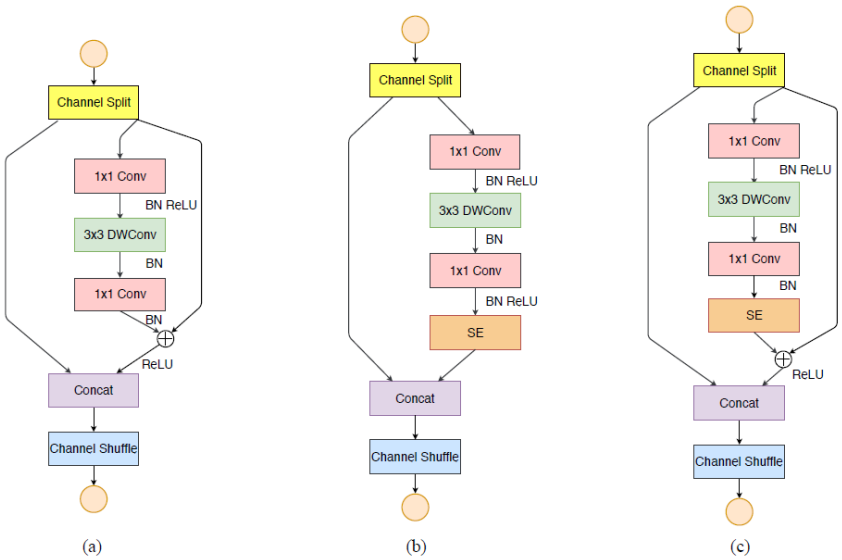


Figure 3.6 (a) ShuffleNet V2 with residual. (b) ShuffleNet V2 with SE. (c) ShuffleNet V2 with SE and residual. [18]

The ShuffleNetV2 is designed for memory and power efficiency. It has a channel split operator that divide the channels into 2 groups. One part of the channel passes into convolutional layers to extract features, but other part is directly injecting into the feature pools. Due to the author, this way decreases memory usage because of equal channel width and increases the efficiency.

3.1.4. VGG16

Besides small sized models considered in previous sections, I also picked large models in this study. The aim for this selection is to observe the effect of the proposed feature injection method on them. Therefore, I picked the VGG16 model proposed by [19]. This is the fourth model used in this study. The unique part of VGG16 is the resizing input images while training and the testing. The explanation of the VGG16 model can be seen on Figure 3.7. The VGG16 is focused on the convolutional layers which followed by “Max Pooling” every time. Max pooling is a pooling operator that outputs maximum value of given window. Whole architecture has, padding with 3x3 convolutional layers and pooling layer with max feature. The tail part consists of 2 fully connected layers with “SoftMax” activation function.

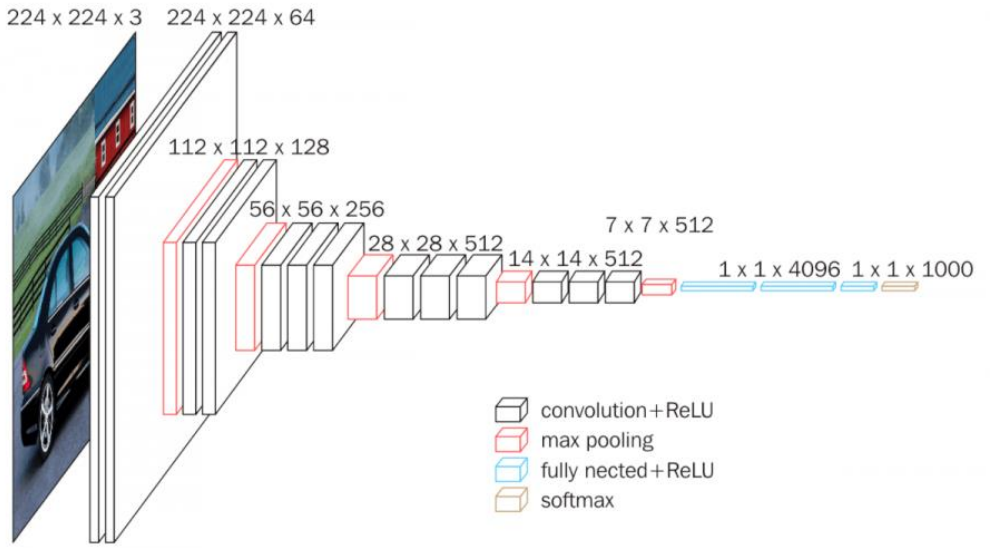


Figure 3.7 VGG16 Model Explanation [19]

3.1.5. ResNet50V2

As in VGG16, I picked the ResNet50V2 model proposed by [20]. The reason for choosing this model was to see how the biggest model that I have selected, would behave in the I developed. This is the fifth and final CNN model considered in this study. As we can understand from its name, the ResNet50V2 is the enhanced version of ResNet50. The unique part of this architecture is coming from skipping layers zone. The ResNet has an ability to skip some layers, thanks to this, they train deeper ResNet also known as Residual Network. In this ResNet50V2 version, each convolutional layer followed by Batch Normalization layers. The explanation of the ResNet50V2 model can be seen on Figure 3.8.

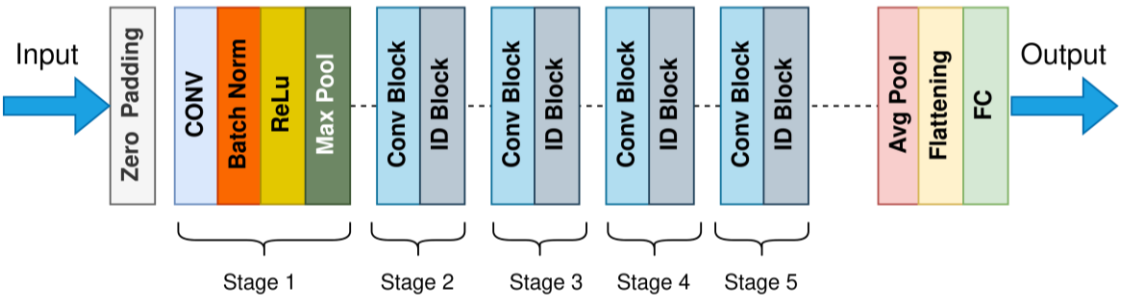


Figure 3.8 ResNet50V2 Model Explanation [20]

3.2. Traditional Feature Extraction Methods used in This Study

Our aim in this study is improving the performance of known CNN models by injecting features extracted by traditional methods. Therefore, I summarize the candidate traditional methods in this section. To note here, the proposed system is not limited by the traditional methods introduced in this section. Other methods can also be used in the proposed framework.

3.2.1. Sample Mean

The first traditional feature extraction method used in this study is the sample mean. A mean value can be considered as a first-order statistical feature extraction method that can be used to summarize information in color bands. The formula can be seen below.

$$mean = \frac{(\sum_{n=1}^{N} x)}{N}$$

Therefore, I obtained the average value for each band in our color image. I have 3 features from each image in dataset. In final step, I normalized these values by using the formula can be seen below. Hence, these features will be ready to be used in injection.

$$norm = \sqrt{(\sum_{n=1}^{N} x^2)}$$

3.2.2. Gray Level Co-occurrence Matrix Features

Another traditional feature extraction method considered in this study is gray level co-occurrence matrix (GLCM) proposed by [21]. GLCM has been proposed for texture analysis for a long time. It has been generalized to other problems, including land use classification in satellite images, since then. Haralick et al. introduced several second-order statistical features based on GLCM. In 1973 Haralick has been introduced the GLCMs and divide these by 6 groups. They are correlation, contrast, homogeneity, energy, and ASM matrix features. The calculation of matrix features is simple. First, the image converted to grayscale images. So, the number of channels is decreased to one.

