

SAINT PETERSBURG STATE UNIVERSITY

As a manuscript

Samarin Aleksei Vladimirovich

**COMBINED NEURAL NETWORK MODELS FOR  
SPECIFIC IMAGES CLASSIFICATION**

Scientific specialty 2.3.5. Mathematical and software support for computing  
systems, complexes and computer networks

Dissertation of Candidate of Physical and Mathematical Sciences

Translation from Russian

Scientific adviser: Doctor of Technical Sciences,  
Candidate of Physical and Mathematical Sciences  
Koznov Dmitrij Vladimirovich

St. Petersburg — 2023

## Table of contents

	Page
<b>Introduction</b> . . . . .	4
<b>1. Domain overview</b> . . . . .	12
1.1 Basic neural network architectures . . . . .	13
1.2 Modern deep neural network classifiers of general images . . . . .	23
1.3 Text detection . . . . .	27
1.4 Optical text recognition . . . . .	29
1.5 Combined neural network architectures . . . . .	32
<b>2. Combined neural network architecture for image recognition without explicit text representation</b> . . . . .	34
2.1 Architecture, principles of construction and operation of the combined neural network model VCA . . . . .	36
2.2 Encoders for extracting classifying features . . . . .	38
2.3 Autoencoder for highlighting the characteristic features of contours .	40
2.4 Consistency regularization adaptation . . . . .	41
2.5 Implementation of VCA for solving the problem of document classification . . . . .	42
2.6 Experiments . . . . .	43
2.7 Chapter Conclusion . . . . .	48
<b>3. Combined neural network architecture for image recognition with explicit use of textual information</b> . . . . .	50
3.1 Statement of the problem . . . . .	51
3.2 Image pre-processing. . . . .	52
3.3 CCIT architecture with OCR module . . . . .	55
3.4 CCIT architecture with visual descriptors . . . . .	56
3.4.1 Type A descriptor . . . . .	57
3.4.2 Type B descriptor . . . . .	60
3.4.3 Type C descriptor . . . . .	64

3.5	Photo recognition system for facades of commercial buildings with advertising signs . . . . .	66
3.6	Experiments . . . . .	67
3.7	Chapter Conclusion . . . . .	70
<b>Conclusion . . . . .</b>		<b>72</b>
<b>References . . . . .</b>		<b>73</b>
<b>Appendix A. Acts on the implementation of the results of dissertation work . . . . .</b>		<b>83</b>

## Introduction

**Research rationale.** Currently, in the field of computer vision, a number of problems related to the classification of images in difficult conditions are relevant. In particular, many Internet services (from payment systems to services for restoring access to accounts in social networks) actively use document recognition and analysis tools. For example, in such systems as «Gosuslugi»<sup>1</sup> resource and «Webmoney»<sup>2</sup> payment system, documents are processed to obtain personal information about the client. In the case when such a document is a scanned passport, the last name, first name, patronymic of the owner, as well as the series and number of the document, place and date of issue, division code, and so on are extracted from it. In addition to extracting direct information, it is also important to establish the authenticity of the document or the presence of the fact of its modification (using Adobe Photoshop and other similar tools).

It should be noted that such services are usually based on the analysis of fragments of personal data, watermarks, etc., taking into account their relative position on the document page. All this information is placed in the frame in a strictly defined way to simplify the task of automatic recognition of personal information. Therefore, strict requirements are imposed on photographs of documents. For example, the government online services portal<sup>3</sup> regulates the angle, lighting conditions and other parameters that determine the quality of recognition of inscriptions (such as «Tesseract»<sup>4</sup>). In the social network «VKontakte»<sup>5</sup>, in order to restore a user's access to an account, all personal information specified on the page of an identity document is not required. In most cases, this is enough to check the image of a certain part of the submitted document, which must fully contain the user's photo, as well as his first and last name.

Thus, it is possible to single out a group of approaches to document recognition that extract semantic information from significant areas of the image (photographs of a person, key inscriptions, watermarks) and their relative position [1–4]. However, these methods are not applicable to solving the problem of analyzing areas of docu-

---

<sup>1</sup><https://www.gosuslugi.ru>

<sup>2</sup><https://www.webmoney.ru>

<sup>3</sup><https://51.мвд.рф/folder/2411299>

<sup>4</sup><https://github.com/tesseract-ocr/tesseract>

<sup>5</sup><https://vk.com/>

ment images in the vicinity of user photos under the assumption of various lighting conditions and artificial modifications of the original images that deform some of the personal data. It should also be noted that in the context of the recognition task, only the information that is necessary to confirm that the user has a genuine identity document is required.

An example of another formulation of the problem of image classification in complex conditions is the task of classifying photographs of facades of commercial buildings by type services provided [5–10].

This task in general is quite difficult due to the presence on photos with unique fonts, text colors, sizes and styles design of advertising signs. Also, the quality of recognition is affected by the shooting conditions. It should be noted that the decision on whether a sample belongs to a particular category can be made on the basis of information obtained from the text of an advertising sign, but also using visual features, without isolating textual information.

At the moment, there are many general approaches to solving the problem of image classification [11]. However, their application is difficult in the case of image classification in complex and non-standard conditions, which is well known on the example of the problem of classifying building facades in the presence of advertising signs [6; 12–14]. An important characteristic feature of advertising posters, which significantly complicates their classification, is the absence of convex elements and the presence of printed text.

**Research background.** A significant contribution to the development of the theory and practice of solving problems of image classification was made by K. He, X. Zhang, S. Ren и J. Sun [15], K. Simonyan and A. Zisserman [16], M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. Chen [17], M. Tan and Q. Le [18]. It should also be noted the contribution of J. Redmon, S. Divvala, R. Girshick and A. Farhadi [19], W. Liu, D. Anguelov, D. Erhan, C. Szegedy and S. Reed [20] in solving the problem of localization of objects in the image. When considering specific text detectors, the work of M. Liao, B. Shi, X. Bai, X. Wang and W. Liu [21], Z. Tian, W. Huang, H. Tong, P. He and Y. Qiao [22], X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He и J. Liang [23] should also be highlighted. A great contribution to the development of image descriptors was made by T. Dittimi и C. Suen [24], J. Sun, Z. Shisong и W. Xiaosheng [25], C. Huang и J. Huang [26].

Note that the problems of image analysis with text fragments are very specific, which largely excludes the possibility of reusing the proposed mechanisms for solving a wider class of problems. Also, as will be shown in this dissertation, general image classification methods are ineffective in solving problems of classifying images with text, since most of the significant information can be contained in the text. Moreover, methods based only on optical text recognition also cannot provide full coverage of all characterizing features. There are a number of studies devoted to this topic, for example, T. Intasuwan, J. Kaewthong и S. Vittayakorn [5], as well as T. Tsai, W. Cheng, C. You, M. Hu, A. W. Tsui и H. Chi [8].

This paper demonstrates the effectiveness of using combined models in comparison with classical neural network methods of computer vision in solving tasks under consideration.

**Research goal.** The main **purpose** of this work is the development and implementation of mechanisms for classifying images with text fragments in complex and non-standard conditions: varying the shooting angle, different lighting conditions, changing the scale, the presence of highlights and shadows, and the presence of various artifacts in images.

To achieve this purpose, the following **research main objectives** were formulated:

- perform analysis of modern methods for classifying images, as well as methods for optical text recognition and extraction of diverse descriptors, including an analysis of the constructive components of deep neural network classifiers applicable to solving problems of image analysis with visual representations of text;
- develop and implement a neural network model for classifying images with the presence of text on the stage in conditions where explicit Optical Character Recognition (OCR) is impossible;
- develop and implement a neural network architecture for classifying images with the presence of text on the stage under conditions that allow the use of OCR technologies;
- perform an experimental study of the proposed neural network architecture to assess the quality of classification.

**Research outcomes.**

- Combined neural network architecture VCA (VGG Combined with Autoencoders) for image classification, which does not perform explicit recognition of textual information. The proposed architecture was used to solve the problem of determining the suitability of an image of an identity document, subject to varying shooting conditions.
- Combined neural network architecture CCIT (Combined Classifier of Images with Text) for image classification with explicit recognition of textual information. A number of additional descriptors were also developed to extract style characteristics and visual design details from the text, which significantly improved the quality of image recognition. The proposed approaches were used to solve the problem of classifying photographs of facades of commercial buildings according to the type of services provided.
- An experimental study of the classification quality of developed architectures in comparison with modern methods of general image classification in the context of the tasks under consideration.

**Methods and methodology.** The subject of this dissertation is the problem of classifying images with text fragments. The object of the study is a set of approaches, methods, models and tools for classifying images with the presence of textual information in complex conditions. The work used methods of data analysis and machine learning, various principles for constructing neural network architectures, numerical methods for solving optimization problems, theory of algorithms and mathematical statistics, as well as software engineering tools.

**Research correspondence to the scientific specialty.** The content of the dissertation research corresponds to the passport of the scientific specialty 2.3.5 (mathematical and software computing machines, complexes and computer networks):

- P. 4 (intelligent systems of machine learning, database management data and knowledge, tools for developing digital products), since the dissertation proposed tools for building intelligent systems (neural network architectures) for solving the problems of classifying some specific types of images;
- P. 7 (models, methods, architectures, algorithms, formats, protocols and human-machine interface software, computer graphics, visualization, image and video data processing, systems virtual reality, multimodal interaction in sociocyberphysical systems), since within the framework of this work new

image processing systems are proposed, including models, architectures and algorithms for their processing.

**Research novelty.** Within the framework of this work, the problem of image classification with restrictions on the quality of shooting, as well as methods for processing texts included in images, is solved. A new neural network architecture is proposed for the case when optical recognition of the text included in the image is not performed. The new architecture uses the combination of several image descriptors - intermediate representations of some deep neural network encoders working with the original image, and the results of the work of deep encoders on the image obtained from the original by special preprocessing methods. This type of preprocessing is based on a modified idea of the autoencoder.

Also, within the framework of this dissertation work, a new neural network architecture for image classification with optical text recognition was proposed. The architecture is based on a special combinatorial scheme that uses image area descriptors with text.

**Theoretical and practical value of the work.** The theoretical value of this work lies in the proposal of new neural network architectures for narrow classes of tasks that general image classification methods cannot cope with. These architectures are formulated in the most general form and can be used to create new neural network architectures aimed at different classes of problems. An important theoretical aspect is the proof of linearity in image sizes and used time hyperparameters for building image descriptors for the CCIT architecture.

The practical significance of the work lies in the use of the proposed architectures for creating models and target software services that solve specific practical problems: document recognition when restoring access rights, as well as recognition of signs of commercial buildings.

**Work approbation and publications.** The results of the dissertation were reported at a number of international scientific conferences: 8th International Conference «Analysis of Images, Social Networks and Texts, AIST», (July 17-19, 2019, Kazan, Russia), 5th conference «Software Engineering and Information Management, SEIM», (May 16, 2020, St. Petersburg, Russia), 14th international conference «Baltic Conference on Databases and Information Systems, DBIS», (June 16-19 2020, Tallinn, Estonia), 9th international conference «Analysis of Images, Social Networks and Texts, AIST», (October 15-16, 2020), international conference



«Science and Artificial Intelligence conference, SAIence», ( November 14-15, 2020, Novosibirsk, Russia), 25th International Conference «International Conference on Pattern Recognition, ICPR» (January 10-15, 2021, Milan, Italy), 8th International Conference «Image Mining. Theory and Applications, IMTA» (August 21–25, 2022, Montreal Quebec, Canada), 14th International Conference “International Conference on Data Analytics and Management in Data Intensive Domains, DAMDID/RCDL” (October 4-7, 2022, St. Petersburg, Russia).

The models developed in the course of the dissertation work are integrated and successfully work as part of the services of the Russian social network «VKontakte», which is confirmed by implementation acts.

**Publications on the topic of dissertation.** The results of the dissertation were published in 11 papers, of which 1 publication [27] was presented in a journal included in the list of peer-reviewed scientific journals in which the main scientific results of a dissertation for the degree of candidate of sciences should be published; 10 articles [6; 12; 13; 28–34] are published in foreign journals indexed by the Scopus scientometric system. Articles [6; 12; 13; 27–34] were written in collaboration.

The **personal contribution** of the author in these publications is as follows. In the article [27], the author developed a combined neural network model for recognizing photos of identity cards in difficult shooting conditions, developed a research plan and an experimental scheme, independently implemented classifier models based on VCA, integrated them into the access recovery system and other internal algorithms of the «VKontakte» social network , conducted the planned experiments to assess the quality of the classification. The co-authors participated in writing and correcting the text of the article, organized the collection and labeling of data sets for experiments. In articles [6; 12; 13; 28–34], the author proposed a deep neural network architecture CCIT with an OCR module and various modifications using descriptors. The author implemented the training and testing infrastructure, deployment of models within this architecture, proposed special methods for image preprocessing, adaptation of model training techniques, implementation of the basic version of the CCIT architecture and development of interfaces for connecting additional functional blocks, as well as the development of algorithms for constructing special type A, B descriptors , C. The co-authors performed the software implementation of the proposed descriptor calculation algorithms, connected the OCR mechanisms and vectorization of text representations, ensured the collection and

labeling of data sets, participated in setting up experiments and writing the text of articles.

**Scope and structure of the dissertation.** The dissertation work consists of an introduction, 3 chapters, a conclusion, a list of references and an appendix. The total volume of the work is 84 pages. The list of references contains 108 titles.

The **first chapter** describes the most effective and frequently used modern approaches to general image classification, text recognition methods, as well as ways to get image descriptors. These solutions are components of combined neural network classifiers. This chapter also describes the tasks of classifying images with text fragments in complex conditions. The greatest attention is paid to the task of classifying identity cards in the context of the system for restoring access to accounts of the «VKontakte» social network and the task of determining the type of services provided by photographs of the facades of commercial buildings.

The **second chapter** presents a description of the VCA neural network architecture is presented. The basic principles of the construction and operation of the VCA, the encoders used to extract classifying features (including motivation and considered alternatives) are described. An autoencoder is also described for feature extraction of contours and adaptation of the consistency regularization. The chapter also briefly presents the implementation of the VCA architecture for solving the document classification problem. At the end of the chapter, an experimental study of the created architecture is given, which showed its superiority over its analogues in terms of the metric *Precision* on the main (mixed) data set.

The **third chapter** contains a description of the CCIT architecture. A detailed statement of the problem and the procedure for pre-processing images, as well as an OCR module are presented. Algorithms for constructing image descriptors of type A, B, and C are described, and the properties of the linearity of the construction time of these descriptors on the image size and hyperparameters are proved. The implementation of the CCIT architecture in the form of a model for the classification of advertising signs of commercial buildings is described. Finally, an experimental study of the CCIT architecture is presented, which showed its superiority over analogues in the considered class of problems.

The **conclusion** presents the results of the study, as well as prospects for further work.

The **Appendix** contains acts of implementation of the methods developed within the framework of this thesis.

## 1. Domain overview

In this study, we will use three different, but close in meaning terms. We will use the term *neural network* in introductory or general descriptions. The term *model* will denote a specific neural network that has an implementation and is ready for use. In this case, the model settings may vary. Using the term *architecture*, we will denote a certain family of models - in fact, the architecture provides a certain abstract view (view point) on a set of different neural networks (existing or implemented in the future), within which certain structural elements, mechanisms and principles of operation of a neural network common to of all models with a given architecture, and the remaining components of these models, which may also be essential for implementation, are omitted and, therefore, may differ for these models. When creating specific models, it is possible to combine different architectures. Finally, we will use the term *classifier* to refer to neural network architectures designed to classify images.

To train all the neural network models mentioned below, the stochastic gradient descent method [35] and its numerous modifications [36–41] are used. Currently, this group of algorithms is actually the only and uncontested group of methods for optimizing the weights of deep neural networks. All these methods are developments of the basic method [42] of gradient descent, implementing various heuristics that improve the convergence process. In particular, the stochastic gradient descent algorithm [35] analyzes at each step of the algorithm not the entire manifold given by the data set, but only the surface given by some subsample (learning from the manifold given by the entire training set is impossible in practice due to limited resources). Algorithms based on the method of moments [40; 41], use the heuristic of inertia, which allows you to separate important details or, on the contrary, ignore noise when choosing the direction of optimization. Good results in solving a wide class of problems are demonstrated by optimization methods using change history accumulation and normalization, such as [36–39]. To train all the models presented in this paper, we used a modification of stochastic gradient descent [36].

## 1.1 Basic neural network architectures

Currently, the most effective image classification methods are based on deep neural networks [15; 18; 43–45], which use the following important concepts: convolution, attention, combination of different modalities, as well as various ensemble techniques. Further in the chapter, we will consider the main constructive components of deep neural networks, which were used in this work to build the architectures of image classification models.

**Neurons, layers and blocks.** An *artificial neuron* (hereinafter referred to as a neuron) is an atomic element of neural network<sup>1</sup>. Structurally, an artificial neuron is extremely simple and has a natural biological basis, since, to some extent, it models the work of one neuron from the human nervous system. The principle of operation of a neuron is shown in Fig. 1.1.

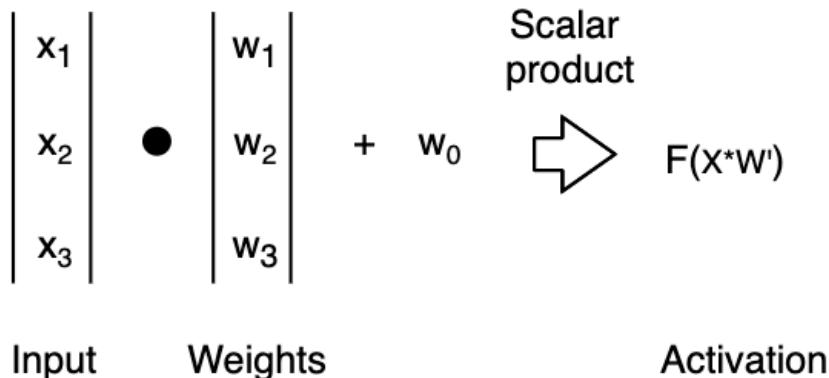


Figure 1.1 — *The principle of operation of an artificial neuron.*

Each component of the input signal of the neuron is multiplied by the weight corresponding to it, after which the activation function is applied to the result of the scalar product of the input vector and the weight vector, the value of which is transmitted further as a result of processing the input vector by the neuron. The weights in this case are the learning parameters of the neuron.

*Fully connected layers* (hereinafter, layers) and *fully connected blocks* (hereinafter, blocks) are used in deep neural network architectures as structural elements [46–49].

<sup>1</sup>about convolutional networks

*Layers* are the level of organization of the neural network following the neuron, containing a set of neurons. A *block* is a set of layers and additional links between layers. A *fully connected layer* is a layer in which all neurons are independent and when two fully connected layers are connected (this connection is directed), the output of each neuron from the previous fully connected layer is fed to the input of each neuron from the second fully connected layer (see Fig. 1.2). A *fully connected block* is a group of polyconnected layers. A neural network composed of fully connected layers is called a *multilayer perceptron* (Multilayer Perceptron, MLP).

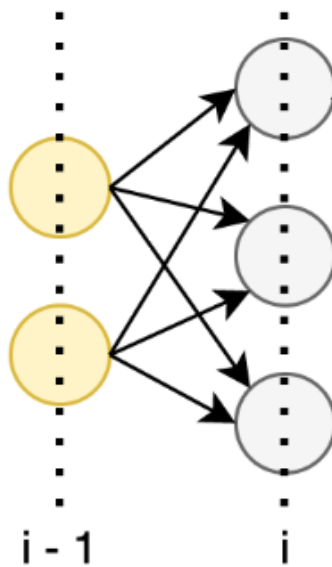


Figure 1.2 — *Linking fully connected layers.*

**Activation functions** [50] are applied inside the neuron, as mentioned above, as well as at the outputs of blocks and layers. The main purpose of such functions is to convert ranges of values. For example, the input range is from minus infinity to plus infinity, and the output range is from 0 to 1, which is necessary to predict various probabilities.

Next, we will consider the most common activation functions used in building deep neural network architectures — logistic function, hyperbolic tangent, ReLU (linear rectifier), SoftMax.

The *logistic function* is given as follows [51; 52]:

$$\sigma(z) = \frac{1}{1 + e^{-z}}.$$

It increases monotonically, amplifying weak signals and saturating strong ones, and is a smooth step function approximator (see Fig. 1.3).

