The relevance of hyper-parameters in convolutional neural networks: a case study with Fashion-MNIST

Rubén Sánchez Acosta
Universidad Católica del Norte
Av. Angamos 0610, Antofagasta, Chile
ruben.sanchez01@ucn.cl

Brian Keith Norambuena
Universidad Católica del Norte
Av. Angamos 0610, Antofagasta, Chile
brian.keith@ucn.cl

ABSTRACT

Artificial vision usually extracts features of images and tries to give an interpretation of them for their application in the solution of certain problems. One of the main applications of computer vision is object detection, which consists in identifying whether an object is present in an image. In this work, a convolutional neural network is used to detect the type of garment present in an image. The purpose of the study is to verify how relevant a correct choice of hyper-parameters can be for convolutional neural networks. The Fashion-MNIST dataset is used for the training and testing phases.

Keywords: Artificial vision, object detection, artificial neural networks, convolutional neural networks, hyper-parameters.

INTRODUCTION

Artificial intelligence has had a great impact on current everyday life. This area of technology is responsible for providing machines with features that are fundamental for the functioning and development of human beings, such as vision, language and intelligence in general. In many fields of research, great advances have been made thanks to artificial intelligence.

Artificial vision or computer vision is an area within artificial intelligence that has been widely studied and where significant results have been achieved. Computer vision allows machines to give meaning to images for countless applications. The present work attempts to solve a problem of artificial vision using convolutional neural networks (CNN), a tool that has proven its usefulness for computer vision, reaching very good results in the state of the art for this field. The performance of some models of convolutional networks is compared by varying their hyper-parameters.

The problem to be solved is the classification of garments: given the image of an article, the aim is to identify what type of garment it is. There are 10 different categories: t-shirt, trousers, sweatshirt, dress, jacket, sandal, shirt, sneaker, bag and boots. The Fashion-MNIST dataset [1] is used, consisting of a training set of 60,000 images and a test set of 10,000 samples. Each instance is a grayscale image of 28x28 pixels.

The rest of this document in structured as follows, first, artificial vision and CNNs in general are detailed. Then, the experiments carried out are described and the results achieved in the Fashion-MNIST set for the implemented CNN are presented. Finally, the results are analyzed, and the conclusions are provided.

ARTIFICIAL VISION

Artificial vision or computer vision is a scientific discipline that includes methods to acquire, process, analyze and understand real-world images in order to produce numerical or symbolic information so that they can be treated by a computer [2].

Within artificial vision, this article focuses on object detection. This area studies how to detect the presence of objects in an image. Generally, two parts can be distinguished in the detection process: the extraction of features from the content of an image and the search for objects based on those features. One of the most used methods in artificial vision are artificial neural networks (ANN) [3], due to two fundamental reasons: firstly, in many cases the ANN are able to extract significant features in the images without the intervention of an expert, turning the first step of object detection into a trivial phase; Secondly, ANNs have achieved the best results in the state of the art of object detection [4].

CONVOLUTIONAL NEURAL NETWORKS

A convolutional neural network (CNN) is a type of multi-layered ANN consisting of alternating convolutional and pooling layers, then it has fully connected layers like basic feedforward networks [5].

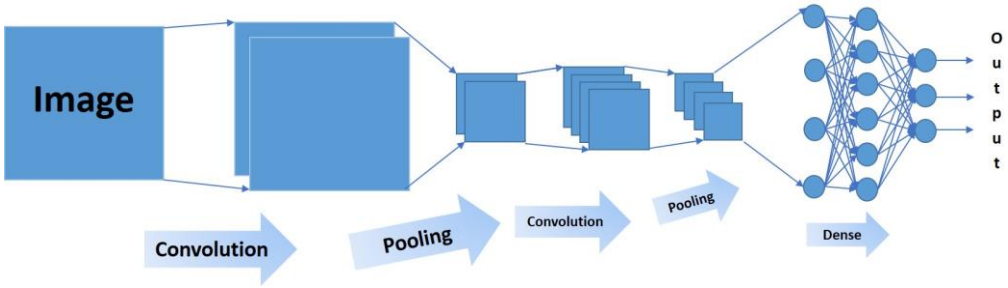

Figure 1: Structure of a convolutional neural network.

