Object recognition in images is used in many areas of practical use. Very often, progress in its application largely depends on the ratio of the quality of object recognition and the required amount of calculations. Recent advances in recognition are related to the development of neural network architectures with a very significant amount of computing that are trained on large data sets over a very long time on state-of-the-art computers. For many practical applications, it is not possible to collect such large datasets for training and only computing machines with limited computing power can be used. Therefore, the search for solutions that meet these practical restrictions is relevant. This paper reports an ensemble classifier, which uses stacking in the second stage. The use of significantly different classifiers in the first stage and the multilayer perceptron in the second stage has made it possible to significantly improve the ratio of the quality of classification and the required volume of calculations when training on small data sets. The current study showed that the use of a multilayer perceptron in the second stage makes it possible to reduce the error compared to the use of the second stage of majority voting. On the MNIST dataset, the error reduction was 29–39 %. On the CIFAR-10 dataset, the error reduction was 13–17 %. A comparison of the proposed architecture of the ensemble classifier with the architecture of the transformer-type classifier demonstrated a decrease in the volume of calculations while reducing the error. For the CIFAR-10 dataset, an error reduction of 8 % was achieved with a calculation volume of less than 22 times. For the MNIST dataset, the error reduction was 62 % when winning by the volume of calculations by 50 times

Keywords: multilayer perceptron, neural network, ensemble classifier, weights, classification of images

UDC 681.3.07: 004.8
DOI: 10.15587/1729-4061.2022.258187

USING A NEURAL NETWORK IN THE SECOND STAGE OF THE ENSEMBLE CLASSIFIER TO IMPROVE THE QUALITY OF CLASSIFICATION OF OBJECTS IN IMAGES

Oleg Galchonkov
Corresponding author
PhD, Associate Professor*
E-mail: o.n.galchenkov@gmail.com
Mykola Babych
PhD, Associate Professor*
Andrii Zasidko*
Serhii Poberezhnyi*
*Department of Information Systems
Institute of Computer Systems
Odessa Polytechnic National University
Shevchenka ave., 1, Odessa, Ukraine, 65044

Received date 11.04.2022
Accepted date 13.06.2022
Published date 30.06.2022

***How to Cite**: Galchonkov, O., Babych, M., Zasidko, A., Poberezhnyi, S. (2022). Using a neural network in the second stage of the ensemble classifier to improve the quality of classification of objects in images. Eastern-European Journal of Enterprise Technologies, 3 (9 (117)), 15–21. doi: https://doi.org/10.15587/1729-4061.2022.258187*

1. Introduction

One of the main trends in the development of technology in the modern world is the increasing introduction of artificial intelligence approaches in all areas of human activity. One of the areas where significant success has been achieved is the recognition of objects in images. This is due to the high practical effectiveness of modern methods of object recognition in such areas as, for example, unmanned vehicles, medicine, and robotic production. Requirements for the quality of object recognition are constantly growing due to the growing need for practical use. However, the main direction of improving the indicators is the development of deeper and deeper neural networks, the training of which is carried out on increasingly large data sets and requires an increasing amount of calculations. Initially, the best results were demonstrated by deep convolutional neural networks, such as described in [1], which in 2012 showed the best results on the ImageNet dataset [2] that contained more than 15 million images. The number of configurable weights in that network exceeds 60 million. The next stage in improving the performance of neural networks was the development of architectures called transformers [3]. One of the most efficient variants of this architecture, ViT-Huge, contains more than 632 million customizable weights. Among the datasets on which that neural network was trained was JFT [4] containing 303 million images. For many practical applications, it is not possible to form such a large set of data. Some relief of the situation is transfer training [5] when a neural network already trained on a data set is taken and completed on a set of data related to the required application. However, the amount of computation to find the output signals of such a neural network remains as large, which makes it difficult to use them in practice. Relevant for practice is the development of such classifier architectures, which, with high classification quality, require a small amount of calculations and can be quickly trained on a small amount of data.

Simultaneously with the development of new architectures of neural networks with an increasing amount of calculations, it turned out that these neural networks have a lot of redundancy. It is possible to reduce the number of weighting coefficients by an order of magnitude without losing the resulting recognition quality [6] or even with its slight improvement [7]. This explains the emergence of new neural network architectures, which, with a significantly smaller number of tunable weights, provide comparable quality of

object recognition in images, compared to more resource-intensive neural networks [8]. The effective integration of such neural networks into ensemble classifiers [9] is an important scientific direction since it opens up new opportunities for constructing classifiers with a better ratio of classification quality and calculation volume.

