Research Article

Hyperparameter Optimization in Convolutional Neural Networks for Maize Seed Classification

Sertuğ FİDAN[1*], Ali Murat TİRYAKİ[2]

Çanakkale Onsekiz Mart Üniversitesi , Orcid ID: 0000-0002-3458-7618 , sertugfidan@outlook.com,
Çanakkale Onsekiz Mart Üniversitesi , Orcid ID:0000-0001-8224-6319 , tiryaki@comu.edu.tr,
Correspondence: sertugfidan@outlook.com; +90 554 434 76 40

(First received January 12, 2023 and in final form March 19, 2023)

Reference: Hyperparameter Optimization in Convolutional Neural Networks for Maize Seed Classification. The European Journal of Research and Development,3(1), 139-149.

Abstract

Corn farming is of great importance for the continuity of our society. Because corn is a cheap and efficient food, especially for animal feeding. However, with the Doubled-haploid technique, the selection of the haploid seeds necessary for this job to be done efficiently creates a problem. Today, the selection of haploid seeds is usually done by trained technicians. With the development of machine learning methods, the parts expected from technicians can be made by machines. In this study, a new model architecture based on a convolutional neural network (CNN) was produced to perform the selection of haploid seeds and the hyperparameters of this model were optimized with the use of tree-structured parzen estimator algorithm. The newly produced model achieved a 94.66% validation score, higher than the VGG-19 model, which proved to be relatively efficient.

Keywords: Hyperparameter optimization, convolutional neural networks, machine learning, tree-structured parzen estimators.

1. Introduction

Corn is of great importance in terms of feeding people and animals and meeting the raw material needs of the industry. Plant breeders have resorted to new technologies to shorten the time required to produce high-yielding and quality hybrid seeds. Techniques to obtain haploid plants provide significant advantages in shortening the seed breeding period [1]. The starting point of maize breeding programs is to produce homozygous lines that will provide hybrid diversity. While crosses with traditional methods require a mating process that lasts approximately five to eight generations, the same process can be performed in an average of two to three generations using only haploid seeds [2]. The selection of haploid seeds is of great importance as it greatly reduces the cost of homozygous line production.

Corn seeds can be either haploid or diploid. Haploid seeds are found in nature with a very low probability of one in a thousand [2]. Considering this possibility, it can be said that haploid seeds cannot be obtained from nature in required quantities. In vivo induced haploids and double haploids are routinely used in modern maize breeding for the production of homozygous baselines [3]. A standart hybrid breeding scheme consist of:

- Selection among the doubled haploid (DH) lines
- One or several stages of testcross selection
- Chromosome doubling procedures

To create doubled haploid lines, it is necessary to know whether the seeds are haploid or diploid. There are multiple ways of differentiating haploid and diploid seeds including flow cytometry, color markers, etc. A most successful and common method used for differentiating haploid and diploid seeds is the R1-nj color marker. This color marker creates a purple tint in the kernel of diploid seeds so it can be separated from haploid seeds.

Today, trained technicians are used to separate haploid kernels from diploid kernels by mainly visual confirmation. Considering the number of kernels that need to be divided into categories, the cost of this work creates a big problem. Different solutions are offered by researchers to reduce this cost and automate the solution to the problem. One of these solutions, which is popular today and produces efficient results, is Convolution Neural Networks (CNN). Two main ideas behind CNN are to use local connectivity and weight sharing to reduce the number of parameters needed to represent images and the usage of the backpropagation algorithm during training [4]. Thanks to the classification of images by CNN, costs can be greatly reduced by eliminating the need for specialists. CNNs have also been the subject of previous articles such as [4][5][6] in the process of distinguishing haploid kernels from diploid kernels. The main differences in the studies in this area are due to the depth of the CNNs used, the activation functions, and the hyperparameters of the neural networks.

