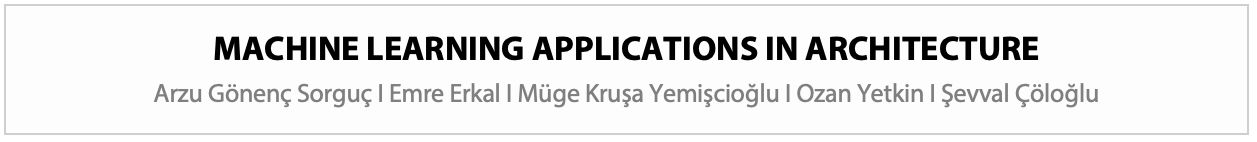




**2021-2022 SPRING**

****

**FINAL REPORT**

|  |
| --- |
| **ORIGAMI GRADING** |

**ŞEYMA NUR ÖZ**

* Clear definition of the problem

Origami, the Japanese art of folding paper into decorative shapes and figures, has difficulty levels. For beginners, starting from the difficult ones is compelling and discouraging, while professionals generally prefer complex configurations. The problem is deciding whether an image is an origami and, if it is, determining its difficulty level.

* The relation of the problem with data file & Interpretation of data file

Data is collected from oriwiki and giladorigami origami databases and classified as easy, medium, and hard by D. Ma, G. Friedlan, and MM. Krell (OrigamiSet1.0: Two New Datasets for Origami Classification and Difficulty Estimation, D Ma, G Friedland, MM Krell (2018), In Proceedings of Origami Science Maths Education, 7OSME, Oxford UK).

There are two sections in the data file: origami/no origami used for phase1 and difficulty levels ( for phase2)

To use the photographs in a CNN algorithm, they were resized to 100 x 100 px.

As the last step of data preprocessing, files are split to train (70%), validation (20%), and test (~10%) datasets.

* Expected outcome based on data file and problem

The expectation from the model is to determine the difficulty level of origami images if it is an origami.

* Clear definition of the model

The model has Convolutional Neural Network(CNN) architecture, which is implemented with TensorFlow, Keras, open-source libraries for machine learning models / neural network implementations (Keras could run on CPU and GPU)

The model has two distinct parts since the problem is solved in 2 phases. Model1 is trained with the first dataset(origami-no-origami) and tested in the first phase. Then Model2 is trained with the second dataset (difficulty levels).

Firstly, the datasets were downloaded, preprocessed, shuffled, and split into training, validation, and test datasets. Then with batching, data within the dataset data files become ready.

For model implantation, the data pass through several layers;(multiple times)

1. Convolutional layer: Convolution is basically dot product multiplication of matrices. In CNN, convolutional operations act on the filter/kernels and image.

The convolutional layer only looks at specific parts of an image.

1. MaxPooling Layer: Pooling is downscaling the image taken from the previous layers, reducing its dimensionality and allowing for the assumption about features.

Max pooling is done by applying a max filter to non-overlapping sub-regions of the initial representation.

1. Flatten Layer: Flattening the input image data into a 1D array.

1. Dense Layer: Dense layers evaluate all the pixels and use that information to generate some output.

Softmax Activation Function:  After combining features to create a model, there should be an activation function like softmax for classifying the outputs.

After that, we have a” fully connected layer”.

Lastly, to train the model, we have to compile it with loss function, optimization algorithm, and learning algorithm.

While training, the model was exposed to all data number of times (called epoch).

The trained model was tested with the dataset.

Results are represented by accuracy, f1 score, confusion matrix etc.

* What is the relation of your model with the problem and your expectancy?

The expectancy from the model is to understand the difficulty level of origami from its picture. Since expectancy is image recognition, CNN model best fits the problem.

* In depth explanation of model qualities (model performance criteria)

The model's number of epochs and color mode I think will affect the result. Furthermore, to compare results and understand the relations, I try different batch sizes and image sizes as parameters in the model.

