Web-Based Application for Digital Image Classification using TensorFlow

Ivan Videv[1] and Ivo Draganov[2]

Abstract – **In this paper, we investigate the capabilities of applying TensorFlow for digital image classification. A complete web-based application is created with easy portability to various operating systems, which cover all the functionalities incorporated into the library. It follows the client-server model and take advantage of all the options for tuning, such as random crop, random scaling, etc. Testing with a set of 772 images of 3 types of objects taken in natural environment prove the applicability of the framework for classification purposes. The system can be extended to solve general and strictly specialized tasks.**

Keywords – **Digital Image Classification, TensorFlow, Deep Learning, Convolutional Neural Network.**

I. Introduction

The algorithms for digital image classification employ two types of preliminary training schemes with labelled datasets:

- Unsupervised learning – the input information is not divided and described prior to its use – there are no categories for association;
- Supervised learning – there are means (teacher) for describing the initial data for training, so the categories exist before the process takes place. Classification and regression are the two main stages here.

K-means clustering [1] is one of the very popular unsupervised learning techniques. Main measure for pertaining of an object to certain cluster is the distance to its center:

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} \left(x_j - \mu_i\right)^2. \quad (1)$$

The number of clusters is initially known but their centers can change their position during training.

Other kind of unsupervised learning is the associative type [2]. Its primary application is directed towards finding relations between variables in databases with large volume. If a certain sequence occurs more frequently than others do, then its sub-sequences would also occur more frequently that other shorter series.

Unsupervised learning is often realized by regression analysis [3]. It relies on estimating the dependence between input and output parameters the latter of which are not discrete.

Decision trees [4] are the foundation of another model where the main feature is hierarchy of rules represented by connected branches. Each node of the tree is a test for a particular property of the data.

Classification and regression could be embedded together and one such approach that takes benefit of this incorporation is the k-Nearest Neighbors (k-NN) [5]. The output data are k in number closest samples. They are property of the object to a certain class, which is expressed more often among all the other k classes.

Support Vector Machines (SVM) [6] are another example of unsupervised learning algorithm. The idea laying behind this method is to find hyper-plane, which separates most accurately two classes of data. If the data clusters are not separable by linear function in n-dimensional space, a transition to (n+1)-dimensional space is made and then a search for a linear function is attempted.

Artificial Neural Networks (ANN) [7] are another example of self-adaptive structures that make unsupervised learning possible. They consist of a network of artificial neurons modeling the real neurons from the human brain. Interaction with the outside world relies on adaptation of input weights depending on expected output obtained through properly selected activation function.

An example of specialized neural network modelling the perception ability of the human visual system is the neocognitron [8]. There are input layer of cells imitating the operation of the photosensitive region from the human cortex followed by cascade of modular structures built from 2 types of cells – simple and complex. The first type react to particular (fixed) stripes of light while the second are sensitive to wider variety of linearly described patterns of illumination. There is a third type of cells as well – the hyper-complex one, which are capable of making analysis of geometric forms.

Hopfield networks [9] introduce bidirectional connections among neurons allowing repetitions. Asynchronous and synchronous mode of updating are possible depending on whether a single neuron, randomly selected according to a rule, or all neurons are going to be updated at once.

Deep learning [10] is the next point of the natural evolution of the classification techniques without supervisor. Since SVM is faster in execution that ANN until recently it was the main approach for practical implementations. With the growth of computing power and the introduction of GPUs (Graphical Processing Unit) for parallel processing ANN become attractive for wider use too. Deep Neural Networks (DNN) have higher levels of abstraction – multiple hidden layers that allow detection of properties going closer to the output. Thus, modelling of more complex data with fewer computations becomes possible.

Neural networks could be constructed with memory [11] using Long Short-Term Memory (LSTM) cells. There is resemblance to Hopfield networks for them. These cells

[1]Ivan Videv is with the Faculty of Telecommunications at Technical University of Sofia, 8 Kl. Ohridski Blvd, Sofia 1000, Bulgaria, E-mail: noxicul@gmail.com.