2. Literature review and problem statement

Conditionally, the structure of the ensemble classifier can be represented in the form of two stages. In the first stage, we have M separate classifiers, at the outputs of which, for each input signal x, a vector of output signals is obtained.

In the second stage, the results of individual classifiers are combined to derive a more accurate result [10]. Methods of constructing the second stage of the ensemble classifier are divided into methods of weighing and meta-learning. Weighting methods show good results when classifiers in the first stage perform the same task and have comparable results. The simplest of these methods is majoritarian voting. In this case, the output signal of each of the classifiers of the first stage is defined as the output number with the maximum value. The output signal of the second stage is defined as the result of finding the maximum number of matches of the output signals of the classifiers of the first stage.

A large number of different modifications of this method are known, which include weighting by characteristics, methods for optimizing the linear combination of basic classifiers (wagging), the use of the naïve Bayesian approach, entropy weighting, and many others [10]. However, these methods are used mainly in economics, and when recognizing objects in images, the majoritarian rule 2 out of 3, 3 out of 4, 3 out of 5, etc. is typically applied. In other words, if the number of coinciding solutions of primary classifiers is more than half the number of classifiers, then this decision is taken as a final solution, otherwise, a random value is taken [11].

The use of meta-learning at the second stage involves using not only the output numbers of the first stage classifiers with maximum values but all the output signals of these classifiers. The most popular approaches from this group are boosting, bagging, and stacking [10]. The success of second-stage training depends significantly on the diversity of models in the first stage. Boosting and bagging imply the use of the same algorithm at the first stage, the variety of models is obtained by forming different subsets from the entire data set and training each of the models on its data set. Stacking involves the use of different algorithms for different models, each of which is trained on a full set of input data. Paper [12] shows that meta-learning provides better performance than majoritarian voting. However, very small datasets were used for comparison in the cited work and simple linear regression was used for stacking. In this regard, the use of a neural network for staking is of interest. To train a neural network, larger datasets than in those [12] are needed but their size should be comparable to that which can be provided in a large number of practical applications. Some of the most popular datasets that meet these conditions are MNIST [13] and CIFAR-10 [14]. MNIST is a set of 60,000 black-and-white images of handwritten digits for training and 10,000 images for testing; each image has a size of 28×28 pixels. CIFAR-10 is a set of 50,000 color images of 10 classes of objects, such as an airplane, horse, dog, etc., and 10,000 images for testing; each image has a size of 32×32 pixels.

Ensuring the effectiveness of stacking in the second stage of the ensemble classifier requires the use of classifiers at the first stage that implement various algorithms. The second requirement for first-stage classifiers is a high quality of classification with a relatively small number of adjustable coefficients. Such requirements are met by a number of classifier structures. In [8], MLP-Mixer is proposed, the architecture of which is built exclusively on perceptrons (MLP). The input images are divided into a lattice of fragments. Further, the data are sequentially processed by *N* MixerLayer blocks. Each of these blocks contains two groups of perceptrons. The perceptrons of the first group process the vertical rows of fragments, and the other group of perceptrons processes the results of the work of the first group of perceptrons horizontally. At the output of the classifier, a fully connected layer is used according to the classical scheme, which generates support signals for each of the feature classes.

Paper [15] proposes the architecture of CCT (Compact Convolutional Transformer). It simultaneously uses blocks of transformers [3] and convolutional neural networks (CNN). It is shown that with the correct choice of dimensionalities, high-quality classification is ensured with a small number of tunable coefficients both on large data sets and on small MNIST and CIFAR-10. In [16], to build the EANet network (External Attention Transformer), three types of blocks were used: MLPs, external attention blocks, and transformers. The mechanism of external attention proposed in this architecture takes into consideration the correlations between all data samples and makes it possible to get a high-quality classification with a small volume of calculations. A similarly high ratio of classification quality and calculation volume characterizes the FNet network [17]. It also has a transformer in its architecture but, in addition, it uses the Fourier transform.