1. Color mode: Color modes are the settings designers use to show colors consistently across devices and materials.

*(RGB and Grayscale)*

1. Number of epochs:A numeric value indicates the number of time a network has been exposed to all the data points within a training dataset.

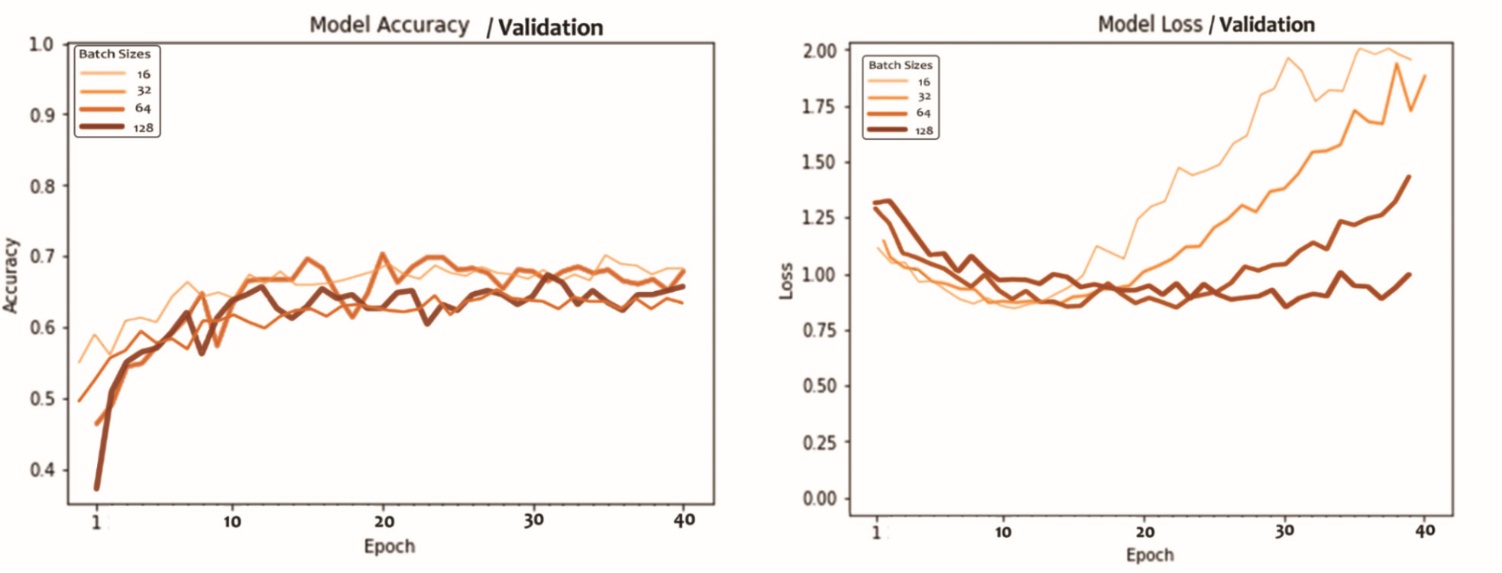
*(0-40)*

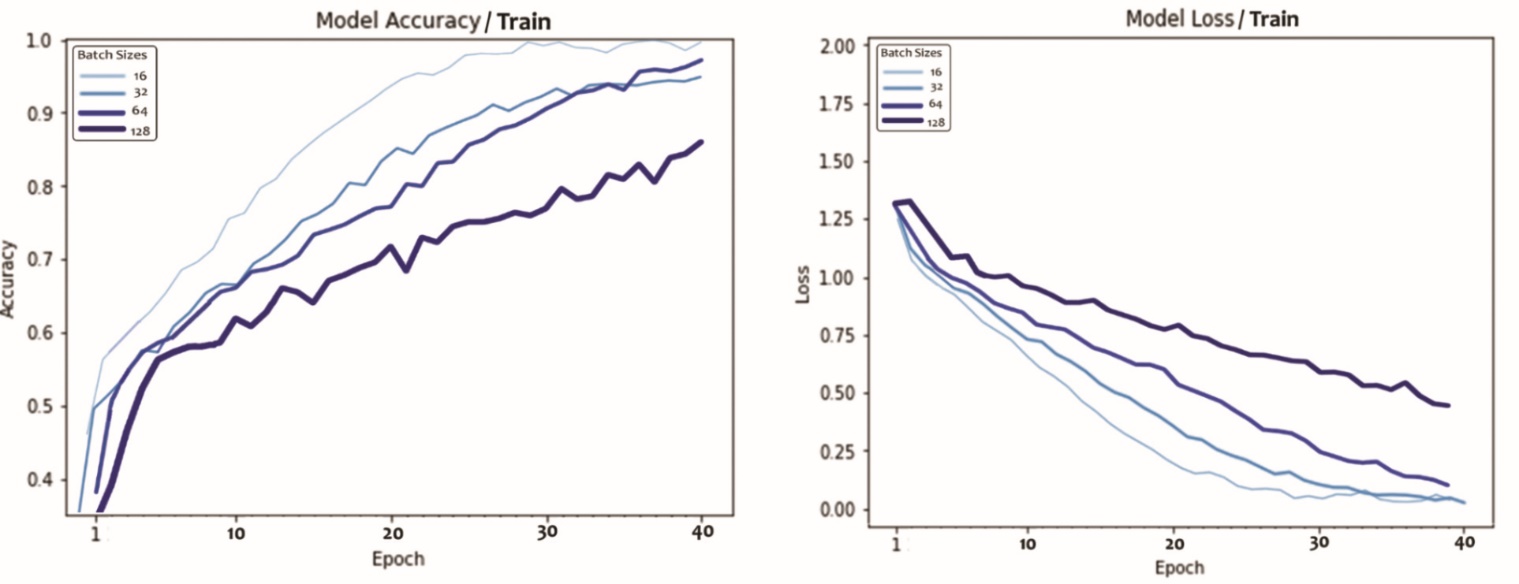
