https://doi.org/10.31891/csit-2025-1-20
UDC 004.932:[004.6:004.85]

Dmytro DASHENKOV, Kyrylo SMELYAKOV
Kharkiv National University of Radio Electronics

METHOD FOR EXTENDING IMAGE CLASSIFICATOR VIA TEXT METADATA STATISTICAL ANALYSIS

The goal of this article is to create a new method for obtaining a neural network for image classification that would work with classes for which examples are not available at the time of training. The task of image classification involves assigning one or more labels to an image based on the objects present in the image. The current state of the art method for creating such neural networks is to train models on the necessary data in a fine-tuning manner. The research methodology is to use existing machine learning models and expand the set of classes that the model operates on by manipulating the weight coefficients of the existing classifier model. The proposed method uses text metadata related to the images and descriptions of object classes to build assumptions about the relationship between different image classes. The method involves, using simple statistical calculations on text data, based on the existing weights of the neural network classifier, generating additional weights for recognizing new classes of objects in the image. The result of the research is the development of an algorithm for obtaining a classifier model that works with a class or classes that are not available during training. The model shows a classification accuracy result higher than the basic random one. At the same time, the classification accuracy for new classes, expressed in the F-score measure, is approximately 0.66, which is lower than the corresponding F-score measure for classes that were present during training, which is approximately 0.93. Also, the paper shows the limitations of the statistics-based approach to fine tuning, highlighting that it is not a full replacement for the classical model training. The scientific novelty lies in the development of methods for expanding image classifier models using statistical analysis of text metadata. The practical significance of the research lies in two aspects. The first aspect is obtaining a more stable base line of classification quality for classes that are added to the models after training using more sophisticated methods. The second aspect is obtaining a method for expanding the classifier for cases when extra data for additional training is not available and the training process itself is not possible due to a lack of computational resources.

Keywords: neural networks, natural language processing, machine learning, finetuning, image classification, multimodal data, statistical data processing.

Дмитро ДАШЕНКОВ, Кирило СМЕЛЯКОВ
Харківський національний університет радіоелектроніки

МЕТОД РОЗШИРЕННЯ КЛАСИФІКАТОРА ЗОБРАЖЕНЬ НА ОСНОВІ СТАТИСТИЧНОГО АНАЛІЗУ ТЕКСТОВИХ МЕТАДАНИХ

Метою цієї статті є створення нового методу отримання нейронної мережі для класифікації зображень, яка б працювала із класами, приклади для яких не доступні на момент навчання. Задача класифікації зображень передбачає призначення зображенню однієї чи декількох міток на основі об'єктів що присутні на зображенні. Поточний загальноприйнятий спосіб створення подібних нейронних мереж полягає в донавчанні моделей на потрібних даних. Методика дослідження полягає в тому, щоб використовувати існуючі моделі машинного навчання і розширює множину класів, якими оперує модель, проводячи маніпуляції із ваговими коефіцієнтами існуючої моделі-класифікатора. Метод що пропонується використовує текстові метадані про зображення та описи об'єктів, що представляють класи зображень для побудови припущень щодо зв'язки між різними класами зображень. Метод передбачає, за допомогою простих статистичних обчислень на основі текстових даних, на основі існуючих вагів нейронної мережі класифікатора, генерацію додаткових вагів для розпізнавання нових класів об'єктів на зображенні. Результатом дослідження є розробка алгоритму отримання моделі класифікатора, що працює із класом чи класами, не доступними під час навчання. Модель показує результат точності класифікації вищий за базовий випадковий. При цьому, точність класифікації для нових класів, виражена в мірі F-score дорівнює приблизно 0,66, що нижче ніж відповідна міра F-score для класів, що були присутні під час тренування — 0,9. Також, робота висвітлює обмеження статистичного підходу до донавчання моделей, підкреслюючи, що він не є повноцінною заміною власне навчанню. Наукова новизна полягає у розвитку методів розширення моделей-класифікаторів зображень за допомогою статистичного аналізу текстових метаданих. Практична значимість дослідження полягає у двох аспектах. Перший аспект — отримання більш стійкої базової міри якості класифікації для класів що додані до моделей пізніше тим чи іншим методом. Другий аспект — отримання методу розширення класифікатора для випадків коли додаткові дані для донавчання недоступні і сам процес тренування не можливий через брак обчислювальних ресурсів.