The data is calculated by averaging of these 4 different directed vectors of image. The GLCMs are calculating the how often a pixel value occurs vertically, horizontally or given array direction. The arrays can be seen on figure 3.9.

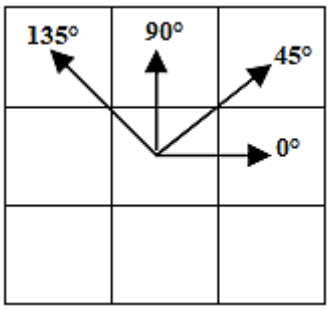


Figure 3.9 GLCM Vector Samples and directions

3.2.2.1. Contrast Feature

Contrast is responsible for measuring the spatial frequency of an image. Squares the difference between the highest and lowest values of a set of pixels. Contrast texture measures local variations found in the image.

$$cont = (\sum_{i,j=0}^{N} P_{ij}(i-j)^2)$$

3.2.2.2. Homogeneity Feature

As can be understood from the name of homogeneity, it is higher when the values of the pixels in the image are close to each other or the same. Homogeneity is more susceptible to the presence of elements close to the diagonals. They contain contrast and opposing logic in GLCM applications and are widely used.

$$homo = (\frac{\sum_{i,j=0}^{N} P_{ij}}{1+(i-j)^2})$$

3.2.2.3. Dissimilarity Feature

The measure of local variations in the image is called dissimilarity.

$$diss = (\sum_{i,j=0}^{N} P_{ij})|i-j|)$$

3.2.2.4. Angular second Moment (ASM) Feature

ASM measures repetitions in selected pixels and defects in textures. Since it is used in gray scale images, it gives higher results compared to other attributes.

$$ASM = (\sum_{i,j=0}^{N} P_{ij}^2)$$

3.2.2.5. Energy Feature

Energy is computed as the square root of an angular second moment. When the selected pixels are same or similar, energy has higher value.

$$energy = \sqrt{ASM}$$

3.2.2.6. Correlation Feature

Correlation feature is measuring linear dependencies between the selected pixels.

$$corr = (\sum_{i,j=0}^{N} P_{ij}[\frac{(i-u_i)(j-u_j)}{\sqrt{(o_i^2)(o_j^2)}}])$$

3.2.3. Hu Moments

The third traditional feature extraction method used in this study is Hu moments proposed by Hu et al [22]. Hu moments are invariant to transformations in grayscale images. In this study, I calculated seven moment invariants from the grayscale image

$$M_{ij} = \sum_x \sum_y x^i y^i I(x,y)$$

$$\eta_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q I(x,y)$$

$$barx = \frac{M_{10}}{M_{00}}, \bar{y} = \frac{M_{01}}{M_{00}}$$

$$mu_{pq} = \frac{\eta_{pq}}{\eta_{00}^{\gamma}}, \gamma = \frac{p+q}{2}$$

$$H_1 = \mu_{20} + \mu_{02}$$

$$H_2 = (\mu_{20} - \mu_{02})^2 + 4(\mu_{11})^2$$

obtained from the corresponding color image. The results can be seen Figure 3.10.

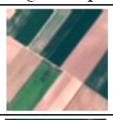
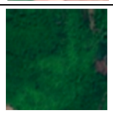
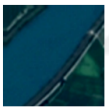
Image Samples	Class Name	H[0]	H[1]	H[2]	H[3]	H[4]	H[5]	H[6]
	Annual Crop	4,20501379	5,08667008	4,36691273	2,02687392	-1,41380867	-4,39255744	-6,02848188
	Forest	5,42071394	2,59517427	4,13966762	1,90893742	-1,55119832	-2,18456772	-6,88069302
	Herbaceous Vegatation	3,37844077	4,66975726	4,54909498	1,52532281	-3,94067806	-7,79810920	7,84235106
	Highway	6,74707283	1,41548819	1,07507123	1,68415508	-5,45441054	4,39703126	-4,64806347
	Industrial	2,77073300	6,73469690	5,84423031	6,83916563	-4,29881926	-3,80441761	-4,64366130
	Pasture	5,28343107	7,31841500	1,48302308	4,67460114	-1,22977445	-1,19519071	5,04630323
	Permanent Crop	3,04284940	5,49846549	1,69226789	2,32621448	-7,07451630	2,46365525	1,27659797
	Residential	5,01773394	1,57431213	2,09217024	2,15362120	1,12017254	2,62381304	-9,13791372
	River	5,25499052	1,15630634	2,07039210	3,01811642	-6,23654871	3,20690217	4,24557340
	Sea Lake	5,90804667	9,47409139	2,08184621	3,97333943	7,50454664	3,40824137	-8,61820749

Figure 3.10 Hu Moments class-based result table for EuroSAT images

3.2.4. Histogram of Oriented Gradients

Another traditional feature extraction method used in this study is the histogram of oriented gradients (HOG). This method has been proposed by Dalal et al [23]. It has been widely used in computer vision to solve different object detection problems. HOG counts the occurrence of gradient orientations of a portion of the image. After using the grayscale data in the HOG method, I collected HOG values created for the single-color band in a one-dimensional array, as I did in the LBP. As a result, I obtain a 64-dimensional feature vector for each image in the dataset. The example of the HOG can be seen on the Figure 3.11.

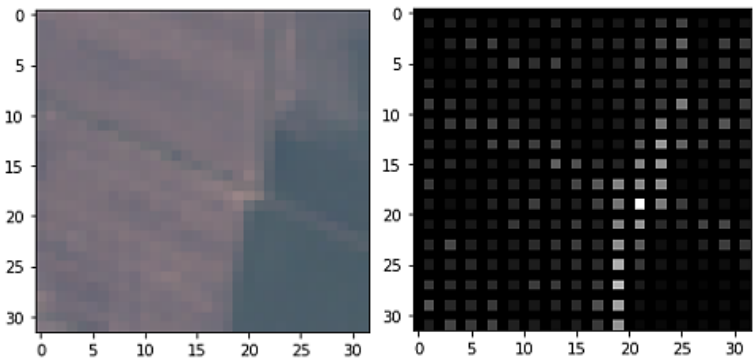


Figure 3.11 Histogram of oriented gradients result matrix(left) the original image(right)

3.2.5. Local Binary Patterns

Another traditional feature extraction method used in this study is the local binary patterns proposed by Ojaja et al [24]. This method has been initially developed for texture analysis. As in other methods, it has also been applied to land use classification studies.