Unlike MLP-Mixer where fragments of the input image are mixed using horizontal and vertical processing, in the FNet network these fragments are mixed using the Fourier transform. At the output of FNet are fully connected layers (perceptron), which form support output signals for all classes of objects. It should be noted that in deep convolutional neural networks, spatial connections are searched simultaneously throughout the image at the same time. The main driving motive in the development of more computationally efficient neural networks is the understanding of the redundancy of convolutional networks. Therefore, the search for spatial connections is not carried out throughout the image at once but by splitting the image into a lattice of fragments and searching for connections between them. In [18], it is proposed to use channel projections and spatial projections with multiplicative gating for this purpose. These neural networks are called gMLP because they are built on the basis of layers of MLP with gating. At the input of gMLP, as well as in previous networks, the image is divided into a lattice of fragments, in the middle, there are L blocks in series, and the output signals of class support are formed by means of a fully connected perceptron.

The architecture of the SwinTr neural network (Swin-Transformer) is based on the search for spatial connections using the self-attention mechanism only inside each of the shifted windows [19]. The use of this network on various data sets showed high classification quality with a relatively small amount of calculations. This is because self-attention is calculated only inside the windows, and not throughout the image, as is done in the standard architecture of the transformer [3].

It should be noted that the neural network architectures proposed in works [8, 15–19] are aimed primarily at classifying objects in large data sets. The characteristics of these networks when working with small data sets as part of ensemble classifiers have not been investigated.

CCT, EANet, FNet, gMLP, MLP-Mixer, and SwinTr neural networks implement original algorithms, which are significantly different from each other and from the classical convolutional neural network (CNN). Therefore, at the first stage of the ensemble classifier, it is advisable to use these networks and CNN, provided that architecture with a small number of configurable coefficients is used.

Thus, at present, a large number of neural networks have been developed that are characterized by the high quality of classification of objects in images with a relatively small amount of tunable coefficients that work according to significantly different algorithms. This opens up the possibility of using them at the first stage of the ensemble classifier, which implements the stacking mechanism at the second stage. However, the use of these networks as part of ensemble classifiers is unexplored. In addition, it seems appropriate to investigate the possibility of using all the output signals of these networks at the second stage of the ensemble classifier within the framework of the stacking mechanism, and not only the maximum ones. To implement stacking, it seems promising to use a neural network, which, firstly, will allow the most complete use of all the outputs of the primary neural networks of the first stage, and secondly, can be trained specifically for the necessary data sets for practice.

3. The aim and objectives of the study

The aim of this work is to build an ensemble classifier that uses modern neural networks at the first stage and implements the stacking mechanism at the second stage using a multilayer perceptron. The classifier should have a relatively small amount of computation and the ability to learn from small datasets MNIST and CIFAR-10. This will make it possible to use such a classifier in many practical applications.

To accomplish the aim, the following tasks have been set:

– to determine the impact of the use of a multilayer perceptron at the second stage of the ensemble classifier on the quality of classification and compare it with the use of majoritarian voting at the second stage;

– to compare the characteristics of the constructed ensemble classifier and transformer.

4. The study materials and methods

The generalized architecture of the ensemble classifier, which uses a multilayer perceptron to implement stacking at the second stage, is shown in Fig. 1. In the first stage, it contains M classifiers, each of which receives an input image x. At the output of each of these classifiers, a vector is obtained

$$P_i(x)=\left(p_{i1}(x),p_{i2}(x),...,p_{ik}(x)\right), \quad (1)$$

where i is the number of the classifier,

k is the number of data classes,

$p_{ij}(x)$) is the support of the i-th classifier that the signal x belongs to the j-th class.

When using majoritarian voting, the maximum is selected from all the output signals of each of the first stage classifiers, and the number of this output is taken as the signal class x, which determined the corresponding classifier

$$y_i(x)=\arg\left\{\max_j\left\{p_{ij}(x)\right\}\right\}. \quad (2)$$

The output signal of the second stage of the ensemble classifier when using majoritarian voting [10]

$$class(x)=\arg\left\{\max_{c_i\in dom(y)}\left(\sum_{i=1}^{k}g\left(y_i(x),c_i\right)\right)\right\}, \quad (3)$$

where the indicator function

$$g(y,c)=\begin{cases}1 \text{ if } y=c,\\ 0 \text{ if } y\neq c,\end{cases}$$

$dom(y)=\{c_1,c_2,...,c_k\}$ is a set of class labels.

