UDC 004.93’14.032.26.05 **doi: 10.32620/reks.2022.3.07**

Viacheslav MOSKALENKO[1,2], Alona MOSKALENKO[1]

[1] ***Sumy State University, Sumy, Ukraine***
[2] ***National Aerospace University “Kharkiv Aviation Institute”, Kharkiv, Ukraine***

NEURAL NETWORK BASED IMAGE CLASSIFIER RESILIENT TO DESTRUCTIVE PERTURBATION INFLUENCES – ARCHITECTURE AND TRAINING METHOD

Modern methods of image recognition are sensitive to various types of disturbances, which actualize the development of resilient intelligent algorithms for safety-critical applications. The current article develops a model and method of training a classifier that exhibits characteristics of resilience to adversarial attacks, fault injection, and concept drift. The proposed model has a hierarchical structure of prototypes and hyperspherical boundaries of classes formed in the space of high-level features. Class boundaries are optimized during training and provide perturbation absorption and graceful degradation. The proposed learning method involves the use of a combined loss function, which allows the use of both labeled and unlabeled data, implements the compression of the feature representation to a discrete form and ensures the compactness of the distribution of classes and the maximization of the buffer zone between classes. The main component of the loss function is the value of the normalized modification of Shannon's information measure, averaged over the alphabet of the classes, expressed as a function of accuracy characteristics. Simultaneously, accuracy characteristics are calculated on the basis of smoothed versions of the distribution of statistical hypothesis testing results. It is experimentally confirmed that the proposed approach provides a certain level of disturbance absorption, graceful degradation and recovery. During testing of the proposed algorithm on the Cifar10 data set, it was established that the integral metric of resilience to tensor damage by inversion of one randomly selected bit is about 0.95 if the share of damaged tensors does not exceed 30%. Also, during testing of the proposed algorithm, it was established that an adversarial attack with a disturbance that does not exceed the L_∞-norm threshold equal to 3 provides resilience that exceeds the value of 0.95 according to the integral metric. Additionally, the integral metric of resilience during adaptation to the appearance of two new classes is 0.959. The integral metric of resilience to the real drift of concepts between the two classes is 0.973. The ability to adapt to the appearance of new classes or the concept drift has been confirmed 8 times faster than learning from scratch.

Keywords: *image classification; robustness; resilience; graceful degradation; adversarial attacks; fault injection; concept drift.*

Introduction

Image classification is one of the most common tasks in the field of artificial intelligence. Classification analysis of visual objects is often a component of security-critical applications. Examples of such applications are autopilots of transport and combat drones, medical diagnostics, production processes, monitoring of traffic flows and inspection of infrastructure or production facilities, and the like. Therefore, the urgency of solving the problem of ensuring the resilience of image classification analysis models to the influence of destructive disturbing factors increases.

In works [1, 2], the vulnerability of artificial intelligence algorithms was investigated and the following destructive effects were identified: noise and adversarial attacks, faults and faults injection into the deployment environment of an intelligent algorithm, concept drift and emergence novelties, that is, test samples that are outside the distribution of training data.

The resilience of the data classifier model to destructive influences is primarily ensured by achieving robustness to absorb a certain level of destructive influences and implementing the graceful degradation mechanism to achieve the most effective behaviour in conditions of incomplete certainty [2, 3]. However, in practice, data analysis models must be continuously improved taking into account the non-stationary environment and new problems. That is why an equally important component of resilience is the ability of the model to quickly restore productivity by adapting to the destructive impact and improving the intelligent algorithm to increase the efficiency of subsequent adaptations [4, 5]. Recovery and improvement mechanisms are developed within the frameworks of continuous and meta-learning [6, 7].

There are many methods and approaches to improve robustness against adversarial attacks. Some researchers classify methods for ensuring robustness against adversarial attacks into the following categories [8]: gradient masking methods [9, 10]; robustness optimization methods [11]; methods of detecting adversarial samples [12]. But the work [13] demonstrated the inefficiency of gradient masking methods. Approaches to optimize robustness include adversarial training and regularization methods that minimize the impact of small disturbances in input data [11]. The last category includes methods of detecting adversarial samples at the input of the model to reject them, however, Carlini and Wagner [12] rigorously proved that detecting the properties of adversarial samples is very difficult and resource-consuming. In works [8, 14], it was proposed to divide the methods of protection against adversarial attacks into two groups that implement two separate principles: increasing intraclass compactness and interclass separation in the feature space; increasing the robustness of features and eliminating non-robust features of the image. The first group includes the well-known methods of knowledge distillation, as they contribute to the increase of interclass distances in the space of features [15]. The work [16] develops the principle of the information bottleneck, which proves the high robustness of the discrete characteristic representation and the efficiency of the loss function based on information measures.

Faults are often modeled by generating random or directed inversions (bit flips) in the memory that stores the weights or the output value of the neuron. To ensure robustness to the injection of faults into the computing environment, deployment of neural networks uses three main approaches: introduction of explicit redundancy [17]; modification of the learning algorithm [18]; architecture optimization [18]. The introduction of obvious redundancy is implemented, as a rule, by duplication of critical neurons and synapses, uniform distribution of synaptic weights and removal of unimportant weights and neurons. It is possible to increase the robustness of the neural network before fault injection at the stage of machine learning by adding noise, disturbances or directly faults during training. A regularization component (penalization) is also introduced for the indirect inclusion of faults in the conventional algorithm [19]. Optimization of the architecture to increase robustness consists in minimizing the maximum error at the output of the neural network for a given number of inverted bits in the memory where the weights or results of intermediate calculations are stored. In research [19], this problem was effectively solved on the basis of evolutionary search algorithms. However, architecture optimization is traditionally a rather resource-intensive process.

Domain randomization methods [20] are proposed in these papers, which increase the robustness of the model to limited shifts in the data distribution. Domain randomization consists in generating synthetic data from a large number of variations, so that real-world data can be considered as another variant of the domain [20]. Randomization of viewing angles, textures, shapes, shaders, camera effects, scaling and other effects can be used for this. In other papers [21], the use of self-supervised learning algorithms on data from different domains is proposed to achieve generalization beyond the boundaries of each of the specific domains. The method of Transfer Learning and Multi-Task Learning strengthen the resistance to out-of-distribution disturbing [22]. However, if real concept drift appears in the data stream, then there is a need to implement reactive mechanisms for the purpose of quick adaptation [23]. There is research on adapting to real concept drift, but the task of adapting in the absence of markup for test data or a significant delay in obtaining it is still a challenge.

Adversarial attacks, injection of errors, drift of concepts and samples outside the learning distribution cannot always be absorbed, therefore the development of reactive resilience mechanisms, namely graceful degradation, recovery and improvement [2, 4] is relevant. The implementation of these mechanisms is often associated with the need to detect a disturbing factor. The most successful methods of detecting adversarial samples, samples outside the training distribution (out-of-distribution) or concept drift are based on the analysis of the space of high-level features. At the same time, confidence metrics based on distances and classification algorithms based on prototypes are used [24, 25]. In work [26], the mechanism for detecting faults affecting the prediction result is based on the calculation of the reference value of the contrastive loss function on test diagnostic data samples in the absence of faults. To detect faults, the current value of the contrastive loss function for diagnostic data is compared with the reference value. Algorithms of nested learning and hierarchical classification were proposed in [27], which is useful for implementing the sophisticated degradation mechanism.

Algorithms for adapting models to destructive factors are considered in papers [28], where the principles of active learning or contrastive learning are used to increase the speed of adaptation by reducing the requirements for the volume of labeled data. The works [29] proposed training methods with semi-supervised learning for the simultaneous use of both labeled and unlabeled data to speed up adaptation to the drift of concepts. The papers [7] consider continuous learning methods that allow continuous accumulation of knowledge on various tasks and improvements, as well as various reminder mechanisms to avoid the problem of catastrophic forgetting. Various approaches to the implementation of meta-learning to increase the efficiency of adaptation are investigated in [6].

Thus, there are studies of individual principles of the resilience of data classification analysis models, but there are practically no works that consider their simultaneous combination.

The purpose of the article is to develop the architecture and training method of the image classifier, which contains the main components of mechanisms of resilience to destructive factors. Under the goal, the following research tasks are formulated:

– propose an architecture and training method of an image classifier to ensure resilience to adversarial attacks, faults injection and concept drift;

– to verify that the classifier has the properties of disturbance absorption (robustness), graceful degradation, recovery and improvement;

– evaluate the resilience of the classifier to destructive factors.