In general, LBP tags the neighboring pixel with a binary number. First, I am converting our images into grayscale. For each pixel(x) of image has a P neighbor that is near the selected pixel. Then, I select the center pixel

and set a threshold for neighbors. In final step, I am setting 1 if the neighbor value is equal or greater than the center pixel. These operations can be seen below. The G_C represents the central pixel intensity value and G_P represents the neighbor pixel intensity value.

$$LBP = (\sum_{p=0}^{P-1} s(G_p - G_c) \times 2^p)$$

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

In result, I have two LBP arrays for each image with image size of 32x32 pixels. Afterward, I combined these two LBP matrices one by one using the projection of the histogram. The histogram projection formula can be seen below. Hence, I obtained 64 features per image. The histogram results can be seen on Figure 3.12 and Figure 3.13.

$$HistProjection_{xy} = (\sum_{x,y=0}^{N} \sum_{x,y=0}^{N})$$

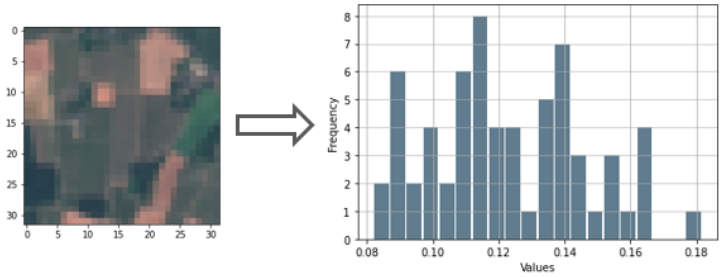


Figure 3.13 Original image on left and histogram projection of local binary pattern results of pasture image on right**.**

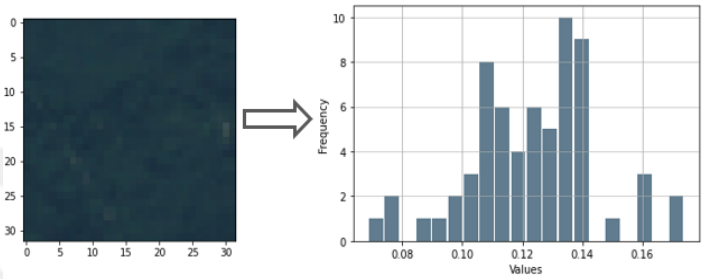


Figure 3.12 Original image on left and histogram projection of local binary pattern results of forest image on right**.**

3.2.6. Color Invariants

The sixth and final traditional feature extraction method used in this study is based on color information. Color information can help us extract information from an image and create unique features. The most important feature that distinguishes this technique from others is that it uses colored data instead of grayscale. In this study, I made use of the color invariants suggested and explained by [25]. As a result, I obtain a 64-dimensional feature vector for each image in the dataset. The example of each band color invariants can be seen on the Figure 3.14.

$$r = \frac{R}{R+G+B'}$$

$$g = \frac{G}{R+G+B'}$$

$$b = \frac{B}{R+G+B'}$$

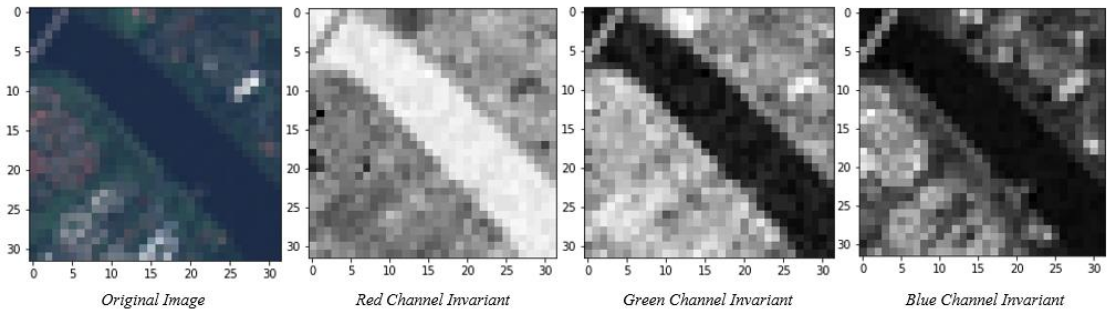


Figure 3.14 Color invariant results for each band of RGB

3.3. Proposed Feature Injection Method for Classification

I can partition a CNN model into two parts feature extraction and classification. The feature extraction part extracts a feature from a given source via successive filtering and nonlinear operations. The main advantage of this block is obtaining filter coefficients via training. Hence, an adaptive structure can be formed for the image set at hand. The classification part is composed of fully connected neural networks most of the time. Just before the fully connected layers, I have some weights that represent the possibilities of the classes. Thanks to the transfer learning method that I am using, I do not create the weights again and again, so the training time is drastically decreased.

CNN methods have been proved to be very powerful in solving computer vision problems. Their only drawback is the model size and number of parameters to be trained for a given problem. Although these issues do not cast a major problem for systems running on powerful servers or desktop computers, they become extremely important for resource-limited embedded systems such as the ones to be deployed on drones. Therefore, I propose injecting traditional features into the CNN models to improve their performance while keeping the overall model size as small as possible.

I first provide the visual representation of the proposed method in Figure 3.15. As can be seen in this figure, the CNN feature extraction method calculates its features. The selected traditional feature extraction methods extract their features. Then, these two feature sets are merged with the concatenating method of our framework Keras. Concatenate layer takes a list of tensors and returns a single tensor that is the merging of all inputs. The concatenated traditional and deep inputs are ready for Dense layers which are responsible for implementing activation function operations. It computes the dot product between given inputs and kernel and added bias value passes another dense layer till reaches the final step. The number of dense layers is changed by different test

cases and models. The formula of dense layers can be seen below.

$$\text{output} = Activation((input \times kernel) + bias)$$

The final step is classification, as I are using the 10-layer classification, I used "SoftMax" activation function. It is abnormalized exponential function and mostly used in the last layer of multiple output neural networks.

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

Just before the classification step of the CNN model. Hence, I inject the traditional features to the CNN model.

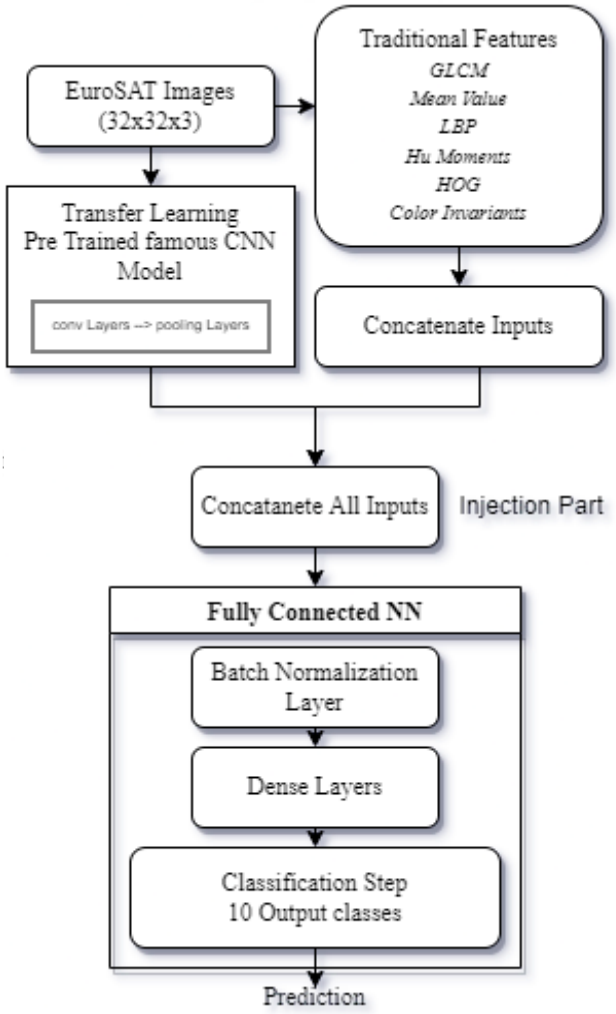


Figure 3.15 The Injecting Traditional Inputs to the CNN Models flowchart

4. EXPERIMENTAL STUDY

I conducted experiments to measure the performance of the proposed method in this section. The body of the methods are written in Python language to run on a Jupyter notebook. The hardware power comes from the Google Colaboratory environment. I have used Keras with TensorFlow framework in all tests. In these, the epoch number is set to 16 with batch size 64. In optimizer side "Adam" is used in training. For our object detection problem, instead of retraining the selected CNN models, I used pre-trained models with "ImageNet" weights using transfer learning method. I used the max pooling and batch normalization just before concatenating the traditional inputs. Each test has been repeated five times and the average values are listed in the following sections. The test code block can be accessed from GitHub [26].

