Identification of Client Profile Using Convolutional Neural Networks

Victor Ribeiro de Azevedo, Nadia Nedjah, and Luiza de Macedo Mourelle(✉)

Pos-graduation Program in Electronics Engineering,
State University of Rio de Janeiro, Rio de Janeiro, Brazil
v.azevedo9@gmail.com, {nadia,ldmm}@eng.uerj.br

Abstract. In this work, a convolutional neural network is used to predict the interest of social networks users in certain product categories. The goal is to make a multi-class image classification to target social networks users as potential products consumers. In this paper, we compare the performance of several artificial neural network training algorithms using adaptive learning: stochastic gradient descent, adaptive gradient descent, adaptive moment estimation and its version based on infinity norm and root mean square prop. The comparison of the training algorithms shows that the algorithm based on adaptive moment estimation is the most appropriate to predict user's interest and profile, achieving about 99% classification accuracy.

Keywords: Convolutional Neural Networks · Deep learning · Social media image classification · Customer profile identification

1 Introduction

The large volume of information on social networks, such as images and videos, makes it necessary to develop new techniques to extract relevant information from users. Thousands of photos that are being published in daily social networks shows itself to be a relevant process for companies. With a segmentation of customers, it is possible to predict user's interests or create advertisements directed to different publics.

Instagram is one of the most popular social network with more than 1 billion active users monthly, as mentioned by specialists of [2] estimated by the end of 2018. This amount is more than *Twitter* users, but it is behind the amount of users of *WhatsApp* and *Facebook Messenger*. Every day, *Instagram* users post more than 100 million photos as related in [2]. It is important to mention that 72% of the users of *Instagram* say they bought a product they saw in the application as mentioned in [6]. This shows that identifying users by interest is a relevant sales strategy.

Deep learning is a subcategory of machine learning where high complexity neural network models are used to recognize patterns. Recognition of patterns in images and videos can be considered one of the most widely used applications

O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12251, pp. 103–118, 2020.
https://doi.org/10.1007/978-3-030-58808-3_9

of deep learning. This work contributes to the scientific community, in the field of computational vision and image classification, with exploratory and experimental analysis of the performance of Convolutional Neural Networks (CNNs). Different training algorithms are applied to a group of images in the categories: animals, electronics, games, vehicles and clothes. To improve the performance of the deep learning system, experiments are performed varying the structure of deep learning training algorithms and the initialization of hyperparameters, such as the number of epochs and the rate of learning. In this work, we compare the performance of the following training algorithms: SGD, AdaGrad, Adamax, RMSProp and Adam.

In Sect. 2, we present some related works. In Sect. 3, we introduce the concept of convolutional neural networks. In Sect. 4, we show our proposed solution and the methodology adopted. In Sect. 5, we present the results obtained. In Sect. 6, we drawn some conclusions and future work.

2 Related Works

One of the first projects using CNN was LeNet [7]. The aim was handwriting recognition by implementing the back-propagation algorithm in a Multi Layer Perceptron (MLP), demonstrating the great potential of CNN as extractors of image characteristics.

In [3], an inference of users interests in the social network *Pinterest* is made, based on the classification of texts and images. The work shows that it is possible to recommend products of certain categories based on the classification of images posted by users. The authors report that one of the project's challenges is the interpretation of user's interest based on the image or comment posted on the social network, but the user refers to another subject. In this way, the results were compared using a multi-modal space, combining texts and images, and uni-modal, with texts or images, in order to achieve a better accuracy in the evaluation. The authors use CNNs for image classification, the Bag of Words (BoW) technique for textual representation and the Bag of Visual Words (BoVW) technique for image representation as a feature map. BoW is often used in document and text classification methods, where the occurrence frequency of each word is seen as a characteristic used to train a classifier. Similarly in BoVW, each image is represented as a histogram of occurrence of visual words (small portions of the image, capable of satisfactory describing the image as a whole).

In [14], the binary classification of users of the social network *Pinterest* between masculine and feminine is based on their images. The differential of this project is that it uses the Scalar Invariant Feature (SIF) algorithm to discover local characteristics in each image. To obtain the images for data source of the users, a program is carried out covering 1500 users of the *Pinterest* to obtain the *Twitter* profile. From these profiles, it is possible to obtain the sex of a randomly selected sample with 160 users (80 male and 80 female). Thirty categories are defined from these users.

In [12], the inference of the users' personality of the *Flickr* network is made from images of their galleries. The interaction between the posts, that people

leave on social networks, and the traits of their personality are investigated. Experiments with 60.000 images, added to favorites by 300 users, allow them to extract features and segment users.

3 Convolutional Neural Networks

Convolutional Neural Networks are deep learning models usually used for pattern recognition and classification applications in images and videos. A CNN is usually formed by an input layer, which receives the images, a series of layers, which perform image processing operations with characteristics map, and a last step that contains a classification neural network, which receives the characteristics map and outputs the result of the classification.