Problem statement

Exist a set of unlabeled images $\{x_j^U \mid j=\overline{1,n}\}$ and a set of labeled images $\{x_{m,j}^S \mid m=\overline{1,M};\ j=\overline{1,n_m}\}$ for training and testing the algorithm of image classification analysis, where n and n_m are the size of the set of unlabeled images and the size of the set of labeled images of m-th class, respectively. Let be given the set of classes $\{X_m^o \mid m=\overline{1,M}\}$ and the structure of parameters vector g of data analysis algorithm, that has be given as

$$g =< e_1,..,e_{\xi_1},..,e_{\Xi_1}, f_1,..,f_{\xi_2},..,f_{\Xi_2} >, \qquad (1)$$
$$\Xi_1+\Xi_2=\Xi,$$

where e_{ξ_1} – ξ_1-th parameter of the algorithm affecting the formation of the feature representation of observations, $\xi_1=\overline{1,\Xi_1}$;

f_{ξ_2} – ξ_2-th parameter of the algorithm affecting the accuracy characteristics of the classifier, $\xi_2=\overline{1,\Xi_2}$.

At the same time, the restrictions imposed on the parameters in the form of formulas are known $R_{\xi_1}(e_1,...,e_{\xi_1},...,e_{\Xi_1})\le 0$, $R_{\xi_2}(f_1,...,f_{\xi_2},...,f_{\Xi_2})\le 0$.

It is necessary to find the optimal values of the parameters vector g (1) which provide a compromise between the maximum value of the information criterion of efficiency averaged over the set of the classes $\bar{J}$ in normal conditions (that is, before the disturbance or after adaptation) and the maximum value of the integral metric of the resilience R of the classifier to disturbances during the given control period [4, 30]:

$$\bar{J}=\frac{1}{M}\sum_{m=1}^{M} J(\alpha_m,\beta_m,D_{1,m},D_{2,m}), \qquad (2)$$

where J is the function of the information criterion of efficiency for the two-alternative system of decision evaluations;

α_m is an estimation of the probability of errors of the first kind for the decision rule of the m-th class;

β_m is an estimation of the probability of errors of the second kind for the decision rule of the m-th class;

$D_{1,m}$ is a sensitivity (first reliability) of the decision rule of the m-th class;

$D_{2,m}$ is a specificity (second reliability) of the decision rule of the m-th class;

$$R=\frac{\frac{1}{|P|}\sum_P \int_{t=0}^{T_c}\bar{J}(t)dt}{\int_{t=0}^{T_c}\bar{J}_0(t)dt}, \qquad (3)$$

where $\bar{J}(t)$ is dependence of the current value of the information criterion of system efficiency on time (number of the test sample or mini-package);

T_c is control period, the values of which are chosen based on the results of a preliminary assessment of the average duration of the interval between the events of destructive effects or the maximum permissible recovery time [4];

P is a set of destructive events during the interval of the control period [0, Tc];

$\bar{J}_0(t)$ is normal functional state (steady state without disturbances), which is entered into the formula for displaying the values of the integral index of resilience in the interval [0, 1];

$$g^* = \arg\max_G \left\{\eta\bar{J}(g)+(1-\eta)R(g)\right\}, \qquad (4)$$

where η is the coefficient regulating the trade-off between the information criterion of efficiency and the resilience of the algorithm during the control period.

Architecture of the resilient classifier model

During the construction of the model, we aim to implement the main characteristics of resilience: robustness, controlled degradation, recovery and improvement. For this, the model will be based on the following principles:

– formation of a prototype and a compact spherical container for each class to detect out-of-distribution samples of training data and concept drift;

– the use of a hierarchical system of sample labeling and hierarchical classification to implement the principles of graceful degradation;

– the organization of FIFO memory buffers with a limited size for storing labeled and unlabeled data that was fed to the input of the neural network and has the corresponding values of labels or loss functions, for the implementation of diagnostic and recovery mechanisms.

Fig. 1 shows the architecture of the resilient classifier. It is shown that the classifier model consists of a feature extractor built on the basis of four ResBlocks modules of the well-known ResNet50 architecture, and a classification module based on prototypes [25]. A set of vector-prototypes is built for the classification analysis of the feature presentation. Prototype vectors are not fixed, they are determined in the learning process as well as the weighting coefficients (filters) of the feature extractor. At the same time, to ensure the principle of graceful degradation, prototypes can belong to different levels of hierarchy in accordance with the hierarchy of labeling. In addition to prototypes, the radii of hyperspherical separating surfaces (containers) for each of the classes are optimized during the training process. Container radii are stored in memory to detect a high level of uncertainty in decision making.

To increase the immunity and robustness of the characteristic description, it is suggested to use one of the options for implementing the information bottleneck principle [16]. For this purpose, it is assumed to compress the feature representation by bringing it closer to a discrete form. Therefore, the output of the feature extractor uses a sigmoid layer and the corresponding regularization in the learning algorithm.

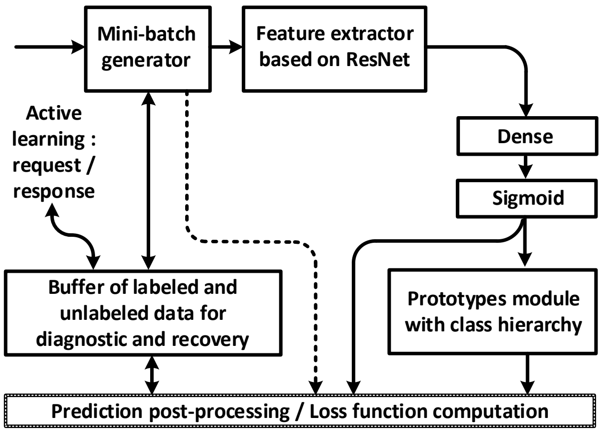

Mini-batch generator
Feature extractor based on ResNet
Active learning : request / response
Dense
Sigmoid
Buffer of labeled and unlabeled data for diagnostic and recovery
Prototypes module with class hierarchy
Prediction post-processing / Loss function computation

Fig. 1. Architecture of the resilient image classifier model

Confidence in the prediction for the i-th sample is estimated by the value of the function of belonging to the k-th recognition class, which has the form

$$\mu_k(z_i) = 1 - \frac{\mathrm{dist}(z_i, \bar{z}_k)}{N \cdot r_k}, \tag{5}$$

where z_i is binary feature representation of the i-th sample at the feature extractor output;

$\bar{z}_k$ is a prototype for the k-th class in the feature space;

N is dimension of the feature vector z_i;

r_k is scaling factor for the radius of the hyperspherical separating surface (container) of the k-th class, $r_k \in (0;1)$;

$\mathrm{dist}(\cdot)$ – squared Euclidean distance.

If the maximum value of the membership function (1) for the input unlabeled sample x_i with a feature representation z_i is less than zero, then such a prediction should not be trusted, and this sample should be added to the buffer of unlabeled data outside the training distribution. If the incoming unlabeled sample falls into one of the containers of recognition classes, then it should be added to the buffer of unlabeled data belonging to the distribution of one of the classes. Buffers of unlabeled samples can be used for semi-supervised learning based on pseudo-labeling, soft-labeling or consistency regularization [29]. In addition, unlabeled samples are candidates for a request for manual labeling (active learning). At the same time, data from the buffer of unlabeled data can be moved to the buffer of labeled data after receiving feedback about their real belonging to recognition classes. The priority of recommending samples for labeling depends on the value of the membership function.

If a labeled sample falls outside the boundaries of its class container, then this may indicate novelty in the data or virtual drift of concepts. Control of falling of labeled samples into a container not of its class can be used to detect the real drift of concepts. To avoid catastrophic forgetting in the conditions of concept drift or the appearance of a new class of recognition, the reminder function is implicitly implemented based on data buffers and reference vectors in the feature space.

After updating the weight of the model, it is necessary to store (or update) the diagnostic sample of data and the corresponding value of the loss function in the memory. After that, together with the test samples in each mini-batch, it is necessary to pass a sample of diagnostic data to compare the past and current value of the loss function to detect faults (injection of faults) in the memory of the network parameters. If the difference in the loss function exceeds a certain threshold α, it is necessary to initiate the network fine-tuning algorithm based on diagnostic data to eliminate this discrepancy, not below the threshold β.