To improve wide input feature extraction parts, the methods have been investigated and mentioned in following sub-chapters.

4.1. Dataset used in Experiments

The most basic resource required for tree detection and feeding our model is the dataset. I obtained the dataset that we will use in our wide and deep learning model in tree detection from various sources. I tested the proposed method on the EuroSAT, Planes and Ships in satellite images datasets, announced respectively [27,28,29] as a benchmark for land use and land cover classification tasks.

Many satellite images can be used in areas such as data detection from satellite images. One of them is the Sentinel satellite. Grovel, Kumar et al. Research by \cite{ship-sentinel} is an example of this.

4.1.1. EuroSAT Dataset

The EuroSAT dataset consists of 27 K RGB images with a resolution of 10 meters and hold labels with information of 10 different classes. These are annual crop, forest, herbaceous vegetation, highway, industrial, pasture, permanent crop, residential, river, and sea lake. I picked 80% of data for training and rest is used by tests. Hence, I have 27000 images that 21600 are used by training. I did not use the data augmentation methods in our tests, since these methods mostly aim to increase number of images randomly (rotating, adding noising and so on), I thought that the data I have would be sufficient since there may be differences in accuracy between the tests. The dataset

image samples can be seen on Figure 4.1.

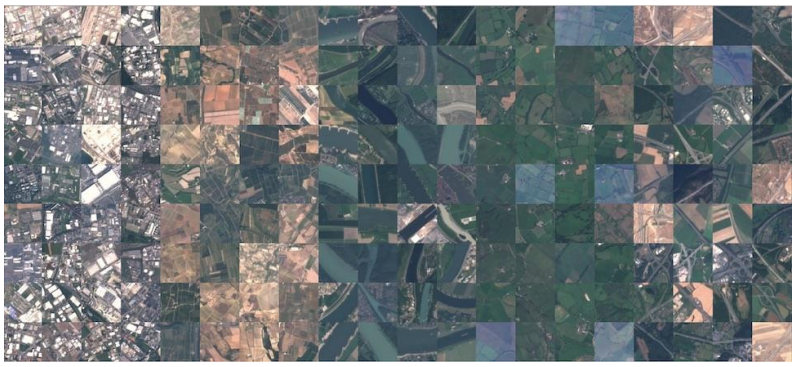


Figure 4.1 EuroSAT Data Samples

4.1.2. Ships in Satellite Imagery

Also, the datasets are known to be fragments from the Planet's "Open California" dataset obtained by the Planet satellite. The data sets consist of 3.7-meter resolution RGB images, and they are provided with ground truth data which hold labels for ships and airplanes. The obtained data Ire collected on the Kaggle website, the first of which was a different dataset. This dataset consists of satellite images collected from the Planet's Open California dataset. It contains 4000 pieces of 80x80 resolution 3-channel RGB images labeled with "ship" or "no ship" classification designed as binary classification. The dataset image samples can be seen on Figure 4.2.

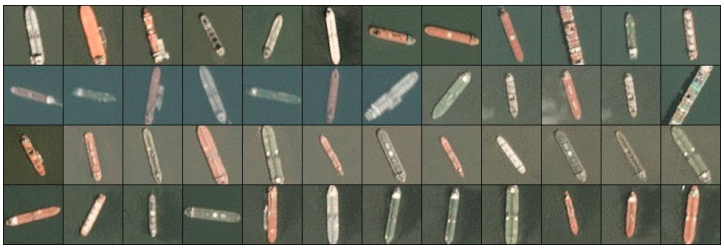


Figure 4.2 Ships in Satellite Imagery Data Samples

4.1.3. Planes in Satellite Imagery

As like in ships dataset, planes dataset is gathered from same source. So that the image properties are same. It has 4000 pieces of 20x20 resolution 3-channel RGB images labeled with "plane" or "no plane" classification designed as binary classification. The

dataset image samples can be seen on Figure 4.3.

Figure 4.3 Planes in Satellite Imagery Data Samples

4.1.4. Trees in Satellite Imagery

The trees dataset contains samples of the images from EuroSAT. Like in EuroSAT it contains images from Sentinel-2 satellite. So that the properties are same. It is gathered from Kaggle. It has 10400 pieces of 32x32 resolution 3-channel RGB images labeled with "trees" or "no trees" classification designed as binary classification. The dataset image samples are same as the EuroSAT images so that the samples can be seen on Figure 4.1.

4.2. Hardware and Software Requirements

In the experiments, I used the hardware from the Google Colaboratory environment. This environment, which includes GPU and TPU parallelization, was chosen for this research in terms of providing Python programming language and a Jupyter programming environment. The Google Colaboratory environment is an environment that helps researchers and students share the hardware power they need for free with the help of cloud technology. In this thesis, I advantage of the GPU power of this environment. These GPUs consist of Nvidia's K80 (GDDR6 12GB) or T4 (GDDR6 16GB), and the processing power reaches up to 8.1 TFLOPS. Thanks to the hardware specifications, it helps a lot during the training steps, due to the research “Performance Analysis of Google Colaboratory as a Tool for Accelerating Deep Learning Applications” [30].

The usage of the Google Colaboratory environment is simple. I used TensorFlow 2 versions and the Keras API on the framework side. The environment fully supports the Keras. In dataset implementation, I used Google Drive and upload whole images there. First, to detect the GPU specifications in the environment I can type “!nvidia-smi -L” command. If Colaboratory allocated a GPU for me, I can continue working. I need to get the dataset from google drive to synchronize and authenticate by

"drive.mount('/content/drive/')" command. When the credentials are correct, I need to extract the traditional features from the given dataset. After, I can feed the CNN models with the same data and concatenate the wide inputs in the final.

4.3. EuroSAT Tests

4.3.1. The Effect on using traditional features without CNN Models

In this test, I would like to test the effect of traditional features without using any CNN model. In order to perform that, I have connected my features directly to the fully connected neural network. The maximum accuracy is around 60% which is quite good for using the system without power of CNN. The results can be seen on Figure 4.4.

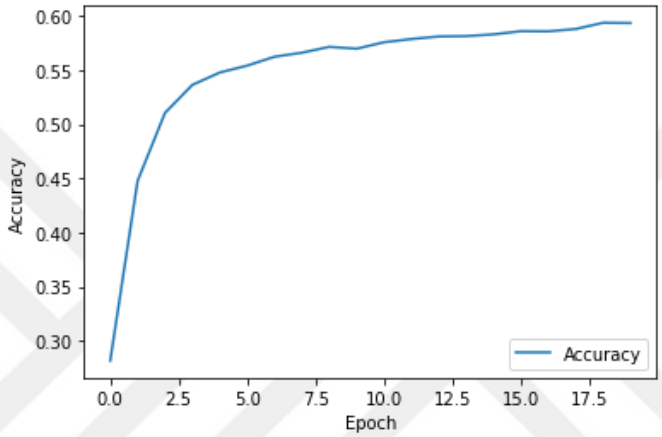


Figure 4.4 The accuracy graph of traditional inputs using with fully connected NN.