For certainty, we will further assume that in majoritarian voting, the output signal of the ensemble classifier is determined by (3) if the number of matches is greater than half the number of classifiers in the first stage. Otherwise, a random class number is taken as the output signal.

Neural networks used as classifiers of the first stage of the ensemble classifier are given in Table 1. The software implementation and all parameters of the first stage classifiers are taken from [20]. All classifiers have been trained on MNIST and CIFAR-10 sets over 50 epochs. The results of training and the number of weighting coefficients are given in Table 1. The number of data classes in both sets k=10.

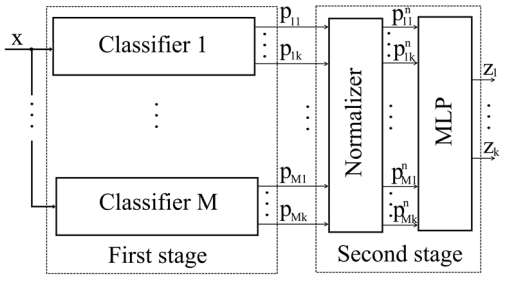

Fig. 1. The generalized architecture of ensemble classifier

Table 1

Parameters of first-stage classifiers

Neural network	MNIST		CIFAR-10	
	Accuracy of classification	Number of weighting factors	Accuracy of classification	Number of weighting factors
CNN	0.9929	34826	0.7687	343306
CCT	0.9879	406987	0.8021	408139
EANet	0.8517	358090	0.6788	355530
FNet	0.9881	992010	0.7572	582410
gMLP	0.9876	1271818	0.7405	862218
MLP-Mixer	0.9574	629258	0.7674	219658
SwinTr	0.9711	147034	0.7128	151386

Because a small number of signals are received in the second stage of the ensemble classifier, a multilayer perceptron with one hidden layer was chosen as a neural network at the second stage. This makes it possible to get comparable results when increasing the number of classifiers at the first stage.

When using a multilayer perceptron ensemble classifier in the second stage, the output signals of the first stage classifiers are preliminarily normalized. The normalizer brings the output signals of each of the classifiers separately to the range [−1, 1] as follows

$$a_i(x)=\max_j\left\{\left|p_{ij}(x)\right|\right\}, \quad (4)$$

$$p^n_{ij}(x)=p_{ij}(x)/a_i(x), \quad (5)$$

where i is the number of the classifier, i=1,...,M,

$p_{ij}(x)$ is the j-th output of the i-th classifier.

The input of the multilayer perceptron receives a signal vector

$$P^n(x)=\left(p^n_{11}(x),...,p^n_{1k}(x),...,p^n_{Mk}(x)\right). \quad (6)$$

Multilayer perceptron (MLP) contains 3 layers [21]:

– input layer; dimensionality, $k\times M$;

– hidden layer; dimensionality $(k-1)\times M$, Relu activation function;

– output layer; dimensionality k, softmax activation function.

The decision on the class to which the input signal belongs is made according to the rule

$$class(x)=\arg\left\{\max_j\left(z_l(x)\right)\right\}, \quad (7)$$

where $z_l(x)$ is the signals at the MLP output (l=1,...,k) at the input image x.

MLP training uses the Adam optimizer, a gradient averaging parameter for training batch_size=128. During MLP training, first-stage classifiers were not trained but we simply calculated $P_i(x)$ vectors using pre-trained weighting factors. For each variant, the MLP was trained 100 times, starting each time with random values for the weightings according to Xavier's initialization [22]. The maximum achieved classification quality was selected for use in the comparison.

The number of adjustable weighting coefficients in the MLP, depending on the number of classifiers in the first stage, is given in Table 2.

Table 2

Number of adjustable weights in MLP

Number of classifiers in the first stage	2	3	4	5	6
Number of adjustable weights in MLP	630	830	1540	2450	3560

The study program was written in Python using the Num.py, Tensorflow, and Keras libraries, and run on the Colab cloud platform [23]. MLP training involved data for training. Classification quality was defined as the ratio of the number of correctly recognized images to the total number of images in the test suite of 10,000.

5. Results of studying the ensemble classifier with a multilayer perceptron in the second stage

5. 1. Determining the influence of the use of a multilayer perceptron on the second stage of the ensemble classifier on the quality of classification

The results of using only two classifiers for the MNIST dataset at the first stage are given in Table 3, and for CIFAR-10 – in Table 4. Based on these results, the best pair was selected for use in the first stage and the best third classifier was selected for use in the first stage. For the top three classifiers, the best fourth classifier was selected, etc. The results for the MNIST dataset are given in Table 5, and for CIFAR-10 – in Table 6.