1. Batch size: Number of divisions in the dataset to batches. Then a single batch is presented to the network at each iteration.

*(16, 32, 64,128)*

1. Image size: Size of the image (pixels)

*(100 ,200)*

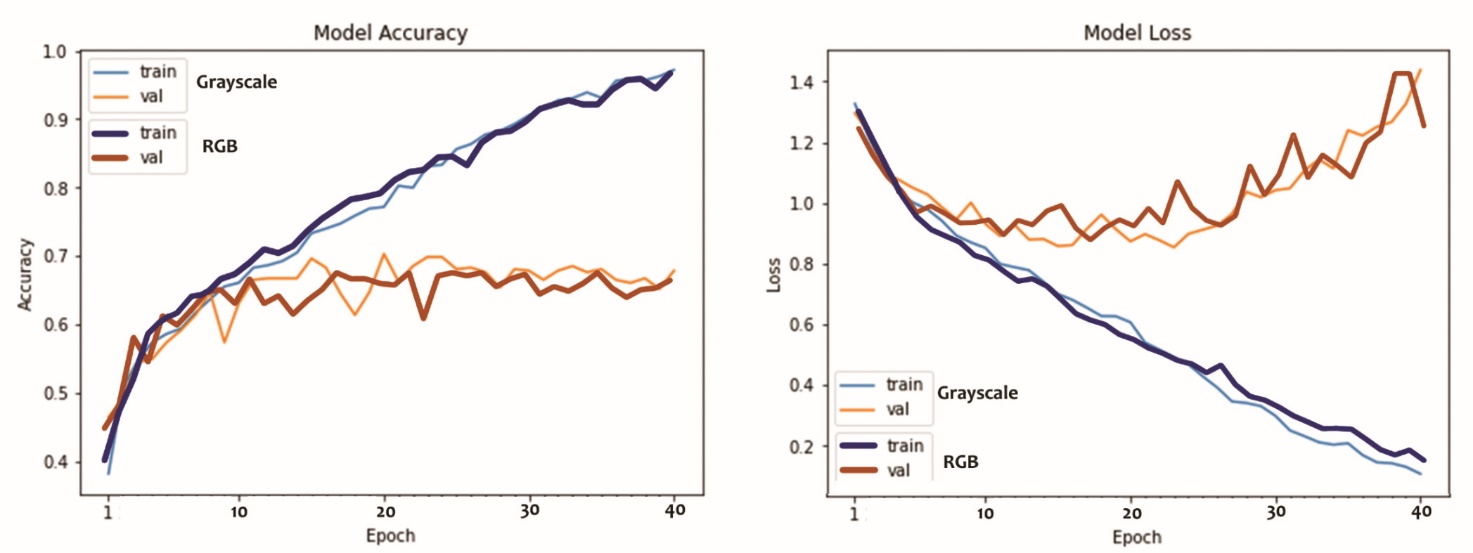
In the first step, the color mode is taken as grayscale, image size is taken as 100 x 100 px, while the number of epoch and batch size are variable.



