CNG483 INTRODUCTION TO COMPUTER VISION PROJECT 3: IRISDEEP

Burak Kaan Kahraman, 2315315
İlbey Evcil, 2079242

ABSTRACT

In this document, it makes a guess as to which person's iris eye images of certain people belong. Apart from the data set in the project, this data set is available in a file with parameters. Using this dataset, we trained different models with different hyperparameters and analyzed the results. Discussions about the methods are also presented in this report. Top performing models achieved 52% Accuracy on the given dataset.

Index Terms:— Iris, CNN, Extract Features, RELU, Softmax, RMSProp, Categorical CrossEntropy

I. INTRODUCTION

This project was carried out by examining under 4 main headings and stages. These; Preprocessing operations: At this stage, it includes the operations we have done in order to process the data given to us, adapt to the model and increase the accuracy rates. First of all, we took the data in the order we determined and after performing the necessary crop operations, we carried out preprocessing operations such as normalization and one hot to make the data more understandable and to adapt to the model we determined.

Creating the CNN Model: While creating our CNN model, we tried to establish a structure that could work in harmony with the requirements requested from us. Requirements; A sequiential structure should be established, convolution and pooling operations should be performed. Our report includes the results and discussions we have obtained from various configurations. While creating this model, various configurations were worked on. For example; such as whether there is a hidden layer or not, or changing the learning rate value.

Testing the model with the validation set: We divided the given data sets into three as train, test and validation at certain rates in order to train, test our model and change our parameters. In addition to these tests, we were able to determine that much better accuracy can be obtained by changing the ratios of validation, train and test sets.

Testing the model with the best configuration we specified: We performed our validation tests and as a result of these tests, we were able to reach the best test accuracy of our model with the configuration with the highest accuracy.

II. PREPROCESSING

Regarding the preparation of the dataset we have, we first took the images in the given dataset from the file. Then, we read each line from the parameters File and determined the order in which we would get the images and Persons according to the names of the images. After determining the order in which we will take each Person, we took the eye iris coordinations for each image from the texts in the parameter and recorded them in a separate list. Then, using the coordinates we received for each image, we cut around the images with a radius R. We took only the region we were interested in in the image in 128*128 format and put these images in a separate list for later use. After making sure that all our images are ready, we labeled each Person according to their data. In the second phase of the data preprocessing process, we have put the images we want to use in the format we want. At this stage, we applied normalization to each of our images. Since we use Categorical cross entropy, we applied one-Hot encode operation to our Label set. Since the data we have at the end of these processes is ready for use, we have divided the dataset we have into training validation and test sets. We divided 50% of the dataset as Training set to train our Model, 25% as Validation set to test the model we trained and modify our parameters, and 25% as Test set to finally test our dataset. While we are conducting tests we found that our training data is well sufficient for our model so we divide our training set to allocate some of our data for validation. After division our distribution was 25% Training set, 50% Validation set, 25% Test set (After that we tried this setup again and we get better results). As a result of these processes, we have completed the Data preprocessing part.

III. CNN-ARCHITECTURE

In the CNN Architecture section, we have decided how our model will be, how many convolution layers it will contain, what will be used as kernel size in these layers, what will be used as padding, which pooling will be used, and how many nodes will be used in the fully connector layer and how many hidden layers will be. First, we experimented to decide how many convolution layers we should use. We reached the conclusion that using two convolutional layers in these trials would be best for our model. While making this decision, we experimented with 1,2,3,4 convolutional layers. We got the best results with two convolutional layers. After we determined the number of convolutional layers to be 2, we did different experiments to determine the most suitable filter number for our model. First, we used eight filters on the first layer, 16 filters on the second layer, and then 16 filters on the first layer and 32 filters on the second layer. As a result of these experiments, we concluded that using eight filters for the first layer and 16 filters for the second layer gives better outcomes for our model. For our model, we used Max pooling after each convolutional layer. The reason we chose max-pooling was that we had to get the highest ones among the features so that we could choose the original features in the images we received. The reason for this was that the distinct features were high in value. We saw that Max pooling was the most suitable for this, and we preferred to use Max pooling. While using max pooling, we used 2x2 as kernel size and two as stride value. After the Convolutional layers, we tried two separate configurations while moving to the fully connected layers. First of all, we used one hidden layer. We used 20% Drop out when moving from the hidden layer to the output layer. In the other configuration, we only used the output layer. Between these two configurations, we decided to choose it because the configuration we use only Output Layer gives us better results. After we decided on the Configuration we will use, we tried to determine our learning rate, epoch number, and batch sizes. After our trials, we set the batch size as 64. We decided to try between 0.001 and 0.0005 for the learning rate, between 25 and 50 for the epoch number.