[2]Ivo Draganov is with the Faculty of Telecommunications at Technical University of Sofia, 8 Kl. Ohridski Blvd, Sofia 1000, Bulgaria, E-mail: idraganov@tu-sofia.bg

memorize data for a given time interval. They have body, input, output and delay gates. Each gate may be modelled as a traditional artificial neuron calculating its output by an activation function through input weights. These networks reveal very good recognition results for speech and handwriting.

Convolutional Neural Networks (CNN) [12] use Deep Learning and multiple hidden layers are contained within its structure. They do not form cycles in contrast to the networks with repetitions. Their operational principle relies on minimal amount of preprocessing. A series of filters are applied directly over the image and properties are derived further used for classification.

In this paper a newly developed web-based application is presented for digital image classification using TensorFlow. The purpose of the study is to test the applicability of the environment to the average non-professional user running it from a personal Desktop computer. In Section II the system architecture and the client design are given. Experimental results are included in Section III followed by discussion in Section IV. Then, a conclusion is made in Section V.

II. Implementation Description

A. System architecture

The general architecture of the testing system is given in Fig. 1. The server part is run over Ubuntu Server v. 16.04. The main processing functionalities are included inside php script modules executed by the PHP engine at the requests of the user through the client side. It is done from a web browser via a communication back and forth with the web server – in this instance Apache. Uploading images for classification is made available by FileZilla ftp client to the vsftpd ftp server. All temporary data and the personal information of the users is contained inside the MySQL database.

TensorFlow [13] is essential part of the proposed architecture. It is a library, developed by Google Inc., for mathematical computations using graphs and tensors. Constructing different configurations for neural network implementations is one of its main applications. Some of the qualities it possess is the strong parallelization, multiple central and graphical processing units support, and dedicated API for data exchange with the system.

Inside the proposed implementation, the Inception-v3 model [14] of convolutional neural network is used (Fig. 2). A successor of the GoogleNet architecture it is preliminary trained into 1000 classes [15] on a dedicated server by the development team. Transfer learning method is applied for knowledge transfer from solving a particular problem to propose a solution to another one.

The initial training is accomplished with the ImageNet database and then the outer layer of the network is displaced. There are 42 layers inside with included factorization – the traditional 7x7 convolution from Inception-v1 is factorized to 3x3 convolutions. In addition, there is normalization of the supporting classifiers, which redound for the better convergence of the deeper layers of the neural network. Re-training only the outer layer of the model saves execution time because the necessity of retransforming the inner layers to conform new shapes becomes obsolete.

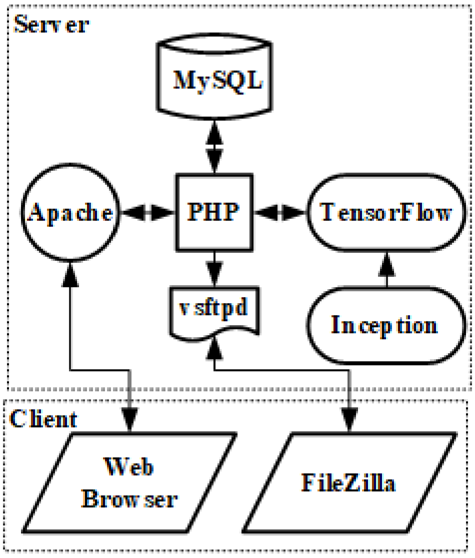
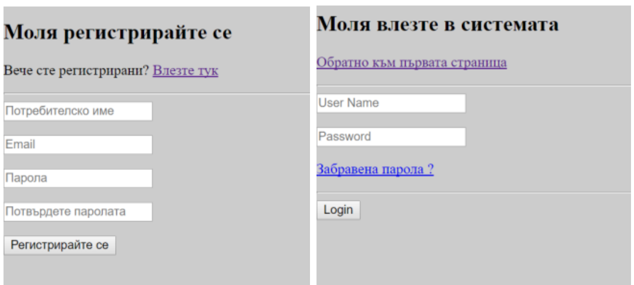


Fig. 1. Classification system architecture

Fig. 2. Inception-v3 structure, [14]

B. Client design

The graphical user interface for registering and admitting a client to the system is given in Fig. 3.



a) b)

Fig. 3. Registration panel a) and login panel of the client interface

Settings selection and initiating the training of the outer layer is placed over another panel of the client application (Fig. 4).