Table 3

Ensemble classifier classification quality when using two classifiers in the first stage and MLP in the second stage for the MNIST dataset

First-stage classifiers	CNN	CCT	EAT	FNet	gMLP	MLP-Mixer
CCT	0.9947	–	0.9890	0.9909	0.9919	0.9903
EAT	0.9934	0.9890	–	0.9858	0.9888	0.9681
FNet	0.9934	0.9909	0.9858	–	0.9910	0.9858
gMLP	0.9938	0.9919	0.9888	0.9910	–	0.9890
MLP-Mixer	0.9934	0.9903	0.9681	0.9858	0.9890	–
Swin Tr	0.9932	0.9893	0.9736	0.9868	0.9901	0.9774

Table 4

Quality of ensemble classifier classification when using two classifiers in the first stage and MLP in the second stage for the CIFAR-10 dataset

First-stage classifiers	CNN	CCT	EAT	FNet	gMLP	MLP-Mixer
CCT	0.8165	–	0.8184	0.8066	0.8077	0.8102
EAT	0.8159	0.8184	–	0.7898	0.7799	0.7912
FNet	0.7905	0.8066	0.7898	–	0.7950	0.7984
gMLP	0.7895	0.8077	0.7799	0.7950	–	0.7890
MLP-Mixer	0.7931	0.8102	0.7912	0.7984	0.7890	–
SwinTr	0.7839	0.8162	0.7707	0.7908	0.7832	0.7917

The results given in Tables 5, 6 show that the use of MLP in the second stage of the ensemble classifier provides an error reduction for the MNIST set by 29–39 %, and for the CIFAR-10 set – by 13–17 %. It should be noted that the quality of the classification of the ensemble classifier significantly depends on the data set used and on the effectiveness of the first-stage classifiers on this data set. For the MNIST dataset, the first-stage classifiers had high scores, which led to the fact that an improvement in the quality of the classification was observed when the number of first-stage classifiers increased from 2 to 4. The subsequent increase in their number increased the total number of tunable coefficients but did not lead to a decrease in the number of errors. The CIFAR-10 suite is more difficult to classify than MNIST. For it, each addition to the first stage of the next classifier led to a decrease in the number of errors.

Table 5

Quality of the ensemble classifier classification when using *M* classifiers at the first stage and MLP at the second stage for the MNIST dataset, and the results of majoritarian voting (MV) at the second stage

First-stage classifiers	*M*	Number of adjustable weights taking into consideration MLP	Classification quality when using MLP	Classification quality when using MV (MV rule)	Reduced error when using MLPs
CNN+CCT+gMLP	3	1 714 461	0.9954	0.9935 (2 from 3)	29 %
CNN+CCT+gMLP+FNet	4	2 707 181	0.9959	0.9935 (3 from 4)	37 %
CNN+CCT+gMLP+FNet+SwinTr	5	2 855 125	0.9959	0.9933 (3 from 5)	39 %
CNN+CCT+gMLP+FNet+SwinTr+MLP-Mixer	6	3 485 493	0.9959	0.9933 (4 from 6)	39 %

Table 6

Quality of the ensemble classifier classification when using *M* classifiers at the first stage and MLP at the second stage for the CIFAR-10 dataset, and the results of majoritarian voting (MV) at the second stage

First-stage classifiers	*M*	Number of adjustable weights taking into consideration MLP	Classification quality when using MLP	Classification quality when using MV (MV rule)	Reduced error when using MLPs
CCT+EAT+MLP-Mixer	3	984 157	0.8401	0.8076 (2 from 3)	17 %
CCT+EAT+MLP-Mixer+FNet	4	1 567 277	0.8445	0.8178 (3 from 4)	15 %
CCT+EAT+MLP-Mixer+FNet+gMLP	5	2 430 405	0.8457	0.8200 (3 from 5)	14 %
CCT+EAT+MLP-Mixer+FNet+gMLP+SwinTr	6	2 582 901	0.8468	0.8239 (4 from 6)	13 %

5. 2. Comparison of the characteristics of the constructed ensemble classifier and transformer

To compare the characteristics of the ensemble classifier, one of the most effective architectures was chosen – ViT (transformer) [3]. Its software implementation, given in [20], was used. The number of base blocks was taken to be small – 8 so that the quality of the classification was comparable to the quality of the classification of the classifiers used at the first stage of the ensemble classifier. ViT training on the MNIST and CIFAR-10 kits was carried out over 50 epochs. As a result, ViT ensured the quality of classification for MNIST 0.9859 and for CIFAR-10 – 0.8261 with the number of adjustable weights 21724362 and 21666762, respectively. A comparison of the results for MNIST is given in Table 7, for CIFAR-10 – in Table 8.