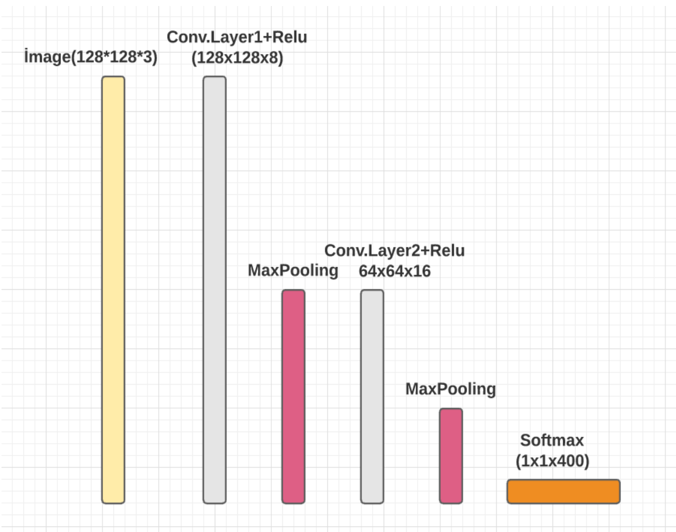

Fig. 1. Our CNN Configuration

IV. TRAINING RESULTS

In this section, we determined the number of Convolution layers for our model and the pooling types we will use. The variables we wanted to control here were the number of epochs, the learning rate, the number of filters and the number of hidden layers that we will use in the fully connected layer. To test these variables, we prepared the Table I. We tested these variables sequentially and recorded their results in the Table I. If we look at Table I, we can reach the conclusion that our best configuration is 0 hidden layer at 0.005 learning rate and 25 epochs.

TABLE I
TEST RESULTS 1 (TRAIN)
HL: HIDDEN LAYER **LR:** LEARNING RATE
EP: EPOCHS **CONV:** CONVOLUTION VALUE **LO:** TRAINING LOSS **VL:** VALIDATION LOSS **TA:** TRAINING ACCURACY **VA:** VALIDATION ACCURACY

HL.	LR.	Conv.	Ep.	Lo.	VL.	TA.	VA.
0	0.001	8.16	25	3.79	3.38	100	51.75
			50	5.77	5.19	100	45.25
		16.32	25	4.76	4.27	100	44.25
			50	6.69	5.99	100	43.50
	0.0005	8.16	25	2.99	2.56	100	56.75
			50	6.21	5.35	100	48.75
		16.32	25	4.92	4.44	100	49.25
			50	7.09	6.52	100	45.50
1	0.001	8.16	25	5.11	4.71	99.99	32.75
			50	4.77	4.23	100	43.0
		16.32	25	5.17	4.77	98.52	32.75
			50	7.20	6.66	100	37.50
	0.0005	8.16	25	4.69	4.35	100	39.75
			50	5.79	5.06	99.64	39.75
		16.32	25	4.33	3.91	99.82	41.25
			50	5.17	4.66	100	43.0

While testing the model we created, we saw that our model reached 100% training accuracy early on. Looking at this situation, we decided to try again by shifting some of the data in the training set to our Validation set. After dividing our dataset as 25% Training, 50% Validation, 25% Test, we reached the results indicated in the Table I. As we can see in these results, the result of our experiments is as follows. When we use 25% training set while training our model, we can test our model more accurately and determine our parameters more accurately, since we have more data for Validation. Therefore, the overall result is better. Table I contains our results.