4.3.2. The Effect on the SqueezeNet Model

In order to measure the performance of our proposed method, we first used the SqueezeNet model as a base. I injected different traditional feature sets to the model and

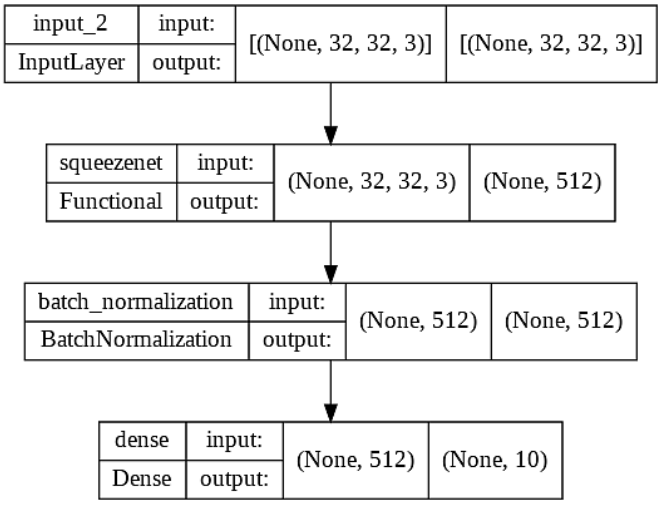


Figure 4.5 The model summary of using SqueezeNet solution

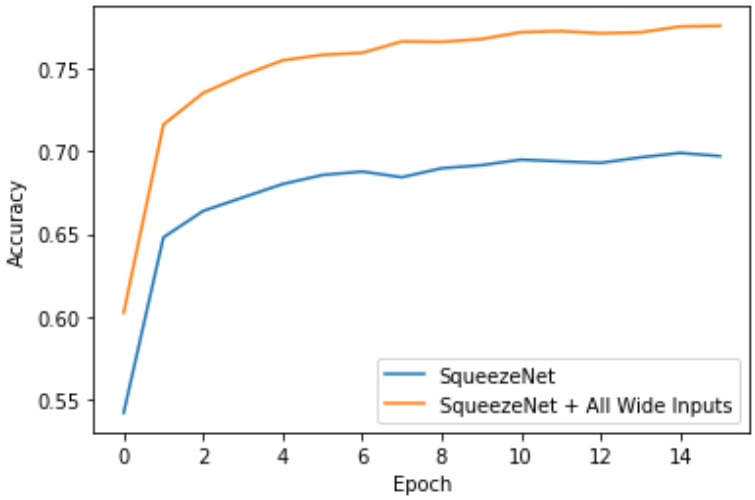


Figure 4.6 Accuracy graph comparison of SqueezeNet

tabulated the obtained results in Table 4.1. As can be seen in this table, the accuracy improvement reaches 8.4% when all traditional features are injected to the SqueezeNet model. This improvement comes with additional 350 parameters to be calculated as traditional features. Hence, the increase in the memory size in case of space allocated by model becomes 66KB.

Table 4.1 The effect of adding wide features to the SqueezeNet model.

Test Scenario	Maximum Accuracy
SqueezeNet	0.6778
SqueezeNet + GLCM features	0.6656
SqueezeNet + color invariants	0.7056
SqueezeNet + Hu moments + HoG + LBP + sample mean	0.7403
SqueezeNet + all traditional features	**0.7618**

4.3.3. The Effect on the MobileNetV2 Model

I next used the MobileNetV2 model as the base to measure the performance of the proposed method. As in the previous section, I injected different traditional feature sets to the model. The tests are retried for EuroSAT and Plane, Ship datasets. The obtained results in Table 4.2. As can be seen in this table, the accuracy improvement reaches 2.27% when all traditional features are injected to the MobileNetV2 model. This improvement comes with additional 350 parameters to be calculated as traditional features. Hence, the increase in the memory size in case of space allocated by model becomes 40KB.

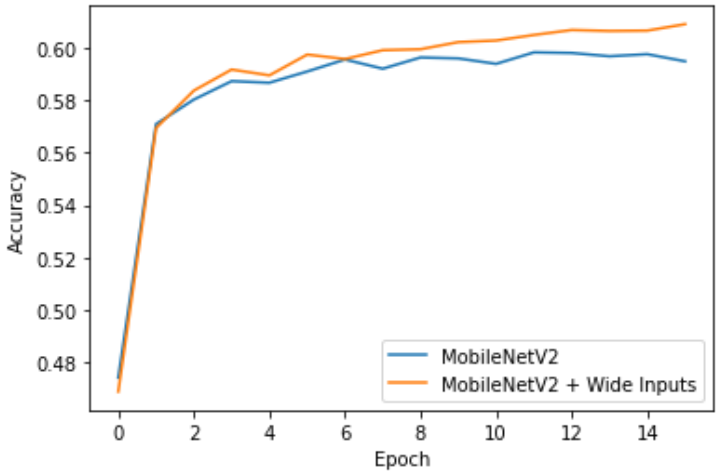


Figure 4.7 Accuracy graph comparison of MobileNetV2

Table 4.2 The effect of adding wide features to the MobileNetV2 model.

Test Scenario	Maximum Accuracy
MobileNetV2	0.6062
MobileNetV2 + GLCM features	0.6101
MobileNetV2 + color invariants	0.6160
MobileNetV2 + Hu moments + HoG + LBP + sample mean	0.6208
MobileNetV2 + all extracted features	**0.6289**

4.3.4. The Effect on the ShuffleNetV2 Model

Third, we used the ShuffleNetV2 model as the base to measure the performance of the proposed method. As in the previous section, we injected different traditional feature sets to the model and tabulated the obtained results in Table 4.3. As can be seen in this table, the accuracy improvement reaches 4.96% when all traditional features are injected to the ShuffleNetV2 model. This improvement comes with additional 350 parameters to be calculated as traditional features. Hence, the increase in the memory size in case of space allocated by model becomes 100KB.

Table 4.3 The effect of adding wide features to the ShuffleNetV2 model.

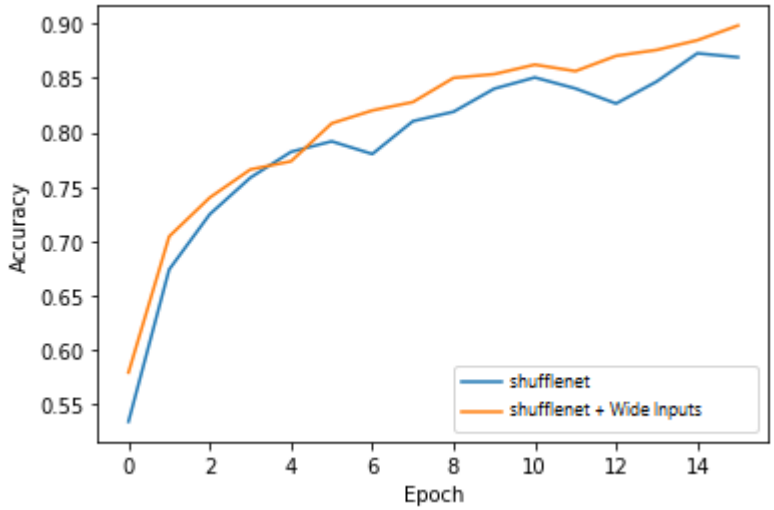


Figure 4.8 Accuracy graph comparison of ShuffleNet

Test Scenario	Maximum Accuracy
ShuffleNetV2	0.8502
ShuffleNetV2 + GLCM features	0.8623
ShuffleNetV2 + color invariants	0.8862
ShuffleNetV2 + Hu moments + HoG + LBP + sample mean	0.8971
ShuffleNetV2 + all extracted features	**0.8998**

4.3.5. The Effect on the VGG16 Model

Next, we used the VGG16 model as a base to measure the performance of our proposed method. As in the previous section, we injected different traditional feature sets to the model and tabulated the obtained results in Table 4.4. As can be seen in this table, the accuracy improvement is only 0.0014% when all traditional features are injected to the VGG16 model. This improvement comes with additional 350 parameters to be calculated as traditional features. Hence, the increase in the memory size in case of space allocated by model becomes 35KB.

