Analysis of Deep Learning Methods in Adaptation to the Small Data Problem Solving

Iurii Krak[1,2][0000−0002−8043−0785], Vladyslav Kuznetsov[1][0000−0002−1068−769X], Serhii Kondratiuk[1,2][0000−0002−5048−2576], Larisa Azarova[3][0000−0002−2631−8151], Olexander Barmak[4][0000−0003−0739−9678], and Pavlo Padiuk[4][0000−0003−3609−112X]

[1] Taras Shevchenko National University of Kyiv, Ukraine
`yuri.krak@gmail.com, sergey.kondrat1990@gmail.com`
[2] Glushkov Institute of Cybernetics of NAS of Ukraine, Kyiv, Ukraine
`kuznetsow.wlad@gmail.com`
[3] Vinnytsia National Technical University, Vinnytsia, Ukraine
`le-azarova@vntu.edu.ua`
[4] Khmelnytskyi National University, Khmelnytskyi, Ukraine
`alexander.barmak@gmail.com, radiukpavlo@gmail.com`

Abstract. This paper discusses a specific problem in the study of deep neural networks – learning on small data. Such issue happens in situation of transfer learning or applying known solutions on new tasks that involves usage of particular small portions of data. Based on previous research, some specific solutions can be applied to various tasks related to machine learning, computer vision, natural language processing, medical data study and many others. These solutions include various methods of general purpose machine and deep learning, being successfully used for these tasks. In order to do so, the paper carefully studies the problems arise in the preparation of data. For benchmark purposes, we also compared "in wild" the methods of machine learning and identified some issues in their practical application, in particular usage of specific hardware. The paper touches some other aspects of machine learning by comparing the similarities and differences of singular value decomposition and deep constrained auto-encoders. In order to test our hypotheses, we carefully studied various deep and machine learning methods on small data. As a result of the study, our paper proposes a set of solutions, which include the selection of appropriate algorithms, data preparation methods, hardware optimized for machine learning, discussion of their practical effectiveness and further improvement of approaches and methods described in the paper. Also, some problems were discussed, which have to be addressed in the following papers.

Keywords: Small data, deep neural networks, machine learning, data dimensionality reduction, anomaly detection, data augmentation, algorithm stability, tensorflow, directml

1 Introduction

One of the main directions of improving the work of modern algorithms of Artificial Intelligence (AI) based on the ideology of Big Data [3] is the use of large datasets, large data centers (computing clouds) and, accordingly, increasing the depth and width of the layers of neural networks and other constructs used in deep learning. This, in turn, creates new advances, but they become inaccessible to the individual researchers and AI developers: first, even a pre-trained model with billions of parameters (such as GPT-3) requires a device to scale this model to new data, and, secondly, the repetition (reproduction) of experiments by other studies becomes possible only for those who have a small supercomputer at the level of a small company in the field of artificial intelligence. If this is done by an individual researcher, the threshold for entering this field becomes insurmountable and the next question arises – do one really need big data to summarize its properties and teach an algorithm with such many parameters? There is an unambiguous answer to this question, but it is partly contained in the history of other methods of intelligent data processing, such as the singular value decomposition and the method of group argumentation. In some cases, close to ideal, several dozen data samples were enough to teach such methods. This example is a possible answer to the question: how to make Big Data and Deep Learning is available to a wide range of researchers? If there is an example of successful learning of some algorithms on small data samples, it means that it is possible to scale the model for deep learning methods so that they can be applied to small data samples. This gives a potential leap for research in this field and makes such methods and models available to a wider range of researchers. In order to address this issue the paper covers both theoretical and practical aspect in software and hardware implementation of these algorithms for particular tasks of learning on small data.

Paper structure. The main results are presented in pargraphs 2-5, as follows:

- The *paragraph 2* discusses related works on machine learning and deep learning, including our works on computer vision, data classification, natural language processing and others. Based upon the problems discussed in that paragraph, the goals and research tasks of the study are presented.
- The following *paragraph 3* discusses pipeline of data preparation and 4 main techniques in data preparation such as anomaly detection, data augmentation, detection of perturbations in data and their affect on the efficiency of learning.
- *Pagargaph 4* focuses mostly on the implementation of the algorithms for different processor architectures and underscores the relationship of code optimization for specific architecture and performance of specific algorithm for particular task.
- According to recommendations given in previous paragraphs, *paragraph 5* focuses on the main goals of the study. Based upon these two group of experiments, the graphic data is provided to illustrate different aspects of supevised and unsupervised deep learning, in particular learning on data, denoising and feature representation.

2 Related Works and Problem Statement

A typical machine learning scheme in approaches that use general-purpose machine learning and in-depth learning tend to differ in substance. If we very briefly define the essence of the first – a cascade of algorithms, the input of which receives data (raw data or processed features), and the output - hypotheses about the belonging of a data element to a particular class. In contrast, the concept of deep learning involves the construction of a neural network architecture that combines different basic elements – ordinary (shallow) layers, convolution layers, thinning, recurrent layers and more. Accordingly, the deep network, with some exceptions, can be constructed simultaneously as one algorithm, even if the structural layers of the network are responsible for different functions – detection of features, grouping, and classification.
Let's focus in more detail on two typical learning pipelines that we used in previous studies [17–19] and which have proven themselves on the example of machine learning (classification of social and textual information) and deep learning, respectively (on the example of classification of images of emotional expressions on the face). A typical machine learning model consists of the following methods:

- transformations of data – e.g. singular value decomposition and integral transformations [1, 7, 31];
- grouping of features – e.g. T-stochastic neighbor embedding [5, 30];
- space compaction – e.g. variation and denoising auto-encoders [10];
- classification in the space of reduced dimension – e.g. decision trees, support vector machines and other methods [11, 21].