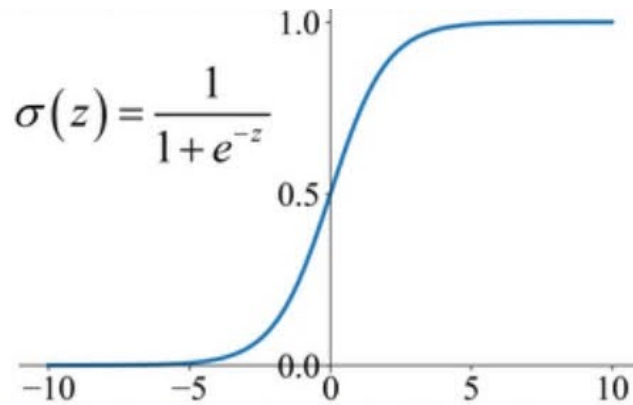


Figure 1.3 – *Graph of the logistic function.*

In deep neural networks, the logistic function is mainly used as an activation for fully connected layers at model outputs. It is usually not used for activation in deep convolutional blocks, as it leads to the problem of damped gradients due to the tendency to quickly saturate when moving away from the origin. These properties prevent the successful application of gradient descent to optimize the values of trainable parameters.

The logistic function has a limited range of values (in the range from 0 to 1) and is convenient to use as an activation function for the outputs of a deep model when solving a classification problem. Also, a convenient property of the logistic function is the ability to express the value of the derivative through the value of the function itself, which saves computational resources when optimizing the model:

$$\sigma'(z) = \sigma(z) * (1 - \sigma(z)).$$

*Hyperbolic tangent* [51; 52] is similar in properties to the logistic function:

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}.$$

The value of its derivative is also expressed in terms of the value of the original function:

$$\tanh'(z) = 1 - \tanh^2(z).$$

Hyperbolic tangent, like logistic regression, tends to boost weak signals and saturate from large input values (see Figure 1.4).

However, the undoubted advantage of the hyperbolic tangent over the logistic function is its centering around zero, which allows the gradients to take both positive

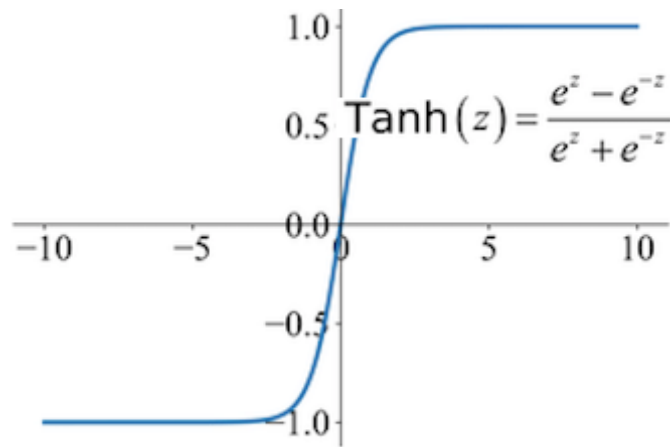


Figure 1.4 — *Graph of the hyperbolic tangent function.*

and negative values in the process of optimizing the model weights, preventing the occurrence of an undesirable zigzag effect during training.

Convolutional layers and blocks in deep neural networks often use the *ReLU* activation function [51; 52] (Rectified Linear Unit):

$$\text{ReLU}(\mathbf{z}) = \max(0, \mathbf{z}).$$

When using ReLU, there is no need to spend resources on finding the numerical value of derivatives when training a neural network, which significantly speeds up the optimization process. But despite the simplicity of the function itself, ReLU has the property of non-linearity (see Fig. 1.5), which allows you to build more powerful approximators than when using linear activation functions.

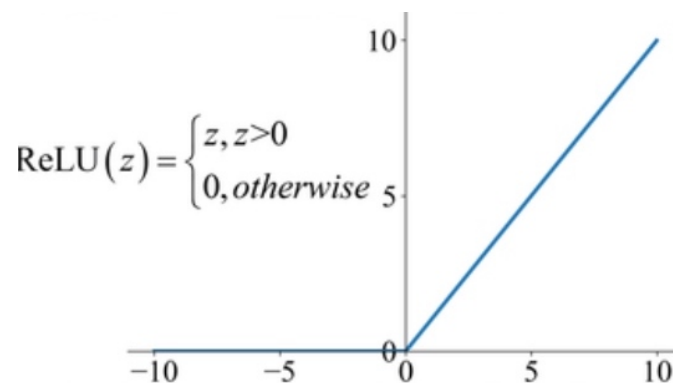


Figure 1.5 — *Graph of the ReLU function.*

The disadvantage of ReLU is that the function is unbounded from above, which entails the lack of saturation and, as a result, the problem of exploding gradients. In addition, negative values do not correct the gradient (a problem called degradation).



To solve the problem of ReLU degradation, its modification, called *Leaky ReLU*, is sometimes used [51; 52] (Leaky Rectified Linear Unit), which is defined as follows:

$$\text{Leaky\_ReLU}(\mathbf{z}) = \max(\beta * \mathbf{z}, \mathbf{z}).$$

Leaky ReLU reduces the risk of degradation of the ReLU function due to non-zero values on the negative semi-axis (see Fig. 1.6), but at the same time, the probability of an explosion of gradients increases.

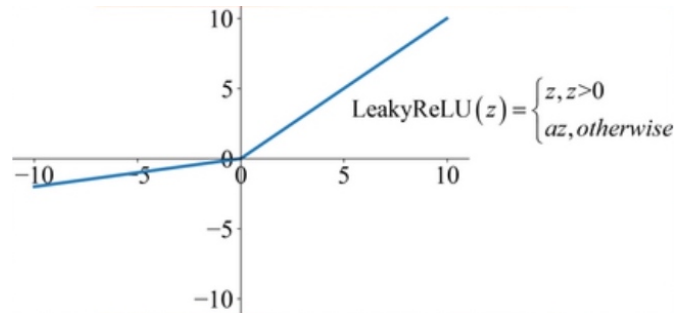


Figure 1.6 — *Graph of the Leaky ReLU function.*

There are many other variations of the ReLU function, and the choice of a specific variant depends on the optimal ratio of the degradation probability, the ease of calculating the derivative, the properties of nonlinearity, and the probability of exploding gradients. The preference for using one or another activation function depends on the context of the task and the neural network architecture used.

*SoftMax* function [51; 52]] allows you to smoothly approximate the maximum and is in some way a generalization of the logistic function into several classes:

$$\text{SoftMax}(z_i) = \frac{\exp(z_i)}{\sum_{k=0}^K \exp(z_k)}.$$

Thus, this function allows you to make a choice from several options and use optimization using gradient descent when learning. Variants are encoded using one-hot-representation [53], the most probable class is represented by a pronounced peak.

**Recurrent architectures (Recurrent Neural Network, RNN)** are designed to process sequential (ordered) data, in particular, texts in natural languages, as well as time series. Currently, these architectures are also used in computer vision for the tasks of processing a video stream, tracking objects for a sequence of frames, optical text recognition, etc. [22; 54].

Recurrent architectures are characterized by the following [55; 56]:

- by reproducing the same functionality for each element of the input sequence,
- dependence of the input of network elements on the output,
- the presence of memory that allows you to take into account the results of already processed information.

Fig. 1.7 demonstrates how the recurrent layer works. Note that during the processing of the input data sequence, the recurrent layer is represented as a chain of repeating elements, although it was possible to depict one single element with a loop.

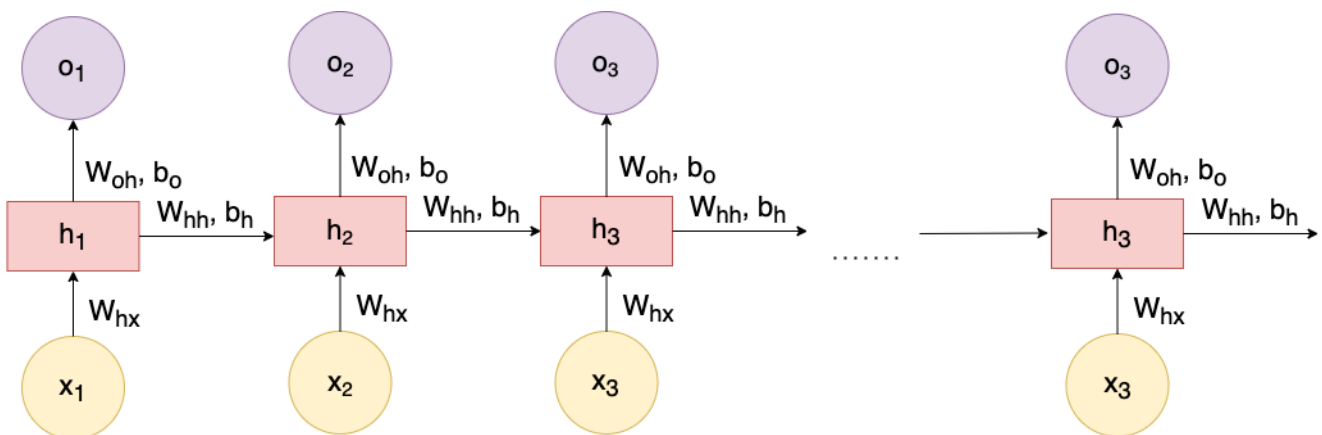


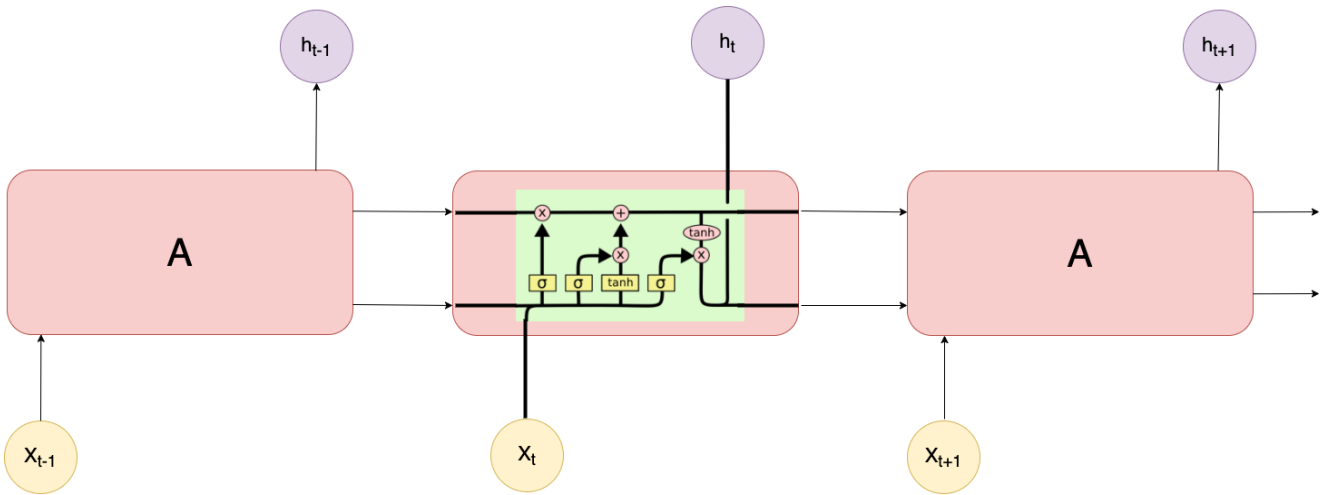
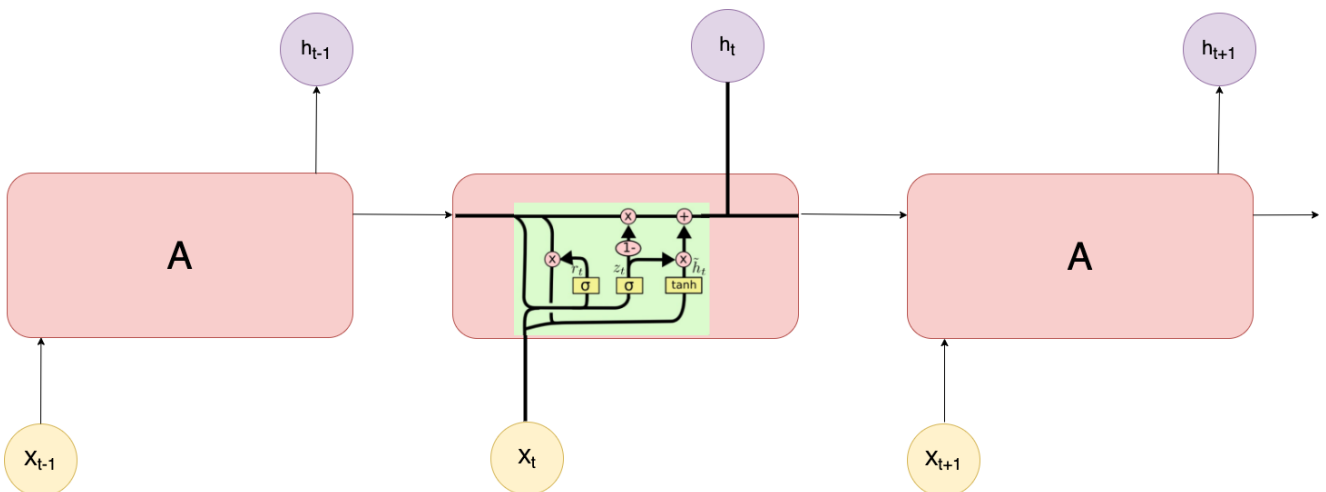
Figure 1.7 — *The device of the simplest recurrent layer.*

Some of the most famous recurrent architectures are LSTM (Long Short-Term Memory) [55; 57] and GRU (Gated Recurrent Unit) [55; 58]. LSTM adds capabilities to the underlying recurrent architecture to handle long-term dependencies. The change of state occurs with the help of special constructions — gateways [55; 57]. The principle of LSTM operation is shown in fig. 1.8.

Among the shortcomings of LSTM, there is a low speed of work, which motivated the appearance of lightweight analogues, such as the GRU [55; 58] architecture (Fig. 1.9).

The GRU architecture, compared to the LSTM, has one less gateway, and the GRU gateways are connected differently. The essential difference between GRU and LSTM is that it is faster and easier to use, but less expressive.

**Deep neural network architectures.** An obvious disadvantage of using neurons as the basic element of a neural network architecture designed for image classification is a large number of weights that arise when trying to build and train a deep fully connected model capable of processing high-resolution images.

Figure 1.8 — *LSTM construction.*Figure 1.9 — *GRU construction.*

*Convolutional Neural Network (CNN) architectures* help to solve this problem, in which the neurons in the layer are replaced by *convolution*. The latter is the following function [59]:

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n - m].$$

This function is used to obtain a feature map by convolving the kernel with the input tensor (see Figure 1.10).

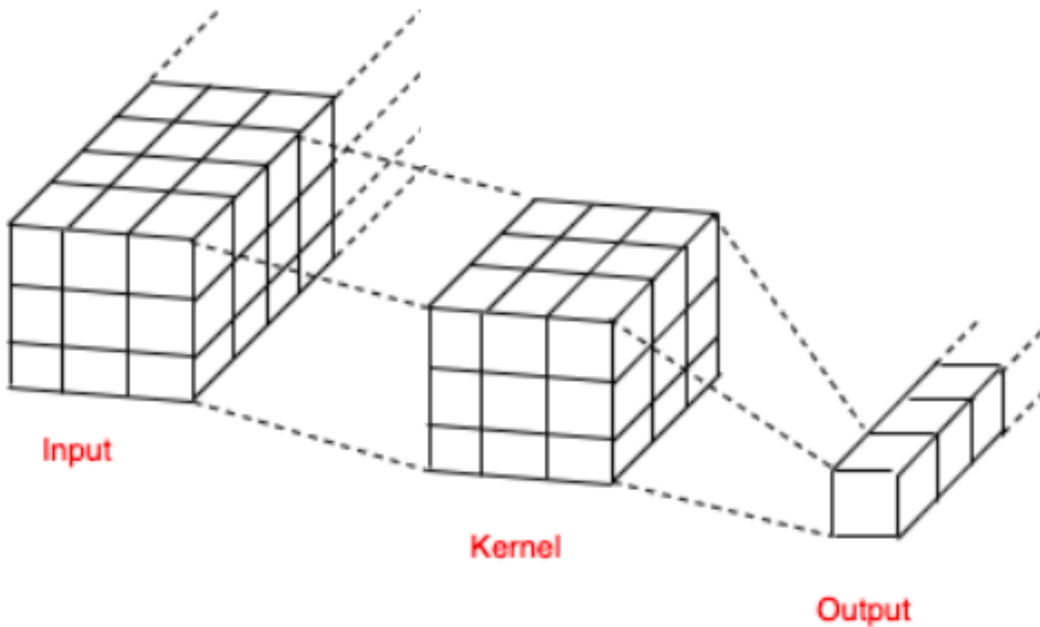


Figure 1.10 — *Illustration of the convolution operation.*

Thus, the number of parameters required to obtain the output tensor is much less than in a fully connected layer and is limited only by the size of the convolution kernel, which makes it possible to effectively train deep convolutional models.

Another important construction of convolutional architectures is *pooling* [59]. In the basic version, it is a function of choosing the maximum value from the region, and thus the input tensor is reduced in dimension. In this case, the division into regions is carried out using a grid (see Fig. 1.11).

*Dropout* operation [59; 60] is intended for regularization at the level of neural network architecture, reducing the effect of network overfitting by preventing com-

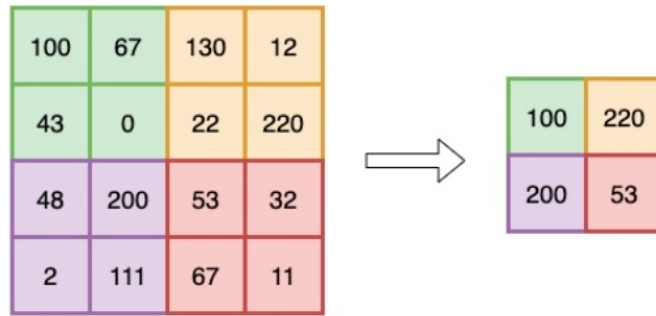


Figure 1.11 — *Pooling operation illustration.*

plex co-adaptations of individual neurons on training data during training. This approach involves the exclusion of a certain percentage of random neurons at different training iterations (see Fig. 1.12). Neuron exclusion is implemented by returning zero in response to any input.

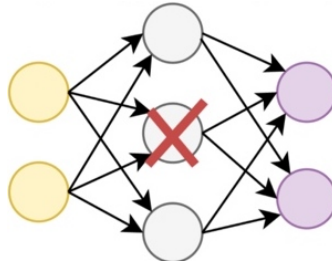


Figure 1.12 — *An illustration of a dropout operation.*

As a result, more trained neurons get more weight in the network, which significantly increases the learning rate, metrics on the training data, and also improves generalization.

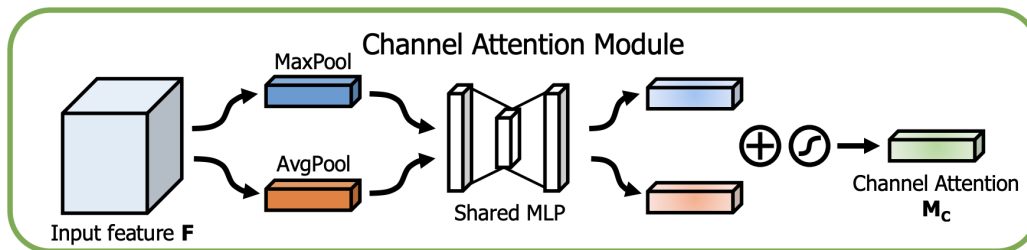


Figure 1.13 — *Block for calculating weights for the channel attention mechanism [61].*

*Mechanism of channel attention [61]* allows you to select the most informative channels from the processed tensor. This effect is achieved by weighting, the coefficients for which are determined during the operation of a special block (see fig. 1.13).

*Spatial Attention Mechanism* [61] does not highlight important channels, but allows you to highlight areas on the image that are most important for solving the target problem. Note that this block is also based on the use of a map for weighting (see fig. 1.14).

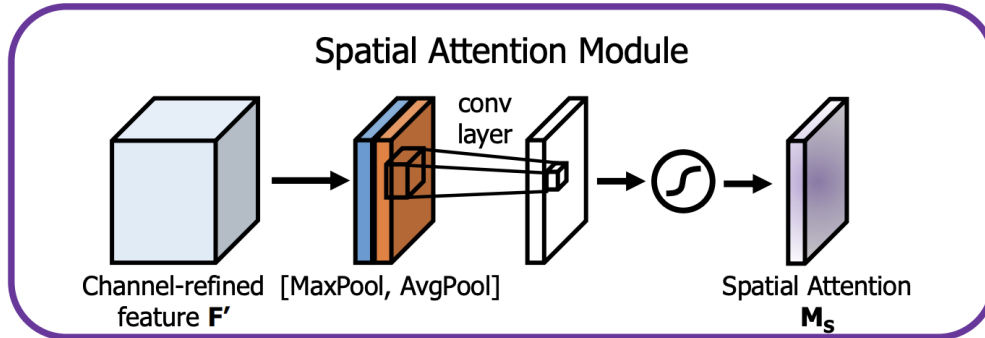


Figure 1.14 — *Block for calculating weights for the mechanism of spatial attention [61].*

Non-local blocks [62] builds on the *concept of self-attention* (see fig. 1.15) and gets rid of the problem of approaches based solely on local convolution operations and pooling operations associated with the locality of the studied patterns. This circumstance is due to the local nature of the mentioned transformations. To achieve nonlocality, a special construction is used, based on comparing different sections of the processed tensor for similarity.