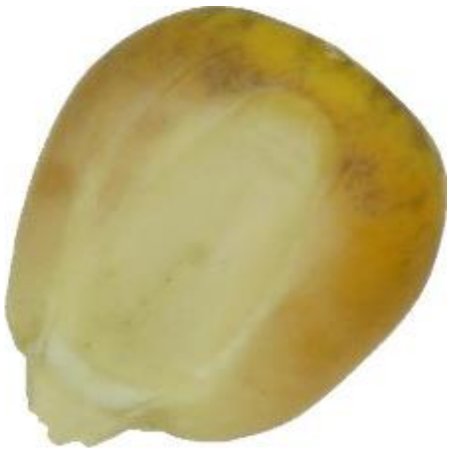
In this research, the hyperparameter space was defined by using the hyperopt library[7], and a new CNN model was created that classifies the seeds as diploid and haploid by making hyperparameter optimization in this space. As a result of our research, a new model was produced using hyperparameter optimization, which produces a more accurate validation score compared to previously produced models.

The remainder of the paper is organized as follows: materials and methods are given in Section 2. The experimental results are given in Section 3. Finally concluding remarks are given in Section 4.

2. Materials and Methods

2.1. Dataset

We have used a publicly available data set which is shared by [2]. The data set includes 1230 haploid and 1770 diploid maize seed images. According to [2] seeds were selected in a way order to make sure to reflect different expression levels of R1-nj marker color(light-dark, dense-less). Then we randomly divided the data into two parts, 20% test data, and 80% training data. We resized the images to 224x224 o make a more fit comparison with the VGG-19 model, which is the best result of [2] with %94.22 validation accuracy. Examples of haploid and diploid seeds from the data set can be seen in Figure 1. The largely distinct purplish color in the kernel of the corn indicates that the seed is diploid.


(a)

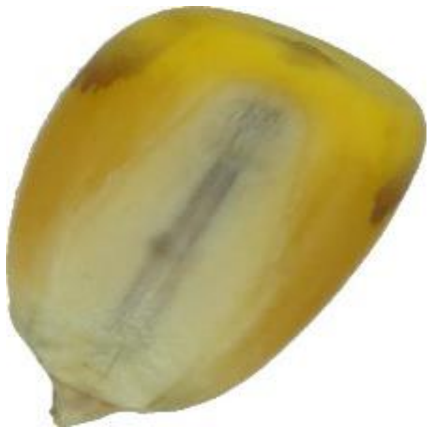

(b)

Figure 1: (a)Haploid seed example[2] (b) Diploid seed example[2]

2.2. Tree-Structured Parzen Estimators(TPE) Algorithm

In this study, we used the TPE[8] algorithm to optimize the quantization hyperparameters with the help of the hyperopt library. Algorithm steps can be summarized as follows:

- Define hyperparameter search space,
- Create objective function which takes hyperparameters and outputs a score(loss, root mean squared error, etc.) which we try to optimize,
- Get a couple of observations using a randomly selected set of hyperparameters,
- Sort collected observations by score and divide them into two groups. The first group (x1) contains observations(randomly selected hyperparameters, winner) that give the best scores, and the second one (x2)- all other observations(loser).
- Two densities(l(x1),g(x2)) get modeled using Parzen Estimators(average of kernels around data points).
- Draw sample hyperparameters from evaluating them in terms of l(x1)/g(x2), and returning the set that yields the minimum value under l(x1)/g(x1) corresponding to the greatest expected improvement. These hyperparameters are then evaluated on the objective function.
- Update the observation list
- Repeat steps between collecting observations and updating observation list with a fixed number of trials or until time limit is reached.

As can be seen from the algorithm, the most important part for the TPE algorithm to work efficiently is the correct definition of the hyperparameter space. we made multiple attempts to define this space correctly. In the continuation of the article, we will talk about the version in which we obtained the best results.