As one can see from the structure of this model, it is possible to perform training step by step to achieve sufficient accuracy in the reconstruction of feature-space, the number of errors of the 1st and 2nd kind, and so on. The most time-consuming method is the method T-stochastic neighbor embedding, as it builds a nonlinear hypothesis about the relative position in the feature-space of reduced dimension; the second execution time is the method of compaction of the space of features, which is essentially a method of deep learning, but due to fewer layers in the context of the task (compared to the task purely deep learning), it is similar to a normal feed-forward neural network. In contrast to the methods of grouping features and compacting the space of features, methods of dimensional reduction are very fast and can be faster than similar methods of deep learning. The most successful results in terms of execution time and efficiency before the emergence of big data were achieved by combining these two groups, where the classification methods consisted of support vector machines and decision trees [12].
If we analyze such a model, we can say that it is possible to build a certain pipeline of deep learning based on neural network constructs and general purpose machine learning, including the same functions – respectively, the dimensional reduction and classification. The difference, however, is that these methods show greater efficiency with increasing data volumes and, accordingly, an increase in the number of parameters taken into account in the model [15]. In the context

of this study, it is important to show threshold (borderline) examples where the use of deep learning methods is appropriate (small data) and, conversely, where it is more appropriate to use a conventional model of machine learning with a cascading combination of classifiers and dimensional transformation methods.
The main goals of the study are: *1) to identify where are the issues in learning on small data*; *2) to investigate practical performance of deep learning algorithms for specific tasks – learning on features and learning on raw data.*
Thus, in order to approach these goals, the datasets should represent different sizes in order to identify the borderline performance for a specific dataset in order to get the best decision plane for classification. This approach, in contrary to studying big datasets, needs less resources, which, as an effect, tends to decrease the computation costs in order to process the data in such way.
Taking in account the problems in small data analysis, **the research tasks are**:

– to create a few datasets which represent different research topics – data preparation, data grouping, clustering and classification;
– to study hidden feature representation in different methods of dimensional reduction, grouping of features and clustering;
– to analyze locations of the biggest density in context of data preparation;
– to study the latent feature space representation and compare such representation with data dimensional reduction methods;
– to conduct tests on algorithms for data classification and clustering.

3 Data Preparation in Machine Learning and Deep Learning

3.1 General View on the Pipeline of Data Preparation

The process of automated data preparation in the two approaches is similar, but in the case of the classic machine learning has certain steps in the analysis of small samples. In this case, the visual analysis of data comes to the fore, which allows assessing the representation of data elements and perform data engineering. Foremost, these are procedures for the selection of informative features, which in combination allows to effectively allocating data classes and conduct classification and clustering procedures [4,20]. Secondly, it is the analysis of data clusters, where it is possible to identify the main classes and sub-classes of data. In contrast, when learning from data in the “deep” approach, layers contain hidden information that cannot always be used to make decisions about both a feature set (the number of features in hidden network layers) and data sets.
In fact, decisions are ultimately made on qualitative indicators – on the convergence of the algorithm and the large number of runs of training procedures, in order to determine the optimal architecture. Except for layer-by-layer learning which to involve deep auto-encoders to initialize network weights (unsupervised learning) [2,28], the informativeness of the features and the data elements themselves is determined by successful data selection (balancing classes in datasets),

using numerous samples formed by affine transformations and data augmentation. Let's focus on the most important approaches in data preparation and its representation for visual analysis.

3.2 Detection of Anomalies in Data

Data anomalies usually occur for several reasons: there is an imbalance of classes, the elements of dataset are scattered or contain poor quality data samples, the presence of hidden relationships between features in the data, non-linearity of feature-space representation that cannot be correctly compacted into reduced dimension and others. This, in turn, means that data elements in the feature space are located at a great distance from the centers of classes outside the areas of data crowding and interfere with the area of crowding belonging to other data classes.
This scenario can be given by an example. The properties of datasets and its features do represent hidden properties only if using specific approaches; hence, the points within original dimension can be very scattered and have a lot of dimensions within feature space. This can be outcome using data dimensional reduction, though they may be highly affected by the size of the dataset; for instance, if the dataset is small the points are scattered, but if it is quite big they form clusters (fig. 1).

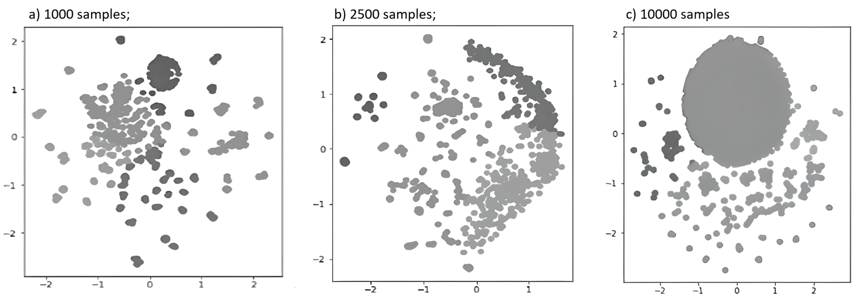

Fig. 1. Evolution of T-SNE feature-space representation using a dataset of reduced number of samples and a full dataset of 10000 samples

According to our investigation, using such quite simple example (shown on figure 1), using the method of T-stochastic neighbor embedding on the datasets of different sizes, but containing same elements (e.g. 10%, 25% and 100% as of figure 1a, 1b, 1c respectively), one can see, that data points begin to shift one towards another, what, in case of augmented dataset may tend to forming another clusters. This interesting feature was found during our studies dedicated to the processing of scientific texts.
This particular example can be explained by following statement. Since this method, in essence, builds such dependencies that in the new, transformed

feature-space, with a relatively large separation band, there is a non-linearity of representations, that can be achieved in relatively large sizes of the datasets. So, having a clusters with relatively large separating band, it is equally possible to use as conventional methods of machine learning and the so-called "wide" neural networks, so as the "deep" and "regular" approach may met. However, the disadvantage of such a representation is the processing time, which potentially requires machine learning accelerators to build a nonlinear data representation. Despite the disadvantages, this approach is very useful in visual analysis of the data since all its features are represented in the space of the low dimension and, most important – it helps to highlight the anomalies in the final representation using a singular value decomposition or T-stochastic neighbor embedding. Knowing their location, it is possible to apply filtering of data by the root-mean-square error (if there is a relatively large number of data representatives), and in the second – to determine the main data axes by constructing regression [32]. Thus, it is possible to note areas with higher data density and, accordingly, areas with fewer anomalies (fig. 2).

