American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)
ISSN (Print) 2313-4410, ISSN (Online) 2313-4402

http://asrjetsjournal.org/

# Binary Image Classification Through an Optimal Topology for Convolutional Neural Networks

Arman Sarraf*

*Department of Electrical and Computer Engineering Islamic Azad University North Tehran Branch*
*Email: armanhs13@gmail.com*

**Abstract**

Deep learning applications in computer vision have expanded over the past years. Image classification, which is the fundamental of most algorithms in the field, has been of interest to many researchers. Advances in hierarchical feature extractions using convolutional neural networks as one of the deep learning architectures have enabled experts to improve the performance of classification significantly. In this work, an optimal binary classifier to distinguish cat and dog images was explored where various architectures and parameters were employed to achieve the best results. To design our experiment, we considered the architectures with two and three convolutional layers using two input image size when models were trained with and without Dropout against an identical dataset. The analysis demonstrated that an accuracy rate of 99.26% for the testing dataset was achieved from a three-layer model with an input image size of 32x32 with Dropout. The classification report of any models was produced to explore other metrics such as precision, recall, and F1-score, and they were aligned with the accuracy rates as this experiment was a balanced data situation.

***Keywords:*** Deep Learning; Computer Vision; Image Classification; Convolutional Neural Networks.

## 1. Introduction

Machine learning, through its simplest definition [1], refers to a branch of computer science where we train our computer with a certain amount of data to do a determined purpose, which is called training or learning procedure. In essence, in machine learning, we can train our software or hardware-based applications, so-called machine learning models, which would predict what we target. Machine learning has several subdivisions that we will discuss in the deep learning section. In principle, researchers consider deep learning as a new branch in machine learning, which was used to be obsolete.

---

* Corresponding author.

The reason for its failure to date was because of the less powerful hardware to process, or abstruse architecture did not exist or were not as advanced as today. Deep learning models training requires a robust parallel processing infrastructure, and advances in computational science and especially hardware have enabled machine learning practitioners to implement deep learning models. Deep Learning has a similar architecture to the human brain (neural network) that is designed with a large number of neurons and deep, capable layers that can be learned and enhance performance [2]. Each neuron processes a particular stage of the learning at a particular point in inputs. Classical neural network architectures could only extract limited features from imaging data in computer vision problems; therefore, explicit feature extraction was required before training neural network models [3]. To resolve such a problem, a three - dimensional neural network (3D Convolutional Neural Network) was designed by scientists, which consists of three components of weight, height, and depth in each layer that could solve the mentioned problem in this architecture by extracting more in-depth features from data. Convolutional Neural Networks so-called CNNs consist of a large number of hidden layers, in which each layer is containing a large number of neurons [4]. Each neuron of a given layer connects with all the neurons in the previous layer but works individually. Each convolutional neural network referring to CNN consists of three components: 1- Convolutional layer (Conv), 2- Pooling layer, 3- Fully Connected layer. Each Conv layer is in charge of breaking down pixels into smaller processable blocks consisting of that are convolved by several filters in each iteration for learning and activating neurons in the higher layers. The pooling layer, which is commonly used between Conv layers, reduces the number of parameters and calculations, which usually downsamples input data (i.e., feature maps) by the size of two. Fully Connected Layer, as its name suggests, is a layer where all its neurons connect with the previous layer, and this is the main bridge between this layer and the Conv layer. In the Conv layer, the neurons are only in touch with a local area of inputs, but in a fully connected layer, all neurons from the previous layer connect with the current layer. Image classification has a broad application in computer vision problems and refers to a set of machine learning or deep learning methods to classify images in a given dataset into different categories. The purpose of classification can either be a yes/no decision so-called binary classification where a given item is predicted as class 0 or class 1, or several classes referring to multi-class classification. Classification is one of the main bases of object recognition and class prediction of each item. These days, a variety of algorithms and methods have been developed to increase productivity and enhance existing classification performance and have shown significant progress [5].

## 2. Related works

Deruja and his colleagues demonstrated in their work that by using two classes (dog/cat) and 8000 samples data for training and 2000 samples to test the model, they succeeded to build a three-layer neural network and they achieved to an efficiency which equals to 0.83% and loss with 0.39% [6]. Sarraf and his colleagues researched image classification usage in MRI and FMRI for brain function, in which the team succeeded in achieving some implemental results [7]. Shanmukhi and his colleagues showed that team members designed a CNN model with 2000 samples of dogs and cats for training and 400 samples for testing, which prints predicted writing of “input photographed is a dog/cat” [8]. Shiddieqy and his colleagues designed several neural network models with two and five layers and set 90 times for epochs, which used 25000 samples for training (12500 were dogs, and the rest were cats) and 12500 samples for testing. Finally, the yield for the two-layer model accuracy was 0.556%, and the loss was 11.409, and the five-layer model accuracy was 0.77%, and the loss value was 11.869 [9]. Liu