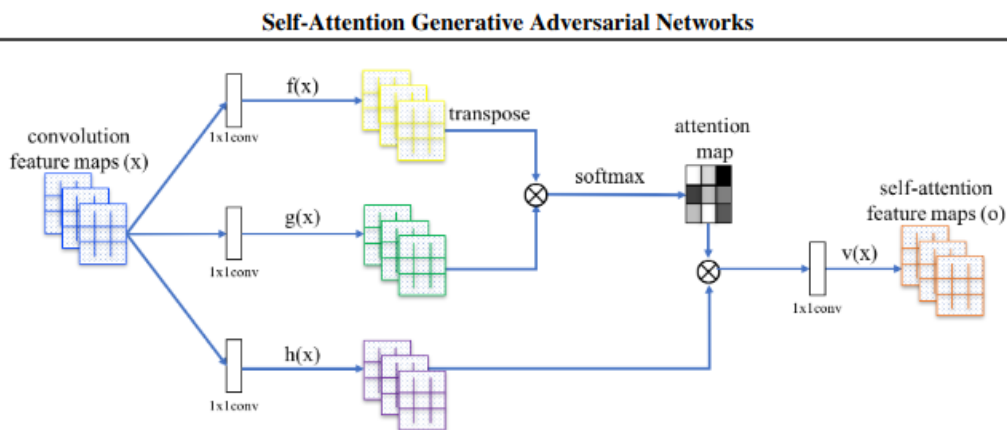


Figure 2. The proposed self-attention module for the SAGAN. The  $\otimes$  denotes matrix multiplication. The softmax operation is performed on each row.

Figure 1.15 — *An example of self-attention block adaptation [63] for image processing.*

## 1.2 Modern deep neural network classifiers of general images

Neural networks of the *VGG* (Visual Geometry Group) [16] family are sequence of convolutional blocks, pooling and fully connected layers (see Fig. 1.16).



Figure 1.16 — *Architecture VGG-16 [64].*

Despite some archaism, these architectures are still actively used in different areas of computer vision for feature extraction [27; 30]. In many ways this is due to their ability to generate interpretable feature maps.

The main disadvantages of VGG are associated with a large number of weights and uncomplicated architecture, which makes the learning process slow and the weight of the network itself large compared to more efficient and modern classifiers.

The main structural element of the architectures of the family *Inception* [65–67] are *Inception-blocks* (see Fig. 1.17), inside which the input tensor is passed in parallel through multiple independent sequences of convolutional layers, the results of which are combined using a  $1 \times 1$  convolution [68].

The main innovation of the family of neural network architectures *ResNet* (Residual Neural Network) [15; 69] are the residual links, which represent the addition of results, obtained on earlier layers to later ones, bypassing the convolutional blocks (Fig. 1.18).

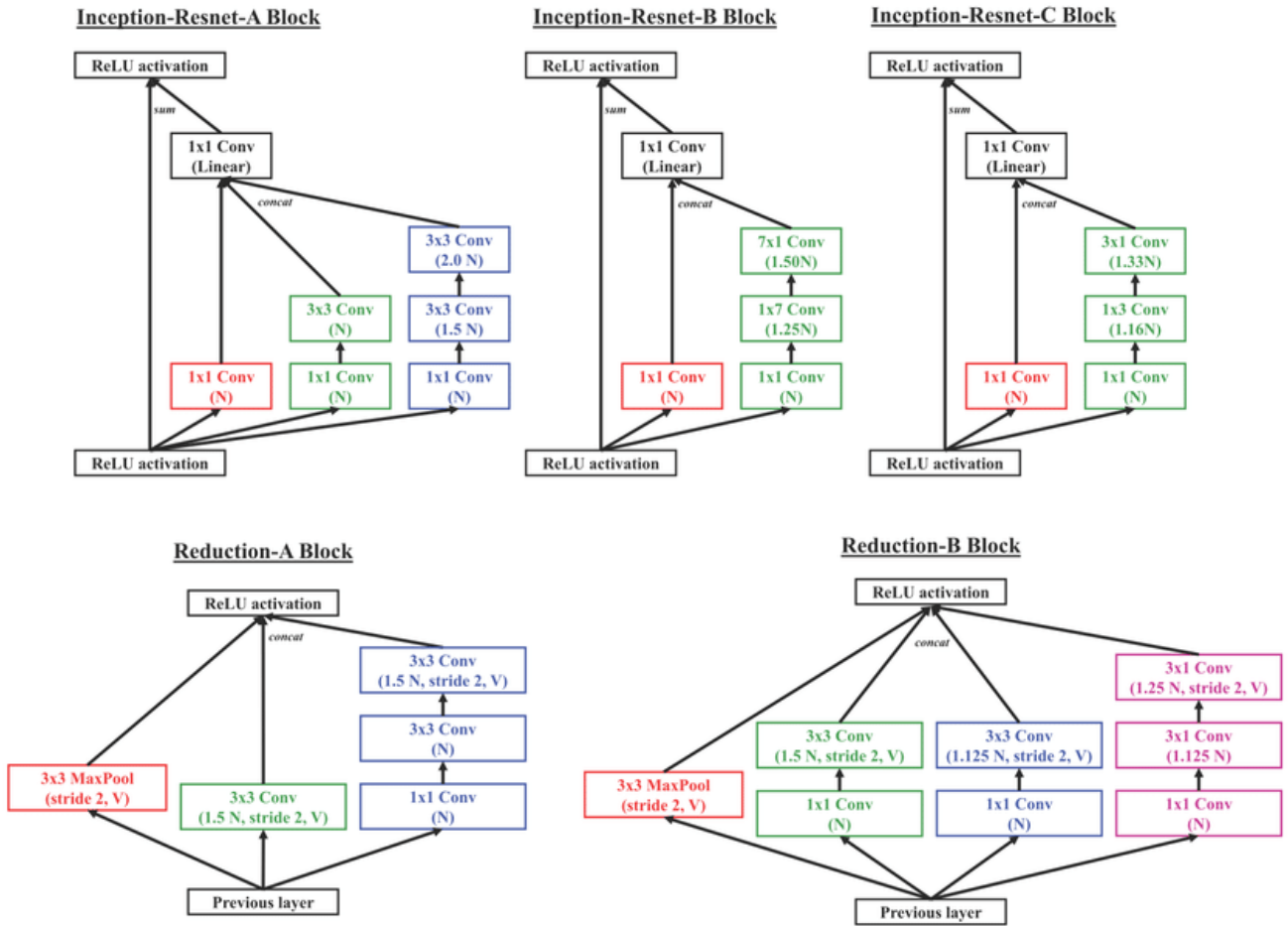


Figure 1.17 — Various configurations of Inception blocks [66].

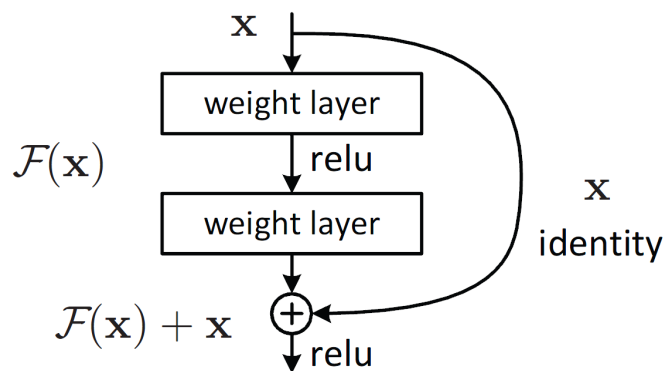


Figure 1.18 — Residual connections in ResNet [15] model.



This technique allows you to effectively deal with the attenuation of gradients, which, in turn, made it possible to build networks of greater depth than before the invention of this concepts. Representatives of this family of architectures are at the heart of many of the most effective architectures in solving various current computer vision problems [70–73].

The *MobileNets* [17; 43] family of neural networks has greatly advanced use of deep convolutional architectures on mobile devices. The authors of these architectures proposed the use of a special type convolution - *depthwise convolutions*(Fig. 1.19).

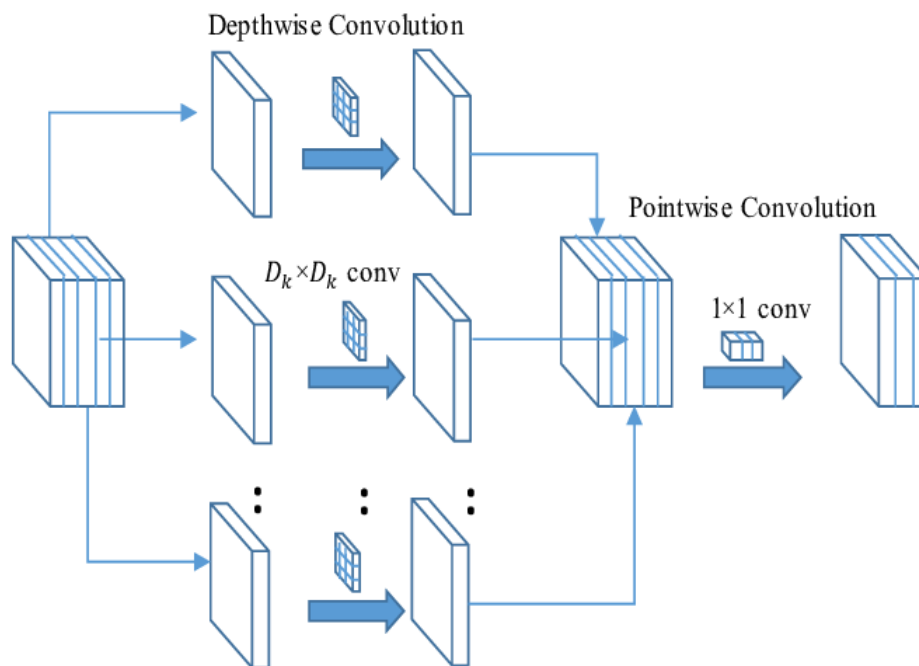


Figure 1.19 — *Depthwise convolutions* [74].

This solution is a kind of approximation of the heavier nucleus by means of the product of lighter ones, which makes it possible to reduce the number of weights in the neural network, while maintaining high quality problem solving. In addition, the authors of this approach have selected the optimal hyperparameters of the width and depth of the neural network, which made it possible to effectively build convolutional feature encoders.

Authors of the *EfficientNet* [18; 75] architecture family proposed an approach that currently demonstrates some of the best results on a set of general image classification problems [11]. The key feature of his work is to find the optimal ratio of hyperparameters that define the shape of tensors (see Fig. 1.20) when extracting

visual features. Thus, when building an encoder, it is possible to extract complex features while expanding the scoped context.

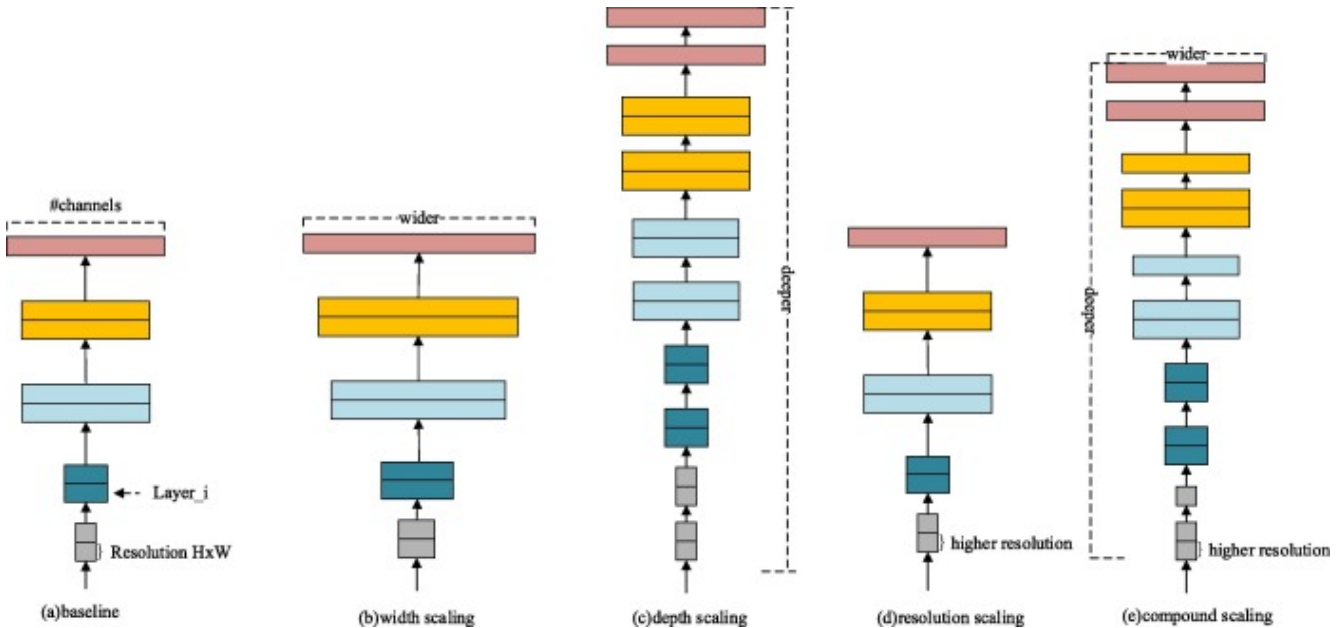


Figure 1.20 — Various tensor parameters in EfficientNet [76] models.

Deep neural network architectures of the *CoAtNet* [77] (Convolution and self-Attention Network) family are built on the idea of combining the attention mechanism and convolutions. Two options for constructing hybrid layers were proposed:

$$y_i^{post} = \sum_{j \in \varrho} \left( \frac{\exp(x_i^T x_j)}{\sum_{k \in \varrho} \exp(x_i^T x_k)} + w_{i-j} \right) x_j,$$

$$y_i^{pre} = \sum_{j \in \varrho} \left( \frac{\exp(x_i^T x_j) + w_{i-j}}{\sum_{k \in \varrho} \exp(x_i^T x_k) + w_{i-j}} \right) x_j,$$

where  $y_i^{pre}$  and  $y_i^{post}$  are the pre-normalization and post-normalization types of convolution and attention matching, respectively,  $x$  is an element of the input space,  $w$  are the training parameters,  $\varrho$  - area under consideration.

The pre-normalization type ( $y_i^{pre}$ ) showed the best performance in experiments conducted by the authors of the architecture [77]. Neural network classifiers built using this approach successfully combine the ability to highlight both global features, thanks to the use of the attention mechanism, and to derive local patterns over the entire image area, thanks to the convolution property.

The architectures of *Vision Transformer* [78; 79] and other variations of *Transformer* [80] adapted for solving computer vision problems are based on multi-headed

attention constructions:

$$MultiHead(Q,K,V) = [head_1, \dots, head_h]W_0,$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V),$$

where  $W_i$  denotes matrices of learning parameters, and  $head_i$  here are represented by *non-local blocks*, described earlier in the dissertation text. The general principle of the transformer operation is as follows: several blocks of attention ( $head_1, \dots, head_h$ ) are built independently, the results of which are combined and weighted by multiplying by the learning weight matrix  $W_0$ . Thus, the transformer block is an ensemble of non-local blocks that implement the self-attention mechanism in deep convolutional neural networks.

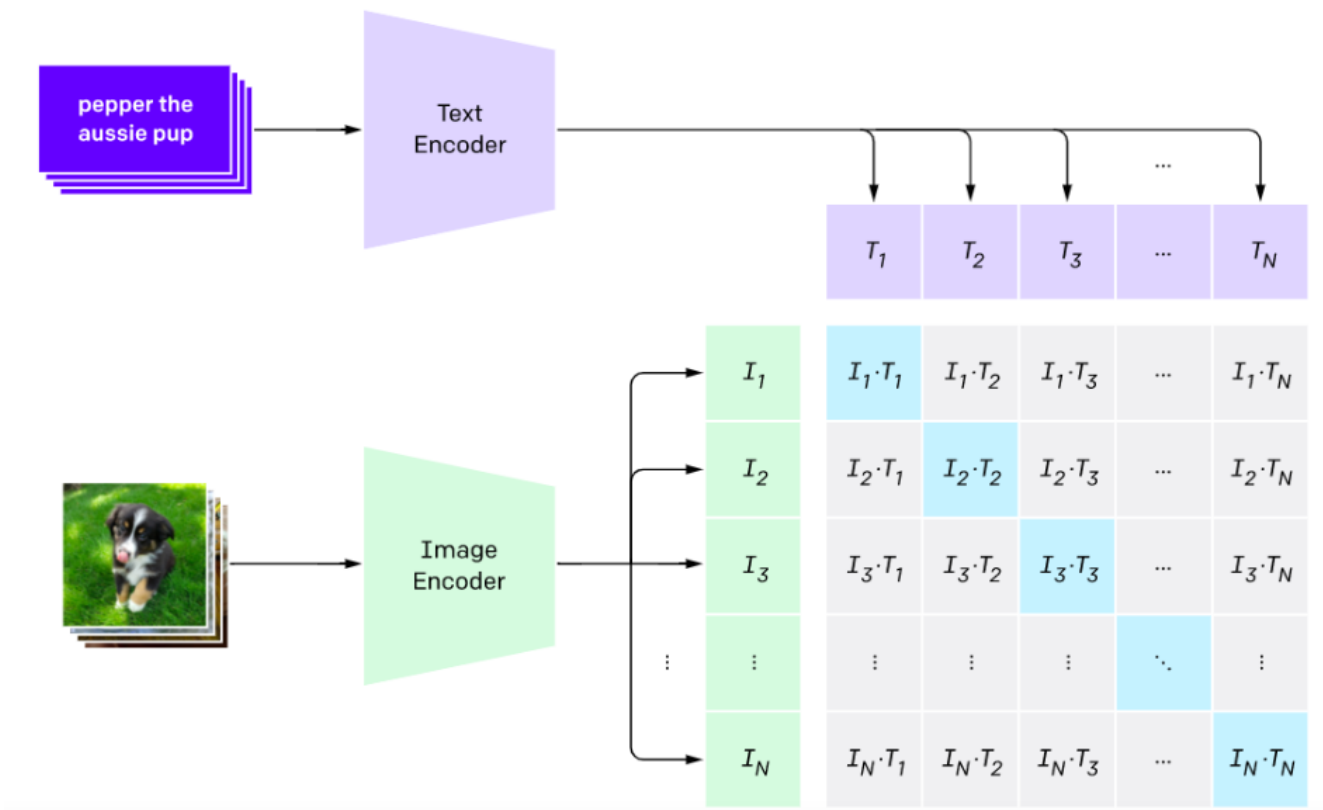
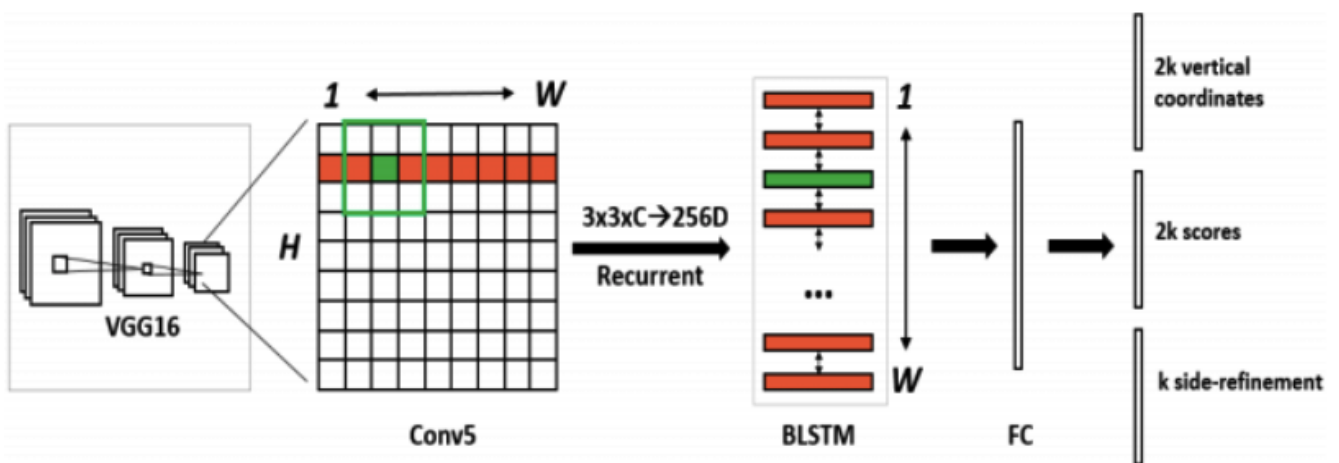
In works related to the use of the transformer architecture in computer vision, in addition to using the attention mechanism, various types of tokenization are used, designed specifically for processing image data. For example, there is tokenization based on segmentation, as well as on the basis of grid image division.