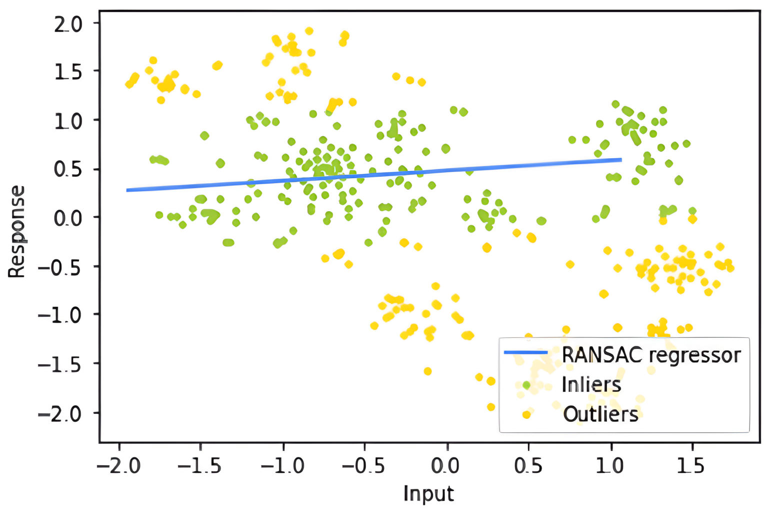

Fig. 2. Principal axis of data and outlier points in reduced feature space

According to the figure (fig. 2) the feature space representation given by T-stochastic neighbor embedding allows to arrange the data in the same way as usage of principal components. The first principal axis of data shows not only the data crowding, but the property of the dataset itself – e.g. if the data points located very far from the regression also have very big mean-square error they also may represent elements of other classes. Usage of such principle in detail (for instance, only on one class) may help to find outlier points and identify anomalies in data (if any).

3.3 Data Augmentation

An important step in data preparation is the creation of a data set in which the representatives of the data classes are balanced both in the number of data elements and in their relative location in the case when the data clusters are unevenly distributed (Table 1). To do this, it is possible to artificially generate data samples that have the same location in the feature space (original or reduced dimension), so that the shape of the clusters and the distribution of data samples will remain the same. This can be achieved by applying certain techniques to balanced learning, including SMOTE, ADASYN and others that generate random data samples [6, 8, 9, 22]. Samples with the involvement of specialized unsupervised methods, in particular variation auto-encoders. This subspecies of neural networks contains a core in the middle, which describes the properties of the data, namely their distribution and relative position in the latent space of features, which is generated by the auto-encoder on the output layer of the encoder. Thus, it is possible to generate as a set of random samples in the latent feature space or, conversely, to obtain encoding in the transformed space generated by the source layer of the decoder and thus obtain new data samples (Table 1).

Table 1. Classification rate on test dataset

precision	recall	f1-score	Support	Total
Class 1	0.59	1.00	0.74	366
Class 2	0.98	0.95	0.96	454
Class 3	0.90	0.64	0.75	332
Class 4	1.00	0.59	0.74	333
accuracy			0.81	1485
macro avg	0.87	0.79	0.80	1485
weighted avg	0.87	0.81	0.81	1485

In a study on the analysis of scientific texts and the impact of sample size on the quality of the algorithm, including convergence, building an effective hypothesis of data separation by other algorithms (decision trees, support vector machines, etc.), we found that the use of augmentation methods to generate new samples data, and the balancing of data samples in general has a positive effect on the quality of data sampling while maintaining the feature-space configuration of data clusters, their distribution in space and the external boundaries on which the data separation band can be built (Table 2).

The comparison of tables listed above (Table 1 and table 2) may show the difference be-tween number of samples within every class and also their proportion one against another. Change in percentage of each data class as well as increasing the size of the dataset increases the recall in worst-case scenario from 59% on class 1 to 81% and from 87% to 92% which is an impressive result.
Main outcome of such technique is to figure out the minimal size of the dataset,

Table 2. Classification rate on augmented dataset

precision	recall	f1-score	Support	Total
Class 1	0.81	0.97	0.88	1313
Class 2	0.92	0.96	0.94	1239
Class 3	0.95	0.76	0.85	1274
Class 4	0.98	0.93	0.96	1254
accuracy			0.91	5080
macro avg	0.92	0.91	0.91	5080
weighted avg	0.91	0.91	0.91	5080

needed to solve a specific problem – studying on small data, not re-creating the dataset from scratch. This may help to see the possible situations where are the deep learning methods can be applied. Also, using same strategy, the other statement can be inferred: if the dataset is separable with desired accuracy in certain feature-space of reduced dimension, it is possible to design a deep learning network that instead of learning from features (representation of lower dimension) can study from the raw data, which, on other hand can be also augmented in original space.

3.4 Perturbation Compensation and Stability of Classification Algorithms

Data preparation always assumes the presence of poor and even corrupted data, which may ultimately affect the construction of a separate plane in hyperspace. As a result of this, the data contains of anomalies, where individual data elements are located outside the main area of data classes, and it affects the hypotheses about data separation towards the location of outlier points, which affects the overall efficiency of classification and clustering algorithms (figure 3).
One approach is to generate as much instances of data as possible, so as the "noise" can be studied by the deep learning approach and anomalies can be omitted (which is not the case in the learning on small data). Another and more effective approach is to detect such anomalies and eliminate them by filtering or by compensating for disturbances that increase the displacement of the endpoints of data clusters. This can be achieved by using denoising auto-encoders, which is one of the methods of deep learning. The auto-encoder reinforces the general trends in data – the placement of clusters with high density and, thus, weakens weak trends, which are anomalies and caused by certain disturbances in the data.
According to the figure 3, it can be seen that the data representation in latent feature space is more "tight" and the separation band becomes very narrow. This is caused by denoising feature of an auto-encoder, because the latent feature space decreases a number of information stored in each feature and thus optimizes for mean-square error of decoded representation and decrease in entropy of the encoded data.
This may give an idea of another application of auto-encoders: since they can