Machine learning method

When developing a machine learning method, we aim to ensure the main characteristics of resilience: robustness, graceful degradation, recovery and improvement. For this, the following principles will be the basis of the learning algorithm:

– increasing the compactness of the distribution of classes and the buffer zone between classes to increase robustness to noise, emissions and adversarial attacks by introducing an additional regularization component to the resulting loss function;

– penalization for the discretization error (compression to binary form) of the feature representation as one of the options for implementing the information bottleneck to increase the robustness and informativeness of the characteristic description;

– taking into account the hierarchical nature of the data labeling and the hierarchical nature of class prototypes by calculating the loss function separately for each level of the hierarchy to provide graceful degradation mechanisms in the recognition mode;

– implementation of reactive performance recovery mechanisms under the influence of disturbances based on fine-tuning of weights on diagnostic data to eliminate the impact of detected faults, new initialization of prototypes of drifting classes or new classes, use of new data without labeling in the consistent regularizing component to quickly adapt to adversarial attacks;

– avoiding catastrophic forgetting during adaptation to changes and attacks without completely retraining the model from scratch by implementing reminder mechanisms based on buffer data and prototype vectors.

Classifier training has two main stages:

– the preparatory stage, when training is carried out on the available amount of data;

– adaptation of the classifier to novelty and disturbances during operation.

The main criterion of learning in both cases is the information measure. The loss function based on the use of the information measure has the form:

$$L_{INF} = 1 - \bar{J} , \qquad (6)$$

where $\bar{J}$ is the value of the information criterion of efficiency averaged over the set of the classes (1).

K. Shannon's normalized entropy information measure is used as an information measure of k-th class recognition efficiency [30] and it calculated by the formula

$$J_k = \frac{H_o - H_\gamma}{H_o} =$$
$$= 1 + \frac{1}{2}\left(\frac{\alpha_k}{\alpha_k + D_{2,k}} \log_2 \frac{\alpha_k}{\alpha_k + D_{2,k}} + \right.$$
$$+ \frac{D_{1,k}}{D_{1,k} + \beta_k} \log_2 \frac{D_{1,k}}{D_{1,k} + \beta_k} +$$
$$+ \frac{\beta_k}{D_{1,k} + \beta_k} \log_2 \frac{\beta_k}{D_{1,k} + \beta_k} +$$
$$\left. + \frac{D_{2,k}}{\alpha_k + D_{2,k}} \log_2 \frac{D_{2,k}}{\alpha_k + D_{2,k}} \right), \qquad (7)$$

where H_o is a priori entropy for two alternative decision systems;

$H(\gamma)$ is posterior entropy, which characterizes the residual uncertainty after decisions are made.

Separaring hyperspherical surfaces are constructed for each class on the radial basis of the feature space and are the basis for decision rules. The accuracy characteristics for decision rules of the k-th class are calculated based on statistical hypothesis testing:

$$D_{1,k} = \frac{TP_k}{TP_k + FN_k + \varepsilon} + \varepsilon , \qquad (8)$$

$$D_{2,k} = \frac{TN_k}{TN_k + FP_k + \varepsilon} + \varepsilon , \qquad (9)$$

$$\alpha_k = \frac{FN_k}{FN_k + TP_k + \varepsilon} + \varepsilon , \qquad (10)$$

$$\beta_k = \frac{FP_k}{FP_k + TN_k + \varepsilon} + \varepsilon, \qquad (11)$$

where TP_k is the number of correct positive classifications of samples for decision rules of the k -th class;

TN_k is the number of correct-negative classifications of samples for decision rules of the ks-th class;

FP_k is the number of false-positive classifications of samples for decision rules of the k -th class;

FN_k is the number of false-negative classifications of samples for decision rules of the k -th class;

ε is a constant that is added for numerical stability, $\varepsilon = 10^{-6}$.

Procedures for calculating statistical tests are non-differentiable, so in training mode, you can use their smoothed versions [31]

$$TP \approx \sum_{i=1}^{n} \hat{y}_i \odot y_i , \qquad (12)$$

$$FP \approx \sum_{i=1}^{n} \hat{y}_i \odot (1-y_i), \qquad (13)$$

$$FN \approx \sum_{i=1}^{n} (1-\hat{y}_i) \odot y_i, \qquad (14)$$

$$TN \approx \sum_{i=1}^{n} (1-\hat{y}_i) \odot (1-y_i), \qquad (15)$$

where $\odot$ is element-wise multiplication operator (Hadamard product);

$y_i = \left\{ y_{i,k} \mid k=\overline{1,K} \right\}$ is a class label for i-th sample in one-hot coding format [31];

$\hat{y}_i = \left\{ \mathrm{relu}(\mu_{i,k}) \mid k=\overline{1,K} \right\}$ is the value of the membership function (5) after the relu() function for the i-th sample.

The admissible domain of criterion (7) is limited by inequalities $D_{1,k} \ge 0.5$ and $D_{2,k} \ge 0.5$, or $\beta_k < 0.5$ and $\alpha_k < 0.5$, then, for the convenience of optimization by the method of error backpropagation, it is suggested to perform the following operations during the calculation of the loss function [30]:

$$D_{1,k} = \max(D_{1,k}, 0.5), \qquad (16)$$

$$D_{2,k} = \max(D_{2,k}, 0.5), \qquad (17)$$

$$\alpha_k = \min(\alpha_k, 0.5), \qquad (18)$$

$$\beta_k = \min(\beta_k, 0.5). \qquad (19)$$

To increase the compactness of the distribution of classes and the interclass gap in the feature space, it is suggested to use the contrastive-centre loss function [32]

$$L_{CCL} = \frac{\mathrm{dist}(z_i, \overline{z}_{y_i})}{\sum\limits_{k=1, k\neq y_i}^{K} \mathrm{dist}(z_i, \overline{z}_k) + 1}. \qquad (20)$$

In order to optimize the boundaries of the class distribution, it is additionally proposed to use a regularizing component that connects the average distance between the prototypes of classes and the average radius of the class containers

$$L_C = \overline{r} / (\overline{d} + 1), \qquad (21)$$

where $\overline{d}$ is the average value of the normalized distance between class prototypes, calculated by the formula

$$\overline{d} = \frac{1}{N(K-1)^2} \sum_{k=1}^{K} \sum_{c=1}^{K} \mathrm{dist}(\overline{z}_c, \overline{z}_k); \qquad (22)$$

$\overline{r}$ is the average value of the scaling factor of the class container radius calculated by the formula

$$\overline{r} = \frac{1}{K} \sum_{k=1}^{K} r_k. \qquad (23)$$

To speed up adaptation to changes, data samples without labeling can be used to form a consistent regularization component in the loss function [29]. At the same time, unlabeled data is divided into two groups: unlabeled data falling into class containers; unlabeled data that does not fall into any of the recognition class containers.

Unlabeled data falling into the containers of existing classes is proposed to be used for consistent regularization by calculating the next component of the loss function

$$L_{UCE}^{in} = CE\left(q^{\mu}(z_i'), q^{\mu}(z_i'') \right), \qquad (24)$$

where z_i', z_i'' – indicative representation of augmented versions of the input sample x_i;

$q(z_i)$ is an estimation of the probability of belonging of the characteristic representation z_i for the sample x_i to the recognition classes after calculating the values of the membership functions

$$q_k^{\mu}(z_i) = \frac{\exp\left(\mu_k(z_i)\right)}{\sum\limits_{c=1}^{K} \exp\left(\mu_c(z_i)\right)}; \qquad (25)$$

$CE(\cdot)$ is the standard cross-entropy function.

For γ fractions (<10 %) of unlabeled samples that fall into class containers and have the largest maximum values of $q(z_i)$, it is possible to form a pseudo-labeling with the corresponding classes. Such pseudo-labeled data can be mixed into each mini-batch during training.

Unlabeled data falling outside existing class containers can be instances of unknown classes, outliers, or the result of concept drift. For such samples, it is suggested to use soft labeling which is based on distances to prototypes of already known classes and is calculated by the formula

$$q_k^{dist}(z_i) = \frac{\exp\left(-dist\left(z_i, \bar{z}_k\right)\right)}{\sum_{c=1}^{K} \exp\left(-dist\left(z_i, \bar{z}_c\right)\right)}. \qquad (26)$$

For samples that are outside the containers of known classes, the component of consistent regularization is calculated according to the formula

$$L_{UCE}^{out} = CE\left(q^{\mu}(z_i), q^{dist}(z_i)\right). \qquad (27)$$

Consistent regularization can be performed not only at the level of the classification module, but also at the level of features. The corresponding regularization component of the loss function is calculated by the formula

$$L_{UL2} = dist\left(z_i', z_i''\right). \qquad (28)$$

To implement the information bottleneck, a regularization component is additionally introduced, which penalizes the discretization error of the feature description [30]

$$L_D = z_i^T(e - z_i), \qquad (29)$$

where e is unit vector.