The following parameters could be defined:

- Number of training steps;
- Random crop size;
- Random intensity value;
- Random scale size;

- Enabling/Disabling a flip;
- Testing set size;
- Validation set size;
- Training rate;
- Training set size;
- Enabling/Disabling training over the full group;
- Enabling/Disabling displaying wrongly classified images.

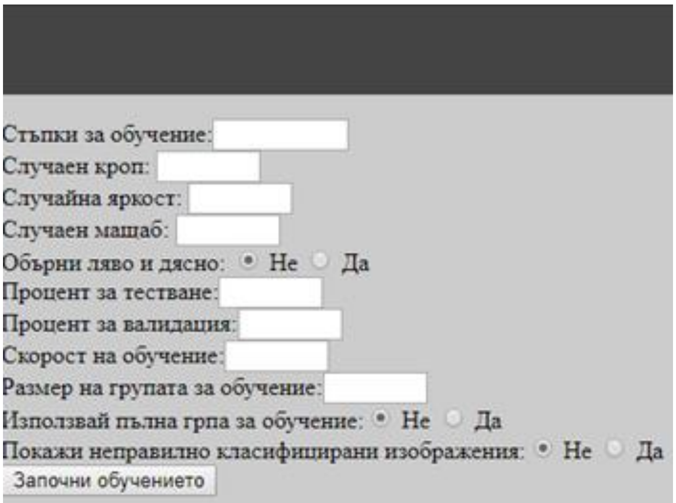


Fig. 4. Settings selection panel

III. Experimental Results

Image test set consists of 3 groups of photos randomly selected from Google Images. They all contain fruits at different stages of ripening in various backgrounds:

- Grapes – 315 images;
- Cherry – 375 images;
- Pineapple – 82 images.

Grapes and cherry are very close in appearance and significantly different from pineapple so it is expected to have lower classification accuracy between the first two groups. Additional image of a car over ocean background is also included in the testing to evaluate the behavior of the system. Totally of ten test images (3 of grapes, 3 of pineapple, 3 of cherry and 1 of the car) are used not being a part of the training set. The hardware test platform is built with Intel Core i5-4440 3.1 GHz CPU with 4 cores, 2 GB RAM. In Fig. 5 are presented some of the training images from the three groups.

Fig. 5 Sample training images

Classification confidence for all 10 test images is presented in Fig. 6 at different training steps.

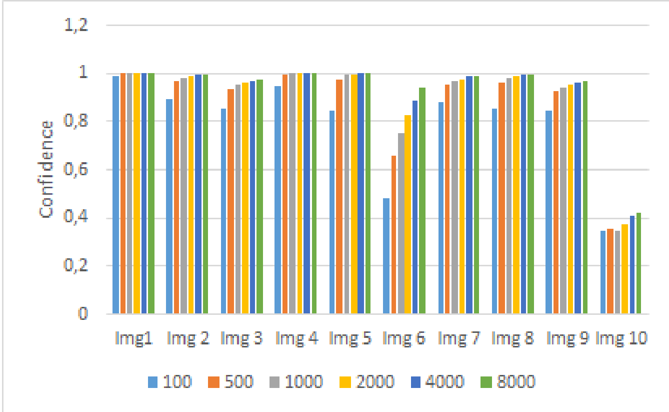


Fig. 6. Classification performance of the system

Initially 4000 steps are performed for the training phase. The execution time is 5 min 30 sec and the overall accuracy is 97.1 %. Comparison between the confidence levels with and without the “random crop”, function is shown in Fig. 7. The random crop introduces random selection of the elements of the tensors describing the images, so a partial matching can be tried.