Ключові слова: нейронні мережі, обробка природної мови, машинне навчання, донавчання моделей, класифікація зображень, мультимодальні дані, статистична обробка даних.

Introduction

The task of image classification is one of the classic and most addressed problems of computer vision. There are many approaches to this problem. This paper considers an approach that uses machine learning models for single- and multi-class classification, i.e. the process that, for a given image, determines one or many classes of objects contained in the image. This approach is characterized by the need to train a neural network on all classes that exist in the domain for which it is necessary to perform classification. Thanks to the latest architectures of neural networks, the problem of image classification has received rapid development in recent years [1, 2].

The approach to image classification using machine learning models has two elements of complexity:

1) A large amount of data is required for training, namely, high-quality, human-annotated images representing the object that the model must recognize in all possible configurations.

2) Quite large computing power is required for training. Depending on the architecture, the process of training a model can take days or even weeks and require specific hardware, such as graphics cards with high RAM.

Also, given the prevalence of the classification problem, there are many trained models with large parameters that have high performance values and are available for free.

Most image classifier models are limited to a certain set of classes with which they work. Extending an image classifier involves increasing the set of classes with which the model works. A typical approach to extending a classifier is to further train the model on new classes, the so-called fine-tuning.

This paper proposes a method for using publicly available classifier models to classify images into classes that the model did not see during training without further training.

State-of-art

Currently, there are several architectures of neural networks that work with images. The main ones are as follows:

1) Convolutional neural networks. This architecture is the oldest among the ones listed here. Images are divided into blocks of pixels which, represented as a tensor of floating-point numbers, are multiplied by another tensor - the convolution kernel. This operation is repeated in each subsequent layer of the neural network [3].

2) Residual neural networks. The architecture of such neural networks is based on the architecture of convolutional networks, with the difference that for certain layers, the input of the convolution operation is not only the result of the previous convolution operation, but also the result of the layer that is several positions higher in the network. These two results are combined by a certain function in a process known as residual combination. This function itself varies in different implementations of this architecture, often the simple vector addition is used [4]. The residual combination operation is presented in formula (1).

$$\boldsymbol{x_i}=\mathrm{F}(\boldsymbol{x_{i-1}})\oplus\boldsymbol{x_{i-k}}, \tag{1}$$

where x_i — result of the i-th neural network layer, k — a small integer number, usually 2 or 3, $\oplus$ — the implementation dependent residual combination operation.

3) Transformer architecture. This architecture has gained popularity in recent years in the field of natural language processing but has also spread to other areas of application of neural networks. Transformers use attention mechanisms to find connections between different elements of an image and solve the specific tasks using these connections [5]. Often, neural networks built on this architecture also use the operation of residual combination of the results of the work of different layers [5, 6].

All of the architectures listed above share the same approach to how the model proceeds from image processing to classification decision-making, namely, using an approach that involves generating a vector representation of the input image — an embedding — and processing this vector using a dense layer of a neural network. All of the described features of the neural network architecture are found in the the encoder model that generates the embedding. The dense classifier layer itself is a simple linear layer of a neural network. Each neuron in the classifier layer generates a single value—the probability that the image contains an object of the class associated with this neuron. Thus, it is fair to assume that all neurons in the classifier layer are independent of each other during image processing. During training, the results of each neuron's operation affect the error function, and therefore the state of all other neurons; however, since this paper considers only the static state of the neural network, that is, the state outside of training, this effect can be neglected.

From the point of view of the mathematical model of a neural network, the classifier layer is an operation of matrix multiplication of the embedding by the weight matrix, as shown in formula (2).

$$\mathbf{P}=\mathbf{E}\cdot\mathbf{W}, \tag{2}$$

where P — probability vector, which describes the classes of objects present on the, E — vector, the result of the last layer of the encoder model — the embedding, W — the weight matrix of the classifier layer.