2.3. Hyperparameter Search Space

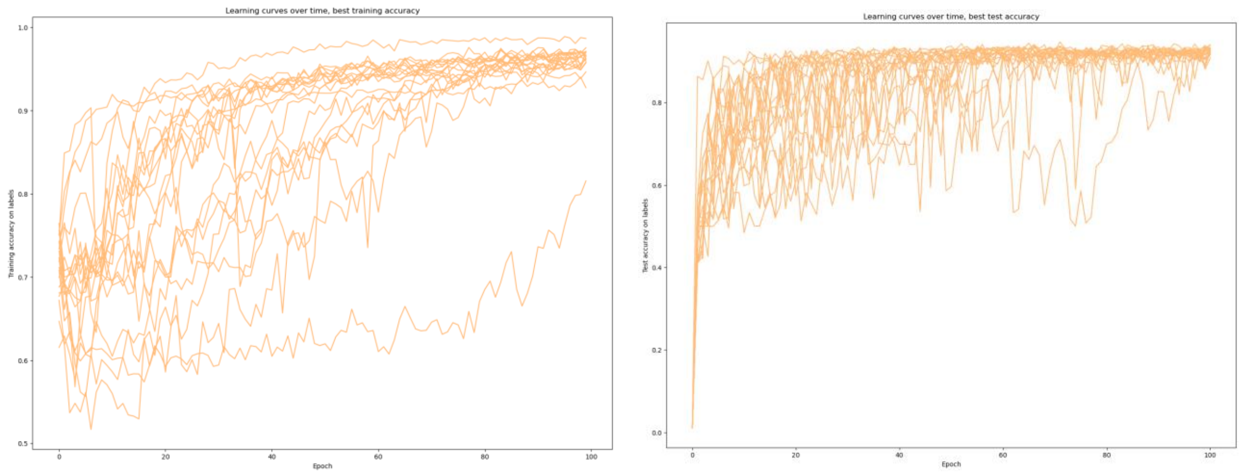
In this section, we will define the parameters that we are trying to optimize, and if they are needed, additional information will be given about the parameters.

- Learning rate multiplier: Decreases, and increases default learning rate(0.007).
- Learning weight decay: Decreases and increases weight decay(0,01).
- Batch size: Decides batch size (Between 10 and 32, step 2).
- Optimizer type: Decides optimizer (Adam, Radam, RMSprop)
- Convolution layer dropout probability: Decides if there is dropout layer after the convolution layer and what its dropout probability should be. (Between 0.0, 0.30) Has a chance to be activated after every layer.

- Fully connected layer dropout probability: Decides if there is a dropout layer after a fully connected layer what its dropout probability should be. (Between 0.0, 0.30)
- Batch normalization usage: Decides if the model should use Batch Normalization. (Yes or No) Has a chance to be activated after every layer.
- Usage of specific convolution layer architecture for first layers: Decides should start of the model have a specific architecture which uses back-to-back convolutions (Can be 0 which means no special architecture, 3 convolution, 4 convolution).
- Usage of residual connections: Decides should the model use residual connections if so how many ?
- Convolution hidden layer units multiplier: Decreases and increases default hidden layer values.
- Number of convolution layers stacked: Decides how many convolution layers in a row can be used(2 or 3).
- Pooling type: Decides pooling type for subsample step (can be max, avg, all convolution[9] or inception[10])
- Convolution kernel size:
- Residual convolution kernel size:
- Fully connected layer neuron multiplier: Increases and decreases fully connect neuron counts.
- Usage of a second fully connected layer neuron multiplier: Decides if the model should use a second fully connected layer after the first one with reduced neuron counts.
- Activation function: Decides which activation function to use (relu, gelu, swish).