and his colleagues in their study, showed that by building a six-layer neural network and using a Kaggle dataset, they could achieve maximum efficiency of 0.94% [10]. Sarraf showed that neural networks could also be useful besides image classification and object detection for text recognition, and he reported some proper results [11]. Cengil and his colleagues conducted a research study and showed in their paper with a similar purpose; an eight-layer model was designed which used 10000 samples for training and 5000 samples for testing, which could achieve 92.1% [12]. Bandhu and his colleagues used a dataset of 20,000 samples of dogs and cats, 12,000 of which were used for training and 8,000 for testing; they succeeded in developing a model with 1.0 accuracy [13]. Jajodia and his colleagues developed a model with 90.1% accuracy with using 8000 samples of dogs and cats for training and 2000 samples for testing the model [14]. Naidoo and his colleagues completed a study using CNNs and, the objective of their project was similar to what we completed in our current study, and the team achieved 80% accuracy with a model they have trained [15]. Sarraf and his colleagues designed and developed a CNN model for recognizing hair color, which had a similar structure and was related to what is done in this article, and his achievement was 99% accuracy [16].

## 3. Data and Methods

The first step of our deep learning model development pipeline was to select a dataset addressing the target of our study. As it is known, two datasets are required for training and testing steps to perform. Therefore, to address this requirement to build the models with the highest efficiency, and eventually, to test and evaluate the models, a large number of different images with specified labels were required to identify the picture belongs to which class (dog or cat).

### *3.1. Data Preparation*

The best way to provide the required datasets was to use data publicly available. 'Kaggle' offers a vast number of datasets that are allowed to be used for research purposes. It is clear that the more variety and number of samples exist, the better the training and the higher the accuracy could be. Searching for 'dogs vs. cats' in Kaggle showed that several datasets exist. The metadata, including several samples for various datasets, were explored, and we found that the best dataset is the one that contains 25000 samples (12500 for dogs/cats) with tagged labels.

### *3.2. Data Normalization*

The metadata of this dataset showed that they had different sizes, colors, and measures. Such data were unable directly used for machine learning model development and fed into our convolutional neural network models that had been designed; therefore, the data required preprocessing steps. In this study, the CNN models were designed based on two categories: the models that process inputs size of 32 * 32 and models with the input size of 64 * 64. In the preprocessing steps, we converted 25,000 samples once to 32 * 32 images and next to 64 * 64. Then, the images stored in the repository were loaded into the memory and vectorized, and the results were saved using the Numpy Python package with "npy" extensions. The format of data was required to be in PNG for a deep learning pipeline enabling our framework to read the data and pass them to the CNN layers and

networks. If such data conversion were not performed, the CNN models would crop the input images so that the models would result in very high loss values in the training step and also a very low accuracy. The reason behind that is a significant amount of information would be removed by cropping the images by then networks if the original image size was used. It should also be considered that each sample has a value between 0 - 255. For the sake of model development, images were normalized to a range of [0,1] to achieve the highest performance. To avoid data generation repeatedly, two normalized datasets of 32 * 32 and 64 * 64 for training the CNN models with the "npy" extension were created and stored. After the models were trained, a set of samples was required to test the models and measure the performance of classification, including accuracy rates [17]. It should be noted that: 1- test data samples should also have labels so that when they are predicted by the models, their labels determine whether or not the prediction is correct; otherwise, the labeling should be done manually. 2- The samples that were used in the training dataset cannot be used in the testing, and so we need new samples. As mentioned, the higher number of samples are used in training, the more developed models are robust, so from the Kaggle the best dataset with the largest number of photos labels was found and used in this work. After we downloaded the testing dataset, the same preprocessing steps as training, including converting images to 32 * 32 and 64 * 64 and saving the results with "npy" extensions, were applied to the testing dataset. In this study, the dataset, including 1011 dogs', and 1012 cats' samples (a total of 2023 images) was used in this project.