The authors of *CLIP* (Contrastive Language-Image Pre-Training) [81] proposed an approach that allows combining text and image modalities in a single vector space (see Fig. 1.21).

### 1.3 Text detection

One of the areas related to the subject of this work and in which deep learning technologies are actively used is text detection. This task is usually formulated as finding the coordinates of a figure with a minimum area covering a piece of text presented in an image. Note that there may be several such areas, in which case it is required to determine several figures, depending on the problem statement and the nature of the input data. Below we consider common approaches to solving the problem of text detection.

*CTPN* (Connectionist Text Proposal Network) [22] is a combined neural network architecture (see Fig. 1.22) that performs text detection.

Figure 1.21 — *CLIP architecture* [81].Figure 1.22 — *CTPN architecture* [82].

The operation of this detector can be divided into several stages. First the original image is skipped through a convolutional encoder, which is VGG16 [16]. Further a scanning window moves along the received feature map, transmitting considered area to the input of a bidirectional recurrent neural network. As a result, the obtained intermediate result is passed through fully connected layer, resulting in the formation of regions covering text.

Among the shortcomings of this architecture, one can single out a non-optimal operating time, the reason for this is the use of a bidirectional recurrent neural network in as one of the modules.

The architecture for text detection *EAST* (Efficient and Accurate Scene Text Detector) [23] is convolutional architecture without the use of recurrent modules (see Fig. 1.23). In this way, the problems of detector performance (presence of CTPN) are solved. In addition, CTPN contains a number of features related to the output data format. *EAST* detects the coordinates of enclosing quadrilaterals considering affine deformations to take into account the position of the surface with the text in space relative to the observer (see Fig. 1.24). To achieve this effect, *EAST* learns to combine affinity-transformed regions with desired area.

*CRAFT* (Character-Region Awareness For Text detection) [83] is a convolutional architecture without recurrent elements, performing text detection using region-level prediction symbols and predicting their mutual affine alignment using special additional type of regions - *affine regions* (Fig. 1.25).

Thanks to this technique, it becomes possible to predict boxes whose deformations exceed beyond the usual affine transformations of rectangles as a result of projective distortions (Fig. 1.26).

## 1.4 Optical text recognition

Currently, text localization methods show quite high efficiency [22; 23; 82–84], however, optical character recognition methods often do not show high efficiency, due to the variation of many factors, such as background, fonts, conditions image capture, etc. [2; 6; 29].

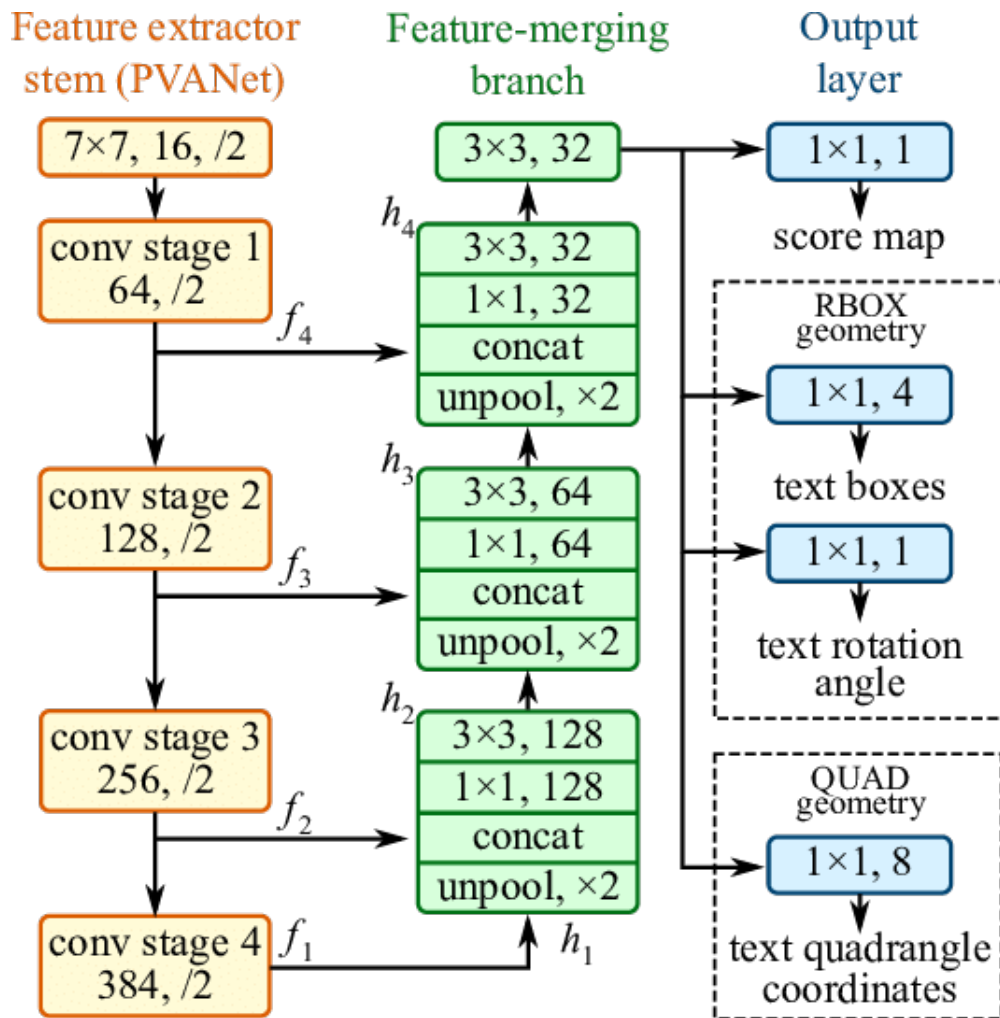


Figure 1.23 — EAST architecture [23].

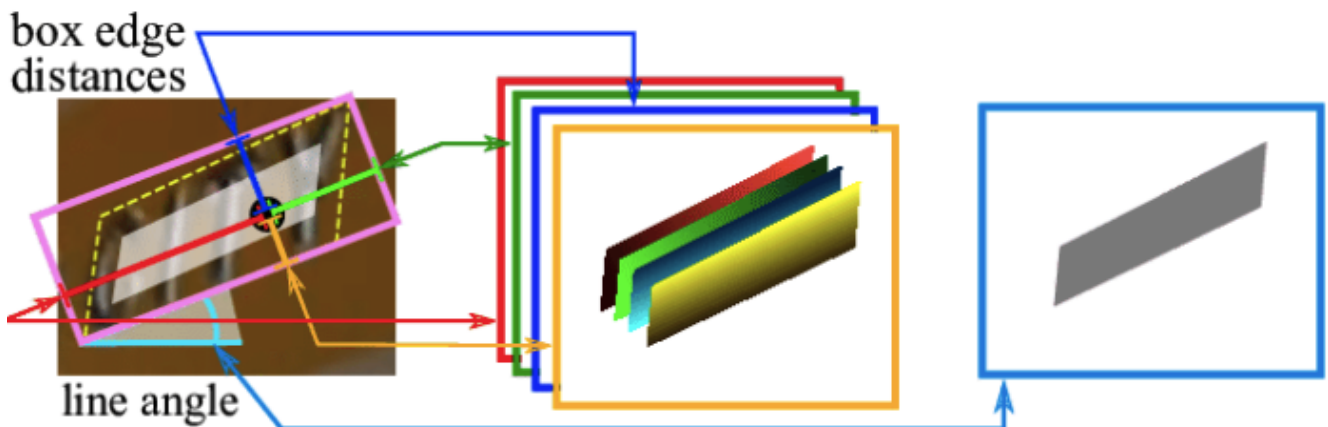


Figure 1.24 — Affine transformations of regions in EAST [23].

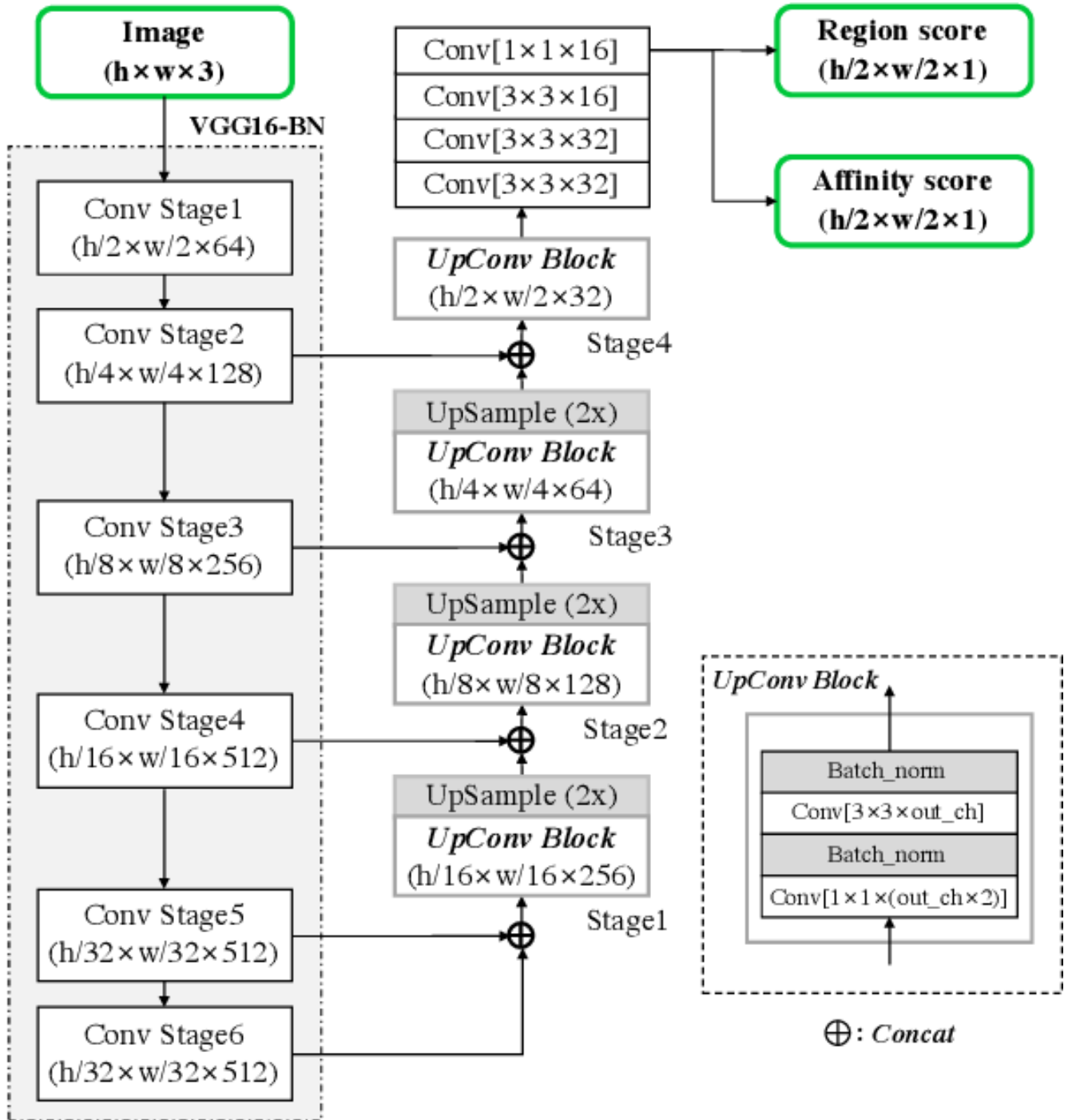


Figure 1.25 – CRAFT architecture [83].



Figure 1.26 — *Types of regions detected by the CRAFT detector [83].*

Let us briefly consider the *Tesseract* [85; 86] software package [85; 86], which is designed to solve the problem of optical text recognition and is often used as part of combined models. This solution demonstrates good performance in character recognition on document scans, however, it demonstrates a low efficiency in solving problems of optical recognition in difficult conditions (for example, on street advertising signs [29]). Moreover, in order to work effectively, *Tesseract* often requires an additional preprocessing step input image. Among the negative aspects, one can also note the low performance due to the use of LSTM [57].

At present, a framework for optical recognition is widely used *EasyOCR* [87]. This solution is a modular architecture using CRAFT [83] to a priori text position detection, as well as a convolutional recurrent neural network *CRNN* [84], which is a combination of a deep convolutional neural network and a recurrent neural network for recognition characters.

## 1.5 Combined neural network architectures

It should be noted that among the previously considered architectures, there are combined neural networks representing is a modular design containing blocks of various functional purpose. For example, when detecting text using the CRAFT [83]



and CTPN [22] methods, with the help of convolutional blocks, the visual cues while the data sequence is being processed using the recursive module. Moreover, when processing data from different domains, multi-arm models are often used, which are feature encoders, consisting of several threads that process heterogeneous data. Ensemble solutions are also widely used, such as MTCNN [88] (Multi-Task Cascaded Convolutional Networks).

Such a model building paradigm makes it possible to effectively solve complex problems, such as object detection, OCR processing (Optical Character Recognition) of multimodal content.

## 2. Combined neural network architecture for image recognition without explicit text representation

Currently, there are a significant number of approaches to document recognition, which are based on the extraction of semantic information from significant areas of the document image — photographs of citizens, key inscriptions, watermarks [1–4; 89]. However, the semantic analysis of image fragments, including text recognition, is a very time-consuming operation and is often necessary only in conditions of a large variability of valid input images. At the same time, there are highly specialized problems of document classification, in which much can be said by superficially analyzing image objects, as well as the general visual style.

In such tasks, often, a deep analysis of the document image is not required, but it is only necessary to assign it to one of the groups (for example, a certain group of identity documents), as well as to guarantee the fulfillment of certain conditions (in the example above, the document belongs to a certain person, and also the authenticity of the document). However, there are specific requirements for the quality of classification results - completeness, accuracy, etc. Finally, another feature of the task may be requirements for the quality of document images - for example, there may be variability in lighting when creating a photograph of a document. All these and other similar restrictions and requirements lead to the fact that the general methods of classifying documents turn out to be ineffective within the framework of this task, and it is required to create a special neural network architecture that is maximally adapted to solve this particular task.

In this chapter, we will consider a specific class of document classification problems, within which the following restrictions are imposed on the analyzed images.

- The document must contain the minimum text fragment that must be recognized — the last name, first name and patronymic of the user, as well as his photo.
- The mutual arrangement of elements in the image affects the classification.
- The image of documents is characterized by the presence of pronounced contours.
- Document image may contain visual defects, such as glare, out-of-focus areas.



Figure 2.1 — *Examples of images of an identity card suitable for user identification:*

- a) an image that does not contain any visible distortion;*
  - b) an image containing valid user add-ons that hide part of personal data;*
  - c) an image containing projective distortions resulting from varying shooting angles;*
  - d) an image containing artifacts that have arisen when varying lighting conditions (glare)*
- (all personal information is hidden with a blur).*

- Image angle, framing, lighting, and color balance are unknown.
- The document image may include areas containing artificial changes made by the user to hide some of the personal data, and this is correct.

Examples of such document images are shown in fig. 2.1. The classification problem can be formulated as follows. The image being checked must be assigned to one of the classes  $C_i$ ,  $i \in [0, N]$ , while  $C_0$  is a class of document images that are not correct, for example, the image is not a photograph of a document, proving the identity.

Finally, a feature of this class of problems is the requirement for high accuracy (precision) of the classification results and the fully automatic operation of the final services - that is, the option of issuing TopN results with subsequent «manual» analysis is excluded.

To solve this class of problems, within the framework of the dissertation research, a neural network architecture VCA (VGG Combined with Autoencoders) is proposed, which allows you to build the most expressive, in the context of this problem, descriptors of document images.

Based on the proposed VCA architecture, a model was created for recognizing identity documents, and the «VKontakte» social network service was developed to restore user access.

## **2.1 Architecture, principles of construction and operation of the combined neural network model VCA**

The VCA architecture takes a document as input and, with the help of an encoder, turns it into a set of features (feature vector), which is then passed to the classifier, and the latter assigns it to one of the predefined classes, thus classifying the document. The architecture includes two parallel branches — a general-purpose encoder based on VGG19 and a combined encoder based on VGG16 with a preliminary use of an autoencoder. The latter is used to work with highly noisy images with a large feature variance, while simultaneously building a contour map of the processed image and then passing this map to an encoder (created based on VGG16), which, in turn, extracts image features from it Image features extracted

by both encoders, are combined into one vector, which, in turn, is fed to the image classifier, and the latter assigns the original image of the document to one of the classes. The general architecture of the VCA is shown in fig. 2.2.

The first branch of VCA consists of a modified VGG19 [16] model reduced to the penultimate subsampling layer (fig. 2.3, column E) and is a weak encoder that allows stable (with respect to noise, angle and lighting conditions) to determine if the provided image is an ID photo. However, as experiments have shown, when using such an encoder, a problem arises in differentiating different types of documents, and this solution does not show proper efficiency when separating photographs of one type of image from another.

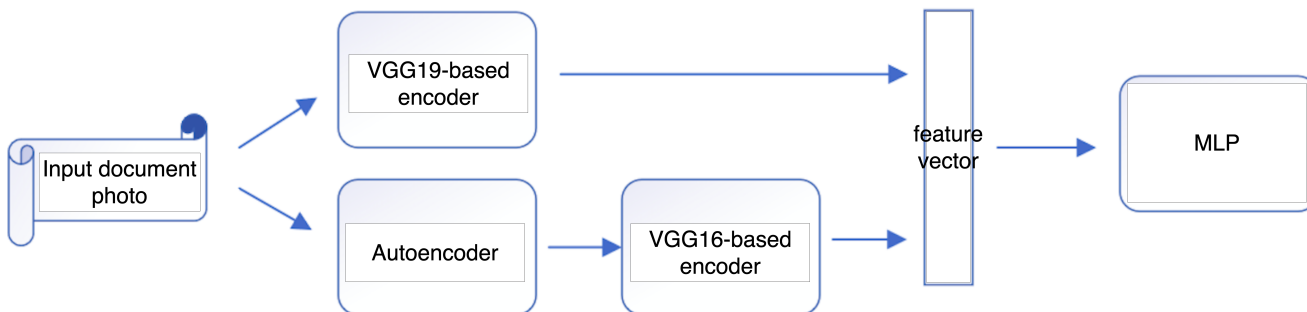


Figure 2.2 — *VCA architecture.*

To increase the ability of the VCA architecture to stably distinguish between different types of images, an additional branch has been added to it, which contains a modified VGG16 encoder (Fig. 2.3, column D), which processes the result of applying the autoencoder [90] to the analyzed image. An auto-encoder is integrated into the architecture to extract information about contours in the presence of noise and variability in lighting factors. VGG16 is reduced here to the penultimate subsampling layer (similar to VGG19 in another branch, fig. 2.3, column E). It should be noted that the combination of auto-encoder and VGG does not allow good recognition of suitable document images under a wide variation in shooting conditions. As a result, when combining the above branches by adding an upper fully connected layer with sigmoidal activation, an architecture is obtained that allows you to verify and differentiate images of various types well.

## 2.2 Encoders for extracting classifying features

An essential step in image classification is feature extraction. The currently most efficient methods for extracting features from images are based on convolutional neural networks, which have significantly outperformed all other approaches on open datasets [15; 16; 43; 65].

The first version of the VCA architecture used well-known general-purpose encoders that performed well on the well-known ImageNet dataset (see table 1). We are talking, first of all, about VGG encoders (not adapted), as well as Inception, ResNet, MobileNet.

The encoder based on Inception [65] proposes to use special blocks that combine convolutional and fully connected layers for the most efficient extraction of key features from the original information and its intermediate representations (see section 1.2).

The ResNet [15] structure assumes the use of residual links, which allows building networks of greater depth (see section 1.2).

MobileNet [43] propose the use of special base block convolution structures that make it possible to make neural network models usable without significant loss in efficiency (see section 1.2).

However, when testing these encoders on a test set of photographs of documents «VKontakte», it was found that the best results on sets of common images do not guarantee a higher accuracy of classification of document images. In particular, the propensity of models was established [15; 16; 43; 65] to false positives. This can be interpreted as a consequence of the specificity of the studied data.

In this regard, it was decided to stop using modified versions of the VGG.

The VGG [16] model is a sequence of convolutional, fully connected, and subsampling [59] layers. In this paper, variants VGG16 and VGG19 are considered, which differ from each other in the number of layers - 16 and 19, respectively (Fig. 2.3 column D and 2.3 column E).