3. Result

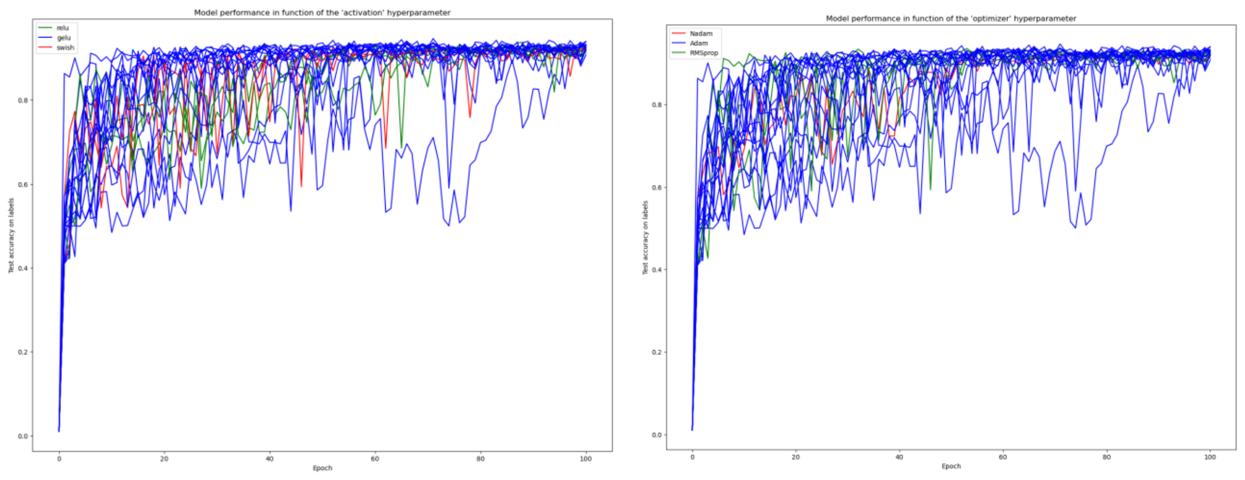
Using the hyperparameter space we defined above, we trained 150 different models over 100 epochs, with a default learning rate of 0.001 and a weight decay of 0.007. All experiments were implemented in Python with the help of libraries: Tensorflow, Keras, Hyperopt, and Numpy. All models are implemented on a lab computer with an RTX 3070 GPU. In order to increase the readability of the graphics we will present, we will only talk about the top 20 models(best validation accuracy) from now on.



(a) (b)

Figure 2: (a) Training accuracy over epochs for CNN models (b) Validation Accuracy over epochs for CNN models

As we can see from Figure 2. even though if a model is started with relatively wrong parameters at the beginning of the training, as the number of epochs increases, it is possible to catch up with the best models. Our best validation accuracy is %94.66 which is higher than best validation scores of [2] and [6].



(a) *(b)*

Figure 3: (a) Usage of activation function for CNN models (b) Usage of optimizer for CNN models

Figure 3. (a) shows that the majority of the activation functions used for the best models are Linear Units of Gaussian Error (GELU) and Figure 3. (b) shows that for our tests Adam optimizer was the best performer overall.