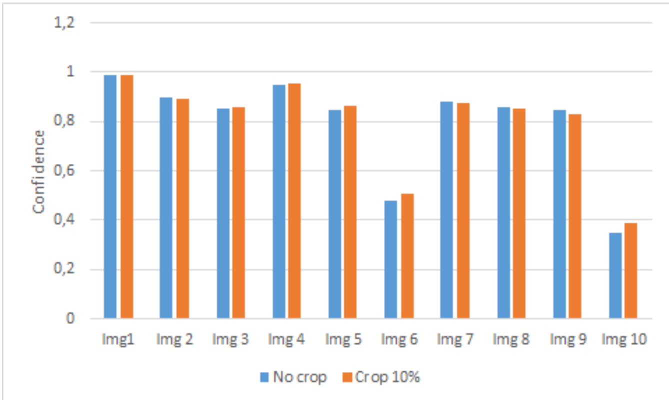


Fig. 7. Influence of the “random crop” function on the success rate of the system

There is one of the test images (Image 6, containing grapes) which returns slightly different results from the other images on average (Fig. 7 and 8).

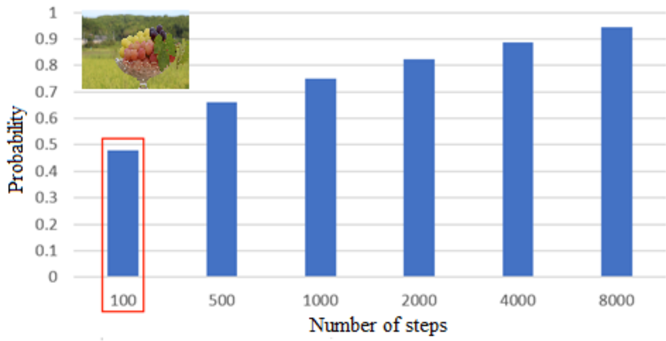


Fig. 8. Lower matching rate for Image 6

The probability for it is 47% at 100 steps, rising up to 66% at 500 steps. Probable cause for its lower matching rate is the mutual disposition with surrounding objects. These are the leaves and grass in the background with different texture and dominant size.

When objects of completely different type are passed to the system, such as the car from Fig. 9, the results are drastically changed. The probabilities for the three classes become - 40.5%, 31.1%, and 28.2%. Still the most probable match is with very low rate – below 50%.

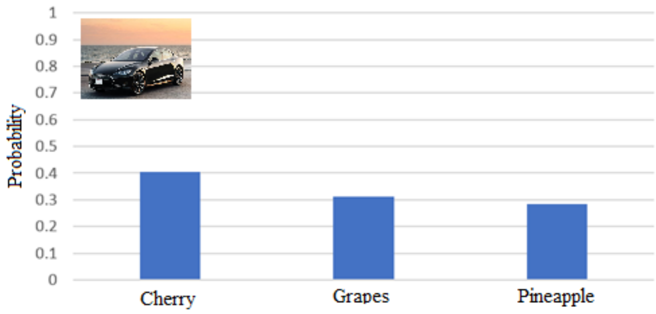


Fig. 9. Matching results for image with unfamiliar objects

IV. DISCUSSION

Efficient use of convolutional neural networks for visual classification purposes is a hard task because of the compromise that need to be done between accuracy of the model and the computational resource involved.

The experimental results show that the trained classifier copes well at the highest test level of 100 steps (iterations). Here is only one wrongly classified image – 6 of 9 passed to the input aside from the train set. The mismatch case possess very low confidence level of 0.47983. At 500 steps, the same image is correctly assigned with higher confidence of 0.66005. The execution time here is below 30 sec. Confidence comparable to that of the other test inputs – 0.94296 is achieved for 8000 steps for image 6. None of the other test images is categorized better in terms of confidence with the considerable rise of the number of steps – 0.95 is the typical value for 2000 steps.