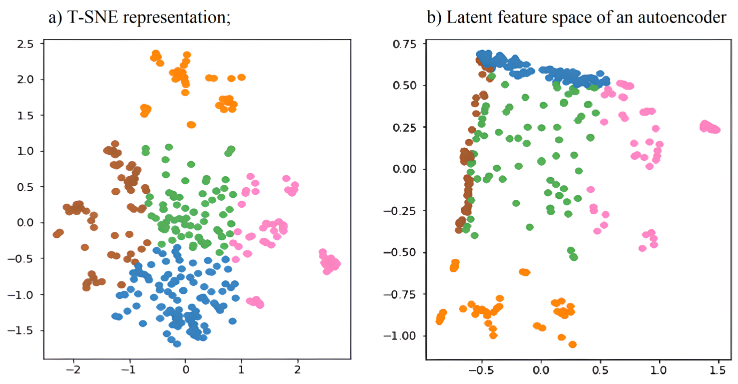

Fig. 3. Comparison of data representations using two different algorithms for data grouping and data dimensionality reduction and their affect on separation band

reduce the entropy in the data and eliminate disturbances, they may be used to test the algorithm stability of classification algorithms, because in the latent space of features between-class margin and separation band is significantly reduced, which affects the increase of errors of the, except for gradient boosting methods and decision trees, what can be clearly seen on the figure 4.

3.5 Orthogonal Transformations in Machine and Deep Learning

The peculiarity of most methods of deep learning and classical neural networks is that the set of features in the hidden layers and, accordingly, the characteristic functions they form are almost equivalent (except for the latest networks such as graph neural network and networks with active involvement of dropout). This means that modifying the architecture to reduce the number of features in the hidden layers is possible only after re-learning the network or (very limited) by visualizing and analyzing images (patterns) of characteristic functions stored in the intermediate layers of the network. In contrast, classical methods, such as Fourier transform, singular value decomposition, wavelets, and others, involve the representation of the original data by a set of linearly independent characteristic functions and allow discarding non-informative features in terms of their power in the final representation. This property is useful when there are hidden links between individual features that affect the redundant information in the features, which ultimately affects the efficiency and speed of classification algorithms.
This feature of integral transformations, as a rule, is not applied at construction of architecture of deep networks for unsupervised and supervised learning. Therefore, the dependencies embedded in the network layers are nonlinear and allow hidden linear relationships between features. On the one hand, this allows to build complex nonlinear hypotheses when classifying data (for example, in autoencoders), on the other hand, the presence of restrictions on the weights of

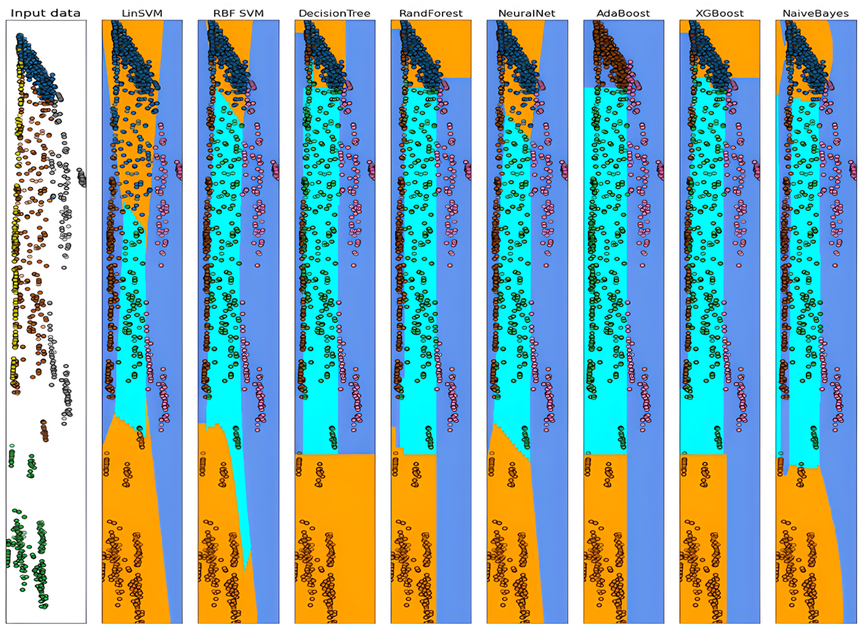

Fig. 4. Hypotheses visualizing for different classifiers in latent feature space

the network allows engineering features in the middle of the network and rank them by importance. To over-come these problems, we proposed a modification of the autoencoder, taking into account the limitations on the orthogonality and linear independence of the weights; thus, the representation of data (forms of data clusters) in the latent space of features is very similar to the representation of features obtained by means of a singular value decomposition or similar integral transformations.

4 Note on Implementation of Algorithms for the Task

4.1 Overview of Libraries for Machine Learning Used in Study

Currently, researchers have quite a lot of machine learning algorithms and their implementations for modern personal computers. The most famous libraries are Scikit-learn and TensorFlow [26] and, in particular its newest implementation TensorFlow-DirectML [27] from Microsoft, which enables usage of hardware machine learning acceleration for various hardware vendors (Intel, AMD, Nvidia). The-se libraries are widely used among researchers and scientists in the field of machine learning and data science due to the widespread use of Python [23]; though, we have to admit that there are many other languages such as C++, Java, C# (Caffe , Theano , Torch , dl4j, Smile and others) used in AI research but in very specific tasks where the software performance is more important.

These libraries for various languages can be used either on central processing units (CPU) or graphic processing units (GPU) or both, which mostly depends on their adaptation (OpenCL, CUDA). Because our research was more focused on Python, we will look at our machine learning and deep learning experience for this programming language, in particular its Intel implementation, Anaconda, which includes pre-built Intel MKL libraries for matrix operations.