Table 4.4 The effect of adding wide features to the VGG16 model.

Test Scenario	Maximum Accuracy
VGG16	0.9091
VGG16 + GLCM features	0.8995
VGG16 + color invariants	0.9018
VGG16 + Hu moments + HoG + LBP + sample mean	0.9003
VGG16 + all extracted features	**0.9105**

4.3.6. The Effect on the ResNet50V2 Model

Finally, we used the ResNet50V2 model as a base in order to measure the performance of our proposed method. As in the previous section, we injected different traditional feature sets to the model and tabulated the obtained results in Table 4.5. As can be seen in this table, the accuracy improvement is only 0.0158% when all traditional features are injected to the ResNet50V2 model. This improvement comes with additional 350 parameters to be calculated as traditional features.

Table 4.5 The effect of adding wide features to the ResNet50V2 model.

Test Scenario	Maximum Accuracy
ResNet50V2 Only	0.6720
ResNet50V2 + GLCM	0.6802
ResNet50V2 + Color Invariants	0.6826
ResNet50V2 + Hu Moments + HoG + LBP + Mean Value	0.6804
ResNet50V2 + All extracted Features	**0.6878**

Hence, the increase in the memory size in case of space allocated by model becomes 20KB.

4.4. Ship-Plane Dataset Results

4.4.1. The Effect on the MobileNetV2 Model

I have tested the MobileNetV2 model with the combined ship and plane dataset. So that the output of the last neural network layer will have 3 elements ship, plane, or none. As I used in EuroSAT tests, I have used transfer learning and the same fully connected layers as before. The accuracy is increased up to 2% and the result can be seen on the Figure 4.9.

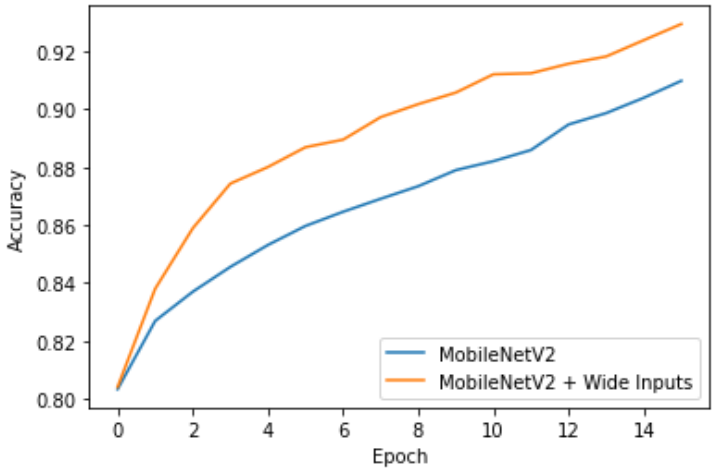


Figure 4.9 Ship & Plane Dataset 8 Accuracy graph comparison of MobileNetV2

4.4.2. The Effect on the VGG16 Model

Next, the other test is based on VGG16. All steps are same with the MobileNetV2 test version, and the results are negligible. Because I could not get important performance difference on this model and results can be seen on Figure 4.10.

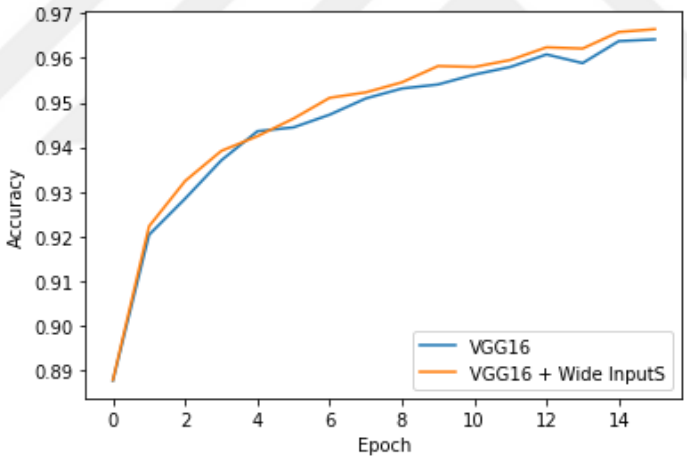


Figure 4.10 Ship & Plane Dataset Accuracy graph comparison of VGG16

4.5. Trees Dataset Results

As In scope of this thesis, the tree detection and classification take an important place in this work. In this section I will summarize the tests and their results. As we are using the same samples with EuroSAT in this section, I would like to test the only models that`s accuracy increased by wide inputs. So that, SqueezeNet, ShuffleNetV2 and MobileNetV2 models are tested.

4.5.1. The Effect on the SqueezeNet Model

When I tested my tree classification task in SqueezeNet model with the same configuration before, I got 3% performance increase in total. The train accuracy graph can be seen on Figure 4.11.

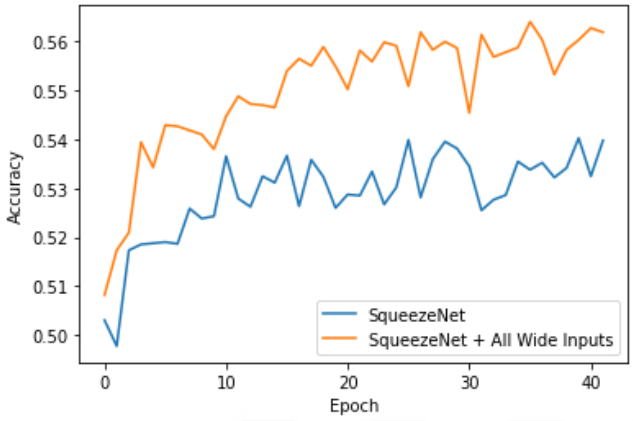


Figure 4.11 Tree Dataset Accuracy graph comparison of SqueezeNet

4.5.2. The Effect on the MobileNetV2 Model

When I tested my tree classification task in MobileNetV2 model with the same configuration before, I could not see notable performance increase in total. The accuracy graph can be seen on Figure 4.12.

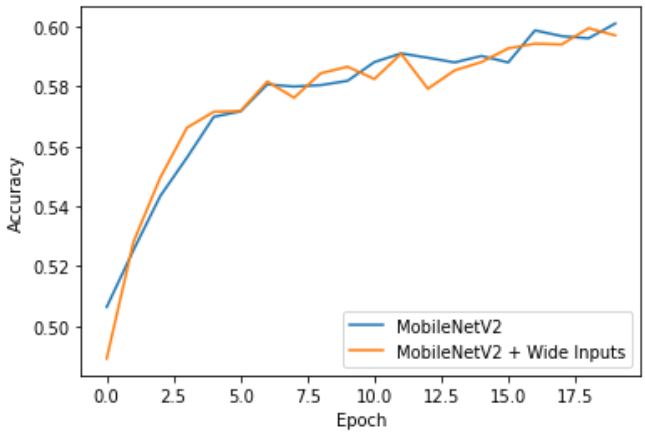


Figure 4.12 Tree Dataset Accuracy graph comparison of MobileNetV2

4.5.3. The Effect on the ShuffleNetV2 Model

When I tested my tree classification task in ShuffleNetV2 model with the same configuration before, I could not see notable performance increase in total.