The matrix W specified in formula (2) has dimensions of the number of elements in the embedding vector to the number of classes the model works with. The probability of the presence of one particular class of objects in the image can be represented as a weighted sum of the elements of the embedding vector, where the weights are the elements of the corresponding column of the matrix W, as shown in formula (3).

$$P_i=\sum_N^{k=1} E_k W_{ki}, \tag{3}$$

where P_i — the probability of objects of the i-th class being present on the image, E_k — k-th element of the embedding vector, W_{ki} — k-th element of i-th column of the weight matrix.

In order to speed up the learning process, the method of retraining models on certain specific data is often used, instead of training the model from scratch. This method involves training only the classifier layer. In this case, the encoder model is in a frozen state, that is, it processes the image and generates embeddings, but its weights are not updated [7, 8].

This approach allows to significantly reduce the training time and the requirements for the hardware for training. However, this approach still requires a large amount of annotated data.

An alternative approach, implemented in the CLIP model, offers a method of classifying images without retraining the model, based on a model that has been trained on a combination of text and graphic data. CLIP accepts as input both an image and a text description of the object class and evaluates whether the given image corresponds to the given description. The advantage of this method is that the model is able to classify images into arbitrary classes that are not available during training. The disadvantage of the method is the high computational complexity of the model, the large amount of data required for training, the complexity of training, and lower accuracy than alternative models [9].

Another, more straightforward approach, is using a multimodal LLM-based model such as OpenAI o1 [10], DeepSeek-R1 [11], or Alibaba QwQ [12], and prompting the model to choose the correct class for a given image. Such approach allows for an even more flexible practical workflow. It also has the advantage of mature HTTP-based APIs for the models, which means that the used does not have to set up any inference infrastructure, such as GPUs and other heavy-loading hardware. The disadvantages of this approach are cost and speed. For use cases when inference speed matters, using a large language is not possible, regardless of whether or not the model is deployed by the used or accessed via an API. Also, the cost of using such APIs or running such a model is orders of magnitude higher than the cost of running a simpler image-processing neural network [13].

Purpose

Despite the widespread use of image classification, the problem of training neural networks to achieve high-quality classification is yet to be solved. As outlined above, training a model requires a large amount of data and significant computing power. Open datasets partially solve the former problem, but since they cannot cover all existing object classes, by their nature they are not exhaustive. Similarly, models available in the public domain are not exhaustive, since such models are aimed at use with a limited number of classes.

Large language and multimodal models capable of perform classification with any number of classes, such as CLIP [9], have been gaining ground recently. However, these models have their own drawbacks, the main of which are the cost and speed of inference.

Such solutions can be compared to the work of human operators who classify images manually. Similarly to large multimodal models, human operators do not need visual examples of objects that need to be recognized but can rely only on a general text description of the class.

Therefore, there is an unsolved problem of classifying images into an arbitrary number of classes that are not represented in some training set of images, which would be based only on image information obtained from a conventional neural network working with images.

The purpose of this work is to develop a method for extending the classifier model to work with previously unseen image classes with help of the statistical analysis of text metadata of the images.

Proposed technique

To develop a method for statistical analysis of text data, a specialized dataset was created. The dataset includes images, text metadata of the images, and text descriptions of objects that define image classes. The dataset is based on the ImageNet dataset, which contains images annotated with 1000 classes [14]. All classes represent objects such as animal species, common objects that can be found in a city (car, ambulance, etc.), clothing items, and other categories. The ImageNet Captions dataset [15] was used for text metadata for images. This dataset contains text descriptions and tags for 389598 images from ImageNet with 999 classes. This text data is represented by image titles, tags, and descriptions. This data is collected from the Flickr site, which is the source of images for the ImageNet dataset. This data is quite dirty, as it contains not only descriptions of images in isolation from context, but also names of locations, technical specifications of cameras, notes from authors of photos, etc. These details create noise in the data.