In the training dataset, accuracy increases, model loss decreases with the number of epochs, and smaller batch sizes give better results.

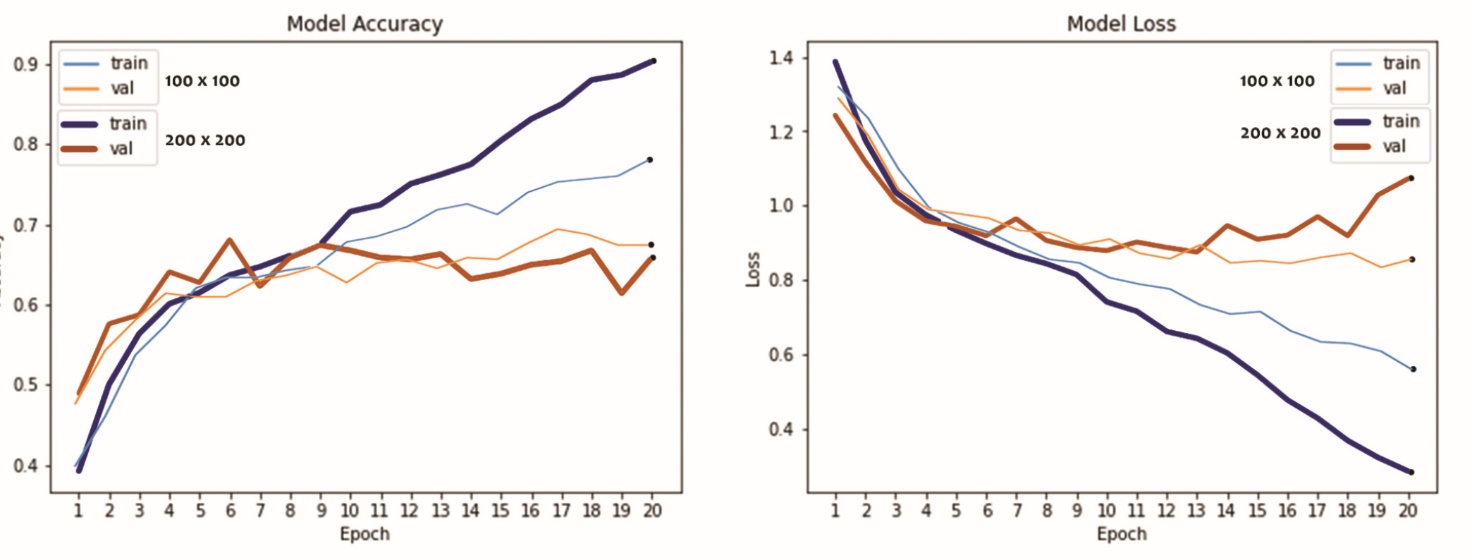
However, I think training datasets results could be misleading. As seen in validation datasets, accuracy starts to fluctuate after 10-15 epochs. Also, in the model loss graph, if the batch size is smaller, it increases after 20 epochs.