The initial values of the parameters of the prototypes of the lower-level classes are proposed to be initialized based on the Hadamard matrix [30]. To do this, the dimension of the Hadamard matrix is first determined $N_{Hadamard} = 2^{ceil(\log_2(N))}$, where $ceil()$ is a function for rounding a number to a larger integer value. Then all values less than 0 are replaced by 0, i.e $Z = \max\left(0, Hadamard(N_{Hadamard})\right)$. To facilitate the process of adapting the prototype to the features of the data structure, label smoothing is used according to the formula $Z' = Z*0.7+0.15$, as a result of which 1s will turn into 0.85, and 0s into 0.15. From the obtained matrix, the first K vectors are chosen, the dimension of which is limited to the first N features, i.e $\bar{z} = Z'[1:K, 1:N]$. At the same time, the initial value of the radius scaling factor r_k for the hyper-spherical container of the k-th class is initialized by the value of half of the Plotkin bound divided by the dimension of the feature space [30]

$$r_k \leftarrow \left(\frac{1}{2}\frac{N}{2}\frac{K}{K-1}\right)\frac{1}{N} = \frac{K}{4\cdot(K-1)}. \qquad (30)$$

In the case of the appearance of a sample with a marking indicating a new ($K+1$)-th class of the lower level, it is necessary to form a new prototype of the class $\bar{z}_{K+1}$ with the corresponding initial value of the radius scaling factor r_{K+1}. To do this, the nearest vector from the remaining unused rows is selected from the modified Adamar matrix Z' where the closeness is determined based on the square of the Euclidean distance. The initial value of the scaling factor of the container radius for the new class is also determined by the formula (30), but taking into account the new number of classes. In case of recognition of real concept drift, prototypes of drifting classes are filled with random numbers from the range [0; 1].

Each coordinate of the prototype-vector of the upper hierarchical level is initialized by copying the corresponding coordinate of one of the prototypes of the lower level, which is selected randomly. The radius of classes of the upper hierarchical level is determined by the formula (30) taking into account the number of classes at this level.

The resulting loss function is formed by the sum of the above components, averaged over the levels of the class hierarchy with coefficients that regulate the impact and priority of individual components.

When training on labeled training data, the loss function is used

$$L_S = \lambda_{INF}\bar{L}_{INF} + \lambda_{CCL}\bar{L}_{CCL} + \lambda_C\bar{L}_C + \lambda_D L_D, \qquad (31)$$

where $\bar{L}_{INF}$, $\bar{L}_{CCL}$, $\bar{L}_C$ are values of the loss functions L_{INF}, L_{CCL} and L_C respectively averaged by H-levels of class-hierarchy;

λ_{INF}, λ_{CCL}, λ_C, λ_D are coefficients that adjust the influence and priority of the components of the resulting loss function.

When new data become available, the labeled samples are combined with unlabeled samples to implement continuous adaptation, and the following loss function is calculated

$$L_{TOTAL} = L_S + \lambda_{UCE}^{out}\bar{L}_{UCE}^{out} + \lambda_{UCE}^{in}\bar{L}_{UCE}^{in} + \lambda_{UL2}\bar{L}_{UL2}, \qquad (32)$$

where $\bar{L}_{UCE}^{out}$, $\bar{L}_{UCE}^{in}$, $\bar{L}_{UL2}$ are values of the loss functions L_{UCE}^{out}, L_{UCE}^{in} and L_{UL2} respectively averaged by H-levels of class-hierarchy;

λ_{UCE}^{out}, λ_{UCE}^{in}, λ_{UL2} are coefficients that adjust the influence and priority of the components L_{UCE}^{out}, L_{UCE}^{in} and L_{UL2} respectively.

The following values of coefficients for the components of the loss function are suggested by default: $\lambda_{INF} = 1.0$, $\lambda_{CCL} = 1.0$, $\lambda_C = 0.001$,

$\lambda_D = 0.001$, $\lambda_{UCE}^{out} = 0.1$, $\lambda_{UCE}^{in} = 0.1$, $\lambda_{UL2} = 0.01$.

Results of machine learning and discussion

The Cifar10 dataset was chosen for experimental research because its images have a small dimensionality, which accelerates experimental research. The classes of this dataset can be arranged in a hierarchical structure. For example, the first top-level class may be the animal class, which includes subclasses bird, cat, deer, dog, frog and horse. The second top-level class may be the vehicle class, which includes subclasses airplane, automobile, ship and truck. Therefore, at the output of the classifier, 12 prototype vectors will be used, of which 2 are for top-level prototypes and 10 are for bottom-level prototypes. The CIFAR-10 dataset consists of 50,000 training and 10,000 test color images of 32x32 pixels. Images are evenly distributed among 10 classes. For the convenience of analysis, we will use 70 % of the training data to train the base model and form dataset_base, and 30 % will be used for additional training dataset_additional.

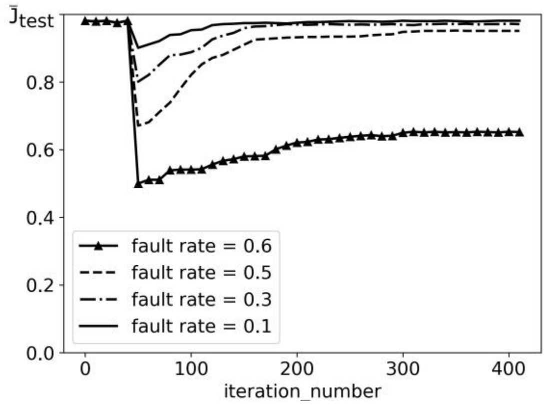
As a result of perturbations there is a decrease in productivity. To test the ability to recover, we consider recovery to be the state of achieving 95% of the performance that was before the perturbing impact. The control interval is equal to 200 iterations to ensure testing on the full amount of test data.

To test the model for fault resilience and recoverability, it is proposed to use the TensorFI2 library, which is capable of simulating software and hardware faults. In the experiment, it is proposed to consider the impact of the most difficult to absorb type of faults, which is to generate a random inversion of bits (bit-flip injection) in each layer of the model. At the same time, a fixed fraction of tensors (fault rate) is randomly selected and 1 bit is randomly selected in them to be inverted. For diagnostics and recovery, diagnostic data is added to each mini-batch along with test data for model input. Diagnostic data are generated from the dataset_additional and their number is equal to the size of 128 samples.

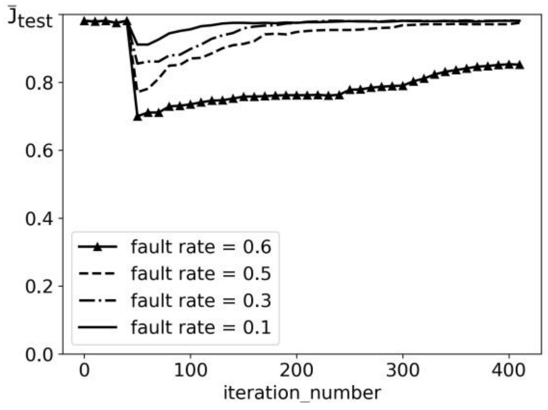
Fig. 2 shows the dependence of the value of the information criterion averaged over set of classes on the number of testing iterations of the model trained on the dataset_base. In this case, the first 50 iterations are performed without fault injection, and on the 51st iteration 4 versions of the model are generated with different fractions of tensors with an inverted bit in a random position, i.e. $\text{fault\_rate} \in \{0.1; 0.3; 0.5; 0.6\}$. Therefore, 4 performance recovery curves of the model are shown. Dependence of $\bar{J}$ (2) and R (3) on the fraction of tensors with inverted bit in random position for different dimensionality of feature representation at the output of feature extractor at different hierarchical level of classifier is shown in Table 1 and Table 2.

The analysis of Fig. 2 shows that the classification analysis algorithm is resilient to error injection if $\text{fault rate} \le 0.5$. However, the larger the greater the loss of classifier efficiency after recovery. Comparison of Fig. 2, a and Fig. 2, b shows that the top-level classifier exhibits a higher level of robustness and resilience, which allows it to be used in the mechanisms of graceful degradation under the influence of faults.