Table 1 — The results of the work of models on ImageNet test set of photos of the general plan

Model	Top-1 Accuracy	Top-5 Accuracy
ResNet50	0.749	0.921
MobileNet	0.704	0.895
VGG16	0.713	0.901
VGG19	0.713	0.900
InceptionV3	0.779	0.937

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input ( $224 \times 224$ RGB image)					
conv3-64	conv3-64 <b>LRN</b>	conv3-64 <b>conv3-64</b>	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 <b>conv3-128</b>	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 <b>conv1-256</b>	conv3-256 conv3-256 <b>conv3-256</b>	conv3-256 conv3-256 conv3-256 <b>conv3-256</b>
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 <b>conv1-512</b>	conv3-512 conv3-512 <b>conv3-512</b>	conv3-512 conv3-512 conv3-512 <b>conv3-512</b>
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 <b>conv1-512</b>	conv3-512 conv3-512 <b>conv3-512</b>	conv3-512 conv3-512 conv3-512 <b>conv3-512</b>
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 2.3 — The original architecture of VGG [16] and its components..

### 2.3 Autoencoder for highlighting the characteristic features of contours

Let's take a closer look at the VCA branch containing the autoencoder. The key ability of the autoencoder is to get a compact representation information contained in the input data. Typically, autoencoders learn restore the original tensor from its hidden representation, however, in VCA this model is used for the stable extraction of additional information about the contours in the original image, which makes it possible to extract information about the relative position of the elements (user photos and areas with inscriptions and signs containing personal information). To achieve this effect, the autoencoder learns to restore from the original image representation obtained from the original using a set of filters, whose task is to extract from the original image the most detailed and free from noise outline drawing.

To achieve this effect, we used filters based on Sobel [91] and Gauss [92] filters. The Gaussian filter allows you to get rid of high-frequency noise by smoothing original image. Subsequent application of Sobel operators with different size kernel, allows you to highlight information about the contours of objects presented on analyzed image. After applying the Sobel operators, the result filtering, an adaptive threshold transformation is applied that takes into account the local the value of the average brightness of the image (to ensure invariance when lighting changes).

It should be noted that, in the general case, the optimal parameters (kernel sizes for Gaussian and Sobel operators, as well as the dimensions of the neighborhood and the activation threshold for threshold transformation) does not exist. Therefore, when creating a training dataset for the autoencoder, we used several options for highlighting the characteristic features of contours with different parameters, that is, each input image is associated with a set  $P$  tensors of the following form:

$$P = \{t_i \circ h_j \circ v_k \circ g_l(I) | t_i \in T, h_j \in H, v_k \in V, g_l \in G\},$$

where  $I$  — original ID image,  $G$  — set of Gaussian smoothing operators with different sizes of kernels,  $H$  is a set of horizontal Sobel operators with different sizes kernels,  $V$  is a set of vertical Sobel operators with different sizes of kernels,  $T$  is a set of binary threshold transformations with different threshold values.



Next, for each of the original image, the best variant is selected from the corresponding set of tensors, according to the contour characteristic, which is the variance of  $L_2$ -norms of domains from the central part of the photo (to avoid the influence of the contour characteristics of the edges of the document page). Thus, each input image is associated with an output tensor  $b$  that satisfies condition:

$$b = \operatorname{argmin}_{p \in \mathcal{P}(I)} \operatorname{Var}[n(s(c(p)))],$$

where  $I$  is the original ID image,  $P$  — set of results of processing the original image using filters and thresholding (as shown above),  $c$  is the mapping that maps image  $p$  to its frameless version (cropped 1/5 in height and 1/5 in width from original image dimensions),  $s$  — display, which splits the original into a set of non-intersecting windows (width and height of the window — 1/10 from the size of the argument - 100 windows in total, respectively),  $n$  is a mapping that takes a set of tensors as input and returns a set of numbers, each of which corresponds to the  $L_2$ -norm of one of the elements of the input set, that is,  $n(S) = \{\|s_i\| \mid s_i \in S\}$ .

## 2.4 Consistency regularization adaptation

In order to train the VCA architecture, consistency regularization, which is a well-known method of teaching with partial involvement of the teacher, was adapted to improve the color gamut of an image. This approach does not affect the size or running time of the model and only slightly complicates the learning process.

The VCA must produce the same results for similar images, which means that the output of the generator must be invariant under any slight changes in the images. For this reason, consistency regularization was used during training. In terms of the notation from the beginning of this section, the  $L_{cr}$  consistency regularization is defined as follows:

$$L_{cr} = \lambda \sum_{i=1}^n \|h_i(I_{so}) - h_i(T(I_{so}))\|^2,$$

where  $\lambda$  — a weighting factor,  $T(\cdot)$  — a random image minor modification function, and  $\|\cdot\|$  denotes the  $L^2$  norm of the given vector.

In this work, a combination of randomly shifting the image by a few pixels, and then randomly flipping the image horizontally and vertically as a minor modification of the image, due to the context of the problem, was used. First of all,  $L_1$  and  $L_{SSIM}$  were used as loss functions, which are defined as follows:

$$L_1 = \|I_e - I_{gt}\|_1,$$

$$L_{SSIM} = 1 - SSIM(I_e, I_{gt}),$$

where  $\|\cdot\|_1$  — is the  $L^1$ -norm of the given vector, and SSIM is the value of the structural similarity index [93],  $I_e$ , and  $I_{gt}$  stands for enhanced image and target image, respectively.

So the best VCA configuration was trained with a combination of  $L_1$ ,  $L_{SSIM}$  and  $L_{cr}$ :

$$L = L_1 + L_{SSIM} + L_{cr}.$$

## 2.5 Implementation of VCA for solving the problem of document classification

The proposed VCA architecture was implemented as a model for recognizing identity documents as part of the service for restoring access to the social network account «VKontakte». In this case, a number of restrictions are imposed on the image of the document, non-compliance with which entails an unambiguous classification of the image as  $C_0$ , i.e. as unsuitable for user identification. These restrictions are presented below.

- An area of the ID containing the user’s photo as well as the user’s name and surname, should not contain artificial changes.
- Glare and noise that overlap the area with the photo, as well as the name and surname, should not interfere with visual identification when the image is recognized by a person (by an employee of the access recovery department of the «VKontakte» social network).

- Eligible for the identity verification process for citizens of the Russian Federation include photographs of documents from list<sup>1</sup>.
- For citizens of other states for which the automated verification procedure is supported, a passport photo is required.

The VCA-based model is used in the final service for restoring access to the «VKontakte» network to implement a filter that operates in a fully automatic mode and rejects unsuitable applications. There are a number of criteria for rejecting an application, but the criteria related to the analysis of the ID card image are verified using a model implemented based on the VCA architecture.

To implement the process of model training and data preparation, the Python programming language, specialized computer vision libraries OpenCV [94], Dlib [95] and PIL [96] were used. The Tensorflow [97] deep learning framework was used to build and train classifier models and data preprocessing. The model was trained on a cluster equipped with Nvidia Tesla t4 16 GB graphics cards.

## 2.6 Experiments

This section presents an experimental study of the proposed VCA neural network architecture.

**Datasets** The model that implements the VCA architecture was trained and tested on the data of the «VKontakte» social network arranged as follows.

Based on many photographs of documents of employees of the «VKontakte» company, a basic set was created. Further, on the basis of this set, additional images were obtained using two different procedures. These procedures simulated two situations that are often encountered in practice: the presence of additional, user-defined changes to photographs and complex conditions (angles) for shooting documents.

The first case is important, since it is on the borderline of fraud (creation by the user of a fictitious document with the aim of hacking the «VKontakte» social network) and the admissible overwriting of personal data by the user in the photo of the document, which are not needed for identification (as mentioned above, the

---

<sup>1</sup><http://www.consultant.ru/cons/cgi/online.cgi?req=doc;base=LAW;n=149244#008807190564687595>

latter requires only Full name and photo of the user, but the number and series of the document, place of issue, date of birth of the user and other information is not necessary). To obtain such photographs, the «manual» work of experts was used, who made acceptable and unacceptable user additions, as well as retouching photos of identity cards. To simulate unacceptable user changes (photo location area and full name), automated procedures were used that perform image gluing, retouching, inserting sections of other images and artificially generated text fragments into the original identity card images. It was also allowed to overlap different areas of the image with third-party background objects.

The second procedure consisted in augmenting some of the photos from the base set by adding artificial elements that imitate artifacts that occur when shooting in different lighting conditions. Affine transformations were also used here to expand the number of supported shooting angles.

It should be noted that the expansion of the data set by adding distortion and other additional artifacts, in addition to the above, was not applied, since the image samples containing strong distortions of this type do not fit the initial requirements for the task, and the presence of weak distortions did not significantly affect the results of the model.

Ultimately, the following three sets of data were received for training and testing:

- *mixed* data set, which includes both the original photos of the base set and photos with additions and changes in angle and shooting conditions; photographs obtained using the two procedures described above were added to the base set in proportion to make the data set relevant to real data;
- data set *with additions* consisted only of images obtained using the procedure described above, emulating user changes;
- the *complex angles* dataset was also created, which included the photos obtained using the second procedure.

Despite the fact that the mixed sample is as close as possible to the data entering the system, which is why it is a priority for analysis, additional samples were selected to study the effectiveness of models in conditions of complex angles and user additions. The fact is that these circumstances are the most common scenarios that make it difficult to recognize documents. But it should be noted that in a mixed data set, the number of elements corresponding to complex conditions is

not so large as to unequivocally state the quality of the model when the angle varies and the presence of additions. Moreover, the study of such cases separately allows you to correct and fine-tune the models to work in difficult conditions.

The total number of samples in the total training dataset was 18,000 and was evenly distributed across 12 classes. The test set had the following structure: *mixed* data set — 6000 images, *with additions* set — 1000 images, *complex angles* set — 1000 images.

**Quality metrics** The main metric by which the quality of the model based on the VCA architecture was evaluated is *Precision* (precision,  $P$ ). The service for restoring access to the account of the social network «VKontakte» should not accept invalid documents (but it may sometimes not recognize some correct documents). *Recall* (fullness,  $R$ ) and  $F1$  were also measured. All these metrics were calculated in a standard way [98]. Note that  $P$ ,  $R$ , and  $F1$  are formulated in terms of a binary classification, i.e., for one class. To assess the classification quality for several classes, the metric values for them are averaged over the results obtained for each class separately. It should also be noted that the final values of the metrics are calculated as an average over several iterations with negligible variance.

Depending on the value of the activation threshold, the values of the metrics  $P$ ,  $R$ , and  $F1$ -score for each class will differ. As a result, the problem of choosing a criterion for setting the threshold arises. In this case, there is the following simple criterion for calculating this threshold. Since the most undesirable error is accepting an image that is unsuitable for account recovery as acceptable, the threshold value was chosen according to the maximization of the  $R$  value for the class of user ID photos that are unsuitable for account recovery. However, it should be noted that under the conditions of integration of the model with the final software service, there is a natural restriction on the minimum value  $P = 0.98$  for the class of unsuitable photos due to the maximum allowable load on the operators of the final document verification service — they check «manually» documents deemed unsuitable. As a result, the threshold for the exit responsible for declaring the document unusable was chosen according to the  $R$  maximization principle, with the minimum allowable  $P = 0.98$ , which was achieved with the threshold value  $threshold = 0.956$  (Fig. ??). Note that for VCA outputs corresponding to certain types of ID cards, the threshold was chosen according to the maximization of  $P$  in order to reduce the proportion of false positives.

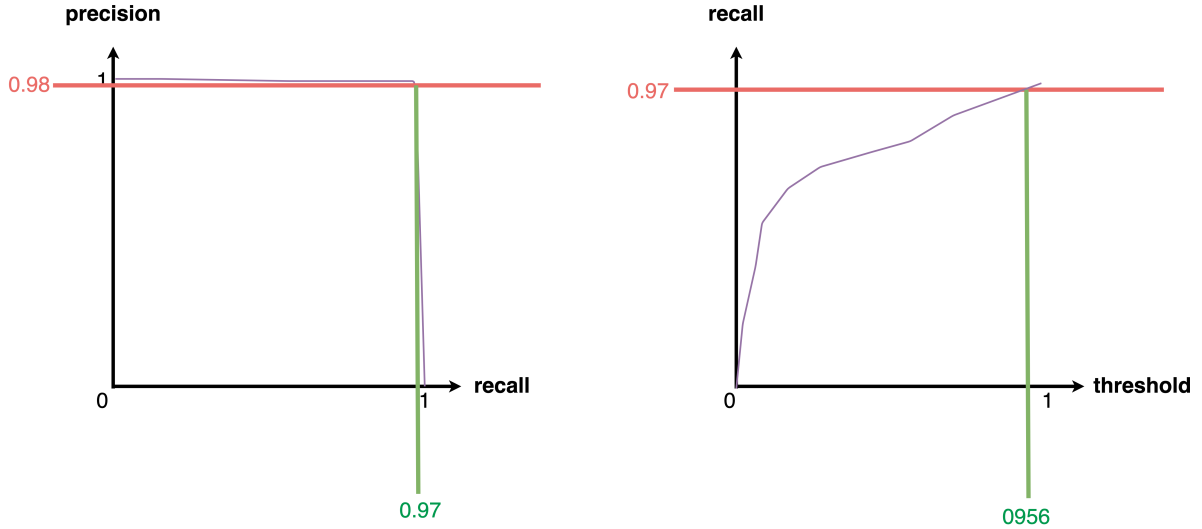


Figure 2.4 — *Selection of the activation threshold for the classifier under conditions of restrictions on Precision.*

Having thus set the threshold value that generates the maximum possible value of *recall* for the class of unsuitable photos, determines the quality metrics presented in the table 2.

**Analogues for comparison** For a comparative analysis of the quality of VCA work, neural network classifiers from the following groups were studied.

- The first group is represented by convolutional classifiers of the families VGG [16], Inception [65], ResNet [15] and MobileNet [43], which was discussed in detail above.
- The second group of approaches is represented by families of neural network classifiers using self-attention mechanisms — CoAttNet [77], ViT [80] (see section 1.2).
- The third group of studied architectures is based on the idea of combining textual and visual descriptors/embeddings within a single feature space (see section 1.2) and is represented by architectures of the CLIP [81], BLIP [**clip2**; 44]. The pretrains for these classifiers were obtained on the open COCO [99] and Flickr30k [100] datasets.
- Also, as analogues for VCA, combined methods were studied, which are a combination of an autoencoder (see section 2.3), the result of which is processed using the above neural network add-ons (AE+VGG16, AE+VGG19, AE+ MobileNet, AE+ResNet, AE+Inception).

Table 2 — Experimental results

Models	Results on datasets								
	Mixed			With additions			Difficult angles		
	$P$	$R$	$F_1$	$P$	$R$	$F_1$	$P$	$R$	$F_1$
VGG16	0.863	0.914	0.888	0.626	0.587	0.606	0.674	0.622	0.647
VGG19	0.871	0.912	0.891	0.631	0.576	0.602	0.672	0.638	0.655
MobileNet	0.897	0.931	0.914	0.678	0.593	0.633	0.691	0.683	0.687
ResNet	0.901	0.933	0.917	0.676	0.624	0.649	0.712	0.698	0.705
Inception	0.913	0.944	0.928	0.699	0.641	0.669	0.722	0.702	0.712
AE+VGG16	0.928	0.810	0.865	0.814	0.733	0.771	0.839	0.708	0.768
AE+VGG19	0.930	0.808	0.865	0.801	0.731	0.764	0.843	0.711	0.771
AE+MobileNet	0.951	0.896	0.922	0.927	0.812	0.866	0.867	0.787	0.825
AE+ResNet	0.962	0.883	0.920	0.920	0.818	0.866	0.865	0.772	0.816
AE+Inception	0.963	0.923	0.943	0.924	0.822	0.870	0.894	0.815	0.853
ViT-B_32(+MLP)	0.970	0.912	0.940	0.963	0.801	0.875	<b>0.965</b>	<b>0.902</b>	<b>0.932</b>
CLIP-ResNet50+MLP	0.989	<b>0.978</b>	<b>0.983</b>	<b>0.970</b>	0.821	0.889	0.942	0.897	0.919
BLIP+MLP	0.993	0.962	0.977	0.969	0.867	0.915	0.935	0.900	0.917
VCA	<b>0.996</b>	0.971	<b>0.983</b>	0.951	<b>0.894</b>	<b>0.922</b>	0.941	0.871	0.905

**Analysis of results.** The results of calculating the quality metrics of the implemented models and analogs that satisfy the physical constraints of the cluster are presented in Table 2.

Despite the fact that a number of modern image classification architectures showed results comparable to VCA, and even surpassed VCA in some cases (for example, the ViT-B\_32(+MLP) architecture turned out to be the best in samples with complex angles), on a mixed on a data set that is as close as possible to real data entering the account access recovery system, VCA demonstrates the best results in terms of the *Precision* metric (as mentioned above, this metric is the main one for the task under consideration).

The reason for the effectiveness of the ViT-B\_32(+MLP) architecture is its ability to focus well on key image fragments, while difficult photography angles greatly hinder this for other architectures.

It should also be noted the efficiency of the CLIP-ResNet50+MLP classifier, which outperformed VCA in terms of the  $R$  metric value on the mixed dataset, as well as in the  $P$  metric value for the dataset with additions. Such high performance for this architecture can be explained by the following circumstances. Pretrains of the CLIP architecture generally perform well on a wide class of image classification problems due to a special modality matching training procedure that is trained on a very large and well-balanced dataset. Moreover, the model based on CLIP-ResNet50+MLP includes an implementation of the concept of attention (though not

as powerful as in ViT-B\_32(+MLP)), which also affects its high efficiency. However, despite all the advantages of the CLIP-ResNet50+MLP architecture, it was originally designed for general-purpose images, therefore it loses in  $P$  on a mixed sample of the VCA architecture. Good performance on the  $R$  metric for CLIP-ResNet50+MLP on a mixed sample also does not indicate superiority over VCA, since in this case it demonstrates a lower  $P$  value, i.e. skips inappropriate photos, which critically affects the operation of the entire service.

Note that VCA, in addition to achieving the best result for the main criterion (the value of  $P$  on the *mixed* dataset), is always in the top four solutions when working on the datasets *with additions* and *complex angles* according to any considered metrics (see Table 2), which indicates the stability of the proposed solution when working in difficult conditions.

## 2.7 Chapter Conclusion

This chapter presents a VCA neural network architecture for classifying documents with specific constraints. The architecture was implemented as a model integrated into the social network service «VKontakte», which restores access to the user’s account based on the provided identity document. The VCA-based model showed an accuracy of 9.6% on the main (*mixed*) dataset. Despite the fact that the result of the nearest predecessor was 99.3%, increasing the accuracy by even a small percentage in this situation is very important, as it reduces the likelihood of accepting incorrect documents (this is important from a security point of view). In addition, in this case, the *Error Reduction* [101] value for  $P$  between the VCA result and the second best result is  $\tilde{43}\%$ , which confirms the significant superiority of VCA over analogues (the Error Reduction metric denotes a measure of uncertainty removal and is useful for small increments of various measurements around the 100% mark).

It should be noted that the VCA architecture, which is capable of extracting a document image descriptor that is stable relative to the stated conditions for obtaining photographs, made it possible to successfully solve the problem.



In the future, it is planned to study the possibility of using the designed architecture for classifying images of documents of various kinds, as well as handwritten texts, in particular, identity cards, the fields of which are filled in by hand.

### 3. Combined neural network architecture for image recognition with explicit use of textual information

This chapter presents the combined neural network architecture CCIT (Combined Classifier of Images with Text) created in the framework of this dissertation research, which is focused on classifying images using explicit text information processing. CCIT is designed to recognize images that have textual information with unique fonts, background and caption colors, sizes, and text styles. Various conditions for shooting image objects were also taken into account. The decision on whether an image belongs to one category or another was made both on the basis of information obtained from the text and on the basis of visual features. Another important feature of the considered images, significantly complicating their classification is the absence of convex elements.

Approaches based on text recognition [85; 86] show good results in recognizing images obtained by scanning documents. However, factors such as the possible lack of information necessary for classification in the text of an image and the complexity of the task of optical character recognition under changing shooting conditions make methods based only on text recognition ineffective in the context of the problem under consideration.

The first version of CCIT [6] took into account the information obtained from the text, as well as the general visual features of the image, but suffered from poor character recognition quality. Therefore, to improve the efficiency of classification, the current version of CCIT performs a visual analysis of the text on the image [12; 13]: visual features of the background, visual features of plates with text, semantic information obtained from the text. In addition to the above, CCIT has additional descriptors [12; 13] calculated from the neighborhoods of image text elements. These descriptors contain implicit information about common label styles and other visual characteristics.

At the moment, many approaches have demonstrated high efficiency in solving general image classification problems [11]. However, the classification of general images differs significantly from the problem with the above restrictions [6; 12–14]. At the same time, such problems are encountered in practice and require the development of specialized approaches to solve them.

So, the CCIT architecture performs a complex analysis of various features of the original image. To do this, it is organized as an ensemble of modules, implying a clear functional division. Each architecture module has its own purpose: extraction of general features, selection of an area with text, extraction of characteristic features of an area with text, analysis of all extracted features and making a final decision. This approach within CCIT is implemented in several ways.

The CCIT architecture was implemented to solve the problem of classifying photographs of commercial building facades with advertising signs [6; 10; 13].