### *3.3. Architecture and Method*

The training step was done by calling and using existing modules (which has a task to train the models). The training process included designing, defining, and setting custom parameters that the models need to make by running the training step. We built several models with a different number of layers and filters to see which model performs better. Therefore, several existing modules were required to build CNN models. Various libraries such as Keras, Tensorflow, TFLearn, Colab, and others have been implemented to build convolutional neural network models, and our project was designed using TFLearn, which has a similar structure with Keras. We utilized modules, including OpenCV, tqdm, TFLearn, Glob, TensorFlow, and imported them to our Jupyter Notebook to complete this study. As it is known, the architecture of the convolutional neural network so-called CNN is composed of three major parts: Conv Layer, Max_pool Layer, and Fully_Connected Layer. Convolutional layer so-called Conv as the first layer is used to scan incoming inputs and distinguish the weight, height, and depth of each image. In other words, this layer is the core of the neural networks, which has hyperparameters that include the main processing, number of filters, the size of each filter, and a task to activate the ReLU layer [18]. The second component is Max_pool, which is the layer that is located between the Conv layers and is in charge of reducing the numbers of network parameters while retaining the maximum values of input [19]. Fully Connected layer (FC) is the third component, and this layer is the last layer of CNN architecture to specify the class probability, and as the layer name suggests, each neuron in this layer is associated with all the neurons in the previous layer [20]. We built the models using several different parameters. In this study, each model was created with and without Dropout. Dropout is a systematic method invented by Google Research to reduce over-connectivity in neural networks by preventing complex synchronization of training data [21]. Dropout is a very efficient and practical method that significantly improves the accuracy and training of the models.

### *3.4. Models*

In this study, for the sake of model comparison, two-layer and three-layer CNN models were designed. In two-layer models, the number of filters was considered as 32 and 64 with a dimension of 3x3. But for three-layer CNN models, the number of filters was increased to 128 with a dimension of 3x3, and at last in CNN architecture, the Fully Connected layer was designed. Based on what was described in Dropout, we implemented this capability as a logical flag where True refers to a model with Dropout, and False refers to without Dropout. TensorFlow has provided a monitoring capability so-called TensorBoard for its users to create the graph of model training with a simple command line to show what is happening at each step. Table 1 illustrate eight CNN models created in this study of dog and cat binary classification.

**Table 1:** Models information and structure designed and built in this project

| Model | Layer | Input img size | Number of Filters | Dropout |
|---|---|---|---|---|
| 1 | 2 | 32*32 | 32,64 | TRUE |
| 2 | 2 | 32*32 | 32,64 | FALSE |
| 3 | 2 | 64*64 | 32,64 | TRUE |
| 4 | 2 | 64*64 | 32,64 | FALSE |
| 5 | 3 | 32*32 | 32,64,128 | TRUE |
| 6 | 3 | 32*32 | 32,64,128 | FALSE |
| 7 | 3 | 64*64 | 32,64,128 | TRUE |
| 8 | 3 | 64*64 | 32,64,128 | FALSE |

After designing the models and roadmap of the project by defining architectures and parameters, we started building the models. We specified the number of epochs to 30 times, as the literature recommended, and this number seems to be sufficient with the number of samples and architecture that has been designed. The higher number of epochs does not necessarily produce a better result for training and accuracy and does not necessarily reduce the number of loss positions. The model's configuration is found at this GitHub address: https://github.com/armansarraf/CNN-Classification-DogsVsCats. Executing the script found in our GitHub allowed training to begin and the models and results to be stored on the disk. Upon the model constructions, information such as efficiency and loss appeared at TensorBoard until the model was completed and stored. After training, eight different convolutional neural network models were developed and trained, where four of them receive an input size of 64 * 64, and the other four models accept an input size of 32 * 32. With a few simple command lines, models could be called, and we could run them for testing purposes. To do this, first, we needed to ensure using the correct image size to input the models whether we want to test 32 * 32 or 64 * 64. After loading the desired models, the task was to test them and evaluate the accuracy of the prediction. For doing this step, we need the test datasets, which were converted appropriately. The first attempt was to use our test data, which had 2023 samples with 64 * 64 image sizes, and assigned it to the input variable defined as 'X,' and then convert them into matrix arrays to be acceptable for the models which Figure 1 explains further.