CNNs are composed of neurons that have weights and bias, which need to be trained. Each neuron receives inputs, to which a scalar product is applied with their respective weights, and provides an output based on a non-linear activation function. A CNN assumes that all inputs are images, which allows us to set properties in the architecture. Traditional artificial neural networks are not scalable to process images, since these require a very high number of weights to be trained.

A CNN consists of a sequence of layers. In addition to the input layer, there are three main layers: convolutional layer, pooling layer and fully connected layer. Besides this, an activation layer is common after the convolutional layer. These layers, when placed in sequence, form the architecture of a CNN, as illustrated in Fig. 1. The convolutional layer is composed of a set of filters that are matrices of real numbers, which can be interpreted as learning weights according to a training. These filters assist in convolution with input data for a feature map. These maps indicate regions in which specific characteristics relative to the filter are found at the entrance. Filter values change throughout the training by having the network learn to identify significant regions to extract features from the data set. The fully connected layer is a common artificial neural network that receives the output of the feature maps and performs the final classification.

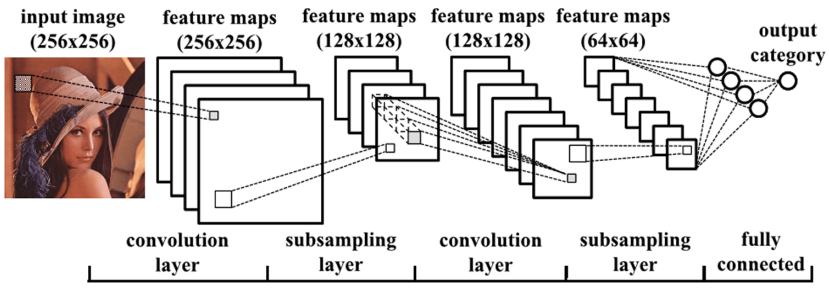

Fig. 1. Architecture of a Convolutional Neural Network

In [1], the pooling layers are mentioned as responsible for reducing the dimensionality of the matrices and the complexity in the neural network. The authors cite the different types of layering aggregation and their advantages. The types of layers cited are: max pooling, average pooling, spatial pyramid pooling, scale dependent pooling.

Activation functions are applied to convolution results matrices. The most used function is the Rectifier Linear Unit - ReLU, used to apply the maximization function to each element of the convolution result. The pooling technique is used after the convolution layer in order to reduce the spatial size of the matrices resulting from the convolution.

The rate of learning is a positive constant, which corresponds to the speed of learning. In the adaptive learning rate (LR) algorithms the value of the learning rate is modified by a term defined in each algorithm. The main adaptive learning algorithms are: AdaGrad, RMSProp and Adam. Besides these, there are others like Adamax, AdaDelta and Nadam.

4 Methodology

For the development of the training model, a virtual environment is created with the container docker. A container docker is a standard encapsulated environment that runs applications through virtualization at the operating system level. An image of docker is a file used to execute code in a docker container. An image is an immutable file that is essentially the representation of a container. Images are created with the compilation command and they will produce a container when started with execution. The image of a docker container is used with the following tools: Jupyter, Matplotlib, Pandas, Tensorflow, Keras and OpenCV [9]. In this work, the Tensorflow and the Keras are used with the language Python to perform the training.

Tensorflow is an open source software library for machine intelligence created by Google in 2015. With Tensorflow, it is possible to define machine learning models, train them with data, and export them. Tensorflow operates with tensors that are multidimensional vectors running through the nodes of a graph.

At the end of the experiments, the performance of each algorithm is analyzed based on the following metrics: accuracy, which is the success rate that the model obtained; precision, which is the rate among those classified as positive; specificity, which is the rate of negative proportion among those that are negative; recall, which is the rate of positive proportion among those who are positive; fmeasure, which is the harmonic mean between precision and recall.

The open source library sklearn [11] is used to measure these performance metrics. According to the documentation of sklearn, it is possible to define the metrics, varying from a minimum of 0 to a maximum of 1 in dimensionless form, as defined in Eq. 1:

$$\begin{cases} precision = \frac{TP}{TP+FP}, \\ recall = \frac{TP}{TP+FN}, \\ fmeasure = \frac{2 \times precision}{precision+recall}, \\ accuracy = \frac{TP+TN}{TP+FN+TN+FP}, \\ specificity = \frac{TN}{TN+FP}, \end{cases} \tag{1}$$

where TP is the number of true positives, FP are the false positive, TN is the number of true negatives and FN are the false negatives.

The model is tested with 5 learning algorithms: Stochastic Gradient Descent (SGD), AdaGrad, RMSprop, Adam and Adamax. The results of each approach are then compared and ordered according to their performance in relation to accuracy and error.