4.2 Implementations for Processors

Machine learning and deep learning libraries for Python are mostly high-level (C and C++) compiled code that is built as a library (DLL) in the structure of a Python interpreter that executes instructions on a Python listing. In the case of libraries such as Open CV or DLib, the source text is compiled on the user's machine for the processor architecture where the program will be executed and according to a set of processor instructions. In the case of more universal libraries, such as Scikit-learn, a ready-made binary file is loaded, which is optimized for the architecture of the whole processor family, and in the case of code generation can be more optimized for a specific architecture (AMD or Intel). According to a recent study, we tried to evaluate the relationships between the number of CPU cores, its theoretical performance, and actual performance on various machine and deep learning tasks — dimension reduction, feature grouping, classification, and clustering (table 3) in order to find the most optimal architecture for specific task (data preparation, feature space reduction, classification etc.).

Table 3. Performance indicators of machine learning procedures for different architectures of processors

Procedure name	max, s	min,s	max/min	Average
Read	1.45	0.139	10.37	0.6081
Reshape	0.006	0.001	6.01	0.0029
Dimensionality reduction	2.173	0.075	28.93	0.6348
Grouping of features	65.57	10.81	6.06	39.07
Clustering of features	0.6	0.072	8.24	0.2387
Sample preparation	0.007	0.002	2.68	0.0051
Deep ANN init	0.269	0.05	4.83	0.1585
Deep ANN learning	117.643	24.10	4.88	71.0883
Deep ANN inference	2.221	0.583	3.81	1.2514
Decision trees training	0.826	0.148	5.59	0.3859

There are a few practical outcomes of the experimental results shown in table 3:

– the tasks that involve simple manipulations with memory (e.g. reshape of an array) display some difference in performance, but it is insignificant both in absolute and relative means and mostly affected by number of computing cores;

- interestingly, the relative difference (28x) in such task as singular value decomposition is more prominent, than of data grouping or learning of the deep network. It means that the matrix operations including multiplication are performed better on newer versions of processors, though it depends on implementation;
- the practical efficiency of learning rely on the speed of the all components included in the task starting from the reading from disk till machine learning tasks and inference on new data, in case if there is a weak link. For instance, slow operation of grouping of features or deep learning, which are most time consuming tasks may affect the overall time and the proper architecture has to be used in the experiment.

5 Evaluation of the Practical Effectiveness of Deep Learning Methods on Real Data

5.1 Data Used in Experiments and Previous Study in Area

In order to evaluate the algorithms of deep learning, a series of experiments were conducted on different data – textual information, video images, medical data, sound samples etc. (figure 5). Some data, in particular scientific texts and video images, were obtained in our research and used to test the concept of deep learning and improve classification methods involving visual analysis of data clusters, transforming the dimensional of data, features creation, face recognition etc [16, 19]. Let's dwell in more detail on another aspect – the analysis of

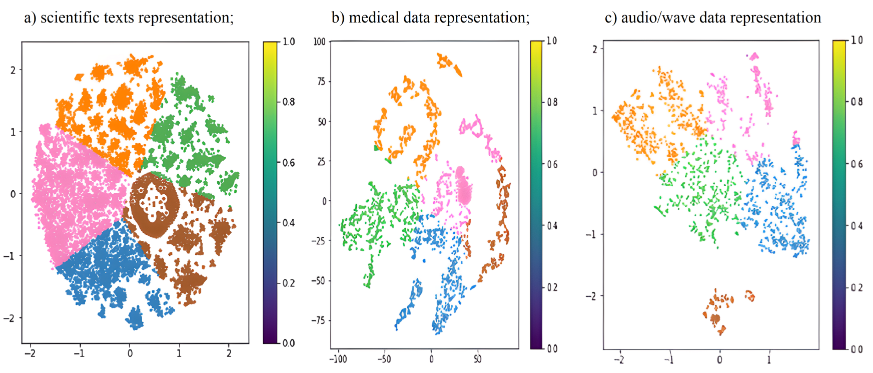

Fig. 5. Data representation in feature space of lower dimensionality for a datasets used in the experimental study

applicability on different types of data and, most importantly – on the impact of sample size on the overall effectiveness of deep learning methods . The peculiarity of deep neural networks and derived constructs is that they can in some cases work with raw data without the use of third-party methods; they have shown themselves best in the analysis of digital images. Instead, when studying more complex data, such as sound or textual information, the input of the neural network receives already converted information. In our research, the vast majority of information is the feature vectors obtained by analysis of methods of feature extraction (detection) – in the case of analysis of facial expressions we used preprocessing by computer vision, in the analysis of sound information – integral transformations, in the analysis of scientific texts – methods of text mining (figure 6).

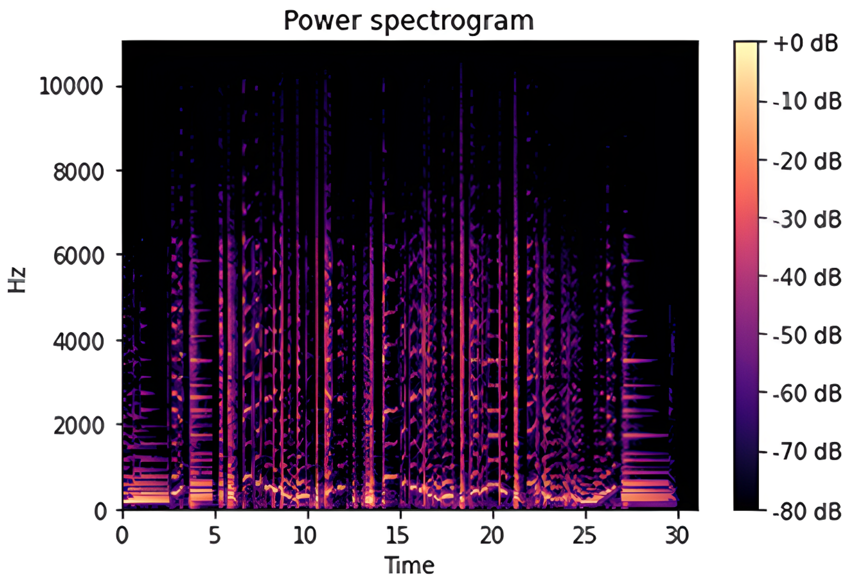

Fig. 6. Fourier transform of sound sample from dataset used in the experiments

5.2 Application of Computer Vision Techniques for Neural Networks

In the work [19], devoted to the study of emotional expressions on the human face, the technology of obtaining samples of trajectories of features on the face with the involvement of computer vision methods was proposed. This allowed to transform the multidimensional space of features (which is the image) on the face into a set of coordinates of the centers of features and to analyze both their static positions (instantaneous states) and changes in their position over time. We studied such a set of features by methods of dimensional reduction such as singular decomposition and classification methods such as support vector machines. The sample size, which at the time of publication was about 170 samples did not allow the full use of shallow neural networks and even more so – deep learning methods due to slightly worse (in the context of the problem and based on the size of the training sample) performance indicators - errors 1 and 2nd kind; constructs such as the deep belief network [13], the convolution neural network [14, 24, 25, 29], and the varieties of cascading denoising auto-encoders were tested (table 4).