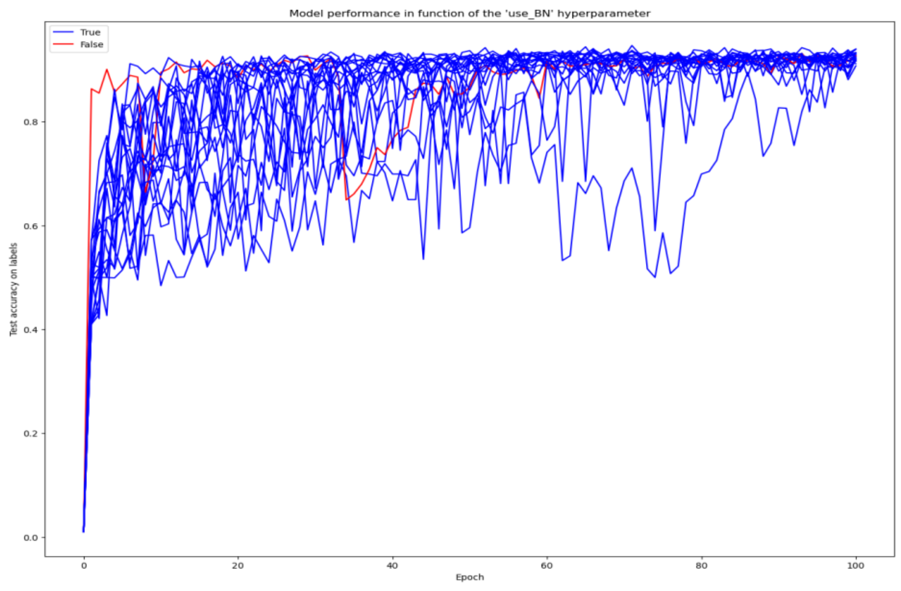


Figure 4: Usage of batch normalization for CNN models

Lastly, as we can see from Figure 4. out of the 20 best-performing models, only one does not use batch normalization. With this graph, we have largely proven the positive effect of batch normalization on CNN's. The best model which have created has %94.66 validation accuracy and %97.3 training accuracy, uses GELU activation function, has batch size of 16, has special first convolution architecture with 4 convolution at start, uses Adam optimizer, uses inception for pooling type, uses batch normalization. Models using pooling in the Inception type generally achieved more efficient results than the others. Finally, the total number of parameters belonging to the best model we created was 9,022,299. 9,022,065 of these parameters are trainable. Compared to VGG-19's 138 million trainable parameters our model has only 9 million parameters and can get similar validation accuracy. Although our model has a low number of trainable parameters, we think that it produces results similar to VGG-19 because our model uses batch normalization, unlike VGG-19. The architecture of the model can be seen in Figure 5.