There are several thoughts on why convolutional networks work so well in recognizing objects in images. One of them is that each convolution layer observes incrementally a larger part of the image as it propagated through the network. In this way, it looks for features in the image at different scales and abstraction levels, this resembles how the visual system of humans works [6].

Convolution layers

In the convolution, product and sum operations are performed between the input layer and the *n* filters (or kernel) that generate a feature map. The extracted features correspond to each possible location of the filter in the original image.

The advantage is that the same filter serves to extract the same feature in any part of the input, with this it is possible to reduce the number of connections and the number of parameters to train in comparison with a fully connected multilayer network. Figure 4 illustrates the convolution process.

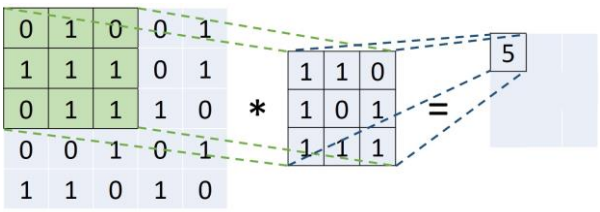

Figure 2: Functioning of a convolution layer.

After performing the convolution, an activation function is applied to the feature maps. The most recommended activation function is the sigmoid ReLU [7].

Pooling layers

In the pooling layers, the number of parameters is reduced by keeping the most common features. The parameters are reduced by extracting statistics such as the average or maximum of a fixed region of the feature map. By reducing features, the method loses accuracy, although it makes training much less expensive.

Figure 3 shows an example of pooling using a 2x2 filter in the form of a sliding window, where the maximum of the region is chosen. You can also use other criteria to summarize the region such as finding the average.

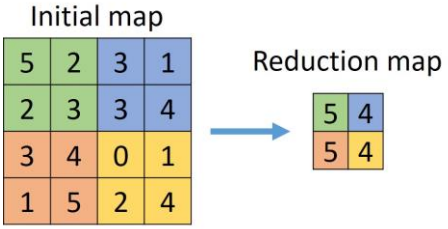

Figure 3: Functioning of a Pooling layer.

EXPERIMENT

To implement the CNN, the TensorFlow library was used together with the Python programming language. Several neural networks with different structures and hyper-parameters were trained to analyze how the variation of hyper-parameters and structure affected their performance.

Two structures of convolutional networks were tested which are explained below. In the first model, the network has a convolution layer, then a pooling layer and finally a fully connected hidden layer that in turn connects to an output layer of ten nodes, one for each type of garment. The second model is similar to the first, but a second convolution layer and another pooling layer are added before the densely connected layer.

In the convolution layers, 5x5 pixel filters are applied, with a ReLU activation function. The pooling layers use 2x2 filters, which implies a reduction of 50% of the size of the vectors. To perform the pooling, the average of the indicated region is chosen. In the hidden layer, the dropout technique is used with a probability of 0.2 to avoid overfitting. The output layer uses the activation function softmax and cross entropy to calculate the error. The network was trained with a stochastic gradient descent, with a size of 500 instances in each batch. The training was carried out until reaching 20,000 iterations.

The hyper-parameters modified are the number of filters in the convolution layers and the number of neurons in the hidden layer. They take values between 32 and 64 filters, as well as between 256, 512 and 1024 neurons in the hidden layer.

RESULTS

Table 1 shows the results achieved by the CNN for their different combinations of hyper-parameters. The CNNs that use a single convolution layer are in the upper part, and then those that use two, all with the respective number of neurons in the hidden layer. For each one, the required training time and the accuracy achieved on the test set are shown.

Table 2 shows the correlation between the tested hyper-parameters and the accuracy of the convolutional neural network.

After obtaining the previous results, the criterion to use in the pooling layers was changed in the CNN with better performance, from Average Pooling to Maximum Pooling (i.e., choosing the maximum of the region), the result is shown in Table 3.