Table 4. Performance indicators of machine learning algorithms on facial expressions

Type of method	Accuracy and size of vector
Multi-layer neural network	75 %, 400x1
Denoising auto-encoder	98 %, 400x1
Deep belief network	75 %, 400x1
Convolution neural network	75 %, 400x1
Singular value decomposition and decision trees	75 %, 140 eigen values

These experiments (table 4) showed that on a given number of samples (small dataset), the accuracy of recognition reached up to 95% for the support vector machines method and about 75% for the deep learning methods noticed in above. However, when using the initiation of the weights with a cascading denoising auto-encoder and layer-by-layer learning of shallow neural networks, the accuracy of recognition approached the accuracy of the support vector machines method and indicated the potential for use of auto-encoders to reduce data dimensionality and grouping of features.

5.3 Latent Representation of Features as an Alternative to Data Dimensionality Reduction Methods

The widespread use of integrated methods of data analysis, such as singular de-composition, wavelets, Fourier transform, allowed to present data as a combination of linearly independent characteristic functions, where the first main decomposition coefficients could be used as points on a two-dimensional diagram for visual analysis of the data and, accordingly, the separation of entries into separate classes.
With the advent of constructs such as auto-encoders and their further development in terms of data representation (e.g., linear auto-encoder with constrained weights and variation auto-encoder) it became possible to similarly encode information and obtain a data representation that meets the specified conditions (dimensionality of the feature vector, orthogonality of weights, error of reconstruction) and, accordingly, to have an alternative representation of the data, including nonlinear (figure 7). The peculiarity of the autoencoder is that after each iteration a new unique location of data points is built in a space of reduced dimension (which is also called latent feature space), which meets the

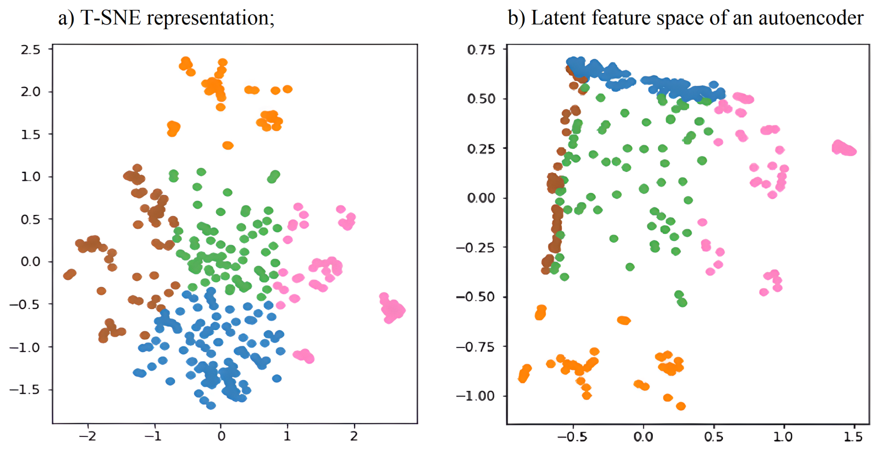

Fig. 7. Comparison of data representations of different methods of dimensionality reduction

basic requirements of the optimization algorithm – minimizing mean-square data reconstruction error. In this space, the peculiarities of the auto-encoder are revealed – the distance between the centers of data accumulation decreases and the value of the total and between-the-class standard deviation decreases, which can be used for noise reduction.

As part of the tasks solved in the analysis of scientific texts, facial expressions and other data, we found that the autoencoder can be used for various tasks not directly related to reducing the dimensionality of data – first, visual analysis of data, and secondly (because in the latent space it is possible to compensate for noise), as well as generate new data using variational autoencoders (figure 8).

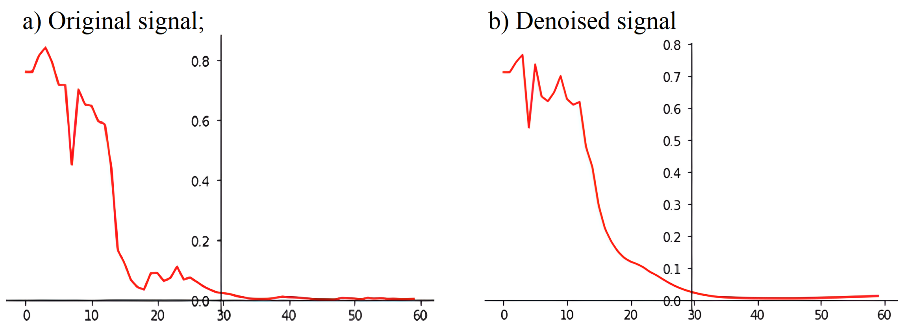

Fig. 8. Denoising effect of an autoencoder on data. The vertical axis represents the relative value of the signal in range [0;1], the bottom axis – the number of measurement (time)

5.4 Performance of Deep Learning Methods on Small and Large Datasets

As part of the experiments to solve the problems of small dataset analysis, both processors and GPU were tested, which differed in both architecture and generation, and which allowed us to assess the performance taking into account the aging of equipment. The main purpose of this experiment was to establish the limit cases of the use of processors and, accordingly, GPUs for the analysis of small data samples.
Based on a series of experiments, we found that data samples known as "toy da-tasets" do not significantly accelerate learning on the GPUs compared to modern processors, but significantly lose in speed. The execution time of the iteration of the algorithm on CPU is a fixed value. However, with the increasing complexity of the calculation there is another trend – a gradual increase in the efficiency of the machine learning accelerator to the estimated maximum value (table 5).