### 3.1 Statement of the problem

Source data — classified images — have the following properties and limitations:

- all images were taken under different conditions, from different cameras that have undergone different calibrations (internal calibration parameters of cameras may vary and are not known to the classification system);
- input images may contain various visual defects, such as glare, glare, noise of various nature, including those that hinder optical recognition of text characters presented in images;
- shooting angles, positioning and alignment of objects in the frame, lighting, white balance and other imaging parameters are unknown and may vary significantly between different samples.

Accordingly, the task of classifying images with the above properties can be formalized as follows: the original image  $Q$  must be assigned to one of the following predefined classes  $C_i$ , where  $i \in [0..N]$ . These classes are parameters of the model that implements the CCIT architecture.

### 3.2 Image pre-processing.

In order to maximally level the influence of artifacts that have arisen due to difficult shooting conditions, the stage of image pre-processing, represented by a combined neural network architecture for color gamut correction, was added to the classification stack [1]. Such a neural network consists of several independent blocks, the number of which depends on the number of filters used. In each  $i$ -th block, a reduced version of the original image  $I_{so}$  passes through the  $h_i$  parameter generator, which matches the  $p_i$  parameters to the corresponding  $f_i$  filter. After applying the filters to the original  $I_o$  image, the final enhanced version of the  $I_e$  image is the sum of the original images and the output results after applying the custom filters. In general terms, the color gamut correction module looks like this:

$$I_e = I_o + \sum_{i=1}^n f_i(I_o, h_i(I_{so})),$$

where  $n$  — the number of selected filters.

Let's take a closer look at the structure of the parameterized filters that were used in the color gamut correction module.

As the first filter, automatic correction *color saturation* was used, which is implemented with using the following formula:

$$\Delta[x,y] = \begin{cases} (m - I_{in}[x,y]) \cdot (1 - \frac{1}{1-p}), & \text{if } p > 0 \\ -(m - I_{in}[x,y]) \cdot p, & \text{otherwise.} \end{cases}$$

$$I_{out}[x,y] = I_{in}[x,y] + \Delta[x,y],$$

where  $p \in [-1,1]$  is a trainable parameter characterizing the transformation applied to each pixel of the input image, and  $m$  is the average value of the input image pixels across channels.

In addition to automatic color saturation correction, automatic *contrast* correction was used, which is set by the transformation of the original image presented below:

$$I_{out}[x,y] = \begin{cases} (I_{in}[x,y] - 0.5) \cdot \frac{1}{1-r}, & \text{if } r > 0 \\ (I_{in}[x,y] - 0.5) \cdot (1 - r), & \text{otherwise.} \end{cases}$$

Here,  $p \in [-1,1]$ , is the only trainable parameter that specifies the transformation to apply to each pixel of the input image.

*White balance* auto-correction is implemented by multiplying the value of the color channel of each pixel of the input image by the training parameter in the range  $[0, 1]$ :

$$I_{out}^c[x, y] = I_{in}[x, y] \cdot s_c.$$

For this transformation, three learning parameters were used for the red, green, and blue channels, respectively.

The automatic *exposure* correction was emulated as follows:

$$I_{out}[x, y] = I_{in}[x, y] \cdot 2^t.$$

Convolution-based filters with a special kernel were also used. The most notable results of these types of transformations have been obtained using a combined kernel with a mask and a fully trainable convolution. For brevity, we call this type of kernel a universal kernel. By varying the mask, it is possible to modify the transformation based on the universal kernel. An example of this is the use of a special kernel that adds sharpness to the original image.

The classic representation of the *universal kernel* convolutional filter looks like this:

$$I_{out} = I_{in} \circledast \frac{1}{\nu} P,$$

where  $P \in \mathbb{R}^{3 \times 3}$  is the fully trained filter kernel matrix and  $\nu$  is the sum of  $P$  elements to normalize the matrix.

The *sharpness filter* is defined using the helper formula:

$$I_{out} = I_{in} \circledast \frac{1}{\nu} (K + M \cdot q),$$

where  $K$  is the filter kernel matrix,  $M$  is the mapping matrix of the same form, as  $K$  and  $\nu$  is the sum of the elements of  $(K + M \cdot q)$  to normalize the kernel matrix. The above formula applies to the red, green and blue channels independently with

their own training parameter. Thus, the sharpness modification is determined by the following parameters:

$$K = \begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & -476 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix},$$

$$M = \begin{pmatrix} 0.8 & 0.8 & 0.8 & 0.8 & 0.8 \\ 0.8 & 0.9 & 0.9 & 0.9 & 0.8 \\ 0.8 & 0.9 & 1 & 0.9 & 0.8 \\ 0.8 & 0.9 & 0.9 & 0.9 & 0.8 \\ 0.8 & 0.8 & 0.8 & 0.8 & 0.8 \end{pmatrix}.$$

The trained *linear transform* image was also used as one of the filters. This transformation can be described by the following expression:

$$I_{out}[x,y] = P \cdot I_{in}[x,y] + b,$$

where  $P \in \mathbb{R}^{3 \times 3}$  denotes the affine transformation matrix to be trained,  $b \in \mathbb{R}^3$  is the trainable vector in RGB color space.

A mapping was also used describing *per-channel color transform*. This transformation consists of a trio of functions that are applied to the red, green, and blue color channels, respectively. Each function is a linear combination of elements  $f_1, f_2, \dots, f_n$  n-dimensional basis, the coefficients for which are calculated from the output of the neural network. Therefore, the channel value for each pixel of the input image is calculated by the following formula:

$$I_{out}^c[x,y] = I_{in}^c[x,y] + \sum_{i=1}^n u_{ic} \cdot f_i(I_{in}^c[x,y]),$$

where  $f_1, f_2, \dots, f_n$  is the functional basis mentioned above,  $u_c$  is the learning parameters (one parameter for each channel).

In this case, only a piecewise basis with the following set of functions was used:

$$f_i(x) = \max(0, 1 - |(n - 1) \cdot x - i + 1|), i \in \{1, 2, \dots, n\},$$

where  $x$  is the value of the current pixel in the input image.

### 3.3 CCIT architecture with OCR module

**Basic architecture.** Consider the basic version of the combined neural network classifier (Fig. 3.1). This model consists of two encoders. The visual feature encoder for the entire image is represented by a convolutional neural network [17; 43; 59; 102]. The text information encoder consists of [6] blocks sequentially applied to the input image. The first block is a convolutional text detector EAST [23]. Further, all the detected text passes through the optical recognition module, which is based on Tesseract [85]. Then the resulting text is transformed into a vector representation RoVe [103]. RoVe is a text encoding method that provides some resistance to typos [103], which allows you to neutralize errors in optical character recognition. At the next step, the obtained features are projected onto the class space by supplying a fully connected block [104] with SoftMax activation to the input.

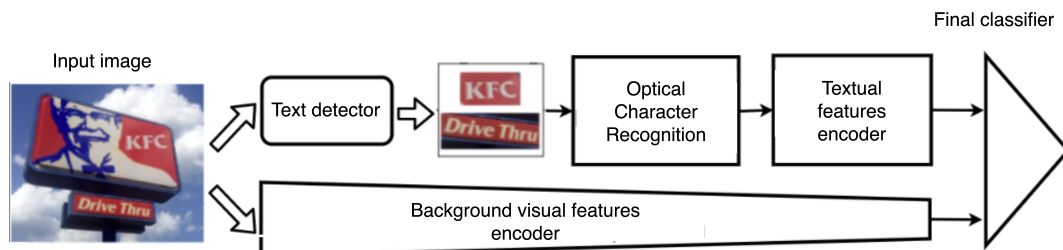


Figure 3.1 — General CCIT architecture with explicit text recognition.

**Combined model with optical text recognition and poster visual features analysis module.** To improve the quality of the combined classifier let's add an additional neural network encoder for the advertising sign image (Fig. 3.2). This modification allows to significantly improve the result when solving the classification problem (Section 3.6).

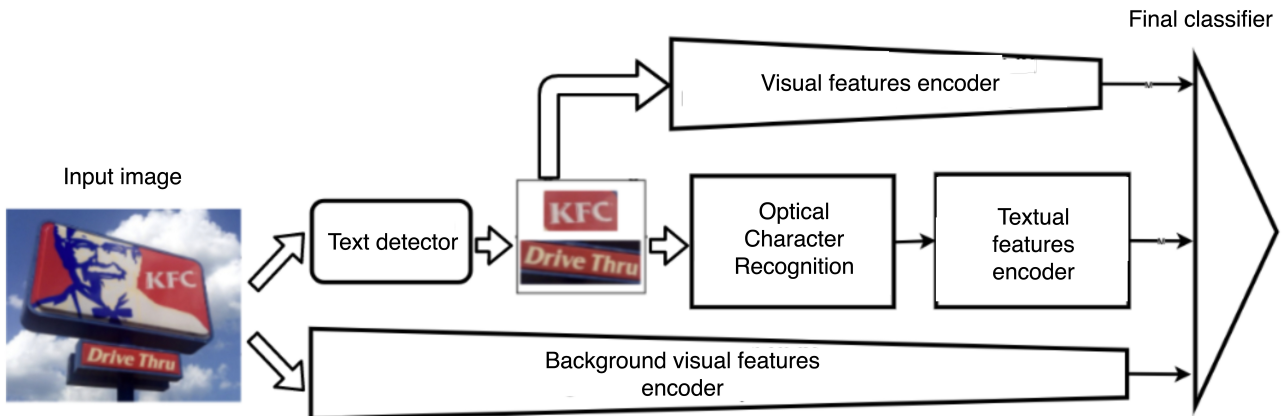


Figure 3.2 — Model architecture with optical text recognition and a separate module for analyzing the visual features of an advertising sign.

### 3.4 CCIT architecture with visual descriptors

In some contexts (for example, when classifying the facades of commercial buildings), when classifying images, not only the semantics of the text contained in the image, but also the visual features of the text (fonts, background, etc.) turn out to be significant [12; 13]. To extract from the image the most important visual features of the text presented in the images, you can create special descriptors for areas of the image with text (i.e. visual descriptors). A general diagram of the CCIT architecture with visual descriptors is shown in Fig. 3.3.

The algorithms for creating such descriptors are described below.

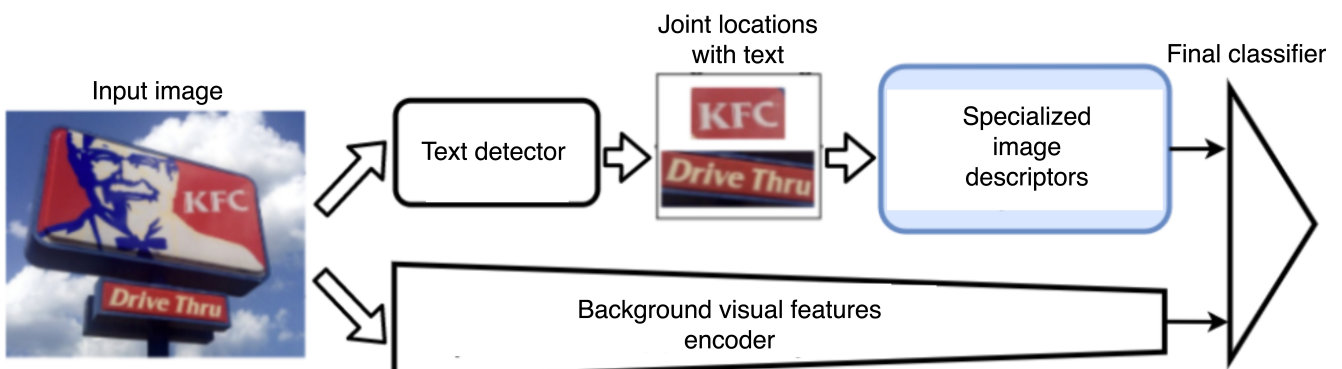


Figure 3.3 — CCIT architecture with visual descriptors.

As can be seen from fig. 3.3, the descriptor calculation algorithm is run on an additional image composed of areas of the original image where text was detected. As a result of this algorithm, a vector is obtained, which is concatenated with the



vector obtained using the encoder of visual features of the entire image. Further, the vector combined in this way is fed to the input of the resulting classifier.

### 3.4.1 Type A descriptor

The first descriptor type created by the CCIT architecture is the A [13] type descriptor. The algorithm for its construction is based on the idea of extracting the maximum amount of information from the relative position of areas with the maximum brightness difference. Another important feature of the algorithm for constructing descriptor A is the independent processing of different words and symbols of the image due to the repetitive nature of printed text.

In order to implement the ideas stated above, it was decided to set the algorithm for calculating type A descriptors as a trace from the movement of objects of a special type (hereinafter we will call such objects *agents*). Agents move across the image independently of each other in steps of a limited size. At each step, each agent decides on the further direction of movement, according to a certain rule given to it during initialization. The rules for choosing the direction of movement for each subsequent step of the agent themselves reflect heuristics, according to which the trace of the movement of each object will contain useful information when solving the classification problem under consideration.

As a heuristic for type A descriptor, the idea of moving objects from the initial position in the direction of maximizing the change in brightness at each step was used. An example of trajectories, movement of agents with such a movement strategy is shown in Fig. 3.4.

It should be noted that each agent has a predefined priority direction to avoid deviation from the main direction of movement due to the presence of contours in the image that are not associated with text characters. Examples of such contours are borders and the non-uniform nature of the image background.

Next, consider the *algorithm for computing the type descriptor A*. The main component of the descriptors are traces of agents, each of which can be represented as follows:

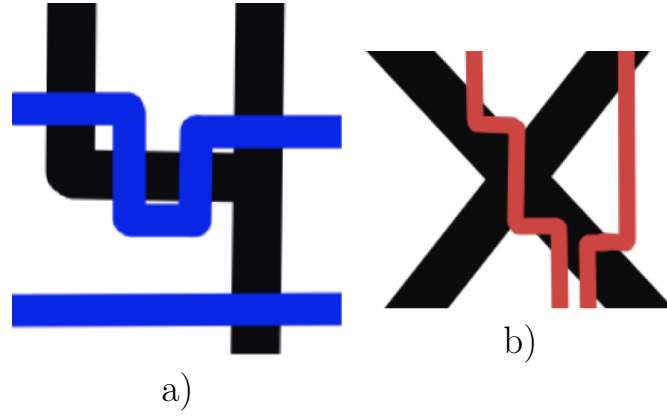


Figure 3.4 – *Agents movement trajectories for constructing a type A descriptor based on the image (original symbols of the advertising poster are marked in black): a) An example of a horizontal track (marked in blue); b) an example of a vertical track (indicated in red).*

$$T^v(x_0, y_0) = (m_1^v(x_0, y_0), \dots, m_N^v(x_0, y_0)),$$

where  $T^v(x_0, y_0)$  is the trace of an agent with a priority direction of movement  $v$  and initial position  $(x_0, y_0)$ ;  $m_i^v(x_0, y_0)$  stands for direction of movement (belonging to the set  $\{up, down, left, right\}$ ) for the step number  $i$  with priority direction  $v$  and initial position  $(x_0, y_0)$ .  $N$  means the number of steps in the tracked trace and is equal to  $W/s + H/s$ .

At each step, the direction of movement is chosen according to the following rule:

$$m_{i+1}^v = \underset{m \in M}{\operatorname{argmax}} (\operatorname{Var}[I[x, y]] + c(m, v)),$$

where  $(x, y) \in P((x_i, y_i), s)$ ,  $v \in \{up, down, left, right\}$ , and  $m_i^v(x_0, y_0)$  denotes the direction of movement for the step  $i$  with priority direction  $v$  and initial position  $(x_0, y_0)$ , as well as a bonus for moving in the priority direction —  $c$ . The pair  $P((x_i, y_i), s)$  denotes the set of coordinates that can be reached with a single step from the position  $(x_i, y_i)$  with a step length equal to  $s$  pixels.  $M$  denotes the set of possible directions of steps, i.e. set  $(\{start, finish, up, down, left, right\})$ . This rule for choosing the direction of the step gives preference to the option in which the sum of the variance of the values of the stepped pixels ( $\operatorname{Var}[I[x, y]]$ ) and the additional bonus for matching the priority direction ( $c(m, v)$ ) is maximum. At the same time,  $c$  is defined as a positive constant equal to 0.1 if  $m$  and  $v$  match, and

0 — otherwise. Thus, we aim to maximize the number of boundaries to be crossed, moving as orthogonally as possible to them at each step.

Thus, we calculate the final descriptors for each priority direction of movement:

$$\begin{aligned} T^{up} &= (T^{up}(x_0, H), \dots, T^{up}(x_A, H)), \\ T^{down} &= (T^{down}(x_0, 0), \dots, T^{down}(x_A, 0)), \\ T^{left} &= (T^{left}(W, y_0), \dots, T^{left}(W, y_B)), \\ T^{right} &= (T^{right}(0, y_0), \dots, T^{right}(0, y_B)), \end{aligned}$$

where  $A$  and  $B$  denote the number of agents with horizontal and vertical priority directions of traffic, respectively.  $W$  and  $H$  stand for the width and height of the original image. As a result, descriptors for all priority directions are combined (using concatenation) into a resultant, which can be described as follows:

$$T = (T^{up}, T^{down}, T^{left}, T^{right}).$$

**Statement 1.** *The type A descriptor algorithm presented above takes  $O(W + H)$  steps, where  $W$  and  $H$  denote the width and height of the original image.*

**Proof.** Note that the decision-making process of each agent at each step is solved by enumeration of all possible options (the number of which is limited by the power of the set of possible directions of movement — there are exactly 4 of them), that is, it is a constant.

We also note that the number of steps for each agent is bounded from above by the image dimensions and depends linearly on them. That is, for horizontal movements, the number of steps does not exceed  $W/s$ , where  $s$  is the step size in pixels. For vertical movements of each agent, the number of steps is limited by the constant  $H/s$ . Thus, the length of each trajectory is bounded from above by the number  $N = W/s + H/s$ .

The number of agents itself is equal to a fixed value  $N_a$  for each direction of movement.

Thus, the total number of agent steps to construct a type descriptor A does not exceed  $= 4 * 4 * N_a * N$ .

Based on the above, the algorithm that constructs the type A descriptor according to the description provided in the section has a computational complexity of  $O(W + H)$ , i.e. takes  $O(W + H)$  steps.  $\square$

**Statement 2.** *The presented algorithm for computing the type A descriptor makes the number of steps linearly dependent on  $N_a$  and  $s$  (hyperparameters of the algorithm for constructing the type A descriptor), where  $N_a$  is the number of agents for each priority direction of movement,  $s$  — step size in pixels.*

**Proof.** The correctness of this assertion follows directly from the proof of assertion 1.  $\square$

Thus, it follows from statements 1 and 2 that the running time of the algorithm for constructing an image descriptor of type A turns out to be linear in the size of the image itself and its hyperparameters. Most similar descriptors have quadratic complexity [24; 26; 105]. At the same time, image sizes in practice turn out to be very large: modern smartphones have 12 to 108 megapixels, smartphones supporting 200 megapixel images will appear in the near future (recall that  $W$  and  $H$  are measured in pixels). The linearity of the algorithm in hyperparameters is also important, since, for example, SIFT [106] and HOG [26] are non-linear, and they use additional methods to speed up that are not applicable to this problem (in our case, we need to work at the maximum image resolution to take into account the visual characteristics of the text). Thus, the complexity of the descriptor construction algorithm is one of the key characteristics of the final solution.

In conclusion, we note that the algorithm for constructing the descriptor A can be easily parallelized in view of the independence of the calculation of the trace of each agent.

### 3.4.2 Type B descriptor

Let's consider one more type of descriptor of image area with text [12] — type B descriptor. This descriptor is similar to type A descriptor: the general vectorized representation of the image with text is also represented as a union of traces of agents' movement; each agent also has a priority direction of movement. The main

difference of the construction algorithm lies in the strategy of choosing the direction of movement at each step of the agent.

Let us consider in more detail *the procedure for constructing a type descriptor B*. In this algorithm, the movement strategy of each agent is based on the idea of moving along significant contours. This movement principle is described as follows:

$$m_{i+1}^v = \underset{m \in M_p}{\operatorname{argmax}} (|R_1^i - R_2^i| + c_p * 1_{\{p\}}(m)),$$

$$(R_1^i, R_2^i) = \begin{cases} (R_{up}^i, R_{down}^i), & \text{if } |R_{up}^i - R_{down}^i| > |R_{left}^i - R_{right}^i| \\ (R_{left}^i, R_{right}^i), & \text{otherwise,} \end{cases}$$

$$R_{up}^i = I[x_i - s/2 : x_i + s/2; y_i - s : y_i],$$

$$R_{down}^i = I[x_i - s/2 : x_i + s/2; y_i : y_i + s],$$

$$R_{left}^i = I[x_i - s : x_i; y_i - s/2 : y_i + s/2],$$

$$R_{right}^i = I[x_i : x_i + s; y_i - s/2 : y_i + s/2],$$

where  $m_i^v$  — the direction of movement,  $i$  — the step number,  $I$  is the input image, and  $(x, y)$  — coordinates of a pixel (agent's position) on the considered image. With the help of  $p$  the priority direction of movement is indicated, and with the help of  $M_p$  — a subset of  $\{up, down, left, right\}$  (permissible movements, depending on the set priority direction  $p$ ). To support the priority direction of movement of agents, a bonus  $c_p$  has been introduced. It should also be noted that each agent always moves  $s$  pixels with each step. Thus, at each step, the agent compares the absolute brightness difference in adjacent vertical directions  $(R_{up}^i, R_{down}^i)$  and adjacent horizontal directions  $(R_{left}^i, R_{right}^i)$ , choosing the direction of the next step, taking into account the bonus for moving in the priority direction  $c_p * 1_{\{p\}}(m)$ . This is done by choosing a direction that maximizes the sum of the maximum absolute brightness difference in adjacent directions  $(R_1^i, R_2^i)$  and the bonus term of the priority direction of movement equal to  $c_p$  (0.1 in the case of movement in the priority direction and 0 in otherwise).