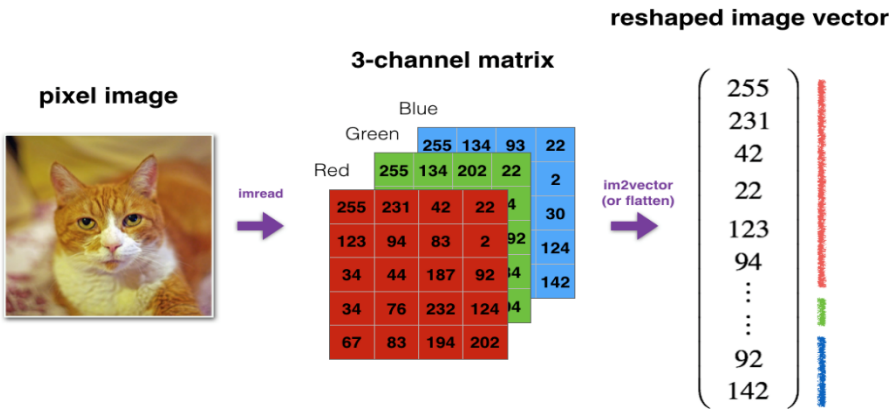
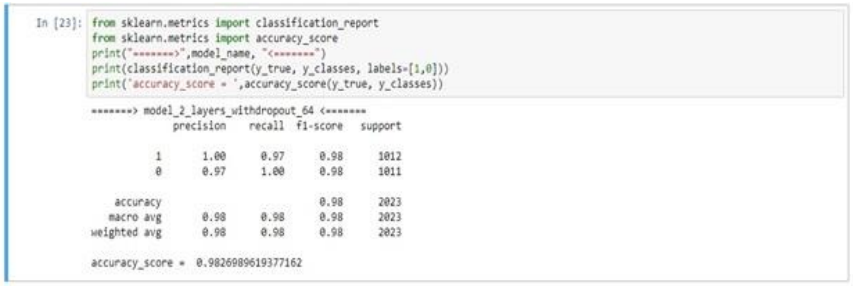



**Figure 1:** Converting inputs to an understandable matrix number for the neural network

Then, the feature vector created in the previous step was to be passed to the model for prediction purposes. Anaconda provides all these features required to users, and all the steps, including prediction and verification of the prediction, can be done with just a few simple command lines. So, in the last block, we implemented the prediction and verification codes to produce the predicted labels for samples. the prediction module produced some information such as accuracy rates, loss, number of samples, and exactitude of each specific dog/cat prediction that could be individually visible and is shown in Figure 2.



```
In [23]: from sklearn.metrics import classification_report
         from sklearn.metrics import accuracy_score
         print("=======>",model_name, "<=======")
         print(classification_report(y_true, y_classes, labels=[1,0]))
         print('accuracy_score = ',accuracy_score(y_true, y_classes))

=======> model_2_layers_withdropout_64 <=======
              precision    recall  f1-score   support

           1       1.00      0.97      0.98      1012
           0       0.97      1.00      0.98      1011

    accuracy                           0.98      2023
   macro avg       0.98      0.98      0.98      2023
weighted avg       0.98      0.98      0.98      2023

accuracy_score =  0.9826989619377162
```

**Figure 2:** Efficiency, model performance

## 4. Results & Discussion

In this step, the predicted labels obtained from our trained models were evaluated. As mentioned, TensorFlow facilitates the evaluation process by building and storing figures of neural network training steps for all the models developed. The results were stored by TensorFlow in a new folder so-called “log” on the disk where all models and details are found. To load the results created by TensorBoard, we used anaconda commands in the terminal, such as (tensorboard -logdir = “your path file location”), which is your file location in the folder that TensorFlow created. By calling such commands, an IP address was generated where users could access the results. TensorFlow has created the localhost to store the results for visualization purposes. Each model performance included two prominent separate figures: 1- accuracy, 2- Loss. We illustrated each metric separately and individually. In Figure 3, as seen, the models’ information such as efficiency during training and testing, and in Figure 4 accuracy of training of the models are shown.