4.5.4. The Effect on the ResNet50V2 Model

When I tested my tree classification task in ResNet50V2 model with the same configuration before, I could not see notable performance increase in total. The train accuracy graph comparison can be seen on Figure 4.13.

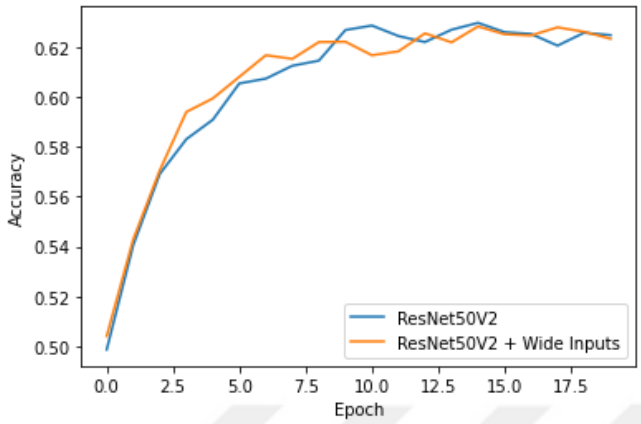


Figure 4.13 Tree Dataset Accuracy graph comparison of ResNet50V2

4.5.5. The Effect on the VGG16 Model

When I tested my tree classification task in VGG16 model with the same configuration before, I could not see notable performance increase in total. Conversely, the added features reduced the accuracy of the model. The train accuracy graph comparison can be seen on Figure 4.14.

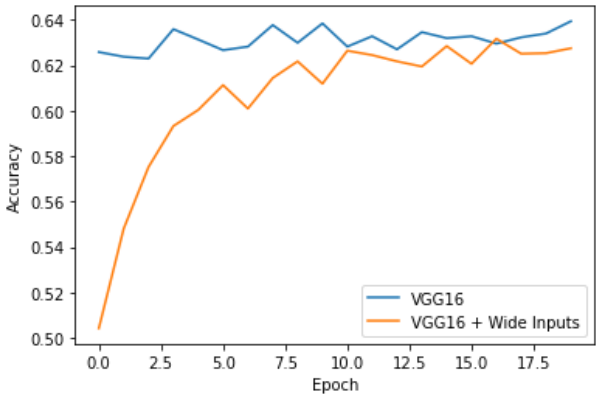


Figure 4.14 Tree Dataset Accuracy graph comparison of VGG16

4.6. Tree Classification Results

Owing to the EuroSAT and the Planes-Ships datasets, have proven the injecting

traditional inputs solution has better effects in terms of accuracy especially in relatively smaller models. In this section I would like to summarize the tree detection results by giving examples. As we know, the EuroSAT dataset has the classes like forest, pasture etc. that might cover the tree detection scenarios. As I experienced on previous test cases, the SqueezeNet gave the best accuracy increase when I inject the wide inputs. In Table 4.6 shows that the SqueezeNet model improved 13% in terms of precision in forest detection.

Table 4.6 SqueezeNet Precision, Recall, F1 Score comparison with and without injecting the wide inputs.

Model Name	Precision	Recall	F1 Score
SqueezeNet	0.737752	0.893543	0.808208
SqueezeNet + Wide Inputs	0.862543	0.788069	0.823626

4.7. Overview of the Injection Performance

I can summarize all the experiments performed in the previous section as follows. When the CNN model size is small, injecting traditional features may increase the accuracy significantly as in the SqueezeNet, MobileNetV2, and ShuffleNetV2. Among these, improvement in the SqueezeNet model reaches up to 8.4%. But the relationship between the parameter numbers and accuracy improvement is not parallel. For example, although MobileNetV2 has lower parameters than ShuffleNetV2, the ShuffleNetV2 performance increase with traditional inputs are higher than the MobileNetV2. Due to the tests, we have understood that the MobileNetV2 is working better while feeding with high resolute images. So that there is an exception exist, while comparing the models on their parameter complexity. The improvement precision, recall, f1 score table can be seen on the Table 4.7. This is a significant improvement. To note here, added parameters to the model (as injecting traditional features) have the size increase of 66KB. This addition can be accepted when the accuracy improvement is considered. On the other hand, the accuracy improvement in the complex models such as VGG16 and ResNet50V2 are almost negligible. One main reason for this result is that these complex models extract almost all information from the image due to the excessive parameter size. Hence, injecting traditional features to such models increase accuracy marginally. I summarized the accuracy improvements for the selected models in Figure 4.14. It means that, the theory of figure further justifies our observations such that the accuracy of relatively

simple CNN models can be improved significantly by injecting traditional features to the model.

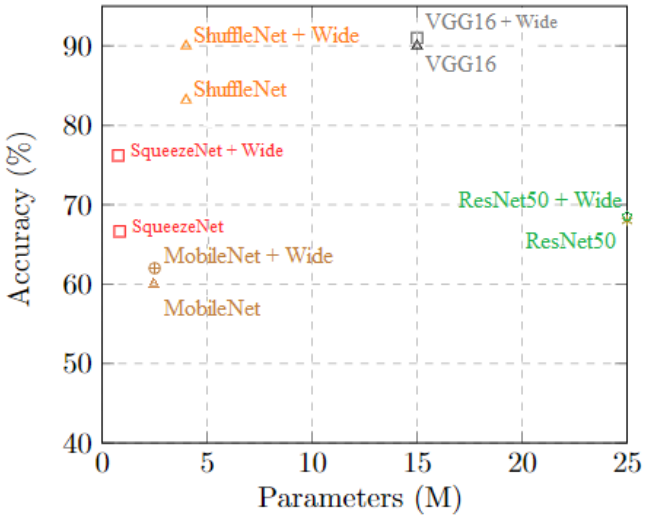


Figure 4.15 Model size versus accuracy comparison for the selected CNN models traditional feature injection method.

Table 4.7 SqueezeNet Precision, Recall, F1 Score comparison difference with and without injecting wide inputs.

EuroSAT Classes	**Precision**	**Recall**	**F1-score**
Annual Crop	0,157496	-0,119933	-0,021414
Forest (Tree)	**0,243933**	**-0,425829**	**-0,174638**
Herbaceous Vegetation	-0,226739	0,211966	-0,082757
Highway	-0,087020	0,332661	0,051487
Industrial	0,156140	-0,044944	0,053856
Pasture	0,028343	0,083969	0,056015
Permanent Crop	0,064218	0,032258	0,043604
Residential	0,148629	-0,189280	-0,036911
River	0,192269	-0,236162	-0,070831
Sea Lake	-0,075221	0,045608	-0,012304

5. CONCLUSION

In this thesis, I presented an injecting traditional feature into the CNN model behavior on land classification and performance effects. Also, I have shown that the proposed method affects, especially in low-powered UAVs. For testing the proposed model, using the models such as MobileNetV2, VGG16, ShuffleNetV2, SqueezeNet, ResNet50V2 which are frequently used in the literature. I also include mean values, gray-level co-occurrence matrices, Hu moments, local binary patterns, histogram-oriented gradients, and color invariants from traditional computer vision feature extraction methods as representing the to be concatenated part of the system. After these features are concatenated into the model, the performance increase is observed distinctly in smaller models.