The choice of model parameters should be carried out taking into account the trade-off between the resilience and performance of the model (4). One of the important hyparameters of the classifier model is the dimensionality of the feature representation. However, the analysis of Table 1 and Table 2 shows that there is no unambiguous relationship between the dimensionality of the feature description, fault resilience and the information criterion of efficiency after recovery. Perhaps this relationship is clearer in the case of disturbances of another type.



a



b

Fig. 2. Dependence of the information criterion of efficiency averaged over set of classes (2) on test data from the number of iterations to the faults and impact of during recovery:
a – lower level of class hierarchy;
b – upper level of class hierarchy

Table 1

Recovery results for different fraction of damaged tensors and different dimensionality of feature representation at the lower hierarchical level of the classifier

Fault rate	R		$\bar{J}$	
	N=64	N=128	N=64	N=128
0.1	0.978	0.979	0.981	0.980
0.3	0.945	0.951	0.970	0.975
0.5	0.881	0.880	0.952	0.961
0.6	0.674	0.511	0.652	0.651

Table 2

Recovery results for different fraction of damaged tensors and different dimensionality of feature representation at the upper hierarchical level of the classifier

Fault_rate	R		$\bar{J}$	
	N=64	N=128	N=64	N=128
0.1	0.981	0.981	0.981	0.980
0.3	0.952	0.955	0.981	0.978
0.5	0.920	0.919	0.975	0.961
0.6	0.815	0.852	0.852	0.851

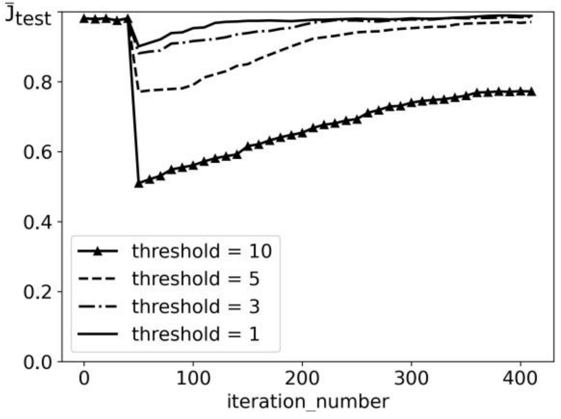
In practice, such metrics as L0-norm, L1-norm, L2-norm and L∞-norm have become widespread. However, only L0-norm and L∞-norm impose restrictions on the spatial distribution of noise, which protects against the formation of distorted samples that are misclassified even by humans. In addition, the choice of perturbation level by the L0-norm or L∞-norm metric does not depend on the image size, which is convenient for comparison. In the experiments, the formation of attacks is proposed to be implemented on the basis of the search algorithm of the covariance matrix adaptation evolution strategy (CMA-ES) using the L∞ metric [33].

Measurements of classifier efficiency are carried out on disturbed test samples, each mini-batch of disturbed test data is formed on the actual data model. At the same time, mini-batches of perturbed data from the dataset_additional set are formed, of which 20 % are provided with data labels to emulate active learning. The perturbed data from dataset_additional are not involved in measuring the model performance, but are used to adapt the model to perturbations of this type.

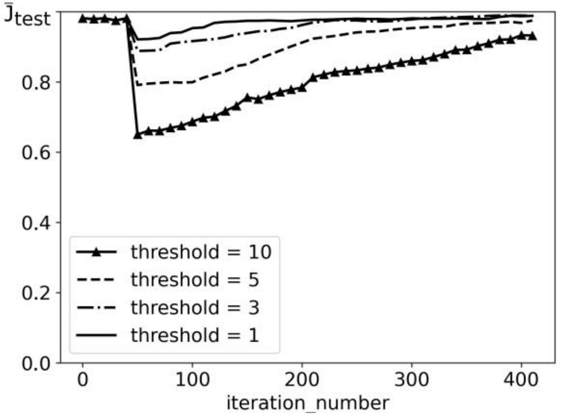
Fig. 3 shows the dependence of value of the information criterion averaged over set of clasess on the number of testing iterations of the model trained on the ataset_base. In this case, the first 50 iterations are performed without fault injection, and on the 51st iteration, data sets with 4 different threshold values of the disturbance level are generated, that is $\text{threshold} \in \{1; 3; 5; 10\}$. Therefore, 4 performance recovery curves are displayed. The dependence of $\bar{J}$ (2) and R (3) on the level of perturbation of samples for different dimensionality of the feature representation at the output of the feature extractor at different hierarchical levels of the classifier is shown in Table 3 and Table 4.

The analysis of Fig. 3 shows that the classification analysis algorithm is resilient to disturbances if $\text{threshold} \le 5$. However, the larger threshold is, the greater the loss of classifier efficiency after recovery. In this case, the comparison of Fig. 3, a and Fig. 3, b shows that the top-level classifier exhibits a higher level of robustness and resilience, which allows it to be used in mechanisms of graceful degradation under the influence of adversarial attacks.

The analysis of Tables 3 and 4 shows that the dimension N = 128, although inferior in terms of the information criterion of efficiency after recovery, provides a better compromise solution according to the form (4) due to a significant improvement in the resilience of the model.



a



b

Fig. 3. Dependence of the value of information criterion of efficiency (2) averaged over set of classes (2) on test data from the number of iterations before and after adversarial attack:
a – lower level of class hierarchy;
b – upper level of class hierarchy

Table 3

Recovery results for different levels of L∞-perturbation and different dimensionality of feature representation at the highest hierarchical level of the classifier

Threshold	R		$\bar{J}$	
	N=64	N=128	N=64	N=128
1	0.978	0.985	0.988	0.980
3	0.954	0.967	0.985	0.980
5	0.879	0.905	0.971	0.967
10	0.693	0.766	0.772	0.670

Table 4

Recovery results for different levels of L∞-perturbation and different dimensionality of feature representation at the lower hierarchical level of the classifier

Threshold	R		$\bar{J}$	
	N=64	N=128	N=64	N=128
1	0.980	0.988	0.988	0.980
3	0.955	0.965	0.988	0.979
5	0.885	0.905	0.975	0.965
10	0.793	0.843	0.932	0.930

In order to test the adaptation to new classes, we first train the model without two lower-level classes, for example, without horse and bird, and then add them as perturbation. To test the adaptation to the concept drift, from a certain point on, samples of the automobile class can be labeled as truck and truck as automobile. Changes in the test dataset occur synchronously with changes in the training dataset. Testing will take place after each training mini-batch on the full amount of test data. To make the analysis easier, each mini-batch will be formed in such a way that it will contain approximately the same number of samples of each of the currently relevant classes. Samples of new and existing classes will be taken from dataset_base, but for existing classes the samples will be selected from the buffer of the last labeled samples (100 samples per class). The mini-batch of labeled data can be extended by 10% with unlabeled samples from dataset_additional. It is assumed that the real concept drift will be detected automatically when the labeled samples from dataset_base with modified labeling are added to the training batches, which will then be added to the buffer of the last labeled data. The threshold for detecting real concept drift is set as 50 samples of one class in a container of another class.

Fig. 4 shows the dependence of the class-wise averaged information efficiency criterion calculated with test dataset on the number of training mini-packages. The first 200 iterations are associated with learning from scratch. Then there is a decline in performance when the labeled (and test) samples of new classes appear, to which the algorithm adapts. The second decline in productivity is associated with the real concept drift, due to the arrival of training (and test) samples with changed labels (truck replaced by automobile and vice versa).

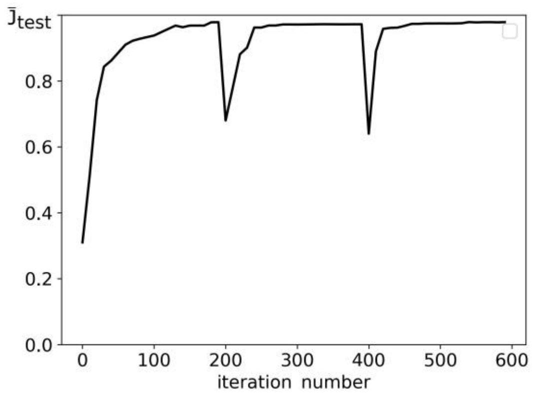


Fig. 4. Dependence of the efficiency criterion of model (2) calculated with test data on the number of training mini-batches during learning from scratch and during adaptation to new classes or to the real concept drift

The analysis of Fig. 4 shows that recovery requires 8 times less data than training from scratch. At the same time, the integrated resilience metric during adaptation to the emergence of two new classes is 0.959. The integral metric of resilience to the real concept drift between the two classes is 0.973. At the same time, after the recovery there was almost no loss of efficiency by the information criterion (2).