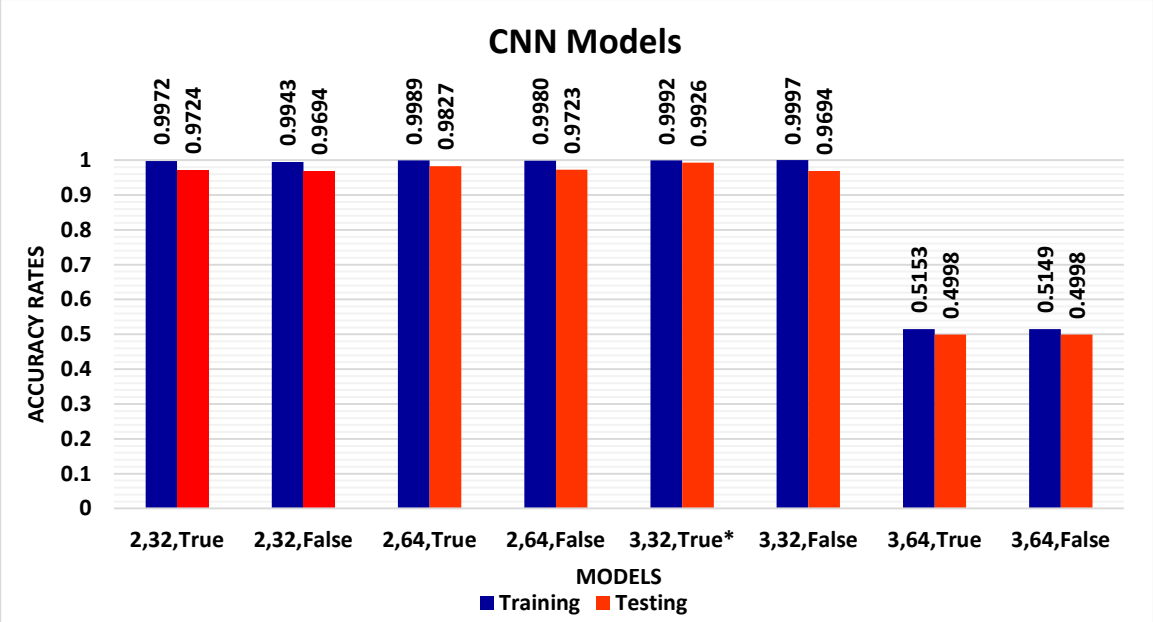



**Figure 3:** Models Information such as efficiency during training and testing, Number of Layers, Size of Images, with dropout=True, Without Dropout=False are visible

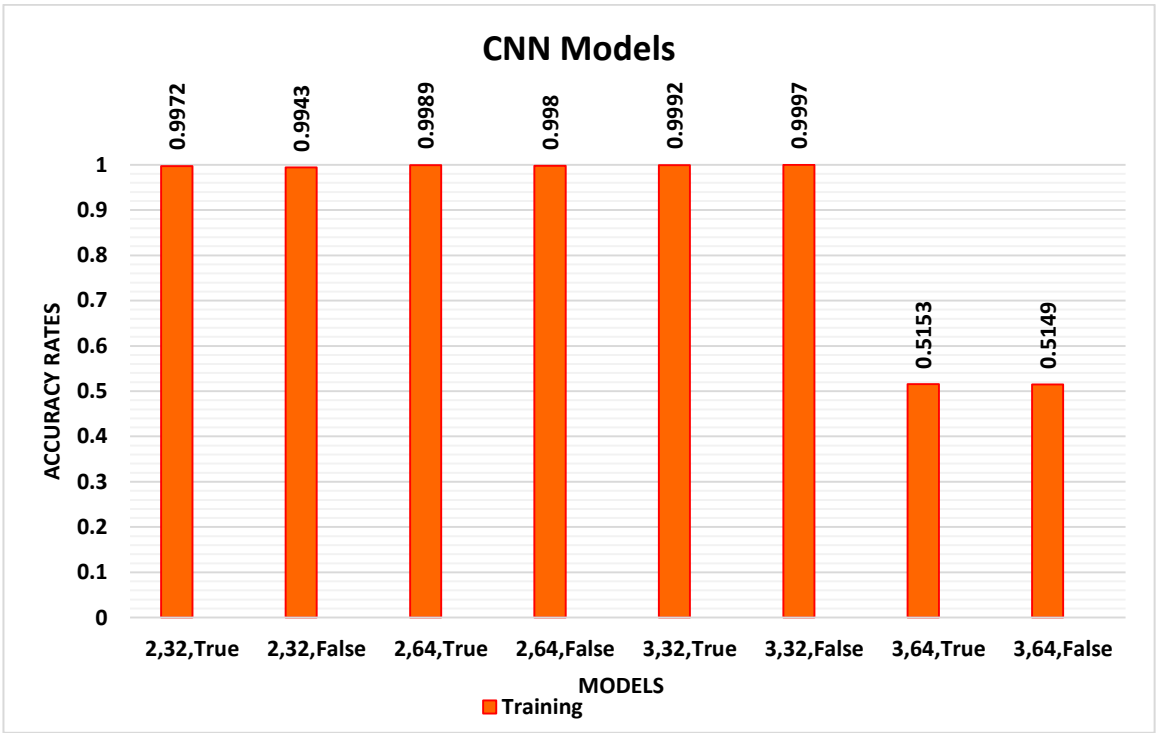



**Figure 4:** Accuracy of training, Number of Layers, Size of Images, with dropout=True, Without Dropout=False are shown

As shown in the figures above, using the dropout technique, the model's performance was improved to some extent, and loss was reduced. Therefore, using and applying the dropout technique during training models enabled us to increase the performance of the models. The information was extracted from each model, Figure

3, and the models were compared. The results indicated that the best model with the highest performance refers to a three-layer model of 32x32 input size with Dropout, and it was marked with an Asterix. Figure 5 and Figure 6 illustrated the best model details.