Afterwards, the behavior of the CNN with the better performance was tested by varying the value of the dropout probability and the following graph in Figure 4 shows the results.

Table 1: Performance of the different CNNs comparison.

Number of convolutional layers	Number of filters in layer 1	Number of filters in layer 2	Number of neurons in the hidden layer	Training time	Accuracy
1	32	-	256	0:41:32	0.815
			512	0:55:23	0.853
			1024	1:14:42	0.869
	64	-	256	0:52:15	0.842
			512	1:18:36	0.870
			1024	1:44:23	0.918
2	32	32	256	0:31:25	0.623
			512	0:40:49	0.682
			1024	0:52:21	0.731
	64	64	256	0:34:56	0.702
			512	0:46:03	0.722
			1024	0:57:26	0.771

Table 2: Correlation coefficient.

	Number of convolutional layers	Number of filters in layer 1	Number of filters in layer 2	Number of neurons in the hidden layer
Accuracy	-0,89	0,24	-0,72	0,35

Table 3: Pooling criterion comparison.

Number of convolutional layers	Number of filters in layer 1	Number of filters in layer 2	Number of neurons in the hidden layer	Pooling strategy	Accuracy
1	64	-	1024	Average	0.918
				Maximum	0.893

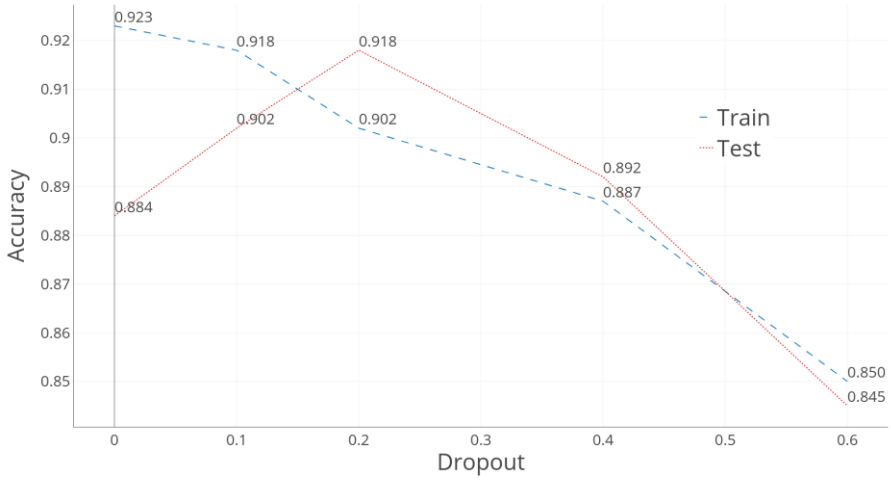

Figure 4: Dropout behavior graph.

DISCUSSION OF RESULTS

Several observations can be made with respect to Table 1. Regarding the time, it took for the training, it can be observed that the increase in the number of neurons present in the hidden layer, in each of the cases, implied an increase in training time. In addition, the increase in the number of filters used in the convolution layers also leads to an increase in training time. The number of convolution layers, on the other hand, produces a reduction in training time, this is due to the reduction of pixels that occurs in the pooling layers, which means that there are fewer connections towards the hidden layer and therefore fewer synaptic weights must be updated in each iteration.

From Table 1, it can also be observed that an increase in the number of neurons in the hidden layer or filters in the convolution layer leads to better accuracy. However, an increase in the number of convolution layers leads to a decrease in accuracy.

The best time was obtained with two convolution layers using 32 filters and 256 neurons in the hidden layer, but it was the worst in terms of accuracy, this is consistent with the observations made. As for the best accuracy, it was obtained with a single convolution layer using 64 filters and 1024 neurons in the hidden layer, which could be predicted from the previous statements.