4.1 Retrieving Data

The selection of training images is based on the *datasets* of the *Kaggle*, *pyimagesearch* images, on images obtained by *Google Images* and *dataset* [8] as described below. About 1.500 images, organized into 5 categories, are divided into 17 subcategories: domestic animals, such as dogs, cats and birds (264 images) [5]; electronics, such as cell phones, notebooks and TVs (366 images) [8]; games, such as consoles and game scenes (188 pictures); vehicles, such as cars, motorcycles, bikes and planes (386 images) [4]; clothing, such as pants, dresses, blouses, shoes and tennis (298 images) [10].

4.2 Creating the Model

The images go through processing techniques aiming to increase the performance of the classifier model and to reduce the computational effort required to process them. The set of images is divided in 80% for training and 20% for testing. The images are resized to a standard 96px wide and 96px high. The intensity of the pixels is normalized by dividing their scalar values by 255, varying in a range of 0 to 1. The matrices of the images are represented with a dimension of $96 \times 96 \times 3$ with 3 color channels according to the color scale RGB.

The CNN architecture is based on [13]. The network architecture is defined with 7 layers of 2D convolution, with reduction of the dimensionality of the matrices using the max pooling function with dimension 2×2 and 3×3, with 3×3 kernel filters and activation functions ReLU. The fully connected final layer is created using the softmax function for classification. Six layers are used with the sequence of convolution operations, activation function ReLU and pooling. Dropout layers are added to the network to control the overfitting problem and the increased error problem in the training. The Dropout adjusts a certain

amount of layer activations based on probabilistic decision making, acting upon activation by a predefined threshold. The network must be able to classify data correctly even without counting on some of its activations at each iteration. This technique becomes important to avoid the overfitting process.

The network then processes the images in the following steps: (1) Obtaining the characteristics: In this first phase the image is processed with the convolutional filters to detect the degree of pertinence of the image with the search characteristics; (2) Positioning the characteristics: In this second phase, the feature maps define what exists in the image and where each characteristic is.

5 Training Results

With the data set prepared, the training routines are established by executing in the *docker* container using the *keras* library with the *python* programming language. The code is run using *Jupyter*, which is a web-based interactive development environment.

5.1 Training Using SGD Algorithm

In the Gradient Descent (GD) algorithm, it is necessary to go through all the samples in the training set to perform parameter updating in a specific iteration. There are three types of Gradient Descent: Batch Gradient Descent, Stochastic Gradient Descent and Mini-batch Gradient Descent.

In the Stochastic Gradient Descent (SGD) algorithm, a training sample is used to update the parameters in a specific iteration. A few samples are selected randomly instead of the whole data set for each iteration. In GD, there is a batch, which denotes the total number of samples from a dataset, that is used for calculating the gradient for each iteration. In typical GD optimization, like Batch Gradient Descent, the batch is taken to be the whole dataset. With large training samples, the use of GD can take a long time. Using SGD, it will be faster, because the performance improvement occurs from the first sample.

Training with the SGD algorithm is carried out using 20 epochs. In Fig. 2, it is possible to observe the decay of the error along the time for the training and testing. Figure 2 shows the error reaching 0.45 for training and 0.46 for test. The Test error curve has some punctual instabilities, but both curves of training and testing have good general relative stability in error decay and in accuracy increasing. In the Fig. 3, it is possible to visualize the growth of training accuracy and validation accuracy reaching performance of 80.80% for training and 82.27% for test. In the Fig. 4, we can see the increase of the hit rate with the visualization of a stable increase of the metrics Fmeasure, precision and recall showing that the relevant results could be correctly classified by the algorithm.