As a result, the trace of each agent with a predefined priority direction of movement can be expressed as follows:

$$T^p(x_0, y_0) = (m_1(x_0, y_0), \dots, m_N(x_0, y_0)),$$

where  $T^p(x_0, y_0)$  – the trace of the agent with priority direction  $p$  and initial position  $(x_0, y_0)$ ;  $m_i(x_0, y_0)$  denotes the movement chosen by the agent at step  $i$  with predefined initial position and movement direction.  $N$  – the length of each trajectory (if the edge of the image is reached before the agent takes  $N$  steps (initialized with  $W/s + H/s$ ), then the remaining steps are filled with a special value). Thus, the trajectories of this type of descriptors tend to lie along the contours of the original image (Fig. 3.5).

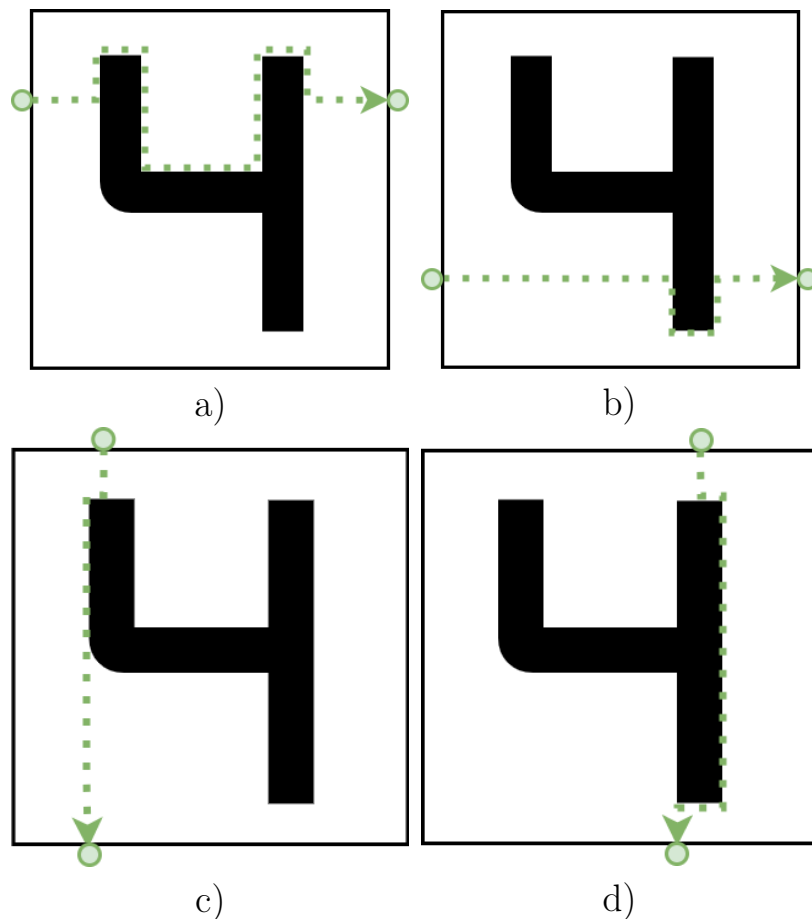


Figure 3.5 — An illustration of the trace of agents with a movement strategy characteristic of type B descriptors:

- a,b) trajectories with a horizontal priority direction;  
 c,d) trajectories with a vertical priority direction of movement.

Further, all trajectories are grouped according to priority directions and initial positions:

$$\begin{aligned} T^{up} &= (T^{up}(x_0, H), \dots, T^{up}(x_A, H)), \\ T^{down} &= (T^{down}(x_0, 0), \dots, T^{down}(x_A, 0)), \\ T^{left} &= (T^{left}(W, y_0), \dots, T^{left}(W, y_B)), \\ T^{right} &= (T^{right}(0, y_0), \dots, T^{right}(0, y_B)). \end{aligned}$$

where  $A$  and  $B$  denote the number of horizontally and vertically oriented agents, respectively.  $W$  — original image width,  $H$  — height.

As a result, all groups of trajectories in priority directions are combined by concatenation into the final representation of the descriptor, which is formalized as follows:

$$T = (T^{up}, T^{down}, T^{left}, T^{right}).$$

**Statement 3.** *The proposed method for constructing a type descriptor  $B$  takes  $O(W + H)$  steps, where  $W$  and  $H$  denote the width and height of the image in question.*

**Proof.** Note that the decision-making process of each agent at each step is solved by enumeration of all possible options (the number of which is limited by the power of the set of possible directions of movement — there are exactly 4 of them), that is, it is a constant.

We also note that the number of steps for each agent is bounded from above by the image dimensions and depends linearly on them. That is, for horizontal movements, the number of steps does not exceed  $W/s$ , where  $s$  is the step size in pixels. For vertical movements of each agent, the number of steps is limited by the constant  $H/s$ . Thus, the length of each trajectory is bounded from above by the number  $N = W/s + H/s$ .

The number of agents itself is equal to a fixed value  $A$  for each vertical direction of movement and  $B$  — for each horizontal direction of movement.

Thus, the total number of agent steps to construct a type descriptor  $B$  does not exceed the value  $= 4 * 2 * (A + B) * N$ .

Based on the foregoing, we can conclude that the algorithm for constructing a type descriptor B has a computational complexity of  $O(W + H)$ , i.e. takes  $O(W + H)$  steps.  $\square$

**Statement 4.** *The presented algorithm for calculating the type B descriptor makes the number of steps linearly dependent on A and B (hyperparameters of the type B descriptor construction algorithm), where A is the number of descriptors for each priority vertical direction of movement, B — the number of handles for each priority vertical direction of movement.*

**Proof.** The correctness of this assertion follows directly from the proof of statement 3.  $\square$

### 3.4.3 Type C descriptor

Consider another type of special descriptor for an image area with text (type C descriptor). The description of the construction procedure of this descriptor almost completely coincides with the description of the type B descriptor. The only difference is that after the agent deviates from the priority direction, the bonus from the agent's movement in the priority direction is equal to zero. Thus, the agent is hung on a meaningful image object (Fig. 3.6).

The rule for choosing the direction of the agent's movement at the next step is specified as follows:

$$m_{i+1}^v = \underset{m \in M_p}{\operatorname{argmax}} (|R_1^i - R_2^i| + \alpha * c_p * 1_{\{p\}}(m)),$$

where  $\alpha = 1$  if the direction of movement did not change, otherwise 0.

Thus, each agent is released from the influence of the priority direction of movement only when the first significant element of the image is reached. That is, the C type descriptor repeats the idea of the B type descriptor, moving along the boundaries of the maximum brightness difference (maximizing the absolute brightness difference in adjacent directions  $|R_1^i - R_2^i|$ ), while the component that makes the agent move in the priority direction  $\alpha * c_p * 1_{\{p\}}(m)$  vanishes due to the



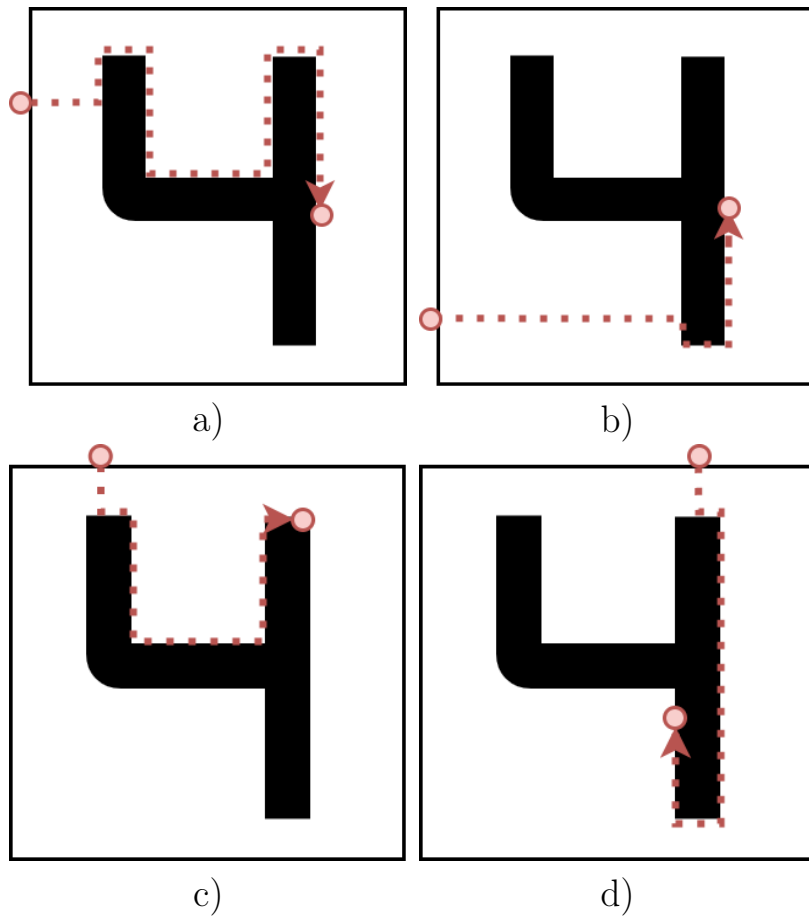


Figure 3.6 — *An illustration of the agent trace when constructing a C type descriptor:*

*a,b) trail of agents with a priority horizontal direction of movement;  
c,d) trail of agents with a priority vertical direction of motion.*

special multiplier  $\alpha$  as soon as the agent is affected by the contour. As a result, the agent is attached to the first detected contour.

**Statement 5.** *The proposed algorithm for constructing a type descriptor  $C$  takes  $O(W + H)$  steps, where  $W$  and  $H$  denote the width and height of the image in question.*

**Proof.** The proof is similar to assertion 3. □

**Proposition 6.** *The presented algorithm for computing the type  $B$  descriptor makes the number of steps linearly dependent on  $A$  and  $B$  (hyperparameters of the type  $A$  descriptor construction algorithm), where  $A$  is the number of descriptors for each priority vertical direction of movement,  $B$  - - the number of handles for each priority vertical direction of movement.*

**Proof.** The correctness of this assertion follows directly from the proof of assertion 5. □

### 3.5 Photo recognition system for facades of commercial buildings with advertising signs

The task of classifying photographs of facades of commercial buildings by the type of services provided is related to the task of recognizing images of identity cards. Both of the listed tasks are image classification tasks with the presence of a visual representation of text, obtained using image capture tools in difficult shooting conditions.

However, images in the context of the task of classifying commercial facades are much more variable, since shooting (creating images) takes place in the street, under different lighting conditions using mobile devices (smartphones) with a wide variety of camera characteristics and settings, etc. Therefore, a more complex solution is required. on the other hand, there were no requirements for high accuracy rates (Precision).

The model was deployed on a server equipped with an Intel i7 3.5 GHz processor, 64 GB, RAM, 2 NVidia Tesla t4 16 GB graphics cards.

The model training server was configured with four 16GB NVidia Tesla t4 graphics cards, 64GB RAM, and an Intel i7 4GHz processor.

The model was trained and worked on the basis of the Tensorflow [97] deep learning framework. To develop models and support additional data processing and preparation operations, the Python programming language and additional computer vision and image processing frameworks — PIL [96] and OpenCV [94] were used. Below is an approbation of the previously described variants of the CCIT architecture on datasets.

### 3.6 Experiments

This section presents an experimental study of the developed image classification methods.

#### Datasets

To investigate the effectiveness of the CCIT architectures options described above, a Signboard Classification Dataset (SCD)<sup>1</sup> was collected, marked up and made publicly available. This set includes photographs of commercial building facades with advertising signs (fig. 3.7). All elements of the SCD were divided into 4 categories according to the type of service provided: restaurants, hotels, shops, other (anything that does not fall into the listed categories). The images for the SCD were taken from the Google Street View [107] public dataset and from the Flickr [108] image host. All samples obtained from the Flickr hosting are distributed under a license allowing their use for research purposes. The performance of all CCIT variants was studied on SCD subsamples containing 357, 1000, 2000, and 3000 images. Each subsample was divided into a training set, a testing set, and a hyperparameter validation set in one of the most commonly used proportions — 5:1:1. It should also be noted that in each sample, all classes are represented by approximately the same set of elements. Previously, augmentations were applied to the images by an angle of up to 10 degrees and a shift of up to 15 pixels in various directions.

**Quality metrics.** To study the effectiveness of the use of various CCIT options in solving the problem of classifying the facades of commercial buildings by the type of services provided, the classification quality metric was used —  $F_1$ . For the binary classification case, formulated similarly to Section 2.6.

---

<sup>1</sup><https://github.com/madrugado/signboard-classification-dataset>



Figure 3.7 — *Illustrations of elements from the SCD dataset:*

- a) *photograph of the hotel facade;*
- b) *a photo of the shop front;*
- c) *the image of the facade of the restaurant with an advertising sign;*
- d) *photograph with a sign that does not belong to the categories listed above (class 'other').*

To evaluate the efficiency of the classifier in solving the problem of multiclass classification, the values of  $F_1$  were averaged over all 4 considered classes. The best results on test samples for each model and dataset configuration are presented in the table 3.

For definiteness, the threshold for calculating metrics was set as follows way to maximize  $F_1$ . Due to the lack of additional conditions in the problem statement, a balanced quality metric was chosen. For each method and all classes, the  $F_1$  was averaged, after that, a threshold that maximizes the value of this metric was selected.

**Comparable analogues.** General image classification models have been tested as alternative approaches for implementations of CCIT architectures.

Similarly to these approaches, implementations of CCIT architectures with an OCR module and various types of descriptors were trained and tested on the given data sets (see sections 3.4, 3.4.1, 3.4.2, 3.4.3).

**Analysis of the results.** The results of the experiments are shown in Table 3.

We can note the low efficiency of all the considered models on the CFD-357 dataset. This circumstance is due to the extreme conditions for training deep models to work with multimodal content — 357 original images to which data augmentation was applied. However, for successful generalization of models, such a volume of data in the training sample was not enough. This justification is supported by the increase in the quality metric values for each of the studied architectures as the number of elements in the studied data sets increases — CFD-1000, CFD-2000,

CFD-3000 contain 1000, 2000 and 2000 original images of commercial building facades, respectively, which significantly enriches the feature space.

When considering the experimental results, the second noteworthy fact is the degradation of the quality metric value for the CCIT-OCR modification. The reason for this circumstance is the insufficient information content of the text presented on the images of the facades of commercial buildings. Note that CCIT modifications that use special descriptors for areas with a visual representation of text instead of explicit OCR showed better results and are more suitable for solving this problem.

We also note that the quality metrics for various variants of descriptors A, B, and C demonstrate a significant increase in the quality of classification. This fact is evidence of the advantage of later heuristics (B over A, C over B), which implement more and more optimal strategies for moving agents. The movement strategy of agents for a descriptor of type B, which regulates movement along the contours, turned out to be more informative than the movement strategy for a descriptor of type A, crossing the contours. This one also speaks in favor of the importance of contour information. The agent movement strategy for the C type descriptor turned out to be even more efficient, since this strategy takes into account the peculiarity of printed text — its division into characters.

Table 3 —  $F_1$  values on test samples for various SCD configurations.

Model	CFD-357	CFD-1000	CFD-2000	CFD-3000
VGG16	0.11	0.19	0.22	0.30
VGG19	0.13	0.25	0.28	0.32
MobileNet	0.12	0.22	0.20	0.28
ResNet	0.18	0.38	0.41	0.43
Inception	0.13	0.33	0.39	0.41
ViT-B_32(+MLP)	0.17	0.40	0.42	0.43
CLIP-ResNet50+MLP	0.20	0.43	0.45	0.44
BLIP+MLP	0.22	0.40	0.46	0.47
CCIT-OCR	0.24	0.47	0.46	0.43
CCIT-type A	0.28	0.51	0.58	0.62
CCIT-type B	0.38	0.59	0.61	0.70
CCIT-type C	0.35	0.63	0.66	0.74

As can be seen from the table 3, the best values of metrics and qualities are achieved by implementing the CCIT architecture with a special type descriptor C.

### 3.7 Chapter Conclusion

In this chapter, the combined neural network architecture CCIT was presented for solving problems of image classification with the presence of text fragments under the condition of different image quality and the presence of defects in them. Variants of the CCIT architecture with additional descriptors of text areas of the image were also presented, which made it possible to significantly improve the quality of image processing. In particular, the application of this approach was studied on the example of solving the problem of classifying photographs of facades of commercial buildings by the type of services provided. To build an experimental base, several versions of data sets were collected for training and testing models.

The results of the experiments show the low efficiency of general image classifiers compared to models based on CCIT architectures. This circumstance may be due to the lack of an explicit analysis of the visual representations of the text, which are very different from general images in terms of the relationship between the semantics of the image and the low-level graphic primitives presented (such as the nature of the contours and brightness differences, for example). This hypothesis is also supported by the high efficiency of architectures with attention compared to other general image classifiers. The ability to clearly focus attention on some areas of the image, which increases the quality metrics, indicates the importance of the idea of localization for solving problems of image processing with visual representations of text.

Thus, the conducted experiments have demonstrated the effectiveness of using CCIT (the best result on SCD at the moment among neural network approaches with  $F_1 = 0.74$ ) and special descriptors for solving problems of image classification with text fragments. It should be noted here that various variants of the CCIT architecture have emerged over time to improve the quality of the models. For these purposes, a number of heuristics were gradually formed, each of which turned out to be more successful in practice than the others. The CCIT-OCR modification used the semantics of explicitly detected text, which made it possible to achieve an advantage over most of the general classification methods studied. However, with the expansion of the data set, degradation of quality metrics began to appear, which demonstrated the insufficient expressive ability of the methods for extracting seman-

tics from the text for this task, which was associated with the presence of a large number of proper names and titles in the texts of signboards. To solve this problem, a modification of CCIT with a special type descriptor A was created, based on the idea of the importance of the visual representation of text. To further improve the quality of the classification, the agent movement strategy (CCIT architecture with a special type B descriptor) was modified in such a way as to force the agents to move along the text contours, which showed an increase in efficiency. This circumstance emphasized the importance of extracting contour information. Subsequently, another idea was formed to achieve even higher quality (CCIT architecture with type C descriptors): separation of contour characteristics for various text elements. That is, for each agent, we modify the movement rule so that the agent is maximally attached to the first found object.

Thus, the use of the ideas of creating special descriptors to take into account the signs of the visual representation of the text made it possible to significantly improve the quality of solving the class of problems under consideration.

## Conclusion

In the course of the dissertation research, the following main results were obtained.

1. A review and analysis of methods for classifying and processing images applicable to the analysis of images containing a visual representation of text has been prepared.
2. The architecture of the combined neural network VCA classifier for classifying scene images with visual text elements without the use of optical character recognition is presented.
3. Architectures of combined CCIT neural network classifiers are presented for classifying images of scenes with visual elements of text, using special methods for extracting semantics from the visual representation of text. For CCIT architectures modules for extracting special descriptors for image areas with a visual representation of text have been developed.
4. An experimental analysis of the developed neural network models has demonstrated their high efficiency, which is superior to the efficiency of general image classification methods in classifying images with a visual representation of text.

Further work on this topic can be directed towards creating expressive and lightweight image descriptors with text and developing more efficient procedures for computing them. It is possible to integrate convolutional image encoders based on new neural network architectures for efficient feature extraction. It is also possible to refine the presented schemes of combined models (adding and editing modules) to achieve greater efficiency. In addition, an interesting direction for further expansion of the VCA architecture may be the analysis of documents whose fields are filled in by hand.