The data given in Tables 7, 8 demonstrate that the ensemble classifier on the MNIST and CIFAR-10 datasets makes it possible to get a better classification quality with a much smaller amount of calculations. For the CIFAR-10 dataset, the best ratio was shown by an ensemble classifier using CCT, EAT, and MLP-Mixer neural networks at the first stage. With 22 times fewer weights, the classification error is 8 % smaller than ViT. For the MNIST dataset, the best ratio was shown by an ensemble classifier using CNN and CCT neural networks at the first stage. With 50 times fewer weights, the classification error is 62 % smaller than ViT.

Table 7

Comparison of ensemble classifier characteristics with ViT for MNIST dataset

First-stage classifiers	*M*	Number of weighting factors with respect to ViT	Classification quality	Error reduction compared to ViT
CNN+CCT	2	0.02	0.9947	62 %
CNN+CCT+gMLP	3	0.08	0.9954	67 %
CNN+CCT+gMLP+FNet	4	0.12	0.9959	71 %
CCT+EAT+MLP-Mixer+FNet+gMLP	5	0.13	0.9959	71 %
CNN+CCT+gMLP+FNet+SwinTr+MLP-Mixer	6	0.16	0.9959	71 %

Table 8

Comparison of ensemble classifier characteristics with ViT for CIFAR-10 dataset

First-stage classifiers	*M*	Number of weighting factors with respect to ViT	Classification quality	Error reduction compared to ViT
CCT+EAT	2	0,035	0.8184	4,5 %
CCT+EAT+MLP-Mixer	3	0,045	0.8401	8,0 %
CCT+EAT+MLP-Mixer+FNet	4	0,072	0.8445	10,5 %
CCT+EAT+MLP-Mixer+FNet+gMLP	5	0,112	0.8457	11,2 %
CCT+EAT+MLP-Mixer+FNet+gMLP+SwinTr	6	0,119	0.8468	11,8 %

6. Discussion of results of studying the effectiveness of using a multilayer perceptron at the second stage of the ensemble classifier

The efficiency of the ensemble classifier, which uses stacking at the second stage, depends significantly on the difference in the algorithms by which the classifiers work at the first stage. Our study showed that for small data sets, the ensemble classifier makes it possible to get better characteristics with a smaller amount of calculations, compared to conventional classifiers with the most modern architecture. The results obtained are determined by two factors. The first is the selection of a large number, significantly different in the algorithm of work, classifiers for the first stage. The second is the best integration of all the output signals of the classifiers of the first stage due to the use of a neural network in the second stage. A comparison of the efficiency of majoritarian voting and the neural network (Tables 5, 6) reveals that all output signals of primary classifiers carry information about the class of the input signal.

The main advantage of the proposed ensemble classifier is the high ratio of the quality of classification and the required amount of calculations. Together with the ability to learn from small amounts of data, this can significantly expand the range of practical applications where such classifiers can be used.

As a limitation of the study, it should be noted that at the first stage of the ensemble classifier, only 6 types of different neural networks were used. In the practical use of the results obtained, it is necessary to take into consideration the possibility of the emergence of new networks with a different architecture from those used, which can also be used at the first stage and provide high characteristics of the ensemble classifier. Further development of the proposed approach primarily implies the development of new architectures of neural networks that provide an expansion of the diversity of classifiers of the first stage. When using a large number of classifiers at the first stage of the ensemble classifier, it seems promising to study the effectiveness of replacing a multilayer perceptron of the second stage with a more complex neural network, for example, a convolutional one.

7. Conclusions

1. It is shown that the use of a multilayer perceptron at the second stage provides a reduction in classification error compared to majoritarian voting. On the MNIST dataset, the reduction in classification error ranged from 29 % to 39 % depending on the number of classifiers in the first stage. On the CIFAR-10 dataset – between 13 % and 17 %.