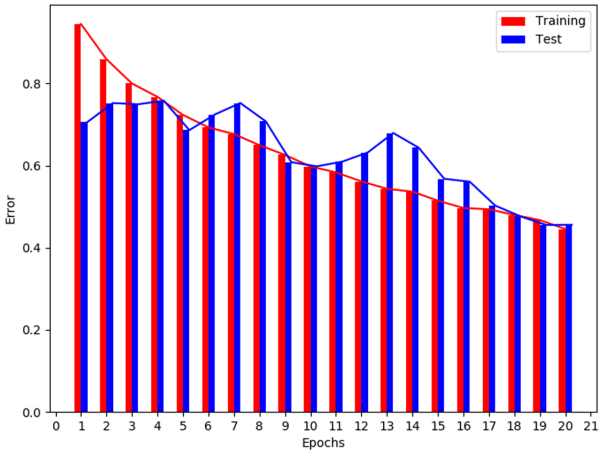

Fig. 2. Error decay during training and test using SGD algorithm

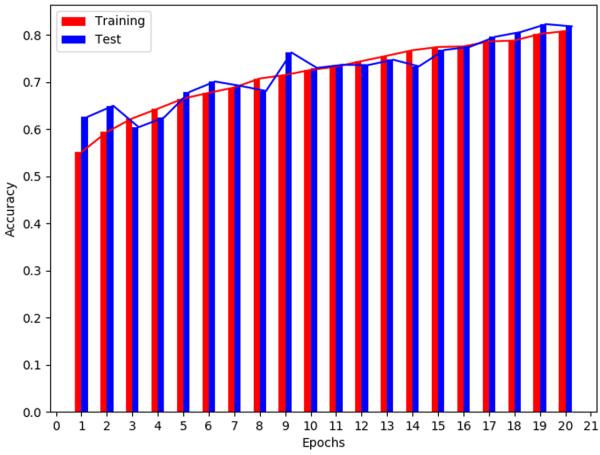

Fig. 3. Accuracy increase during training and test using SGD algorithm

5.2 Training Using AdaGrad Algorithm

The AdaGrad algorithm offers improvements, reducing the learning rate of the weights that receive high gradients and at the same time increasing the rate of learning in the updates of weights. In the AdaGrad algorithm, the learning rate is normalized by the root of the sum of the gradients of each parameter squared.

The results of the trainings and test process are presented using the algorithm AdaGrad with 15 epochs and initial learning rate of 0.01. In Fig. 5, we can see the error decay along the epochs for both training and testing curves. Figure 5 shows some instabilities and peaks for the test curve decay reaching 0.26 and stability for training curve reaching 0.24. Figure 6 shows the growth of training

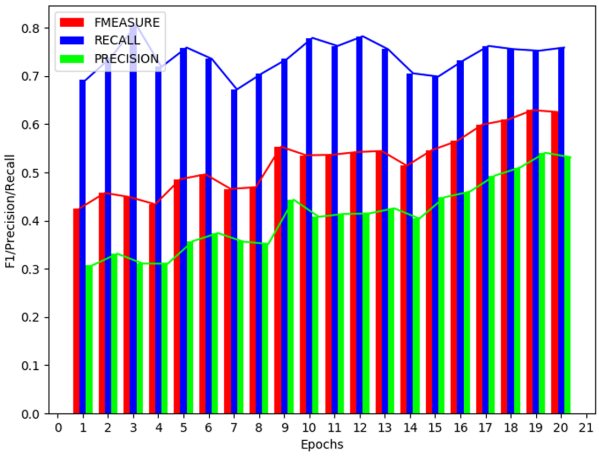

Fig. 4. Metric curves for Fmeasure, Precision, Recall using SGD algorithm

accuracy and validation accuracy reaching performance of 90.21% for training and 89.5% for test offering better results than SGD. In Fig. 7, it is possible to observe an unstable increase of the metrics Fmeasure, Precision and Recall.

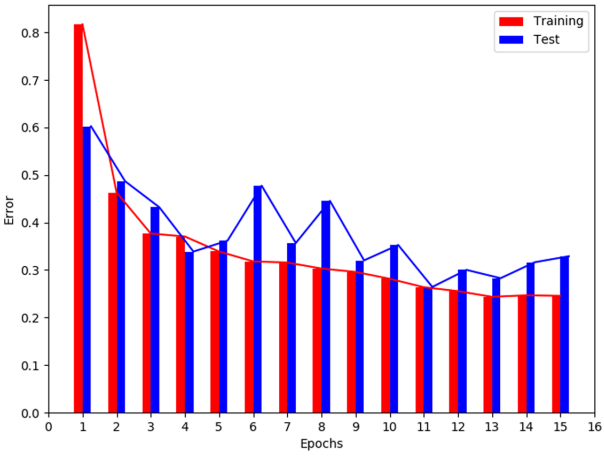

Fig. 5. Error decay during training and test using AdaGrad algorithm

5.3 Training Using RMSprop Algorithm

RMSprop is an algorithm that calculates the magnitudes of the gradients for each parameter and they are used to modify the learning rate individually before applying the gradients. This algorithm modifies the AdaGrad to achieve better