## References

1. *Javidi B., L. Horner J.* Optical pattern recognition for validation and security verification // *Optical Engineering (OPT ENG)*. — Vol. 33(6). — pp. 224-230. — 1994. — DOI: 10.1117/12.170736.
2. *Vizilter Y., Zheltov S., A. Lukin A.* Development of OCR system for portable passport and visa reader // *Proceedings of SPIE - The International Society for Optical Engineering*. — Vol. 3651. — pp. 194-199. — 1999. — DOI: 10.1117/12.335817.
3. Slant rectification in Russian passport OCR system using fast Hough transform / *E. Limonova [et al.]* // *International Conference on Machine Vision*. — Vol. 10341. — pp. 127-131. — 2017. — DOI: 10.1117/12.2268725.
4. *Kim K., Oh A., Woo Y.* PCA-Based Face Verification and Passport Code Recognition Using Improved FKCN Algorithm // *Eighth International Conference on Intelligent Systems Design and Applications*. — Vol. 2. — pp. 51-57. — 2008. — DOI: 10.1109/ISDA.2008.247.
5. *Intasuwan T., Kaewthong J., Vittayakorn S.* Text and Object Detection on Billboards // *10th International Conference on Information Technology and Electrical Engineering (ICITEE)* — pp. 6-11. — 2018. — DOI: 10.1109/ICITEED.2018.8534879.
6. *Malykh V., Samarin A.* Combined Advertising Sign Classifier // *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* — Vol. 11832. — pp. 179-185. — 2019. — DOI: 10.1007/978-3-030-37334-4\_16.
7. *Zhou J., McGuinness K., O'Connor N. E.* A Text Recognition and Retrieval System for e-Business Image Management // *MultiMedia Modeling*. — pp. 23-35. — 2018.
8. Learning and Recognition of On-Premise Signs From Weakly Labeled Street View Images / *T. Tsai [et al.]* // *Transactions on Image Processing*. — Vol. 23(3). — pp. 1047-1059. — 2014. — DOI: 10.1109/TIP.2014.2298982.

9. *Chacra D. A., Zelek J.* Road Segmentation in Street View Images Using Texture Information // 13th Conference on Computer and Robot Vision (CRV). — pp. 424-431. — 2016. — DOI: 10.1109/CRV.2016.47.
10. *Chattopadhyay T., Sinha A.* Recognition of trademarks from sports videos for channel hyperlinking in consumer end // 13th International Symposium on Consumer Electronics. — pp. 943-947. — 2009. — DOI: 10.1109/ISCE.2009.5156881.
11. ImageNet: A Large-Scale Hierarchical Image Database / J. Deng [et al.] // IEEE Conference on Computer Vision and Pattern Recognition — pp. 248-255. — 2009.
12. *Samarin A., Malykh V., Muravyov S.* Specialized Image Descriptors for Signboard Photographs Classification // Communications in Computer and Information Science . — Vol. 1243 — pp. 122-129. — 2020. — DOI: 10.1007/978-3-030-57672-1\_10.
13. *Samarin A., Malykh V.* Worm-like image descriptor for signboard classification // CEUR Workshop Proceedings, — Vol. 2691 — pp. 30-33. — 2020. — DOI: 10.1007/978-3-030-57672-1\_10.
14. *Xue C., Lu S., Zhan F.* Accurate Scene Text Detection Through Border Semantics Awareness and Bootstrapping // Computer Vision – ECCV. — pp. 370-387. — 2018.
15. Deep Residual Learning for Image Recognition / K. He [et al.] // Conference on Computer Vision and Pattern Recognition (CVPR). — pp. 770-778. — 2016. — DOI: 10.1109/CVPR.2016.90.
16. *Simonyan K., Zisserman A.* Very Deep Convolutional Networks for Large-Scale Image Recognition // CoRR. — 2014.
17. MobileNetV2: Inverted Residuals and Linear Bottlenecks / M. Sandler [et al.] // CVF Conference on Computer Vision and Pattern Recognition. — pp. 4510-4520. — 2018.
18. *Tan M., Le Q.* EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks // International conference on machine learning — pp. 6105-6114. — 2019.

19. You Only Look Once: Unified, Real-Time Object Detection / J. Redmon [et al.] // Proceedings of the IEEE conference on computer vision and pattern recognition. — pp. 779-788. — 2015.
20. SSD: Single Shot MultiBox Detector / W. Liu [et al.] // Proceedings, Part I 14. — Springer International Publishing. — pp. 21-37. — 2015.
21. TextBoxes: A Fast Text Detector with a Single Deep Neural Network / M. Liao [et al.] // Proceedings of the AAAI conference on artificial intelligence — Vol. 31. — 2016.
22. Detecting Text in Natural Image with Connectionist Text Proposal Network / Z. Tian [et al.] // Computer Vision—ECCV 2016: 14th European Conference, Amsterdam Proceedings, Part VIII 14. — pp. 56-72. — 2016.
23. EAST: An Efficient and Accurate Scene Text Detector / X. Zhou [et al.] // Conference on Computer Vision and Pattern Recognition (CVPR). — pp. 2642-2651. — 2017. — DOI: 10.1109/CVPR.2017.283.
24. *Dittimi T., Suen C.* Modified HOG Descriptor-Based Banknote Recognition System // Advances in Science, Technology and Engineering Systems Journal. — Vol. 3.(5). — 2018. — DOI: 10.25046/aj030541.
25. *Sun J., Shisong Z., Xiaosheng W.* Image retrieval based on an improved CS-LBP descriptor // Information Management and Engineering (ICIME). — pp. 115-117. — 2010. — DOI: 10.1109/ICIME.2010.5477432.
26. *Huang C., Huang J.* A Fast HOG Descriptor Using Lookup Table and Integral Image // CoRR. — 2017.
27. *A.V. Samarin V.A. Malykh P. K.* ID verification method using limited image area // Proceedings of the Institute for System Analysis of the Russian Academy of Sciences — Vol. 70(1) — pp. 15-23. — 2020. — DOI: 10.14357/20790279200102.
28. One-Stage Classifiers Based on U-Net and Autoencoder with Attention for Recognition of Neoplasms from Single-Channel Monochrome Computed Tomography Images / A. Samarin [et al.] // Pattern Recognition and Image Analysis — Vol. 33 — pp. 132-138. — 2023.

29. *Samarin A., Malykh V.* Ensemble-Based Commercial Buildings Facades Photographs Classifier // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). — Vol. 12602 — pp. 257-265. — 2021. — DOI: 10.1007/978-3-030-72610-2\_19.
30. *Tatanov O., Samarin A.* LFIEM: Lightweight Filter-based Image Enhancement Model // 25th International Conference on Pattern Recognition (ICPR). — pp. 873-878. — 2021. — DOI: 10.1109/ICPR48806.2021.9413138.
31. *Samarin A., Savelev A., Malykh V.* Two-Stage Self-Attention Based Neural Model For Lung Cancer Recognition // Science and Artificial Intelligence conference (S.A.I.ence). — pp. 50-53. — 2020. — DOI: 10.1109/S.A.I.ence50533.2020.9303206.
32. Trainable Agents Movement Strategies for Advertising Sign Visual Descriptors / A. Samarin [et al.] // Pattern Recognition and Image Analysis. — Vol. 32(3) — pp. 651-657. — 2022.
33. One-Stage Attention-Based Neoplasms Recognition Method for Single-Channel Monochrome Computer Tomography Snapshots / A. Samarin [et al.] // Pattern Recognition and Image Analysis. — Vol. 32(3) — pp. 645–650. — 2022.
34. Predictors Based on Convolutional Neural Networks for the Movement Strategy of Trainable Agents for Building Customized Image Descriptors / A. Samarin [et al.] // Pattern Recognition and Image Analysis — Vol. 33. — pp. 139-146. — 2023.
35. *Loshchilov I., Hutter F.* SGDR: Stochastic Gradient Descent with Restarts // CoRR. — 2016.
36. *Lowe D.* Adam: A Method for Stochastic Optimization // CoRR. — 2014.
37. *Reddi S. J., Kale S., Kumar S.* On the Convergence of Adam and Beyond // CoRR. — 2019.
38. *Zeiler M. D.* ADADELTA: An Adaptive Learning Rate Method // CoRR. — 2012.
39. RMSProp and equilibrated adaptive learning rates for non-convex optimization / Y. N. Dauphin [et al.] // CoRR. — 2015.
40. *Liu C., Belkin M.* MaSS: an Accelerated Stochastic Method for Overparametrized Learning // CoRR. — 2018.

41. *Botev A., Lever G., Barber D.* Nesterov's Accelerated Gradient and Momentum as approximations to Regularised Update Descent // International Joint Conference on Neural Networks (IJCNN)s, IEEE, — pp. 1899-1903. — 2016.
42. *Ruder S.* An overview of gradient descent optimization algorithms // CoRR. — 2016.
43. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications / A. G. Howard [et al.] // CoRR. — 2017.
44. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation / J. Li [et al.] // CoRR. — 2022.
45. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models / J. Li [et al.] // CoRR. — 2023.
46. *Gülcü A., Kuş Z.* Hyper-Parameter Selection in Convolutional Neural Networks Using Microcanonical Optimization Algorithm // IEEE Access — Vol. 8. — pp. 52528 - 52540. — 2020. — DOI: 10.1109/ACCESS.2020.2981141.
47. *Atmaja B. T., Akagi M.* Deep Multilayer Perceptrons for Dimensional Speech Emotion Recognition // Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC). — pp. 325-331. — 2020.
48. *Rocha M., Cortez P., Neves J.* Simultaneous Evolution of Neural Network Topologies and Weights for Classification and Regression // International Work-Conference on Artificial Neural Networks. — pp. 59-66. — 2005. — DOI: 10.1007/11494669\_8.
49. *Rosenblatt F.* The perceptron: a probabilistic model for information storage and organization in the brain. // Psychological review — Vol. 65(6) — pp. 386-408. — 1958.
50. *Dubey S. R., Singh S. K., Chaudhuri B. B.* A Comprehensive Survey and Performance Analysis of Activation Functions in Deep Learning // CoRR. — 2021.
51. Activation Functions: Comparison of trends in Practice and Research for Deep Learning / C. Nwankpa [et al.] // CoRR. — 2020.
52. *Szandata T.* Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks // CoRR. — 2020.

53. *Hancock J., Khoshgoftaar T.* Survey on categorical data for neural networks // Journal of Big Data — Vol. 7. — pp. 1-41. — 2020. — DOI: 10.1186/s40537-020-00305-w.
54. Spatially supervised recurrent convolutional neural networks for visual object tracking / G. Ning [et al.] // International Symposium on Circuits and Systems (ISCAS). — pp. 1-4. — 2017. — DOI: 10.1109/ISCAS.2017.8050867.
55. *Sherstinsky A.* Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network // Physica D: Nonlinear Phenomena — Vol. 404. — pp. 132306. — 2020. — DOI: 10.1016/j.physd.2019.132306.
56. *Schmidt R. M.* Recurrent Neural Networks (RNNs): A gentle Introduction and Overview // CoRR. — 2019.
57. *Hochreiter S., Schmidhuber J.* Long Short-term Memory // Neural computation. — Vol. 9(8) — pp. 1735 - 1780. — 1997. — DOI: 10.1162/neco.1997.9.8.1735.
58. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling / J. Chung [et al.] // CoRR. — 2014.
59. Implementation of Training Convolutional Neural Networks / T. Liu [et al.] // CoRR. — 2015.
60. Dropout: A Simple Way to Prevent Neural Networks from Overfitting / N. Srivastava [et al.] // Journal of Machine Learning Research. — Vol. 15(1)—pp. 1929-1958. — 2014.
61. CBAM: Convolutional Block Attention Module / S. Woo [et al.] // Proceedings of the European Conference on Computer Vision (ECCV) — pp. 3-19. — 2018.
62. Non-Local Neural Networks / X. Wang [et al.] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). — pp. 7794-7803. — 2018.
63. Self-Attention Generative Adversarial Networks / H. Zhang [et al.] // CoRR. — 2018.

64. Learning better deep features for the prediction of occult invasive disease in ductal carcinoma in situ through transfer learning / B. Shi [et al.] // Progress in Biomedical Optics and Imaging - Proceedings of SPIE. — pp. 98. — 2018. — DOI: 10.1117/12.2293594.
65. Going deeper with convolutions / C. Szegedy [et al.] // Conference on Computer Vision and Pattern Recognition (CVPR). — pp. 1-9. — 2015. — DOI: 10.1109/CVPR.2015.7298594.
66. Chemception: A Deep Neural Network with Minimal Chemistry Knowledge Matches the Performance of Expert-developed QSAR/QSPR Models / G. Goh [et al.] // CoRR. — 2017.
67. Rethinking the Inception Architecture for Computer Vision / C. Szegedy [et al.] // Conference on Computer Vision and Pattern Recognition (CVPR). — pp. 2818-2826. — 2015.
68. 1D convolutional neural networks and applications: A survey / S. Kiranyaz [et al.] // Mechanical Systems and Signal Processing — Vol. 151. — pp. 107398. — 2021. — DOI: 10.1016/j.ymsp.2020.107398.
69. ResNet-like Architecture with Low Hardware Requirements / E. Limonova [et al.] // CoRR. — 2020.
70. Big Transfer (BiT): General Visual Representation Learning / A. Kolesnikov [et al.] // European Conference on Computer Vision. — pp. 491-507. — 2020. — DOI: 10.1007/978-3-030-58558-7\_29.
71. Fixing the train-test resolution discrepancy / H. Touvron [et al.] // CoRR. — 2019.
72. Scene Segmentation With Dual Relation-Aware Attention Network / J. Fu [et al.] // Transactions on Neural Networks and Learning Systems. — Vol. 32(6) — pp. 1-14. — 2020. — DOI: 10.1109/TNNLS.2020.3006524.
73. CAA : Channelized Axial Attention for Semantic Segmentation / Y. Huang [et al.] // CoRR. — 2021.
74. Real-Time Food Intake Monitoring Using Wearable Egocentric Camera / M. Imtiaz [et al.] // 42nd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC). — pp. 4191-4195. — 2020. — DOI: 10.1109/EMBC44109.2020.9175497.

75. *Hoang V.-T., Jo K.-H.* Practical Analysis on Architecture of EfficientNet // 14th International Conference on Human System Interaction (HSI). — pp. 1-4. — 2021. — DOI: 10.1109/HSI52170.2021.9538782.
76. *Zhang P., Yang L., Li D.* EfficientNet-B4-Ranger: A novel method for greenhouse cucumber disease recognition under natural complex environment // Computers and Electronics in Agriculture — Vol. 176. — pp. 105652. — 2020. — DOI: 10.1016/j.compag.2020.105652.
77. CoAtNet: Marrying Convolution and Attention for All Data Sizes / Z. Dai [et al.] // CoRR. — 2021.
78. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale / A. Dosovitskiy [et al.] // CoRR. — 2020.
79. *Ruan B., Shuai H.-H., Cheng W.-H.* Vision Transformers: State of the Art and Research Challenges // CoRR. — 2022.
80. Attention Is All You Need / A. Vaswani [et al.] // CoRR. — 2017.
81. Learning Transferable Visual Models From Natural Language Supervision / A. Radford [et al.] // International conference on machine learning — pp. 8748-8763. — 2021.
82. *Zeng W., Meng Q., Zhang S.* Natural Scene Chinese Character Text Detection Method Based on Improved CTPN // Journal of Physics: Conference Series — Vol. 1314. — pp. 12200. — 2019. — DOI: 10.1088/1742-6596/1314/1/012200.
83. Character Region Awareness for Text Detection / Y. Baek [et al.] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. — pp. 9357-9366. — 2019. — DOI: 10.1109/CVPR.2019.00959.
84. *Sang D., Cuong L.* Improving CRNN with EfficientNet-like feature extractor and multi-head attention for text recognition // Proceedings of the 10th International Symposium on Information and Communication Technology. — pp. 285-290. — 2019. — DOI: 10.1145/3368926.3369689.
85. Tesseract (OCR). — URL: <https://github.com/tesseract-ocr/tesseract>.
86. *Smith R.* An Overview of the Tesseract OCR Engine // Ninth International Conference on Document Analysis and Recognition. — pp. 629-633. — 2007. — DOI: 10.1109/ICDAR.2007.4376991.



87. Effectiveness of Modern Text Recognition Solutions and Tools for Common Data Sources / K. Smelyakov [et al.] // COLINS — pp. 154-165. — 2021.
88. *Xiang J., Zhu G.* Joint Face Detection and Facial Expression Recognition with MTCNN // 4th International Conference on Information Science and Control Engineering (ICISCE). — pp. 424-427. — 2017. — DOI: 10.1109/ICISCE.2017.95.
89. *Kim K.-B., Kim S.* A passport recognition and face verification using enhanced fuzzy ART based RBF network and PCA algorithm // Neurocomputing. — Vol. 71(6-18) — pp. 3964. — 2009. — DOI: 10.1016/j.neucom.2009.07.001.
90. *Hinton G., Salakhutdinov R.* Reducing the Dimensionality of Data with Neural Networks // Science. — Vol. 313. — pp. 504-507. — 2006. — DOI: 10.1126/science.1127647.
91. *Gupta S., Mazumdar S. G.* Sobel Edge Detection Algorithm // International Journal of Computer Science and Management Research. — Vol. 2. — pp. 1578-1583. — 2013.
92. *Elboher E., Werman M.* Efficient and accurate Gaussian image filtering using running sums // 12th International Conference on Intelligent Systems Design and Applications (ISDA). — pp. 897-902. — 2012.
93. Image quality assessment: from error visibility to structural similarity / Z. Wang [et al.] // IEEE transactions on image processing. — Vol. 12(4). — pp. 600–612. — 2004.
94. OpenCV. — URL: <https://opencv.org>.
95. Dlib. — URL: <http://dlib.net>.
96. Pillow (PIL Fork). — URL: <https://python-pillow.org>.
97. Tensorflow. — URL: <https://www.tensorflow.org>.
98. Comparing published multi-label classifier performance measures to the ones obtained by a simple multi-label baseline classifier / J. Metz [et al.] // CoRR. — 2015.
99. Microsoft COCO: Common Objects in Context / T.-Y. Lin [et al.] // Computer Vision – ECCV. — pp. 740-755. — 2014.

100. Flickr30k Entities: Collecting Region-to-Phrase Correspondences for Richer Image-to-Sentence Models / B. A. Plummer [et al.] // CoRR. — 2015.
101. *Vasiliev R., Koznov D., Chernishev G.* TraceSim: A Method for Calculating Stack Trace Similarity // Proceedings of the 4th ACM SIGSOFT International Workshop on Machine-Learning Techniques for Software-Quality Evaluation — pp. 25-30. — 2020.
102. On-premise signs detection and recognition using fully convolutional networks / Y. Wang [et al.] // International Conference on Multimedia and Expo (ICME). — pp. 1-6. — 2016. — DOI: 10.1109/ICME.2016.7552923.
103. *Malykh V.* Robust word vectors for Russian language // Proceedings of Artificial Intelligence and Natural Language AINL FRUCT Conference. — pp. 10-12. — 2016.
104. Densely Connected Convolutional Networks / G. Huang [et al.] // Conference on Computer Vision and Pattern Recognition (CVPR). — pp. 2261-2269. — 2017.
105. *Bachchan A., Gorai A., Gupta P.* Automatic License Plate Recognition Using Local Binary Pattern and Histogram Matching // Intelligent Computing Theories and Application: 13th International Conference, ICIC — pp. 22-34. — 2017. — DOI: 10.1007/978-3-319-63312-1\_3.
106. *Lowe D.* Object recognition from local scale-invariant features // Proceedings of the Seventh IEEE International Conference on Computer Vision — Vol. 2. — pp. 1150-1157. — 1999.
107. Google Street View Dataset. — URL: <https://github.com/daminiR/GoogleStreetViewDatasetBias>.
108. Flickr. — URL: <https://www.flickr.com>.

## A. Acts on the implementation of the results of dissertation work

**Act**  
**on the implementation of the results of the**  
**dissertation thesis by Samarin Aleksei Vladimirovich,**  
**«Combined neural network classifiers of specific**  
**images»**

The results of the dissertation work of Samarin Aleksei Vladimirovich "Combined neural network classifiers of specific images", presented by a combined neural network image classification model with explicit recognition of textual information, are implemented and used in the commercial activities of «V Kontakte» LLC for classifying images containing textual information, which significantly increases the level of automation tasks of various kinds. Thus, the presented results have significant practical value.

Director of Information Technology and User Privacy Protection at «V Kontakte» LLC.



March 1, 2023




Shvets A. R.

**Act**  
**on the implementation of the results of the**  
**dissertation**  
**thesis of Samarin Aleksei Vladimirovich, «Combined**  
**neural network classifiers of specific images»**

The results of the dissertation thesis of Samarin Aleksei Vladimirovich "Combined neural network classifiers of specific images", presented by a combined neural network model for classifying images without explicit recognition of textual information, are implemented and used in the commercial activities of LLC «V Kontakte» in the system for restoring access to user accounts of the social network «VKontakte». The results are of significant practical value, as they increase the efficiency of the access recovery system by 2 times by automatically validating photos of identity cards provided in user applications.

Director of Information Technology and User Privacy  
Protection at «V Kontakte» LLC.

  
March 1, 2023



Shvets A. R.

---