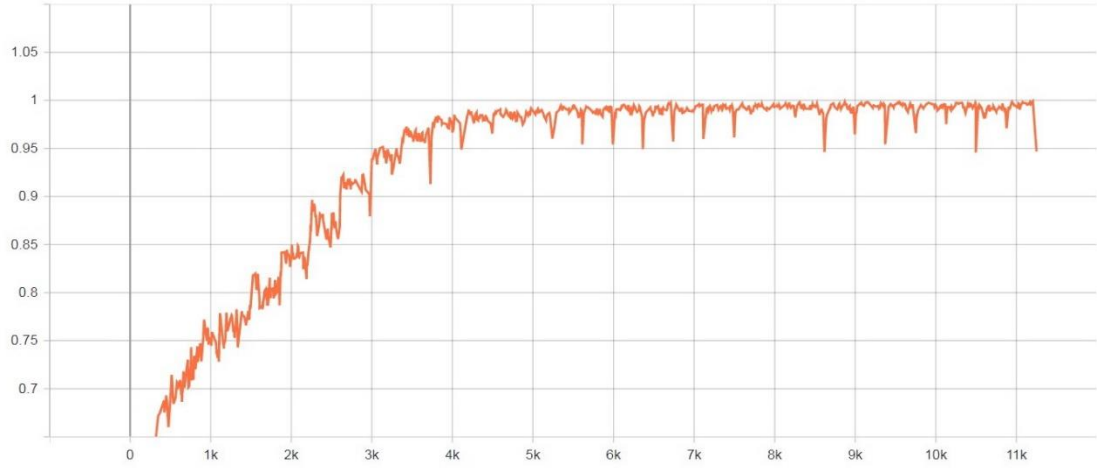



**Figure 5:** This Figure shows the accuracy rate for the three-layer model with a size of 32 x 32 and using Dropout. As is shown in Figure, after the 11.250K training step performance reached 0.992% and outcomes are ideal and acceptable

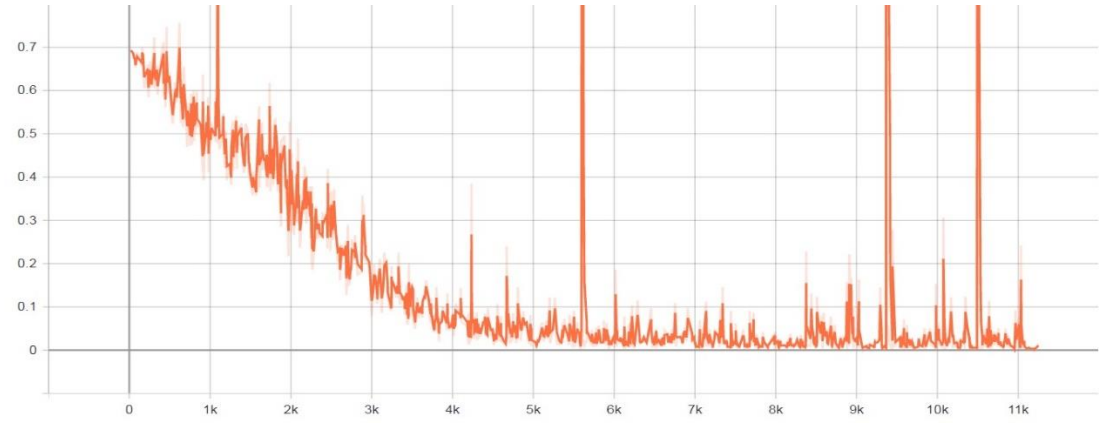



**Figure 6:** This Figure shows the loss of the model, which in step 11.250K training is completed, and the Loss value of all samples is negligible. Batch_size was undefined during this project, so its number was used by default settings, and the cause of 3 jumps explains by this accident which is a normal process and has no negative impact

Last but not least, Table 2 provided the classification report for the testing dataset for the “three-layer” model with dropout and input size of 32 * 32.

**Table 2:** The table above shows more details information about three-layer convnet with an input size of 32*32 and using Dropout

| Information | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| **Cat** | 0.99 | 1 | 0.99 | 1012 |
| **Dog** | 1 | 0.99 | 0.99 | 1011 |
| **Accuracy** | - | - | 0.99 | 2023 |
| **Macro Avg** | 0.99 | 0.99 | 0.99 | 2023 |
| **Weighted Avg** | 0.99 | 0.99 | 0.99 | 2023 |
| **Accuracy Score** | 0.992585 | - | - | - |

We explored the performance of the last two models, which were three-layer with an input size of 64 * 64 with and without using the dropout technique in Figure 3 and Figure 4. Those models were not well performed compared to the rest of the models. The reason behind such a problem could be because of the number of samples for training. When CNN models are built deeper with more layers, more samples are needed for the training process. For example, when we train a two-layer convolutional neural network with an input size of 64 * 64 with 25,000 samples, we need to increase the number of samples. Also, by adding the third layer to the architecture, which increases the number of filters, more data are required for training purposes. Otherwise, the so-called "Underfitting" issue occurs, and the CNN models are not well trained, and their performance is drastically reduced. "Underfitting" occurs when we provide an insufficient number of samples to train a model, and CNN models are unable to be adequately trained. More details about the efficiency and loss of the two mentioned models are discussed in Figure 7 and Figure 8.