In Table 2 it can be seen that the number of convolutional layers is strongly correlated with accuracy in a negative way, i.e. an increase in this hyper-parameter influences the decrease in accuracy. The number of filters in layer 2 also affects the accuracy significantly, but this is directly related to the number of convolutional layers because, when there is only one layer, the number of filters in the second layer is 0. Although the other hyper-parameters affect accuracy as seen in Table 1, they do not show a strong correlation because, when these hyper-parameters vary, the increase in accuracy is small.

For the CNN with better accuracy, a new criterion was used in the pooling layer, choosing the maximum value instead of the average, these results were shown in Table 3. In this case, a better accuracy for the first strategy is observed, this is understandable because the final output is affected by all the pixels when finding its average, therefore, it provides information on all the filtered pixels.

The last graph shows the behavior of the CNN with higher accuracy when varying the dropout probability. Dropout is a technique to prevent overfitting that consists in not activating a neuron, even if it should be activated, with a certain probability. The graph in Figure 4 for the training set shows how, as the dropout probability increases, the accuracy on the training set decreases. This is due to the fact that fewer neurons are activated than they should, causing the CNN to not receive the complete information provided by the dataset during the training phase.

The graph in Figure 4 for the test set shows how increasing the probability of dropout increases accuracy, until it reaches a point where it begins to decrease. There is even a moment where the accuracy for the test set becomes higher than that of the training test. When the dropout is 0, the difference between the accuracy of the training set and the test set is the greatest, which is expected, since all the information of the training set is obtained and therefore a slight overfitting occurs.

FUTURE WORK

Training a neural network is computationally expensive, because of this, others hyperparameters variations wasn't tested. As future work, it is proposed to test the behavior of the variation of the following hyper-parameters:

- Activation function (ReLU, Softmax, tanh, sigmoid).
- Error computation (square error, cross entropy).
- Filter size.
- Training (online, batch).
- Number of hidden layer in the dense network.
- Add a dense layer between the two convolutional layers.

CONCLUSIONS

Structure and hyper-parameters are of crucial importance for the good performance of CNN. A correct definition and choice of these can make the difference between a good and a poor model. Therefore, their choice should be seen as one of the fundamental stages when establishing a model.

As evidenced during the work and contrary to what many believe, in neural networks more is not always better. An increase in the number of convolution layers did not bring about an improvement in performance. Neither an increase nor decrease in the dropout probability improves accuracy in all cases, so a balance should be found.

Often it also happens that to find a model that takes less time to train, accuracy must be sacrificed and vice versa. In this sense, a balance must also be found. In the case of the study used, the network with the best time achieved the worst accuracy, although the one with the best accuracy did not have the worst time. In this sense, what is relevant is what the priority is in the specific case where it is applied, thus there is a trade-off between accuracy and training time.

The present work reaffirms the relevance of hyper-parameters in the proper functioning of a CNN. For the Fashion-MNIST image set, it was proved that there are CNN that do not produce relatively good results and others that achieve excellent performance, the only difference between them was simply their hyper-parameters.

REFERENCES

[1] H. Xiao, K. Rasul, R. Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. Zalando research. 2017.

[2] L. G. Shapiro, G. C. Stockman. Computer Vision. Prentice Hall. 2001

[3] K. Gurney. An introduction to neural networks. UCL Press. 1997.

[4] P. D. Cristea. Application of Neural Networks In Image Processing and Visualization. NATO Science for Peace and Security Series C: Environmental Security. 2009.

[5] D. Svozil, V. Kvasnicka, J. Pospichal. Introduction to multi-layer feed-forward neural networks. Chemometrics and Intelligent Laboratory Systems, Volume 39, Issue 1. 1997.

[6] D. Ciresan, U. Meier, J. Masci, L. M. Gambardella, J. Schmidhuber. Flexible, High Performance Convolutional Neural Networks for Image Classification. Proceedings of the Twenty-Second international joint conference on Artificial Intelligence. 2011.

[7] I. Goodfellow, Y. Bengio, A. Courville. Deep Learning. MIT Press. 2016.

[8] Zalando Benchmark. Zalando research. http://fashion-mnist.s3-website.eu-central-1.amazonaws.com. 2017.