Thus, the ability of the proposed algorithm to absorb a certain level of destructive disturbance and the ability to adapt in order to recover and improve the performance is experimentally proved. At the same time, the higher efficiency and resilience of the algorithm for classes of higher hierarchical level is confirmed, which allows to implement the mechanisms of graceful degradation.

Conclusions

The proposed classifier model implements the mechanisms of perturbation absorption and graceful degradation. A key design aspect of the model is the inclusion of class hierarchical structure and hyperspherical separation surfaces (container) for each class.

A new learning method is proposed that combines maximization of compactness of class distribution and interclass buffer zone, discretization of feature description and regularization of consistency. Regularization of consistency is carried out both at the level of classification output and at the level of features and is used to increase robustness and speed of adaptation to destructive disturbances due to the effective use of unlabeled data.

When testing the proposed algorithm on the Cifar10 dataset, it was found that if the fraction of tensors damaged by the inversion of one randomly selected bit does not exceed 30 %, then the integral metric of resilience to this disturbance is about 0.95. However, at 60 % of damaged tensors the proposed algorithm does not provide resilience.

During the testing of the proposed algorithm, it was found that an adversarial attack with a perturbation that does not exceed the L∞-norm threshold equal to 3 provides a resilience that exceeds the integral value of 0.95. However, when the perturbation exceeds the L∞-norm threshold equal to 10, at the lower hierarchical level the resilience is not provided, and at the upper level the resilience by the integral metric is on the verge of recovery. It was noticed that the larger dimensionality of the feature space provides significantly better resilience to adversarial attacks at higher and lower hierarchical levels.

Experimentally confirmed on the open Cifar10 dataset the ability to adapt to the emergence of new classes or concept drift 8 times faster than learning from scratch. At the same time, the integrated resilience metric during adaptation to the emergence of two new classes is 0.959. The integral metric of resilience to the real concept drift between the two classes is 0.973.

The practical significance of the work is to form a new methodological basis for improving the resilience of image classification systems and quantifing of resilience to perturbing influences such as adversarial attacks, fault injection and concept drift.

Future research will focus on meta-learning algorithms to optimize the performance of data mining models. Particular attention will be devoted to increasing the resilience of the model by improving the architecture of the classifier model.

The research was concluded in the Intellectual Systems Laboratory of Computer Science Department at Sumy State University with the financial support of the Ministry of Education and Science of Ukraine in the framework of research project No 0122U000782 *“Information technology for providing resilience of artificial intelligence systems to protect cyber-physical systems”* (2022-2024).

Contribution of authors: conceptualization of the problem, development of the model and machine learning method, analysis of the results – **Viacheslav Moskalenko**; software implementation of the model and machine learning method, visualization of results and editing of work – **Alona Moskalenko**.

All authors have read and agreed with the published version of the manuscript.

References (GOST 7.1:2006)

1. Jha, S. Trinity: Trust, Resilience and Interpretability of Machine Learning Models [Text] / S. Jha, B. Jalaian, A. Roy, G. Verma // Game Theory and Machine Learning for Cyber Security. – IEEE, 2021. – P. 317-333. DOI: 10.1002/9781119723950.ch16.

2. Kumar, A. A Survey on Resilient Machine Learning [Electronic resource] / A. Kumar, S. Mehta. – 2017. – P. 1 – 9. Available at: https://doi.org/10.48550/arXiv.1707.03184.

3. Mani, G. Incremental Learning through Graceful Degradations in Autonomous Systems [Text] / G. Mani, B. Bhargava, B. Shivakumar // IEEE International Conference on Cognitive Computing (ICCC). – 2018. – P. 25–32. DOI: 10.1109/ICCC.2018.00011.

4. Wied, M. Conceptualizing resilience in engineering systems: Ananalysis of the literature [Text] / M. Wied, J. Oehmen, T. Welo // Systems Engineering. – 2020. – Vol. 23. – P. 3–13. DOI: 10.1002/sys.21491.

5. Поночовний, Ю. Л. Методологія забезпечення гарантоздатності інформаційно-керуючих систем з використанням багатоцільових стратегій обслуговування [Текст] / Ю. Л. Поночовний, В. С. Харченко // Радіоелектронні і комп’ютерні системи. – 2020. – № 3. – С. 43–58. DOI: 10.32620/reks.2020.3.05.

6. Huisman, M. A survey of deep meta-learning [Text] / M. Huisman, J. N. van Rijn, A. Plaat // Artificial Intelligence Review. – 2021. – Vol. 54, No. 6. – P. 4483–4541. DOI: 10.1007/s10462-021-10004-4.

7. Awasthi, A. Continual Learning with Neural Networks: A Review [Text] / A. Awasthi, S. Sarawagi // India Joint International Conference on Data Science and Management of Data. – 2019. – P. 362–365. DOI: 10.1145/3297001.3297062.

8. Smith, L. N. A useful taxonomy for adversarial robustness of Neural Networks [Text] / L. N. Smith // Trends in Computer Science and Information Technology. – 2020. – P. 037–041. DOI: 10.48550/arXiv. 1910.10679.

9. Encoding Generative Adversarial Networks for Defense Against Image Classification Attacks [Text] / J. M. Pérez-Bravo, José A. Rodríguez-Rodríguez, J. García-González, M. A. Molina-Cabello, K. Thurnhofer-Hemsi, E. López-Rubio // Bio-inspired Systems and Applications: from Robotics to Ambient Intelligence. IWINAC 2022. – Springer, Cham, 2022. – Vol. 13259 – P. 163–172. DOI: 10.1007/978-3-031-06527-9_16.

10. Liu, G. GanDef: A GAN Based Adversarial Training Defense for Neural Network Classifier / G. Liu, I. Khalil, A. Khreishah, // SEC 2019. IFIP Advances in Information and Communication Technology.

– Springer, Cham, 2019. – Vol. 562. – P. 19–32. DOI: 10.1007/978-3-030-22312-0_2.

11. Reluplex made more practical: Leaky ReLU [Text] / J. Xu, Z. Li, B. Du, M. Zhang, J. Liu // IEEE Symposium on Computers and Communications (ISCC). – 2020. – 7 p. DOI: 10.1109/ISCC50000.2020. 9219587.

12. Carlini, N. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods [Text] / N. Carlini, D. Wagner // Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security. – 2017. – P. 3–14. DOI: 10.1145/3128572. 3140444.

13. Athalye, A. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples [Text] / A. Athalye, N. Carlini, D. Wagner // 35th International Conference on Machine Learning. – 2018. – 12 p. DOI: 10.48550/arXiv.1802.00420.

14. Silva, S. Opportunities and Challenges in Deep Learning Adversarial Robustness: A Survey. [Text] / S. Silva, P. Najafirad // IEEE Transactions on Knowledge and Data Engineering. – 2020. – 20 p. DOI: 10.48550/arXiv.2007.00753.

15. Goldblum, M. Adversarially Robust Distillation [Text] / M. Goldblum, L. Fowl, S. Feizi, Tom Goldstein // AAAI Technical Track: Machine Learning. – 2020. – Vol. 34, iss. 04. – P. 3996-4003. DOI: 10.1609/ aaai.v34i04.5816.

16. Discrete Infomax Codes for Supervised Representation Learning [Text] / Y. Lee, W. Kim, W. Park, S. Choi // Entropy. – 2022. – No. 24, Iss. 4. – Article ID: 501. – 31 p. DOI: 10.3390/e24040501.

17. Chu, L.-C. Fault tolerant neural networks with hybrid redundancy / L.-C. Chu, B. W. Wah // 1990 IJCNN International Joint Conference on Neural Networks. – San Diego, CA, USA, 1990. – Vol. 2. – P. 639-649. DOI: 10.1109/IJCNN.1990.137773.

18. Training modern deep neural networks for memory-fault robustness [Text] / G. B. Hacene, F. Leduc-Primeau, A. B. Soussia, V. Gripon, F. Gagnon // IEEE International Symposium on Circuits and Systems (ISCAS 2019). – 2019. – 5 p. DOI: 10.1109/ISCAS.2019.8702382.

19. FTT-NAS: Discovering Fault-Tolerant Neural Architecture [Text] / W. Li, X. Ning, G. Ge, X. Chen, Y. Wang, H. Yang // 25th Asia and South Pacific Design Automation Conference (ASP-DAC). – 2020. – P. 211-216. DOI: 10.1109/ASP-DAC47756.2020.9045324.