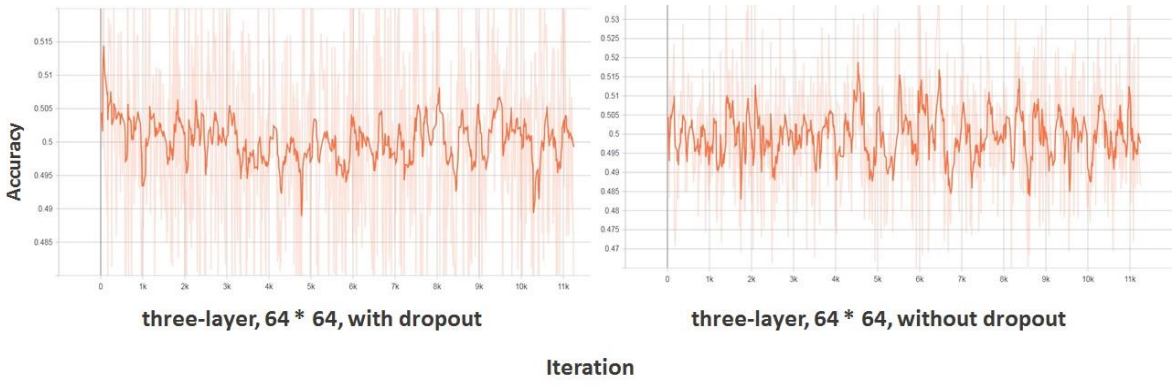



**Figure 7:** As it is shown, the accuracy of both three-layer CNNs could not improve during all the iterations, which, as mentioned, such an occurrence happens when insufficient samples are used for training procedure, and underfitting occurs [22]

As it is evident, neither of both mentioned models are acceptable and have a deficient value of accuracy. But the impact and differences of using the Dropout technique or not as it was said are shown in Figure 8.

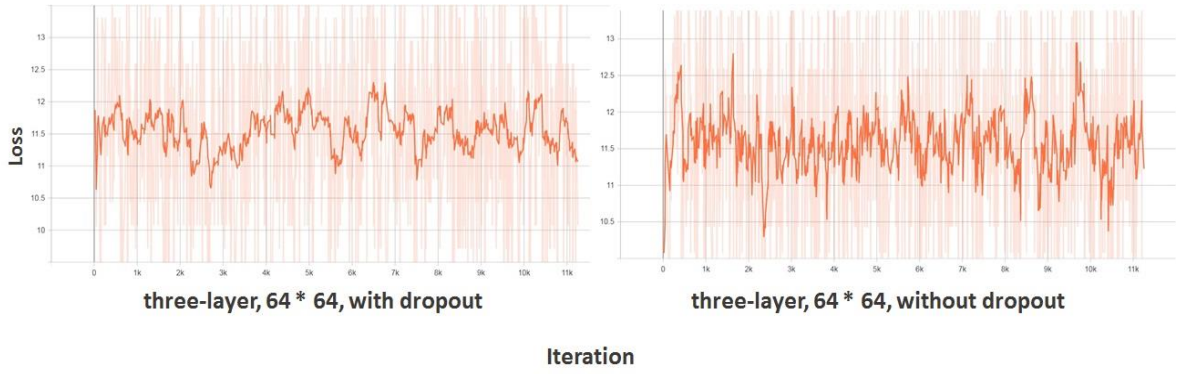



**Figure 8:** without considering that the models were not trained as we expected, it is shown that using Dropout could control and reduce the value of loss even in underfitting.

Therefore, according to the equable training dataset that was used and the different models which were built in this project and the results, it is shown with adding more layers to the CNN models, performance could improve or reduce. As you can see by adding the third layer and considering an input size of 32 * 32, we achieved the highest accuracy and performance, but with training, the same model with raising the input size to 64 * 64 the models with the lowest accuracy were built. So besides the numbers of layers and the number of samples, architecture, and designing of a CNN model plays an important role, and the most important and necessary thing to get the highest performance from a model is to set the best determination for parameters and design an architecture which could be fit with existing datasets.