Another source of test data were English Wikipedia articles, which most closely represent the classes of objects from ImageNet. To create this set, an automatic search by class name was used. If a corresponding article was found, it was used. If a corresponding article was not found automatically, the closest article in content was selected manually. From each article, the text of a short description of the article object was taken. The collected data set, consisting of 1000 elements, is available publicly at https://gist.github.com/ddashenkov/6f399720b8fef309e7711b704bbc8272.

Thus, the data used for the experiment consists of three components: images from the ImageNet set, ImageNet Captions descriptions, and class descriptions collected from Wikipedia.

To digitize the text data, all components of the ImageNet Captions text descriptions for each image were merged into a single string, separated by spaces. All the resulting strings were tokenized using the NLTK library

[16]. Separately, the test descriptions of image classes were also tokenized. Next, all individual tokens are converted to lowercase. Tokens that contain any characters other than Latin letters are removed. Similarly, tokens shorter than 2 letters are removed. Also, all tokens that are present in more than 90% of the examples, i.e. in class descriptions or images up to more than 900 classes, are removed. A diagram of the text processing process is shown in Figure 1. The remaining tokens are the basis for conducting statistical studies of classes.

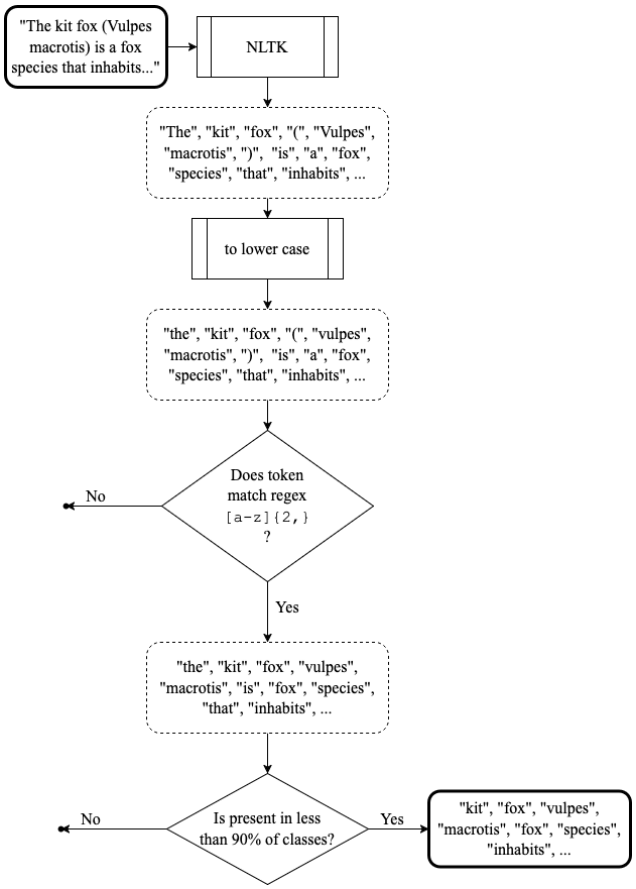

Fig.1. Diagram of the text data preprocessing. The input and the output are highlighted with a bold border, intermediate results are highlighted with a dotted line border

Further processing of text data constitutes grouping all tokens belonging to each of the classes into one list. To do this, each token in the entire set is assigned an identifier from 0 to the size of the set. In total, there are approx. 192K unique tokens. Next, the number of uses of each token is counted for each class. Since the volume of text data is different for different classes, the obtained frequencies are divided by the number of examples for the corresponding class in the ImageNet Captions dataset. Thus, the normalized histograms of the use of certain words in text data associated with ImageNet classes is obtained. These histograms can be represented as multidimensional vectors by placing the normalized frequencies of tokens in places according to the token identifiers. The identifier serves as an index of the frequency value in the vector, indexing starts from 0. The result is 1000 vectors each having 192K values. From now on, let's call them frequency vectors.

In order to establish the relationships between text data and images, a hypothesis was formed: the similarity of frequency vectors is related to the similarity of target vectors — column vectors of the classifier layer weight matrix.