I tested the proposed method on EuroSAT, Plane-Ship, and Tree datasets. EuroSAT tests are 10 classes of land object classification tasks. Plane-Ship has 3 classes of classification tasks. And the Tree dataset is based on binary classification. I compared the classification results of the proposed system with CNN-based models. Extensive testing shows that the proposed wide and deep learning method has significant performance improvement dataset independent compared to selected CNN-based models, especially in relatively smaller and less complex models. For example, the performance increases of models such as SqueezeNet, MobileNetV2, ShuffleNetV2 were much higher than the other famous models such as ResNet, and VGG16, a noticeable performance increase could not be observed in these models. The models have a higher number of parameters and complexity. Since the deep layers are covering and extracting many features based on image, manually feeding extracted traditional features does not increase the model performance.

My experimental results show that it is possible to increase the performance which is lost by using compressing or simplifying the models by adding traditional features in the deep learning phase. Especially in devices with low storage, RAM, and processing power such as embedded devices or mobile devices, the required performance increase is achieved a lot with very little parameter increase. (350 parameters) This gives us an idea that the proposed wide and deep learning-based land classification method may offer opportunities to solve complex object classification tasks on remote sensing images with higher accuracy.

In summary, land classification is taking an important place in the remote sensing field. In this thesis, I introduced my proposed injecting traditional inputs to the CNN models

that helps feature extraction steps and increases the final accuracy. I hope that the performance increase obtained especially in models with low complexity will contribute to the studies in the remote sensing field.

REFERENCES

[1] Nowak, D. J., Hoehn, R., & Crane, D. E. (2007). Oxygen production by urban trees in the United States. *Arboriculture & Urban Forestry. 33 (3): 220-226., 33*(3).

[2] Ramsfield, T. D., Bentz, B. J., Faccoli, M., Jactel, H., & Brockerhoff, E. G. (2016). Forest health in a changing world: effects of globalization and climate change on forest insect and pathogen impacts. Forestry, 89(3), 245-252.

[3] Ozturk, M., Gucel, S., Kucuk, M., & Sakcali, S. (2010). Forest diversity, climate change and forest fires in the Mediterranean region of Turkey. Journal of environmental Biology, 31(1), 1.

[4] Zhu, X. X., Tuia, D., Mou, L., Xia, G. S., Zhang, L., Xu, F., & Fraundorfer, F. (2017). Deep learning in remote sensing: A comprehensive review and list of resources. IEEE Geoscience and Remote Sensing Magazine, 5(4), 8-36.

[5] Ma, L., Liu, Y., Zhang, X., Ye, Y., Yin, G., & Johnson, B. A. (2019). Deep learning in remote sensing applications: A meta-analysis and review. ISPRS journal of photogrammetry and remote sensing, 152, 166-177.

[6] Zhai, L., Sang, H., Gao, Y., An, F., Zhang, Y., & Wei, S. (2015). A new approach for mapping regional land cover and the application of this approach in Australia. Remote Sensing Letters, 6(4), 267-275.

[7] Zhou, Z. H., & Feng, J. (2017, August). Deep Forest: Towards An Alternative to Deep Neural Networks. In IJCAI (pp. 3553-3559).

[8] Weinstein, B. G., Marconi, S., Bohlman, S., Zare, A., & White, E. (2019). Individual tree-crown detection in RGB imagery using semi-supervised deep learning neural networks. Remote Sensing, 11(11), 1309.

[9] Diez, Y., Kentsch, S., Fukuda, M., Caceres, M. L. L., Moritake, K., & Cabezas, M. (2021). Deep learning in forestry using uav-acquired rgb data: A practical review. Remote Sensing, 13(14), 2837.

[10] Iqbal, M. S., Ali, H., Tran, S. N., & Iqbal, T. (2021). Coconut trees detection and segmentation in aerial imagery using mask region-based convolution neural network. IET Computer Vision, 15(6), 428-439.

[11] Cheng, H. T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., ... & Shah, H. (2016, September). Wide & deep learning for recommender systems. In Proceedings of

the 1st workshop on deep learning for recommender systems (pp. 7-10).

[12] Koutrika, G. (2018, May). Modern recommender systems: from computing matrices to thinking with neurons. In Proceedings of the 2018 International Conference on Management of Data (pp. 1651-1654).

[13] Cheng, H., 2016. Wide & Deep Learning: Better Together with TensorFlow. [online] Google AI Blog. Available at: <https://ai.googleblog.com/2016/06/wide-deep-learning-better-together-with.html> [Accessed 9 July 2022].

[14] Jbene, M., El Maliani, A. D., & El Hassouni, M. (2019, November). Fusion of convolutional neural network and statistical features for texture classification. In 2019 International Conference on Wireless Networks and Mobile Communications (WINCOM) (pp. 1-4). IEEE.

[15] Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. Insights into imaging, 9(4), 611-629.

[16] Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. arXiv preprint arXiv:1602.07360.

[17] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4510-4520).

[18] Ma, N., Zhang, X., Zheng, H. T., & Sun, J. (2018). Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Proceedings of the European conference on computer vision (ECCV) (pp. 116-131).

[19] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[20] He, K., Zhang, X., Ren, S., & Sun, J. (2016, October). Identity mappings in deep residual networks. In European conference on computer vision (pp. 630-645). Springer, Cham.

[21] Haralick, R. M., Shanmugam, K., & Dinstein, I. H. (1973). Textural features for image classification. IEEE Transactions on systems, man, and cybernetics, (6), 610-621.

[22] Hu, M. K. (1962). Visual pattern recognition by moment invariants. IRE transactions on information theory, 8(2), 179-187.

[23] Dalal, N., & Triggs, B. (2005, June). Histograms of oriented gradients for human detection. In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05) (Vol. 1, pp. 886-893). Ieee.

[24] Ojala, T., Pietikäinen, M., & Harwood, D. (1996). A comparative study of texture measures with classification based on featured distributions. Pattern recognition, 29(1), 51-59.

[25] Geusebroek, J. M., Van den Boomgaard, R., Smeulders, A. W. M., & Geerts, H. (2001). Color invariance. IEEE Transactions on Pattern analysis and machine intelligence, 23(12), 1338-1350.

[26] Aksoy, M. Ç. Object Recognition in EuroSAT Satellite Images using Wide and Deep Learning Models (Version 1.0) [Computer software]. https://github.com/mcagriaksoy

[27] Helber, P., Bischke, B., Dengel, A., & Borth, D. (2019). Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 12(7), 2217-2226.

[28] rhammell (2018). Planes in Satellite Imagery [Dataset]. https://www.kaggle.com/rhammell/planesnet

[29] rhammell (2018). Ships in Satellite Imagery [Dataset]. https://www.kaggle.com/rhammell/ships-in-satellite-imagery

[30] Carneiro, T., Da Nóbrega, R. V. M., Nepomuceno, T., Bian, G. B., De Albuquerque, V. H. C., & Reboucas Filho, P. P. (2018). Performance analysis of google colaboratory as a tool for accelerating deep learning applications. IEEE Access, 6, 61677-61685.

CURRICULUM VITAE

Mehmet Çağrı Aksoy

EDUCATION

2019-Present MSc in Electrical and Electronics Engineering

Marmara University, Istanbul (Turkey)

Language of Education: English

2014-2019 BSc in Electrical and Electronics Engineering

Anadolu University, Eskişehir (Turkey)

Language of Education: English

WORK EXPERIENCE

2019 July - 2021 July – Embedded Software Engineer – iQmine Turkey GmbH, Istanbul (Turkey)

2021 July - Present – Software Engineer – Huawei R&D Center, Istanbul (Turkey)

SKILLS

Programming Languages

➢ C++, C, Python, Embedded C

Software/Library/API

➢ OpenCV, QT, PyQt, AUTOSAR