## 5. Conclusion

Binary classification of dog and cat images was performed using various CNNs models along with different input image sizes. Also, the impact of the dropout technique was explored in the model development to see whether it improves the performance of classification. The results showed that the best performance of binary classification was achieved from a three-layer model with Dropout and an input image size of 32 x 32. However, the three-layer model showed a low performance with 64 x 64 input size, which suggests that the number of samples used for training was insufficient. Also, the best model achieved can be utilized as a feature extractor in other experiments. For future work, a broader comparison between CNN models to explore the impact of different parameters can be performed.

## Acknowledgment

I would like to express my gratitude towards Dr. Mina Hashemi Nik, Dr. Amir Hatami, Dr. Seyyed Ehsan Zadkhosh, Dr. Ali G., and Dr. Saman Sarraf for extending their help and support in this study.

## References

[1]. Domingos, P., A few useful things to know about machine learning. Communications of the ACM, 2012. 55(10): p. 78-87.

[2]. Sarraf, S., Binary Image Segmentation Using Classification Methods: Support Vector Machines, Artificial Neural Networks and K th Nearest Neighbours. International Journal of Computer (IJC), 2017. 24(1): p. 56-79.

[3]. Shaikh, F., 10 advanced deep learning architectures data scientists should know. 2017.

[4]. Gu, J., et al., Recent advances in convolutional neural networks. Pattern Recognition, 2018. 77: p. 354-377.

[5]. Chen, Y., et al., Deep learning-based classification of hyperspectral data. IEEE Journal of Selected topics in applied earth observations and remote sensing, 2014. 7(6): p. 2094-2107.

[6]. [6] Dureja, A. and P. Pahwa, Analysis of non-linear activation functions for classification tasks using convolutional neural networks. Recent Patents on Computer Science, 2019. 12(3): p. 156-161.

[7]. Sarraf, S., et al. Brain network extraction from probabilistic ICA using functional Magnetic Resonance Images and advanced template matching techniques. in 2014 IEEE 27th Canadian Conference on Electrical and Computer Engineering (CCECE). 2014. IEEE.

[8]. Shanmukhi, M., et al., Convolutional neural network for supervised image classification. International Journal of Pure and Applied Mathematics, 2018. 119(14): p. 77-83.

[9]. Shiddieqy, H.A., F.I. Hariadi, and T. Adiono. Implementation of deep-learning based image classification on single board computer. in 2017 International Symposium on Electronics and Smart Devices (ISESD). 2017. IEEE.

[10]. Liu, B., Y. Liu, and K. Zhou, Image classification for dogs and cats. 2014.

[11]. Sarraf, S. French Word Recognition Through a Quick Survey on Recurrent Neural Networks Using Long-Short Term Memory RNN-LSTM. American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS), 39, no. 1 (2018): 250-267.

[12]. Cengil, E., A. Çinar, and Z. Güler. A GPU-based convolutional neural network approach for image classification. in 2017 International Artificial Intelligence and Data Processing Symposium (IDAP). 2017. IEEE.

[13]. Bandhu, A. and S.S. Roy. Classifying multi-category images using deep learning: A convolutional neural network model. in 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT). 2017. IEEE.

[14]. Jajodia, T. and P. Garg, Image Classification–Cat and Dog Images. Image, 2019. 6(12).

[15]. Naidoo, K., O. Konan, and L. Schwartzkopff. Hardware Accelerated Convolutional Neural Networks. in Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS).

[16]. Sarraf, S., Hair color classification in face recognition using machine learning algorithms. American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS), 2016. 26(3): p. 317-334.

[17]. Sarraf, S., C. Saverino, and A.M. Golestani. A robust and adaptive decision-making algorithm for detecting brain networks using functional mri within the spatial and frequency domain. in 2016 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI). 2016. IEEE.

[18]. Sarraf, S. and G. Tofighi. Deep learning-based pipeline to recognize Alzheimer's disease using fMRI data. in 2016 Future Technologies Conference (FTC). 2016. IEEE.

[19]. Sarraf, S., et al., MCADNNet: Recognizing Stages of Cognitive Impairment Through Efficient

Convolutional fMRI and MRI Neural Network Topology Models. IEEE Access, 2019. 7: p. 155584-155600.

[20]. [20] Sarraf, S., G. Tofighi, and A.s.D.N. Initiative, DeepAD: Alzheimer's disease classification via deep convolutional neural networks using MRI and fMRI. BioRxiv, 2016: p. 070441.

[21]. Hinton, G.E., et al., Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580, 2012.

[22]. Gavrilov, A.D., et al., Preventing model overfitting and underfitting in convolutional neural networks. International Journal of Software Science and Computational Intelligence (IJSSCI), 2018. 10(4): p. 19-28.