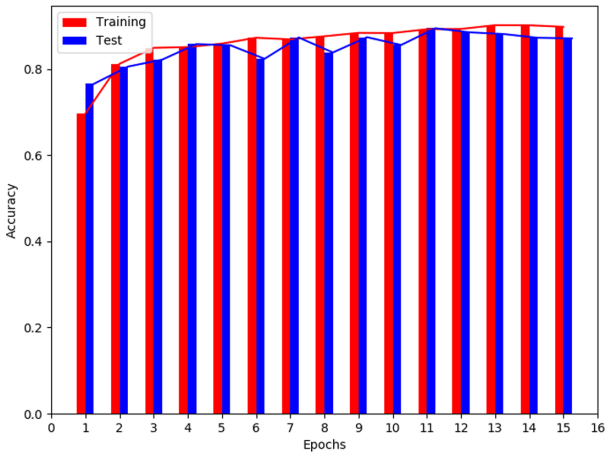

Fig. 6. Accuracy increase during training and test using AdaGrad algorithm

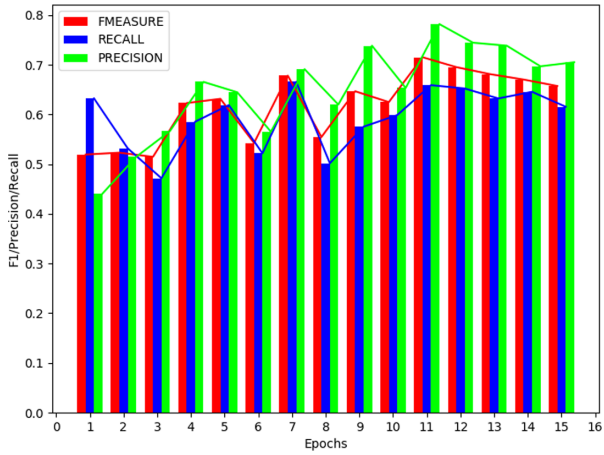

Fig. 7. Metric curves: FMeasure, Precision, Recall using AdaGrad algorithm

performance. The training is performed using the RMSprop with 20 epochs and initial learning rate of 0.001. Figure 8 shows the error decaying to a minimum of 0.20 for training and 0.37 for testing. Both curves of training and testing have good general relative stability in accuracy increasing. In Fig. 9, we can see an unstable increase of the metrics Fmeasure, Precision and Recall. The increase in these metrics along epochs reflects an increase in hit rate. In Fig. 10, the growth in accuracy is shown, reaching a performance of 92.19% for training and 89.16% for testing.

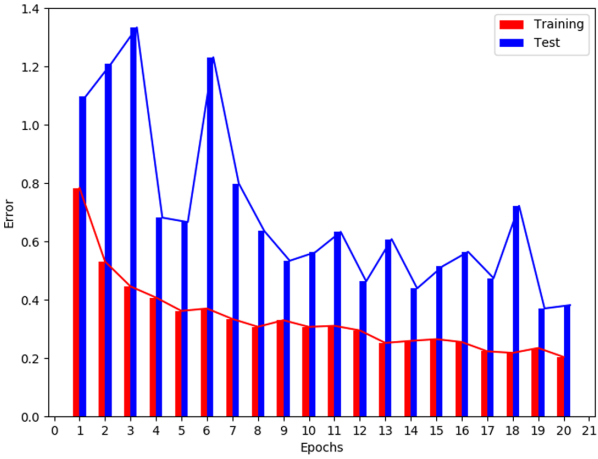

Fig. 8. Error decay during training and test using RMSProp algorithm

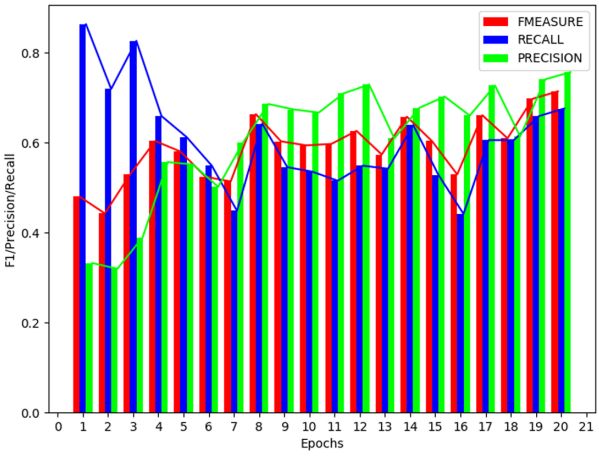

Fig. 9. Metric curves: Fmeasure, Precision, Recall using RMSprop algorithm

5.4 Training Using Adam Algorithm

The Adam training algorithm is a method derived from the RMSprop method that sets the Adagrad method so that the learning rate does not decrease aggressively. The Adam method's contribution is to add a moment in the upgrade and smooth out gradient noises before doing this operation. It inherits from the RMSprop the addition of a decay rate in the sum of the gradients of each parameter while the learning rate is reduced with each step. This method can also be defined as a first order gradient-based optimization of stochastic objective functions.