20. Valtchev, S. Domain randomization for neural network classification [Text] / S. Valtchev, J. Wu // Journal of Big Data. – 2021. – Vol. 8. – Article No. 94. – 12 p. DOI: 10.1186/s40537-021-00455-5.

21. Konkle, T. A self-supervised domain-general learning framework for human ventral stream representation [Text] / T. Konkle, G. Alvarez // Nature Communications. – 2022. – Vol. 13. – Article No. 491. – 12 p. DOI: 10.1038/s41467-022-28091-4.

22. Multi-Task Learning with Knowledge Transfer for Facial Attribute Classification [Text] / X. Fanhe, J. Guo, Z. Huang, W. Qiu, Y. Zhang // IEEE International Conference on Industrial Technology (ICIT), Melbourne, VIC, Australia. – 2019. – P. 877–882. DOI: 10.1109/ICIT.2019.8755180.

23. Priya, S. Deep learning framework for handling concept drift and class imbalanced complex decision-making on streaming data [Text] / S. Priya, R. Uthra // Complex Intell. Syst. – 2021. – 17 p. DOI: 10.1007/s40747-021-00456-0.

24. To Trust Or Not To Trust A Classifier [Text] / H. Jiang, B. Kim, M. Y. Guan, M. R. Gupta // Proceedings of the 32nd International Conference on Neural Information Processing Systems. – 2018. – P. 5546–5557. DOI: 10.48550/arXiv.1805.11783.

25. P-ODN: Prototype-based Open Deep Network for Open Set Recognition [Text] / Y. Shu, Y. Shi, Y. Wang, T. Huang, Y. Tian // Scientific Reports. – 2020. – No. 10. – Article No. 7146. DOI: 10.1038/s41598-020-63649-6.

26. Detection and recovery against deep neural network fault injection attacks based on contrastive learning [Text] / C. Wang, P. Zhao, S. Wang, X. Lin // 3rd Workshop on Adversarial Learning Methods for Machine Learning and Data Mining at KDD. – 2021. – 5 p.

27. Cha, J. Hierarchical Auxiliary Learning [Text] / J. Cha, K. S. Kim, S. Lee // Machine Learning: Science and Technology – 2020. – Vol. 1, No. 4. – P. 1–12. DOI: 10.1088/2632-2153/aba7b3.

28. Active Learning by Acquiring Contrastive Examples [Text] / K. Margatina, G. Vernikos, L. Barrault, N. Aletras // Conference on Empirical Methods in Natural Language Processing. – 2021. – P. 650–663. DOI: 10.48550/arXiv.2109.03764.

29. OpenCoS: Contrastive Semi-supervised Learning for Handling Open-set Unlabeled Data [Text] / J. Park, S. Yun, J. Jeong, J. Shin // International Conference on Learning Representations ICLR 2021. – 2022. – 14 p. DOI: 10.48550/arXiv.2107.08943.

30. Multi-stage deep learning method with self-supervised pretraining for sewer pipe defects classification [Text] / V. Moskalenko, M. Zaretskyi, A. Moskalenko, A. Korobov, Y. Kovalsky // Radioelectronic and computer systems. – 2021. – No. 4. – P. 71–81. DOI: 10.32620/reks.2021.4.06.

31. Optimizing the F-Measure for Threshold-Free Salient Object Detection [Text] / K. Zhao, Sh. Gao, W. Wang, M.-M. Cheng // IEEE International Conference on Computer Vision (ICCV). – 2019. – P. 8848–8856. DOI: 10.1109/ICCV.2019.00894.

32. Qi, C. Contrastive-center loss for deep neural networks [Text] / C. Qi, F. Su // IEEE International Conference on Image Processing (ICIP), Beijing, China. – 2017. – P. 2851–2855. DOI: 10.1109/ICIP. 2017.8296803.

33. Kotyan, S. Adversarial robustness assessment: Why in evaluation both L0 and L∞ attacks are necessary [Text] / S. Kotyan, D. Vargas // PLOS ONE. – 2022. – No. 17(4). – Article No. e0265723. – 22 p. DOI: 10.1371/journal.pone.0265723.

References (BSI)

1. Jha, S., Jalaian, B., Roy, A., Verma, G. Trinity: Trust, Resilience and Interpretability of Machine Learning Models. *Game Theory and Machine Learning for Cyber Security, Chapter 16.* IEEE, 2021, pp. 317-333. DOI: 10.1002/9781119723950.ch16.

2. Kumar, A., Mehta S. A *Survey on Resilient Machine Learning*, 2017, pp. 1–9. Available at: https://doi.org/10.48550/arXiv.1707.03184.

3. Mani, G., Bhargava, B., Shivakumar, B. Incremental Learning through Graceful Degradations in Autonomous Systems. *IEEE International Conference on Cognitive Computing (ICCC)*, 2018, pp. 25–32. DOI: 10.1109/ICCC.2018.00011.

4. Wied, M., Oehmen, J., Welo, T. Conceptualizing resilience in engineering systems: Ananalysis of the literature. *Systems Engineering*, 2020, vol. 23, pp. 3–13. DOI: 10.1002/sys.21491.

5. Ponochovnyi, Yu. L., Kharchenko, V. S. Metodolohiya zabezpechennya harantozdatnosti informatsiyno-keruyuchykh system z vykorystannyam bahatotsil'ovykh stratehiy obsluhovuvannya [Dependability assurance methodology of information and control systems using multipurpose service strategies]. *Radioelektronni i komp'uterni sistemi – Radioelectronic and computer systems*, 2020, no. 3, pp. 43–58. DOI: 10.32620/reks.2020.3.05. (In Ukrainian).

6. Huisman, M., Rijn, J. N., Plaat, A. A survey of deep meta-learning. *Artificial Intelligence Review,* 2021, vol. 54, no. 6, pp. 4483–4541. DOI: 10.1007/s10462-021-10004-4.

7. Awasthi, A.m Sarawagim S. Continual Learning with Neural Networks: A Review. *India Joint International Conference on Data Science and Management of Data,* Kolkata, India, 2019. pp. 362–365. DOI: 10.1145/3297001.3297062.

8. Smith, L. N. A useful taxonomy for adversarial robustness of Neural Networks. *Trends in Computer Science and Information Technology*, 2020, pp. 037-041. DOI: 10.17352/tcsit.000017.

9. Pérez-Bravo, J. M., Rodríguez-Rodríguez, José A., García-González, J., Molina-Cabello, M. A., Thurnhofer-Hemsi, K., López-Rubio, E. Encoding Generative Adversarial Networks for Defense Against Image Classification Attacks. *Bio-inspired Systems and Applications: from Robotics to Ambient Intelligence. IWINAC 2022,* Springer, Cham, 2022, vol. 1325, pp. 163–172. DOI: 10.1007/978-3-031-06527-9_16.

10. Liu, G., Khalil, I., Khreishah, A. GanDef: A GAN Based Adversarial Training Defense for Neural Network Classifier. *SEC 2019. IFIP Advances in Information and Communication Technology*, Springer, Cham, 2019, vol. 562, pp. 19–32. DOI: 10.1007/978-3-030-22312-0_2.

11. Xu, J., Li, Z., Du, B., Zhang, M. Liu, J. Reluplex made more practical: Leaky ReLU. *IEEE Symposium on Computers and Communications (ISCC)*, 2020. 7 p. DOI: 10.1109/ISCC50000.2020.9219587.

12. Carlini, N., Wagner, D. Adversarial Examples Are Not Easily Detected. *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017, pp. 3-14. DOI: 10.1145/3128572.3140444.

13. Athalye, A., Carlini, N., Wagner, D. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. *35th International Conference on Machine Learning*, 2018. 12 p. DOI: 10.48550/arXiv.1802.00420.

14. Silva, S., Najafirad, P. Opportunities and Challenges in Deep Learning Adversarial Robustness: A Survey. *IEEE Transactions on Knowledge and Data Engineering,* 2020. 20 p. DOI: 10.48550/arXiv. 2007.00753.

15. Goldblum, M., Fowl, L., Feizi, S., Goldstein, T. Adversarially Robust Distillation. *AAAI Technical Track: Machine Learning*, 2020, vol. 34 (04). pp. 3996–4003. DOI: 10.1609/aaai.v34i04.5816.

16. Lee, Y., Kim, W., Park, W., Choi, S. Discrete Infomax Codes for Supervised Representation Learning. *Entropy,* 2022, no. 24, iss. 4, article id: 501. 31 p. DOI: 10.3390/e24040501.