Table 5. Performance ratio of auto-encoder training in different tasks

Task name	CPU task time, s	GPU task time, s	Performance ratio
Unconstrained AE, small number of iterations	0.46	0.61	0.75
Slightly constrained AE, small number of iterations	2.19	6.97	0.31
Slightly constrained AE, medium number of iterations	27.55	7.36	3.74
Moderate constrained AE, medium number of iterations	32.98	7.68	4.29
Highly constrained AE, high number of iterations	41.11	7.85	5.24

We figured out that the sample size and the degree of fullness of the RAM of the graphics accelerator significantly affects this amount of delay: it consists of the time of reading from disk or RAM and sending (writing) to the memory of the graphics accelerator. In the presence of narrow data bus with the processor, bottleneck effects can occur, where the weak link is no longer the number of computing cores, and memory itself. In the case of exceeding the system memory of the video accelerator and the presence of slow system memory, the video accelerator in almost all cases loses to the processor. Exceptions to this are cases where these shortcomings are compensated by big number of iterations. In a neural network training experiment dedicated to recognition of individual emotions in a photographic image, we found that execution time affect the overall efficiency ratio to the processor (table 6).

Therefore, as a result of research, we obtained the simple method of definition of the most effective computational means for the task of our research. Initially,

Table 6. Performance of convolution network training using different hardware

Performance parameter	CPU performance	GPU performance
Number of items	448/448	448/448
Time on epoch 1	449 s	107 s
Time on epoch 2	448 s	22 s
Loss on epoch 1	1.7944	1.7978
Loss on epoch 2	1.4812	1.4888
Accuracy on epoch 1	0.3116	0.3112
Accuracy on epoch 2	0.4364	0.4260

the reference AI benchmarking task was calculated, which is slightly smaller in scale, but the amount of system memory involved is satisfactory for running the test task. The next step is to find reference data on the performance of different computing systems on reference tasks, showing the degree to which one device differs from another. Then the target (desired) speed for the given task with a full set of data (operations per second) is selected and the nearest and economically feasible device for this task can be found. In the framework of research we have shown that the theoretical performance rate of machine learning accelerator, calculated from tabular data from open sources and on the results of localization of features on the face is consistent with the actual performance rate, which involves such networks – the classification of images with facial expressions and differs by a small amount for many iterations of the algorithm.

6 Conclusions

The Big Data and Deep Learning is very popular among many researchers in the field of machine learning, but entry in this area is not always possible on a large scale (with really large samples and large models with billions of parameters). That is why acquaintance with it begins with relatively small samples of data obtained by researchers themselves or obtained from the Internet. The rapid development of modern computing devices (processors and graphics accelerators) and the recent rise in their prices due to the global shortage of microelectronics does not always make it economically feasible to use such computing devices, and older generations limit the use of such tools for deep learning.
That is why the tasks of deep learning are scaled to the capabilities of the researcher; this, in turn, poses the problem of analyzing small datasets by deep learning methods. As mentioned in Section 5.3, the analysis of small data does not always give satisfactory results by the deep neural networks. Therefore, based on our own experience of developing neurofunctional information transducers, it is more appropriate to carefully study methods that are similar in idea to the methods of reducing the dimensionality of data: since the autoencoders are very similar to singular value decomposition by the idea, it is possible to create dimensional transformations that are not only similar to singular value decom-

position, but also have some of its properties (paragraph 3.4).
Subject to the preparation of a sample (for example, involving data augmentation methods), which allows the operation of deep learning methods on a real problem, deep learning can be implemented similarly to classical machine learning methods with step-by-step or cascading operations of extraction of features and classification or clustering by deep neural networks (paragraph 2).
In the following works we plan to investigate more carefully other constructs of neural networks – such as recurrent neural networks and to evaluate the possibility of their application for the problems solved in our research, in particular in the analysis of scientific texts and medical data.

References

1. Albtoush, A., Fernández-Delgado, M., Cernadas, E., Barro, S.: Quick extreme learning machine for large-scale classification. Neural Computing and Applications **34**(8), 5923–5938 (2022). https://doi.org/10.1007/s00521-021-06727-8
2. Aloysius, N., Geetha, M.: A review on deep convolutional neural networks. In: 2017 International Conference on Communication and Signal Processing (ICCSP). pp. 588–592. IEEE (2017). https://doi.org/10.1109/iccsp.2017.8286426
3. Alzubaidi, L., Zhang, J., Humaidi, A., et al.: Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. Journal of Big Data **8**(53) (2021). https://doi.org/10.1186/s40537-021-00444-8
4. Babichev, S., Durnyak, B., Zhydetskyy, V., Pikh, I., Senkivskyy, V.: Application of Optics Density-Based Clustering Algorithm Using Inductive Methods of Complex System Analysis. In: 2019 IEEE 14th International Conference on Computer Sciences and Information Technologies (CSIT). vol. 1, pp. 169–172 (2019). https://doi.org/10.1109/STC-CSIT.2019.8929869
5. Chan, D., Rao, R., Huang, F., Canny, J.: T-SNE-CUDA: GPU-Accelerated T-SNE and its Applications to Modern Data. In: 2018 30th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD). pp. 330–338. IEEE (2018). https://doi.org/10.1109/cahpc.2018.8645912
6. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: SMOTE: Synthetic Minority Over-sampling Technique. J. Artif. Intell. Res. (JAIR) **16**, 321–357 (2002). https://doi.org/10.1613/jair.953
7. Dongarra, J., Gates, M., Haidar, A., et al.: The Singular Value Decomposition: Anatomy of Optimizing an Algorithm for Extreme Scale. SIAM Review **60**(4), 808–865 (2018). https://doi.org/10.1137/17m1117732
8. Hast, A., Vast, E.: Word Recognition using Embedded Prototype Subspace Classifiers on a new Imbalanced Dataset. Journal of WSCG **29**(1-2), 39–47 (2021). https://doi.org/10.24132/jwscg.2021.29.5
9. He, H., Bai, Y., Garcia, E., Li, S.: ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence). pp. 1322–1328. IEEE (2008). https://doi.org/10.1109/ijcnn.2008.4633969
10. Huang, G., Chen, D., Li, T., Wu, F., van der Maaten, L., Weinberger, K.Q.: Multi-Scale Dense Networks for Resource Efficient Image Classification (2017). https://doi.org/10.48550/arXiv.1703.09844