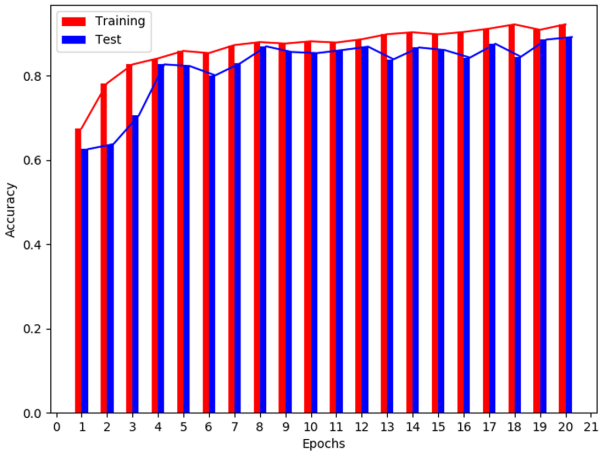

Fig. 10. Accuracy increase during training and test using RMSprop algorithm

The Adam algorithm is run using 50 epochs. Figure 11 shows the error decaying to minimum of 0.12 for training and 0.23 for test showing the good performance of this algorithm. Test error curve has some occurence of instabilities with peaks probability related to some hyperparameter or to some random data chosen in the test step. Figure 12 shows the growth of accuracy, reaching performance of 95.36% for training and 92.12% for test showing a high performance in both steps. In Fig. 13, it is possible to observe some valleys in the increase of the metrics FMeasure, precision and recall. Increasing these metrics along epochs reflects the increase in hit rate. Both curves for training and test have good general relative stability in accuracy increasing.

5.5 Training Using Adamax Algorithm

The Adamax is a variant algorithm of Adam where the second order moment is replaced by the moment of infinite order. Figure 14 shows the error reaching 0.23 for training and 0.36 for test. In Fig. 15, it is possible to visualize the growth of training accuracy and validation accuracy reaching performance of 90.27% for training and 86.15% for test. All curves of error, accuracy, FMeasure, precision and recall are very stables for this algorithm. In Fig. 16, we can observe the increase of the hit rate with the visualization of stable increasing of the metrics Fmeasure, precision and recall.

5.6 Comparative Results

Table 1 shows the maximum accuracy and minimum error for the algorithms used in training and Table 2 shows the corresponding for testing. In addition, data and learning rates are presented for each procedure. In Table 1, the worst accuracy was 80.80% for the SGD algorithm and the two highest accuracy were

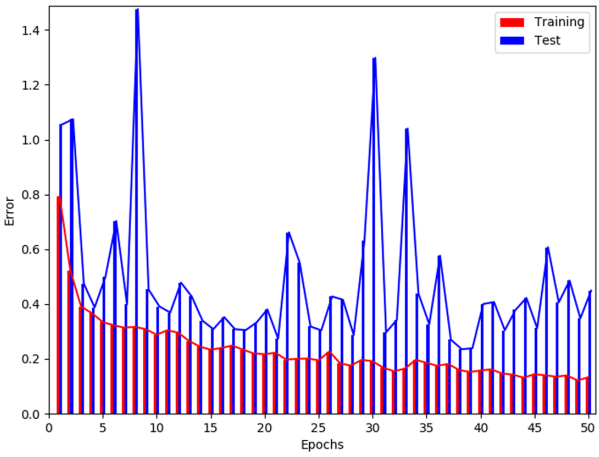

Fig. 11. Error decay during training and test using Adam algorithm

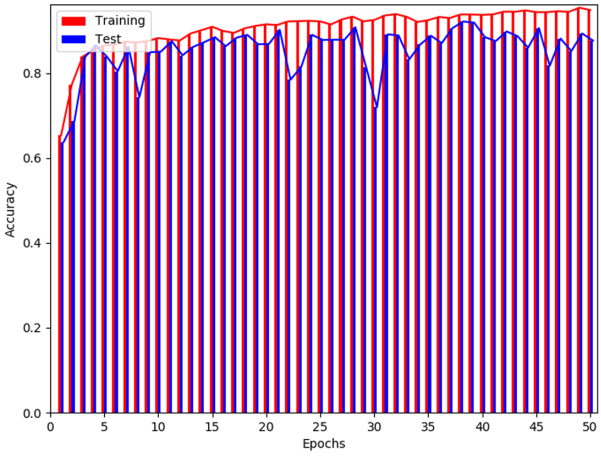

Fig. 12. Accuracy increase during training and test using Adam algorithm

92.19% and 95.36% for the RMSProp and Adam algorithm respectively. The smallest errors observed for training are 0.20 for the RMSProp and 0.12 for the Adam algorithm. Table 2 shows the worst accuracy was 82.27% for the SGD algorithm and the two highest accuracy are 89.50% and 92.10% for the AdaGrad and Adam algorithm respectively. The smallest errors observed for the test are 0.26 for AdaGrad and 0.23 for the Adam algorithm.