17. Chu, L.-C., Wah, B. W. Fault tolerant neural networks with hybrid redundancy. *1990 IJCNN International Joint Conference on Neural Networks.* San Diego, CA, USA, 1990, vol. 2. pp. 639-649. DOI: 10.1109/IJCNN.1990.137773.

18. Hacene, G. B., Leduc-Primeau, F., Soussia, A. B., Gripon, V., Gagnon, F. Training modern deep neural networks for memory-fault robustness. *IEEE International Symposium on Circuits and Systems (ISCAS 2019),* 2019. 5 p. DOI: 10.1109/ISCAS.2019.8702382.

19. Li, W., Ning, X., Ge, G., Chen, X., Wang, Y., Yang, H. FTT-NAS: Discovering Fault-Tolerant Neural Architecture. *25th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2020, pp. 211-216. DOI: 10.1109/ASP-DAC47756.2020.9045324.

20. Valtchev, S., Wu, J. Domain randomization for neural network classification. *Journal of Big Data,*

2021, no. 8, article no. 94. 12 p. DOI: 10.1186/s40537-021-00455-5.

21. Konkle, T., Alvarez, G. A self-supervised domain-general learning framework for human ventral stream representation, *Nature Communications*, 2022, no. 13, article no. 491. 12 p. DOI: 10.1101/2020.06.15.153247.

22. Fanhe, X., Guo, J., Huang, Z., Qiu, W., Zhang, Y. Multi-Task Learning with Knowledge Transfer for Facial Attribute Classification. *IEEE International Conference on Industrial Technology (ICIT)*, Melbourne, VIC, Australia, 2019, pp. 877–882. DOI: 10.1109/ICIT.2019.8755180.

23. Priya, S., Uthra, R. Deep learning framework for handling concept drift and class imbalanced complex decision-making on streaming data, *Complex Intell. Syst.*, 2021. 17 p. DOI: 10.1007/s40747-021-00456-0.

24. Jiang, H, Kim, B., Guan, M. Y., Gupta, M. R. To Trust Or Not To Trust A Classifier. 32nd *International Conference on Neural Information Processing Systems*, 2018, pp. 5546–5557. DOI: 10.48550/arXiv.1805.11783.

25. Shu, Y., Shi, Y., Wang, Y., Huang, T., Tian, Y. P-ODN: Prototype-based Open Deep Network for Open Set Recognition. *Scientific Reports*, 2020, no. 10, article no. 7146. DOI: 10.1038/s41598-020-63649-6.

26. Wang, C., Zhao, P., Wang, S., Lin, X. Detection and recovery against deep neural network fault injection attacks based on contrastive learning. *3rd Workshop on Adversarial Learning Methods for Machine Learning and Data Mining at KDD*, 2021. 5 p.

27. Cha, J., Kim, K. S., Lee, S. Hierarchical Auxiliary Learning. *Machine Learning: Science and Technology*, 2020, vol. 1, no. 4, pp. 1–12. DOI: 10.1088/2632-2153/aba7b3.

28. Margatina, K., Vernikos, G., Barrault, L., Aletras, N. Active Learning by Acquiring Contrastive Examples. *Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 650–663, DOI: 10.48550/arXiv.2109.03764.

29. Park, J., Yun, S., Jeong, J., Shin, J. OpenCoS: Contrastive Semi-supervised Learning for Handling Open-set Unlabeled Data. *International Conference on Learning Representations ICLR 2021*, 2022. 14 p. DOI: 10.48550/arXiv.2107.08943.

30. Moskalenko, V., Zaretskyi, M., Moskalenko, A., Korobov, A., Kovalsky, Y. Multi-stage deep learning method with self-supervised pretraining for sewer pipe defects classification. *Radioelektronni i komp'uterni sistemi – Radioelectronic and computer systems*, 2021, no. 4, pp. 71-81. DOI: 10.32620/reks.2021.4.06.

31. Zhao, K., Gao, Sh., Wang, W., Cheng, Ming-Ming. Optimizing the F-Measure for Threshold-Free Salient Object Detection. *IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 8848–8856, DOI: 10.1109/ICCV.2019.00894.

32. Qi, C., Su, F. Contrastive-center loss for deep neural networks. *IEEE International Conference on Image Processing (ICIP)*, Beijing, China, 2017, pp. 2851-2855. DOI: 10.1109/ICIP.2017.8296803.

33. Kotyan, S., Vargas, D. Adversarial robustness assessment: Why in evaluation both L0 and L∞ attacks are necessary. *PLOS ONE*, 2022, no. 17(4), article no. e0265723. 22 p. DOI: 10.1371/journal.pone.0265723.

Надійшла до редакції 18.06.2022 розглянута на редколегії 25.08.2022

НЕЙРОМЕРЕЖЕВИЙ КЛАСИФІКАТОР ЗОБРАЖЕНЬ ІЗ РЕЗІЛЬЄНТНІСТЮ ДО ДЕСТРУКТИВНИХ ЗБУРЮВАЛЬНИХ ВПЛИВІВ – АРХІТЕКТУРА ТА МЕТОД НАВЧАННЯ

В. В. Москаленко, А. С. Москаленко

Сучасні методи розпізнавання зображень є чутливими до різного типу збурень, що актуалізує розроблення резільєнтних інтелектуальних алгоритмів для критичних до безпеки застосувань. Метою статті є розроблення моделі і методу навчання класифікатора, що проявляє характеристики резільєнтності до протиборчих атак, інжекції несправностей та дрейфу концепцій. Запропонована модель має ієрархічну структуру прототипів та гіперсферичні межі класів, що формуються у просторі високорівневих ознак. Межі класів оптимізуються під час навчання і забезпечують поглинання збурень та витончену деградацію. У запропонованому методі навчання передбачається застосування комбінованої функції втрат, що дозволяє використання як розмічених, так і нерозмічених даних, реалізує компресію ознакового подання до дискретної форми і забезпечує компактність розподілу класів і максимізацію буферної зони між класами. Основною складовою функції втрат є усереднене за алфавітом класів значення нормованої модифікації інформаційної міри Шеннона, що виражена як функціонал точністних характеристик. При цьому точністні характеристики обчислюються на основі згладжених версій розподілу результатів перевірки статистичних гіпотез. Експериментально підтверджено, що запропонований підхід забезпечує певний рівень поглинання збурень, витончену деградацію та відновлення. Під час тестування запропонованого алгоритму на наборі даних Cifar10 встановлено, що інтегральний показник резільєтності до ушкодження тензорів інверсією одного випадково обраного біту становить близько 0,95, якщо частка ушкоджених тензорів не перевищує 30 %. Також, під час тестування запропонованого алгоритму було встановлено, що протиборча атака зі збуренням, що не перевищує за L∞-нормою поріг рівний 3, забезпечує резільєнтність, яка перевищує за інтегральним показником значення 0,95. Крім того інтегральний показник резільєнтності під час адаптації до появи двох нових класів становить 0,959. Інтегральний показник резільєнтності до реального дрейфу концепцій між двома класами становить 0,973. Підтверджено здатність адаптації до появи нових класів або дрейфу концепцій в 8 разів швидше ніж навчання з нуля.

Ключові слова: класифікація зображень; робастність; резільєнтність; витончена деградація; протиборчі атаки; інжекція несправностей; дрейф концепцій.

Москаленко В’ячеслав Васильович – канд. техн. наук, доц., доц. каф. комп’ютерних наук, Сумський державний університет, Суми, Україна; докторант кафедри комп’ютерних систем, мереж та кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського “Харківський авіаційний інститут”, Харків, Україна.

Москаленко Альона Сергіївна – канд. техн. наук, старш. викл. каф. комп’ютерних наук, Сумський державний університет, Суми, Україна.

Viacheslav Moskalenko – PhD, associate professor of Computer Science Department of Sumy State University, Sumy, Ukraine; doctoral student of Department of Computer Systems, Networks and Cybersecurity, National Aerospace University “Kharkiv Aviation Institute”, Kharkiv, Ukraine,
e-mail: v.moskalenko@cs.sumdu.edu.ua, ORCID: 0000-0001-6275-9803, Scopus Author ID: 57189099775.

Alona Moskalenko – PhD, senior lecturer of Computer Sciences Department of Sumy State University, Sumy, Ukraine,
e-mail: a.moskalenko@cs.sumdu.edu.ua, ORCID: 0000-0003-3443-3990, Scopus Author ID: 57148522500.