To test this hypothesis, the similarity of two frequency vectors is determined by two methods: the calculation of the Euclidean distance, illustrated in formula (4), and the calculation of the cosine distance, illustrated in formula (5).

$$d_E^2 = \sum_N^{i=0} (x_i - y_i)^2 \quad (4)$$

$$d_c^2 = \frac{\sum_N^{i=0} (x_i y_i)^2}{\sum_N^{i=0} x_i^2 \sum_N^{i=0} y_i^2}, \quad (5)$$

where d_E — Euclidian distance between vectors x and y, d_c — cosine distance between vectors x and y, N — vector size, approx. 192K.

In order to obtain the target vector, one first needs to collect text data for the corresponding class, process it in the same way as the text data processing described above which was performed for existing image classes, obtain a frequency vector for the new class and calculate the distance between this vector and the frequency vectors of all the other classes. The result of this operation will be 2000 scalar values — one value per each class per distance calculation method. To obtain the target vector, it is necessary to take a weighted average of the vectors. The weights for this operation are the reciprocals of the distance between the vectors, as shown in formula (6).

$$\mathbf{L} = \frac{\sum_{1000}^{i=1} d_i^{-1} \mathbf{a}_i}{\sum_{1000}^{i=1} d_i^{-1}}, \quad (6)$$

where L — target vector, d_i — distance between frequency vectors of the i-th class and the new class, a_i — a column of the classificator layer which corresponds to the i-th class.

Thus, a target vector is generated for the new class, which is added to the weight matrix of the classifier layer. To choose between the vector comparison methods, an experiment is conducted with the existing classes. To do this, a small sample of validation classes is taken. For each of them, a target vector is generated by both methods. When generating a target vector for a particular class, the frequency vector of this class does not participate in calculating the distances, and in formula (6) the sums of 1000 elements are converted into sums of 999 elements.

Experiments

To test the hypothesis of a relationship between the similarity of frequency vectors and the similarity of target vectors, first, it is necessary to choose between the methods of calculating the distances between vectors. For this, four validation classes with the names koala, trailer_truck, bee and cheeseburger were selected. For each of the classes, the values of the distances between the frequency vectors were calculated. Table 1 shows some of the obtained distance values.

Table 1

Several values of frequency vector distances for the validation

Class name	Euclidian distance			Cosine distance		
	Closest class (distance)	Furthermost class (distance)	Mean among all classes	Closest class (distance)	Furthermost class (distance)	Mean among all classes
koala	sloth_bear (0.31)	apron (9.89)	4.74	sloth_bear (0.19)	photocopier (0.98)	0.62
trailer_truck	pickup (0.55)	poncho (14.21)	7.14	pickup (0.2)	bubble (0.96)	0.59
bee	fly (0.21)	mortarboard (9.41)	3.32	dragonfly (0.13)	microwave (0.72)	0.63
cheeseburger	rotisserie (1.35)	umbrella (10.65)	6.04	hotdog (0.12)	umbrella (0.88)	0.41

Next, to obtain the target vector for the validation classes, the weighted average value of the analogous weight vectors for other classes was taken, according to formula (6). To reduce noise, vectors with weights less than the average value for all classes were set to zero. Thus, only vectors with weight values above the average were taken into account.

To evaluate the obtained target vectors, the vectors are tested as weights of the neural network. For this, for each class, 1000 images corresponding to this class and 1000 images corresponding to other classes were randomly selected. These sets were tested on three variants of the model, the original model with trained weights, a model with weights for the corresponding class obtained by the described method using cosine distance, and a model with weights for the corresponding class obtained by the described method using Euclidean distance. The EfficientNet-v2-L model trained on the ImageNet dataset [17] is used as the basis. The effectiveness of the model is assessed by the F-score measure. In this case, the calculation is performed exclusively for the validation classes, i.e., binary classification is considered, other previously trained classes are ignored and each class is evaluated separately. The results obtained show that the use of cosine distance is more effective. The test results are shown in the graph in Figure 2.