Consequently, 64 could be optimum number for batch size, and the number of the epoch can be 20 when we consider all graphs together.



The second group of graphs was created to understand which color mode could give more accurate results with 64 batch size and 100 x 100 px image size.

The graphs show that color mode changes don't create a significant difference.



Lastly, The model was created with two different image sizes.

Accuracy and loss are better in the 200 x 200 px in the training dataset. On the contrary, 100 x 100 px is better in the validation dataset. Since validation accuracy is more realistic, 100 px can be used in the model.

* Explanation of training data & test data

Firstly, there are a clustered image dataset. In order to create and test the model, it was split into 3 parts.

***Training dataset*** 70%

A part of the dataset was used iteratively for training the model.

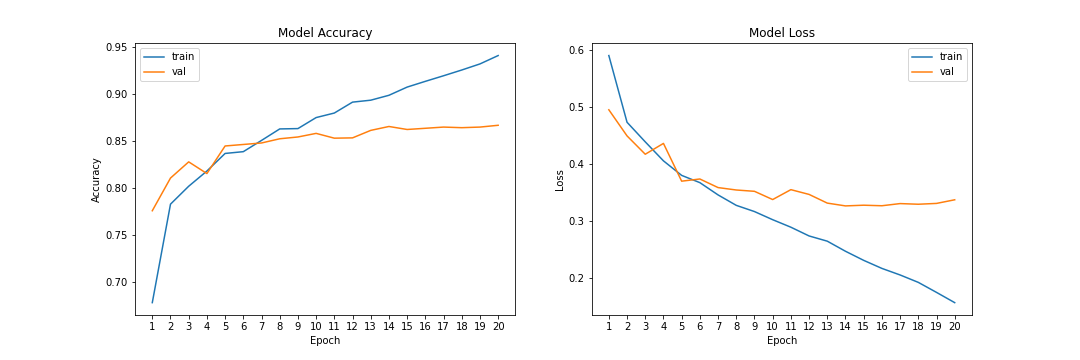
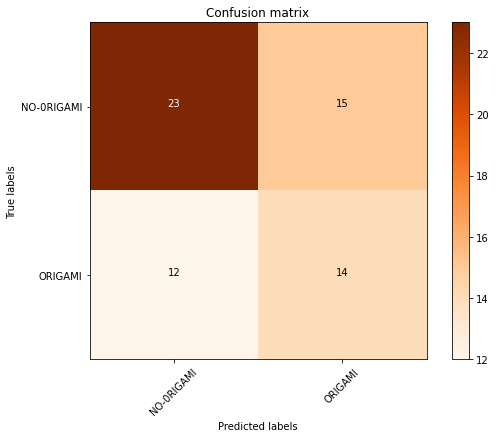
***Validation dataset*** 20%

A part of the dataset was utilized during the training to assess the model's performance at various epochs.

***Test dataset*** 10%

Evaluates the performance of the model after training.