11. Izonin, I., Tkachenko, R., Gregus, M., Duriagina, Z., Shakhovska, N.: PNN-SVM Approach of Ti-Based Powder's Properties Evaluation for Biomedical Implants Production. Computers, Materials & Continua **71**(3), 5933–5947 (2022). https://doi.org/10.32604/cmc.2022.022582
12. Izonin, I., Tkachenko, R., Shakhovska, N., Lotoshynska, N.: The Additive Input-Doubling Method Based on the SVR with Nonlinear Kernels: Small Data Approach. Symmetry **13**(4), 1–18 (2021). https://doi.org/10.3390/sym13040612
13. Jiang, M., Liang, Y., Feng, X., Fan, X., Pei, Z., Xue, Y., Guan, R.: Text classification based on deep belief network and softmax regression. Neural Computing and Applications **29**(1), 61–70 (2018). https://doi.org/10.1007/s00521-016-2401-x
14. Khan, A., Sohail, A., Zahoora, U., Qureshi, A.: A survey of the recent architectures of deep convolutional neural networks. Artificial Intelligence Review **53**(8), 5455–5516 (2020). https://doi.org/10.1007/s10462-020-09825-6
15. Krak, I., Barmak, O., Manziuk, E.: Using visual analytics to develop human and machine-centric models: A review of approaches and proposed information technology. Computational Intelligence pp. 1–26 (2020). https://doi.org/10.1111/coin.12289
16. Krak, Y., Barmak, A., Baraban, E.: Usage of NURBS-approximation for Construction of Spatial Model of Human Face. Journal of Automation and Information Sciences **43**(2), 71–81 (2011). https://doi.org/10.1615/jautomatinfscien.v43.i2.70
17. Krivonos, Y.G., Krak, Y., Barchukova, Y., Trotsenko, B.: Human Hand Motion Parametrization for Dactilemes Modeling. Journal of Automation and Information Sciences **43**(12), 1–11 (2011). https://doi.org/10.1615/JAutomatInfScien.v43.i12.10
18. Kryvonos, I., Krak, I.: Modeling human hand movements, facial expressions, and articulation to synthesize and visualize gesture information. Cybernetics and Systems Analysis **47**(4), 501–505 (2011). https://doi.org/10.1007/s10559-011-9332-4
19. Kryvonos, I., Krak, I., Barmak, O., Ternov, A., Kuznetsov, V.: Information Technology for the Analysis of Mimic Expressions of Human Emotional States. Cybernetics and Systems Analysis **51**(1), 25–33 (2015). https://doi.org/10.1007/s10559-015-9693-1
20. Lytvynenko, V., Lurie, I., Krejcí, J., Voronenko, M., Savina, N., Ali Taif, M.: Two Step Density-Based Object-Inductive Clustering Algorithm. In: Workshop Proceedings of the 8th International Conference on "Mathematics. Information Technologies. Education" (MoMLeT and DS-2019). vol. 2386, pp. 1–19. CEUR-WS, Shatsk, Ukraine (2019), http://ceur-ws.org/Vol-2386/paper10.pdf
21. Lytvynenko, V., Savina, N., Krejcí, J., Voronenko, M., Yakobchuk, M., Kryvoruchko, O.: Bayesian Networks' Development Based on Noisy-MAX Nodes for Modeling Investment Processes in Transport. In: Workshop Proceedings of the 8th International Conference on "Mathematics. Information Technologies. Education" (MoMLeT and DS-2019). vol. 2386, pp. 1–10. CEUR-WS, Shatsk, Ukraine (2019), http://ceur-ws.org/Vol-2386/paper1.pdf
22. Menardi, G., Torelli, N.: Training and assessing classification rules with imbalanced data. Data Mining and Knowledge Discovery **28**(1), 92–122 (2014). https://doi.org/10.1007/s10618-012-0295-5
23. Python: An open-source programming language, environment and interpreter (2022), https://www.python.org/about/
24. Romanuke, V.: An attempt of finding an appropriate number of convolutional layers in cnns based on benchmarks of heterogeneous datasets. Electrical, Control and Communication Engineering **14**(1), 51–57 (2018). https://doi.org/10.2478/ecce-2018-0006

25. Sultana, F., Sufian, A., Dutta, P.: Advancements in Image Classification using Convolutional Neural Network. In: 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN). pp. 122–129. IEEE (2018). https://doi.org/10.1109/icrcicn.2018.8718718
26. TensorFlow: A system for large-scale machine learning (2022), https://www.tensorflow.org/about/
27. TensorFlow-DirectML: Github repository for tensorflow fork accelerated by directml (2022), https://github.com/microsoft/tensorflow-directml
28. Vahdat, A., Kautz, J.: Nvae: A deep hierarchical variational autoencoder (2020). https://doi.org/10.48550/arXiv.2007.03898
29. Wiatowski, T., Bolcskei, H.: A Mathematical Theory of Deep Convolutional Neural Networks for Feature Extraction. IEEE Transactions on Information Theory **64**(3), 1845–1866 (2018). https://doi.org/10.1109/tit.2017.2776228
30. Yona, G., Moran, S., Elidan, G., Globerson, A.: Active Learning with Label Comparisons (2022). https://doi.org/10.48550/ARXIV.2204.04670
31. Zebari, R., Abdulazeez, A., Zeebaree, D., et al.: A Comprehensive Review of Dimensionality Reduction Techniques for Feature Selection and Feature Extraction. Journal of Applied Science and Technology Trends **1**(2), 56–70 (2020). https://doi.org/10.38094/jastt1224
32. Zhang, G., Chen, Y.: More informed random sample consensus. arXiv (2020). https://doi.org/10.48550/ARXIV.2011.09116