Table 3 shows the results of the probabilities of belonging to each category. The tests are presented using the Adam algorithm, which offers the best performance on classifying images of categories animals, electronics, games, vehicles and clothes. It is noteworthy that the performance to classify the categories

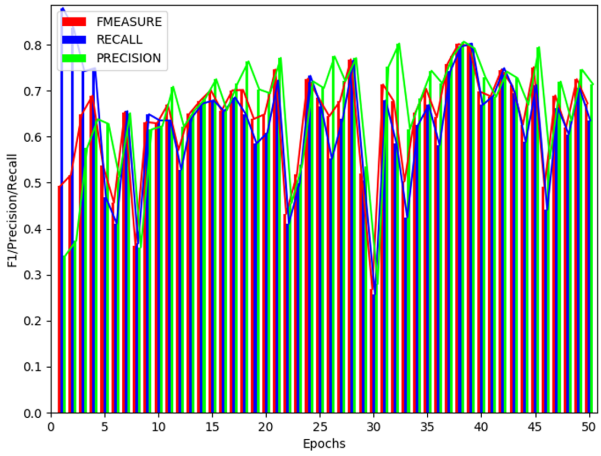

Fig. 13. Metric curves: Fmeasure, Precision, Recall using Adam algorithm

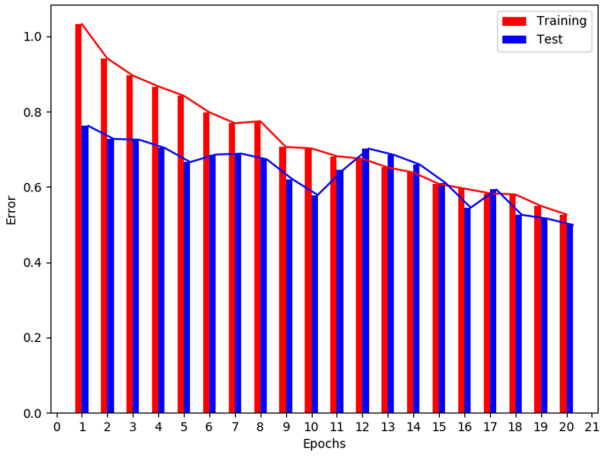

Fig. 14. Error decay during training and test using Adamax algorithm

Table 1. Training performance metrics

Algorithm	Epochs	LR	Max Acc. (%)	Min Error
SGD	20	0.001	80.80	0.45
AdaGrad	15	0.01	90.21	0.24
Adamax	20	0.002	90.27	0.23
RMSProp	20	0.001	92.19	0.20
Adam	50	0.001	95.36	0.12

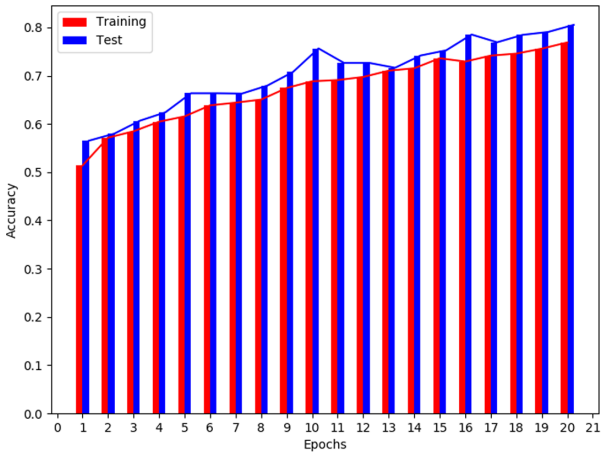

Fig. 15. Accuracy increase during training and test using Adamax algorithm

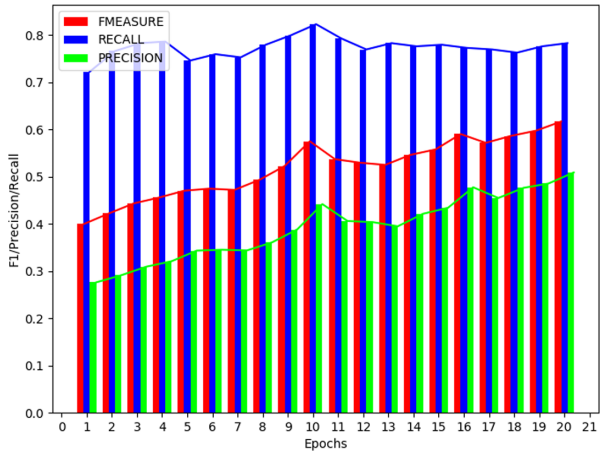

Fig. 16. Metric curves: Fmeasure, Precision, Recall using Adamax algorithm

Table 2. Testing performance metrics

Algorithm	Epochs	L.R.	Max Acc. (%)	Min Error
SGD	20	0.001	82.27	0.46
Adamax	20	0.002	86.15	0.36
RMSProp	20	0.001	89.16	0.37
AdaGrad	15	0.01	89.50	0.26
Adam	50	0.001	92.10	0.23

chair, garden and tower presents maximum probabilities of 66.88%, 0% and 38.42%, respectively, demonstrating that these classes were not used in training. Several images from the trained categories presented performances above 98%, indicating good generalization capacity.