2. A comparison of the proposed ensemble classifier with a classifier having a transformer architecture demonstrated its significant advantage. The best result on the CIFAR-10 dataset was an 8 % reduction in classification error with a lower number of weights by a factor of 22. Similarly, on the MNIST dataset, the classification error was 62 % smaller with 50 times fewer weights.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems 25 (NIPS 2012). Available at: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf
2. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition. doi: https://doi.org/10.1109/cvpr.2009.5206848
3. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T. et. al. (2021). An image is worth 16x16 words: transformers for image recognition at scale. arXiv. Available at: https://arxiv.org/pdf/2010.11929.pdf
4. Sun, C., Shrivastava, A., Singh, S., Gupta, A. (2017). Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. 2017 IEEE International Conference on Computer Vision (ICCV). doi: https://doi.org/10.1109/iccv.2017.97
5. Brownlee, J. Deep Learning for Computer Vision. Image Classification, Object Detection, and Face Recognition in Python. Available at: https://machinelearningmastery.com/deep-learning-for-computer-vision/
6. Denil, M., Shakibi, B., Dinh, L., Ranzato, M. A., Freitas, N. (2014). Predicting Parameters in Deep Learning. arXiv. doi: https://doi.org/10.48550/arXiv.1306.0543
7. Blalock, D., Gonzalez Ortiz, J. J., Frankle, J., Guttag, J. (2020). What is the state of neural network pruning? arXiv. doi: https://doi.org/10.48550/arXiv.2003.03033
8. Tolstikhin, I., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T. et. al. (2021). MLP-Mixer: An all-MLP Architecture for Vision. arXiv. doi: https://doi.org/10.48550/arXiv.2105.01601
9. Aggarwal, C. C., Sathe, S. (2017). Outlier Ensembles. An Introduction. Springer, 276. doi: https://doi.org/10.1007/978-3-319-54765-7
10. Rokach, L. (2019). Ensemble Learning. Pattern Classification Using Ensemble Methods. World Scientific, 300. doi: https://doi.org/10.1142/11325
11. Hirata, D., Takahashi, N. (2020). Ensemble learning in CNN augmented with fully connected subnetworks. arXiv. doi: https://doi.org/10.48550/arXiv.2003.08562
12. Ting, K. M., Witten, I. H. (1999). Issues in Stacked Generalization. Journal of Artificial Intelligence Research, 10, 271–289. doi: https://doi.org/10.1613/jair.594
13. LeCun, Y., Cortes, C., Burges, C. J. C. The MNIST database of handwritten digits. Available at: http://yann.lecun.com/exdb/mnist/
14. Krizhevsky, A. The CIFAR-10 dataset. Available at: https://www.cs.toronto.edu/~kriz/cifar.html

15. Hassani, A., Walton, S., Shah, N., Abuduweili, A., Li, J., Shi, H. (2021). Escaping the Big Data Paradigm with Compact Transformers. arXiv. doi: https://doi.org/10.48550/arXiv.2104.05704
16. Guo, M.-H., Liu, Z.-N., Mu, T.-J., Hu, S.-M. (2021). Beyond Self-attention: External Attention using Two Linear Layers for Visual Tasks. arXiv. doi: https://doi.org/10.48550/arXiv.2105.02358
17. Lee-Thorp, J., Ainslie, J., Eckstein, I., Ontañón, S. (2021). FNet: Mixing Tokens with Fourier Transforms. arXiv. doi: https://doi.org/10.48550/arXiv.2105.03824
18. Liu, H., Dai, Z., So, D. R., Le, Q. V. (2021). Pay Attention to MLPs. arXiv. doi: https://doi.org/10.48550/arXiv.2105.08050
19. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z. et. al. (2021). Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. arXiv. doi: https://doi.org/10.48550/arXiv.2103.14030
20. Code examples. Computer vision. Keras. Available at: https://keras.io/examples/vision/
21. Brownlee, J. Better Deep Learning. Train Faster, Reduce Overfitting, and Make Better Predictions. Available at: https://machinelearningmastery.com/better-deep-learning/
22. Brownlee, J. (2021). Weight Initialization for Deep Learning Neural Networks. Available at: https://machinelearningmastery.com/weight-initialization-for-deep-learning-neural-networks/
23. Colab. Available at: https://colab.research.google.com/notebooks/welcome.ipynb