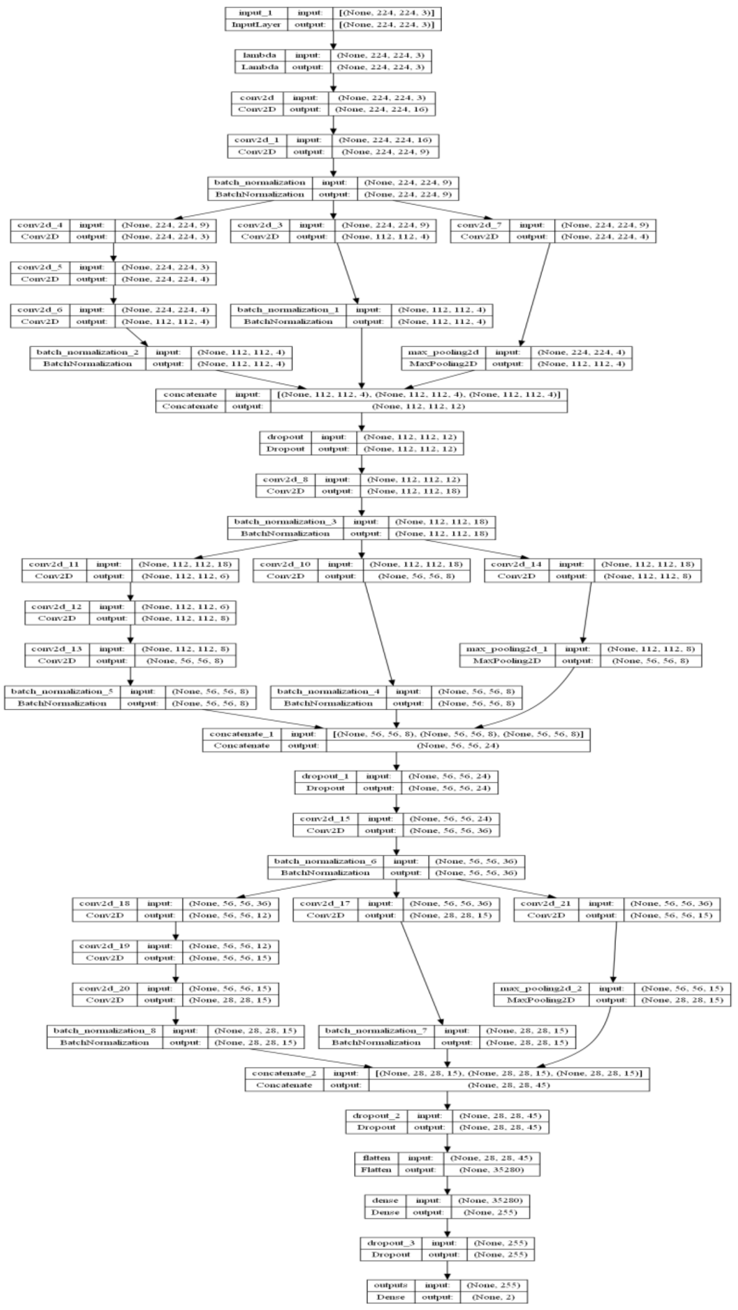


Figure 5: Best model architecture

4. Discussion and Conclusion

For this study, a hyperparameter space that can optimize a CNN is created. We investigated common hyperparameter properties of models with high validation accuracy. Although the importance of the hyperparameters we found depends on the data set we have, this new model we created is able to rule out a relatively cutting-edge model like the VGG-19. According to similar studies on the same dataset [2][6], we can say that we have produced a more efficient model with 94.66% validation accuracy on this data set. As a result of experiments on the number of neurons of the default fully connected layers, the results of the hyperparameter optimization caused the batch normalization to not work efficiently if the number of neurons in the fully connected layer was less than expected or the number of epochs was not sufficient. Although batch normalization is a good method to increase the efficiency of CNNs, it seems that it needs to meet certain conditions in order to obtain optimum results, so it may not be suitable for every model.

During our experiments, we tried to keep the number of parameters in the range of 8m -80m by controlling the number of neurons in the fully connected layers. Next, we plan to build models that use a larger number of trainable parameters and optimize their parameters. We will also check whether the model we created works with similar success on other sample datasets.

References

[1] I. Cerıt, G. Comertpay, R. Oyucu, B. Cakir, R. Hatipoglu, and H. Ozkan. (2016). Melez mısır islahında in-vivo katlanmıs¸haploid tekniginde kullanılan farklı inducer genotiplerin haploid İndirgeme oranların belirlenmesi, Tarla Bitkileri Merkez Araştırma Enstitüsü Dergisi, vol. 25, no. OZEL SAYI-1, pp. 52 – 57.

[2] Y. Altuntas¸, Z. Comert, and A. F. Kocamaz. (2019). Identification of haploid and diploid maize seeds using convolutional neural networks and a transfer learning approach, Computers and Electronics in Agriculture, vol. 163, p. 104874,[Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168169919300481

[3] H. H. Geiger, G. Andr´es Gordillo, and S. Koch. (2013). Genetic correlations among haploids, doubled haploids, and testcrosses in maize, Crop Science, vol. 53, no. 6, pp. 2313–2320, [Online]. Available: https://acsess.onlinelibrary.wiley.com/doi/abs/10.2135/cropsci2013.03.0163

[4] Le Cun BB, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD. (1990). Handwritten digit recognition with a back-propagation network, In: Advances in Neural Information Processing Systems. Denver: Morgan-Kaufmann.

[5] B. Veeramani, J. W. Raymond, and P. Chanda. (2018). Deepsort: deep convolutional networks for sorting haploid maize seeds, BMC bioinformatics, vol. 19, no. 9, pp. 1–9.

[6] E. Dönmez. (2020). Classification of haploid and diploid maize seeds based on pre-trained convolutional neural networks, Celal Bayar University Journal of Science, vol. 16, no. 3, pp. 323–331.

[7] Bergstra, J., Yamins, D., Cox, D. D. (2013). Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures, TProc. of the 30th International Conference on Machine Learning (ICML 2013), pp. I-115 to I-23.

[8] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. (2011). Algorithms for Hyper-Parameter Optimization, in Advances in Neural Information Processing Systems 24, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., pp. 2546–2554.

[9] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. (2014). Striving for simplicity: The all convolutional net, [Online].Available: https://arxiv.org/abs/1412.6806

[10] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. (2016). Inception-v4,inception-resnet and the impact of residual connections on learning,[Online]. Available:https://arxiv.org/abs/1602.07261