Table 3. Result of images classification in percentage for 5 categories

Image	Anim.	Eletr.	Games	Veic.	Clothes
Dog	99.77	0.00	0.01	0.09	0.00
Cat	99.37	0.01	0.05	0.64	0.07
Bird	99.22	0.05	0.14	0.16	0.07
Notebook	0.12	99.59	0.00	0.08	1.87
TV	0.04	82.62	0.03	66.50	0.14
Cellphone	0.00	99.98	0.00	0.01	0.01
Ps4	0.01	3.07	95.84	0.01	10.32
Car	0.00	0.05	7.16	96.92	0.21
Moto	0.00	0.47	0.80	97.53	3.47
Bike	0.00	0.06	0.08	98.63	4.04
Plane	0.02	0.02	0.05	98.78	1.31
Shoes	0.23	1.33	0.21	55.13	95.92
Dress	1.25	0.00	0.00	0.19	99.70
Pants	12.02	2.34	0.28	0.31	85.49
Chair	0.83	3.17	0.04	6.29	66.68
Garden	0.00	0.00	0.00	0.00	0.00
Tower	0.01	32.76	38.42	1.73	1.78

6 Conclusions

In this work, a convolutional neural network is used as a computational intelligence technique to predict the interest of social network users, in certain categories of products, using image classification. This analysis is valid as a source of knowledge in the segmentation of consumers by interest.

Optimizers and training algorithms are a crucial part of the artificial neural network. Understanding their performance can help choosing the best option for the applications. The knowledge of how hyperparameters can influence the performance of the CNN is very important when training the network. The comparison of the training algorithms SGD, AdaGrad, Adamax, RMSProp and Adam allows us to verify that *Adam* obtains higher accuracy and performances than the others. After completion of all the experiments, when classifying some images of the defined categories, it is possible to obtain satisfactory classification performances as presented in the graphs of performance metrics.

For future work, we suggest to vary the images categories in order to obtain classifications with greater generalization capacity. Another suggestion is to go through the images of users of a certain social network in order to validate on a larger set of images.

References

1. Ajeet Ram Pathak, M.P., Rautaray, S.: Application of deep learning for object detection. Procedia Comput. Sci. **132**, 1706–1717 (2018). https://doi.org/10.1016/j.procs.2018.05.144, http://www.sciencedirect.com/science/article/pii/S1877050918308767. International Conference on Computational Intelligence and Data Science
2. Aslam, S.: Omnicoreagency: Instagram statistics & facts for 2019 (2019). https://www.websitehostingrating.com/instagram-statistics/. Instagram by the numbers: usage stats, demographics and facts you need to know
3. Cinar, Y.G., Zoghbi, S., Moens, M.: Inferring user interests on social media from text and images. In: 2015 IEEE International Conference on Data Mining Workshop (ICDMW), pp. 1342–1347, November 2015. https://doi.org/10.1109/ICDMW.2015.208
4. Image classification: car, motorbike, bicycle (2012). https://www.kaggle.com/c/image-classification2/data
5. Competition distinguish dogs from cats (2014). https://www.kaggle.com/c/dogs-vs-cats/overview
6. Keyes, D.: Instagram rolls out shoppable posts for more merchants (2017). https://www.businessinsider.com/instagram-rolls-out-shoppable-posts-for-more-merchants-2017-10
7. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
8. Li, F.F., Andreetto, M., Ranzato, M.A.: Caltech101 image dataset (2003). http://www.vision.caltech.edu/Image_Datasets/Caltech101/
9. Maksimov, A.: Docker container with jupyter, matplotlib, pandas, tensorflow, keras and opencv (2019). https://hub.docker.com/r/amaksimov/python_data_science/
10. Rosebrock, A.: Multi-label classification with keras (2018). https://www.pyimagesearch.com/2018/05/07/
11. SciKit-learn.org: Machine learn in pyton (2019). https://scikit-learn.org/stable/
12. Segalin, C., Perina, A., Cristani, M., Vinciarelli, A.: The pictures we like are our image: continuous mapping of favorite pictures into self-assessed and attributed personality traits. IEEE Trans. Affect. Comput. **8**(2), 268–285 (2017). https://doi.org/10.1109/TAFFC.2016.2516994
13. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (2015)
14. You, Q., Bhatia, S., Sun, T., Luo, J.: The eyes of the beholder: gender prediction using images posted in online social networks. In: 2014 IEEE International Conference on Data Mining Workshop, pp. 1026–1030, December 2014. https://doi.org/10.1109/ICDMW.2014.93