TABLE II
TEST RESULTS 2 (VALIDATION)
HL: HIDDEN LAYER **LR:** LEARNING RATE
EP: EPOCHS **CONV:** CONVOLUTION VALUE **LO:** TRAINING LOSS **VL:** VALIDATION LOSS **TA:** TRAINING ACCURACY **VA:** VALIDATION ACCURACY

HL.	LR.	Conv.	Ep.	Lo.	VL.	TA.	VA.
0	0.001	8.16	25	2.08	3.86	100	35.50
			50	2.65	5.05	100	33.75
		16.32	25	2.12	4.12	100	36.38
			50	3.02	5.61	100	30.88
	0.0005	8.16	25	2.13	4.04	100	37.87
			50	2.61	5.12	100	34.25
		16.32	25	2.14	3.98	100	34.25
			50	4.22	7.91	100	29.12
1	0.001	8.16	25	3.46	5.33	99.46	20.13
			50	3.96	6.16	99.38	17.37
		16.32	25	3.53	5.33	100	21.0
			50	5.85	8.41	99.71	13.5
	0.0005	8.16	25	2.97	4.80	99.21	22.88
			50	3.45	5.88	100	24.88
		16.32	25	3.07	5.14	100	24.88
			50	3.17	5.18	100	23.5

V. TESTING RESULTS

Table III are the accuracy of the models tested using the configurations that give the best Validation Accuracy. Model 1 here comes from Table 1, which is obtained from the results we obtained in line with the requirements given to us, and model 2 comes from Table 2, where the validation set is predominant as a result of the tests performed. Considering the results obtained, it was observed that the test accuracy increased when the loss decreased. Apart from that, the fact that our set allocated for validation was larger increased the accuracy significantly. However, since it does not support the requirements given to us, it is only used as an example. It is thought that the low accuracy results obtained are due to the number of classes in the given dataset and the inadequacy of the set to train it. There are 2 trains, 1 validation and 1 test set for 400 classes.

TABLE III
FINAL RESULTS
MN: MODEL NAME **L:** LOSS
TEA: TEST ACCURACY

MN.	L.	TEA.
Model 1	3.41	45.75
Model 2	1.91	62.5

In addition to these, the best configurations selected by looking at the validation accuracies were obtained from the lesser Epoch numbers in both tables. Because it has been observed that epoch numbers can be overtrained over a certain number. In addition, the lower number of convolution layer filters in both enabled us to obtain better results for accuracy. It was concluded that it would not be beneficial to make a generalization, since the learning rates obtained from the given dataset may vary according to the samples taken from the dataset. In addition, it has been observed that better results are obtained in models without hidden layers than those used.

VI. ADDITIONAL COMMENTS AND DISCUSSION

When we look at all the sections and tables, some of the results we can reach are as follows. First, although the number of persons is high in the given datasets, the data we have for each person may not be enough to get high accuracy. For each person, we have 2 data to train our model, 1 data for validation and 1 data for testing affects the accuracy of our model. The Tests in Table 2, which we have done in addition to the parts requested from us, prove what we have said. The following can be tried to overcome this problem. By making Data Augmentation, the images we have can be reproduced, or as we tried, since our model completes the training quickly, the testing process can be done more healthily by shifting some of the data in our training set to the validation set. We tried to get a higher result by choosing the second option and we were partially successful in this. In addition to these, increasing or decreasing the number of filters or using more convolutional layers than necessary in the configurations we tried affect the accuracy negatively. In addition, increasing the number of hidden layers in the fully connected layer gives bad results for our model. References [1], [2] and [3] were used while creating our models and calculating our related parameters.

REFERENCES

[1] Yan Ding. Calculating parameters of convolutional and fully connected layers with keras. https://dingyan89.medium.com/calculating-parameters-of-convolutional-and-fully-connected-layers-with-keras-186590df36c6, October 2020. Accessed: 2021-6-20.

[2] Matthew Stewart, Phd Researcher, and Phd Researcher Matthew Stewart. Simple introduction to convolutional neural networks. https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac, February 2019. Accessed: 2021-6-20.

[3] Keras Team. Keras: the python deep learning API. https://keras.io/. Accessed: 2021-6-20.