Refining the model in 100 steps and 10 “random crop” takes almost an hour on the test platform. Tangible improvement has taken place with image 6 properly classified. Still, the results prove less accuracy from those obtained at 500 steps and without “random crop” where training takes 30 sec.

Adding noise to the images, as additional option, may be used to increase the accuracy even higher that is not possible only by incrementing the number of steps. TensorFlow supports Graphical Processing Units. In addition, it could be compiled with activated SSE4.1/SSE4.2/AVX/AVX2/FMA instructions sets.

In multiple cases, the accuracy of the model at 2000 steps achieved in 2 to 3 minutes of training is completely satisfactory. Common Desktop and mobile computers have the appropriate hardware for the purpose.

V. CONCLUSION

In this paper a web-based application for digital image classification is presented using TensorFlow. It proves to supply recognition rates enough for most practical cases. Need of proper training with account on the temporal accuracy at each iteration is a key phase of the whole tuning of the system. Execution times appear to be also satisfactory for the particular user working on a mid-class Desktop computer. Further development of the system suppose including a proper web-based interface for image category presentation, organizing topic based albums, etc.

REFERENCES

[1] J. A. Hartigan, M. A. Wong, ”Algorithm AS 136: A k-Means Clustering Algorithm”, Journal of the Royal Statistical Society, Series C (Applied Statistics), vol. 28, no. 1, pp. 100-108, 1979.

[2] J. J. Jenkins, W. A. Russell, “Associative Clustering during Recall”, The Journal of Abnormal and Social Psychology, vol. 47, no. 4, pp. 818, 1952.

[3] S. Menard, *Applied Logistic Regression Analysis*, SAGE publications, 2018.

[4] S. R. Safavian, D. A. Landgrebe, “A Survey of Decision Tree Classifier Methodology”, IEEE Transactions on Systems, Man, and Cybernetics, vol. 21, no. 3, pp. 660-674, 1991.

[5] S. A. Dudani, “The Distance-Weighted k-Nearest-Neighbor Rule”, IEEE Transactions on Systems, Man, and Cybernetics, vol. 4, pp. 325-327, 1976.

[6] M. A. Hearst et al. “Support Vector Machines”, IEEE Intelligent Systems and Their Applications, vol. 13, no. 4, pp. 18-28, 1998.

[7] R. J. Schalkoff, *Artificial neural networks*, New York, McGraw-Hill, 1997.

[8] K. Fukushima, S. Miyake, *Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Visual Pattern Recognition. In: Competition and Cooperation in Neural Nets*, Berlin, Heidelberg, Springer, pp. 267-285, 1982.

[9] J. J. Hopfield, “Neural Networks and Physical Systems with Emergent Collective Computational Abilities”, Proceedings of the National Academy of Sciences, vol. 79, no. 8, pp. 2554-2558, 1982.

[10] Y. Lecun, Y. Bengio, G. Hinton, “Deep Learning“, Nature, vol. 521, no. 7553, pp. 436, 2015.

[11] A. Graves, A.-R. Mohamed, G. Hinton, “Speech Recognition with Deep Recurrent Neural Networks”, In Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2013), pp. 6645-6649, Vancouver, Canada, 2013.

[12] A. Krizhevsky, I. Sutskever, G. Hinton, “Imagenet Classification with Deep Convolutional Neural Networks”, In: Advances in Neural Information Processing Systems, pp. 1097-1105, 2012.

[13] M. Abadi et al. “TensorFlow: A System for Large-Scale Machine Learning”, In: OSDI, pp. 265-283, 2016.

[14] C. Szegedy et al. “Rethinking the Inception Architecture for Computer Vision”, In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 2818-2826, Seattle, WA, USA, 2016.

[15] https://github.com/WillKoehrsen/Machine-Learning-Projects/blob/master/pre%20trained%20models/imagenet_class_names.txt