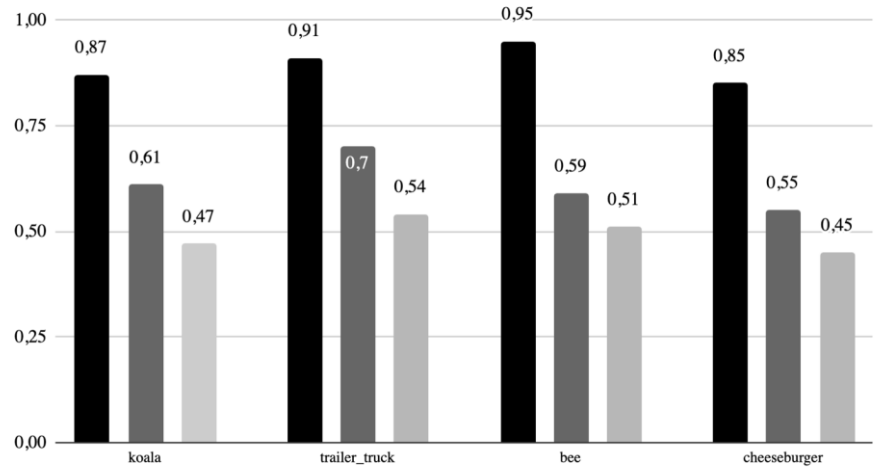

Fig.2. Validation classes F-score graph. Black denotes values for the trained model, dark-gray — values received via generated target vectors obtained with the use of cosine distance between frequency vectors, light-gray — values received via generated target vectors obtained with the use of Euclidian distance between frequency vectors.

The mathematical expectation of the F-score values that could be obtained with random weights for binary classification is 0.5. The graph shows that the empirically obtained values for weights generated using Euclidean distance are close to random. Also, the graph shows that the F-score values obtained when evaluating the model with weights generated using cosine distance are higher than random, but lower than those obtained with the trained model.

To further evaluate the method, the same algorithm for generating the target vector was performed for a new class that the model has never seen before. The class is called “warship”, its representatives are images of modern military ships. To obtain data for this class, data from Wikipedia from the article "Warship" was used, 25 public domain images of warships were collected for testing, and descriptions of these images were written manually. The class description and image descriptions were processed using the method detailed above. For the obtained frequency vector, cosine distances were calculated with the frequency vectors of other classes. Using these distances, the weighted average of the weight vectors was obtained, i.e. the target vector of the warship class. The F-score value of the binary classification obtained for this vector on 25 images is 0.66.

Conclusions

This article proposes a method for expanding a neural network classifier of images to obtain a model that is capable of recognizing new image classes i.e. classes that are not present in the training set. At the same time, the neural network itself can have an arbitrary architecture, and the possibility of expanding the model should not be prepared for at the time of training. This allows the use of almost any modern classifier model, including those that are available publicly for free. At the same time, the method itself does not involve training the neural network, which significantly reduces the requirements for hardware.

The method itself does not provide high quality compared to training a neural network. However, given the complete absence of training data and the need to retrain the model, the obtained measurements can be considered acceptable. The experiment described in this article demonstrates the efficacy of such an approach. The neural network model obtained during the experiment reaches the average F-score value of 0.62 for binary classification among the five test classes.

The application of the proposed method is possible to obtain a high-quality base line for fine-tuning a neural network based on a small amount of data. If the model retrained on several image examples cannot achieve a similar level of efficiency of binary classification as one obtained via the proposed statistical method, it is rational to use the method proposed in this article instead of retraining.

Also, the method can be used in conditions of complete absence of examples for a new class. After all, to obtain similar efficiency from other zero-shot training methods, it is a model of a specific zero-shot-friendly architecture must be used.