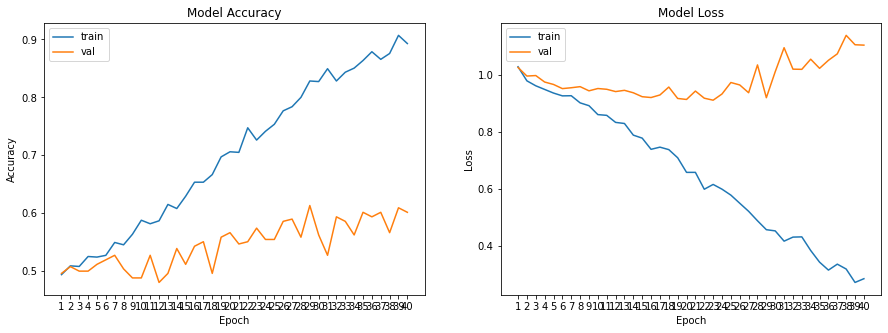
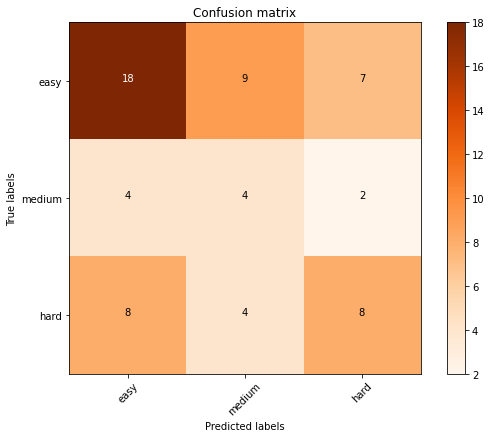
* Clear definition of scores & results
* Assessment of the graphics and scores

***Phase 1/Origami & No-origami***

|  |  |
| --- | --- |
| Accuracy(Train): | 0.94 |
| Accuracy(Validation): | 0.87 |
| Accuracy(Test): | 0.578125 |
| Precision: | 0.586299 |
| Recall: | 0.578125 |
| F1 Score: | 0.580962 |

The line graphs show that accuracy reaches almost 1.0 in the training dataset and approximately about 0.85 in the validation dataset. However, in the test dataset, it remains under 0.6.

And as seen from the confusions matrix, the probability of predicting no-origami images is higher.

***Phase2/Difficulty level***

|  |  |
| --- | --- |
| Accuracy(Train): | 0.88 |
| Accuracy(Validation): | 0.59 |
| Accuracy(Test): | 0.468750 |
| Precision: | 0.502574 |
| Recall: | 0.46875 |
| F1 Score: | 0.480260 |

The second phase has a smaller dataset and a more complicated task (distinguishing the difficulty levels. Consequently, it cannot reach the accuracy level of phase1 despite the higher number of epochs.

On the other hand, there is no useful standard for determining the difficulty levels. And the assigned levels by the users are inconsistent. In the dataset, they were classified according to appearances. (how much they look intricate) But still, it is not an objective parameter.

* The method of representation of the results (which graphic type is suitable?)

Line graphs are used to compare the results, especially in model qualities.

Line graphs are used again in the model evaluation to show the accuracy of the train and validation datasets.

Additionally, heat maps are used for illustrating the confusion matrices.

* Projections of the results

The model's target was to grade the origami images after distinguishing the origami from no-origami images. The first phase gives 0.87 accurate results in the validation dataset, which is quite good. However, due to the lack of standards for difficulty level, phase 2 is not as successful as phase1.

* References

***Dataset:*** *OrigamiSet1.0: Two New Datasets for Origami Classification and Difficulty Estimation, D Ma, G Friedland, MM Krell (2018), In Proceedings of Origami Science Maths Education, 7OSME, Oxford UK*

*Oxford Learner’s Dictionaries | Find definitions, translations, and grammar explanations at Oxford Learner’s Dictionaries*. (n.d.). Oxford Learner’s Dictionaries. <https://www.oxfordlearnersdictionaries.com/>

Alake, R. (2021, December 15). *Implementing AlexNet CNN Architecture Using TensorFlow 2.0+ and Keras*. Medium. https://towardsdatascience.com/implementing-alexnet-cnn-architecture-using-tensorflow-2-0-and-keras-2113e090ad98

*Convolutional Neural Network - javatpoint*. (n.d.). Www.Javatpoint.Com. https://www.javatpoint.com/pytorch-convolutional-neural-network