References

1. Jiahui Yu, Zirui Wang, Vijay Vasudevan. CoCa: Contrastive Captioners are Image-Text Foundation Models. *ArXiv*. 2022. DOI: https://doi.org/10.48550/arXiv.2205.01917.
2. Wortsman M., Ilharco G., Gadre S. Y. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. *ArXiv*. 2022. DOI: https://doi.org/10.48550/arXiv.2203.05482.
3. O'Shea K., Nash R. An Introduction to Convolutional Neural Networks. *ArXiv*. 2015. DOI: https://doi.org/10.48550/arXiv.1511.08458.
4. García-Arias Á. L., Okoshi Y., Hashimoto M. Recurrent Residual Networks Contain Stronger Lottery Tickets / IEEE Access. 2023. Vol. 11. 16588-16604 p. DOI: https://doi.org/10.1109/ACCESS.2023.3245808.
5. Arman M. Optimizing Multimodal Transformers for Medical Image Captioning: Enhancing Automated Descriptions via AI Systems / 2025 IEEE 6th International Conference on Image Processing, Applications and Systems (IPAS), Lyon, France. 2025. 1-5 p. DOI: https://doi.org/10.1109/IPAS63548.2025.10924492.
6. Raffel C., Shazeer N., Roberts A.. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *ArXiv*. 2019. DOI: https://doi.org/10.48550/arXiv.1910.10683.
7. Cetinic E., Lipic T., Grgic S. Fine-tuning Convolutional Neural Networks for fine art classification / Expert Systems with Applications. 2018. V. 114. 107-118 p. ISSN 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2018.07.026.
8. Singh M. K., Kumar B. Fine Tuning the Pre-trained Convolutional Neural Network Models for Hyperspectral Image Classification Using Transfer Learning / Computer Vision and Robotics. 2023. 271-283 p. DOI: http://dx.doi.org/10.1007/978-981-19-7892-0_21.
9. Radford A., Kim J. W., Hallacy C. Learning Transferable Visual Models From Natural Language Supervision. *ArXiv*. 2021. DOI: https://doi.org/10.48550/arXiv.2103.00020.
10. Tianyang Zhong, Zhengliang Liu, Yi Pan. Evaluation of OpenAI o1: Opportunities and Challenges of AGI. *ArXiv*. 2024. DOI: https://doi.org/10.48550/arXiv.2409.18486.
11. Daya Guo, Dejian Yang, Haowei Zhang. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *ArXiv*. 2025. DOI: https://doi.org/10.48550/arXiv.2501.12948.
12. Yuchen Xia, Jiho Kim, Yuhan Chen. Understanding the Performance and Estimating the Cost of LLM Fine-Tuning. *ArXiv*. 2025. DOI: https://doi.org/10.48550/arXiv.2408.04693.
13. An Yang, Baosong Yang, Beichen Zhang. Qwen2.5 Technical Report. *ArXiv*. 2025. DOI: https://doi.org/10.48550/arXiv.2412.15115.
14. ImageNet. URL: https://www.image-net.org/index.php (access date: 23.11.2024).
15. Fang A., Ilharco G., Wortsman M. Data Determines Distributional Robustness in Contrastive Language Image Pre-training (CLIP). *ArXiv*. 2022. DOI: https://doi.org/10.48550/arXiv.2205.01397.
16. NLTK :: Natural Language Toolkit. URL: https://www.nltk.org/ (access date: 11.01.2025).
17. Mingxing Tan, Quoc V. Le. EfficientNetV2: Smaller Models and Faster Training. *ArXiv*. 2021. DOI: https://doi.org/10.48550/arXiv.2104.00298

Dmytro Dashenkov **Дмитро Дашенков**	PhD student, Software Engineering dept., Kharkiv National University of Radio Electronics, Kharkiv, Ukraine, email: dmytro.dashenkov@nure.ua https://orcid.org/0000-0001-9797-1863 Scopus Author ID: 57715601500 ResearcherID: LXA-2271-2024	аспірант кафедри Програмної Інженерії, Харківський національний університет радіоелектроніки, Харків, Україна.
Kyrylo Smelyakov **Кирило Смеляков**	ScD, Prof., Head of the Software Engineering dept., Kharkiv National University of Radio Electronics, Kharkiv, Ukraine, email: kirill.smelyakov@nure.ua https://orcid.org/0000-0001-9938-5489 Scopus Author ID: 57203149663 ResearcherID: LBI-2194-2024	доктор техн. наук, проф., зав. кафедри Програмної Інженерії, Харківський національний університет радіоелектроніки, Харків, Україна.