ISSN 2076-5886

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Черкаський національний університет
імені Богдана Хмельницького

ВІСНИК ЧЕРКАСЬКОГО УНІВЕРСИТЕТУ

Серія ПРИКЛАДНА МАТЕМАТИКА. ІНФОРМАТИКА

Виходить 2 рази на рік

Заснований у березні 1997 року

№ 1 (2019)

ЧЕРКАСИ – 2019

Засновник, редакція, видавець і виготовлювач –
Черкаський національний університет імені Богдана Хмельницького
Свідоцтво про державну перереєстрацію КВ № 21396-11196Р від 26.06.2015

Матеріали «Вісника» присвячені прикладним і фундаментальним науковим дослідженням у галузі інформаційних технологій, моделювання систем, прикладних інформаційних систем в різних сферах діяльності, математичному моделюванню фізичних процесів, математичної фізики, інформаційних і обчислювальних технологій. У публікаціях досліджуються різні аспекти досліджень учених і фахівців в області обчислювальних мереж і систем, елементів і пристроїв обчислювальної техніки та систем управління, програмного забезпечення, Публікації в журналі охоплюють широкий спектр проблем, пов'язаних з прикладними і фундаментальними науковими дослідженнями в галузі інформаційних технологій.

Наукові статті збірника рекомендовані аспірантам, студентам, викладачам вищої школи, а також усім, хто цікавиться проблематикою прикладної математики та інформатики.

Журнал входить до переліку наукових фахових видань України з фізико-математичних, технічних наук (Наказ МОН України від 12 травня 2015 р. № 528).

Випуск № 1.2019 наукового журналу Вісник Черкаського університету, серія: «Прикладна математика. Інформатика» рекомендовано до друку та до поширення через мережу Інтернет Вченою радою Черкаського національного університету імені Богдана Хмельницького (протокол № 5 від 07.02.2019).

Журнал реферується Українським реферативним журналом "Джерело" (засновники: Інститут проблем реєстрації інформації НАН України та Національна бібліотека України імені В. І. Вернадського).

Адреса редакційної колегії:
18000, Черкаси, бульвар Шевченка, 81,
Черкаський національний університет ім. Б. Хмельницького,
кафедра прикладної математики та інформатики. Тел. (0472) 36-13-55
web-сайт: http://ami-ejournal.cdu.edu.ua/index.
e-mail: (herald_pm@ukr.net)

СЕКЦІЯ «ПРИКЛАДНА МАТЕМАТИКА»

UDK 519.24, 530.19, 330.46

DOI 10.31651/2076-5886-2019-1-3-16

PACS 02.50.Cw, 02.70.Rr, 03.65.−w, 89.65.Gh

SOLOVIEV Vladimir,
Head of the Department of Informatics and Applied Mathematics, Kryvyi Rih State Pedagogical University, Ukraine
e-mail: vnsoloviev2016@gmail.com
ORCID 0000-0002-4945-202X

SERDIUK Oleksandr,
Department of Informatics and Applied Mathematics, The Bohdan Khmelnytsky National University of Cherkasy, Ukraine
e-mail: serdyuk@ukr.net
ORCID 0000-0002-3919-4661

QUANTUM ECONOPHYSICAL PRECURSORS OF CRYPTOCURRENCY CRASHES

This article demonstrates the possibility of constructing indicators of critical and crash phenomena in the volatile market of cryptocurrency.

The possibility of constructing dynamic measures of complexity as quantum econophysical behaving in a proper way during actual pre-crash periods has been shown. This fact is used to build predictors of crashes and critical events phenomena on the examples of all the patterns recorded in the time series of the key cryptocurrency Bitcoin, the effectiveness of the proposed indicators-precursors of these falls has been identified. From positions, attained by modern theoretical physics the concept of economic Planck's constant has been proposed.

The theory on the economic dynamic time series related to the cryptocurrencies market has been approved. Then, combining the empirical cross-correlation matrix with the Random Matrix Theory, we mainly examine the statistical properties of cross-correlation coefficient, the evolution of the distribution of eigenvalues and corresponding eigenvectors of the global cryptocurrency market using the daily returns of cryptocurrencies price time series all over the world from 2013 to 2018.

The result has indicated that the largest eigenvalue reflects a collective effect of the whole market, and is very sensitive to the crash phenomena. It has been shown that both the introduced economic mass and the largest eigenvalue of the matrix of correlations can act like quantum indicators-predictors of falls in the market of cryptocurrencies.

Keywords: *Cryptocurrency, Bitcoin, complex system, measures of complexity, crash, critical events, complex networks, quantum econophysics, Heisenberg uncertainty principle, Random Matrix Theory, indicator-precursor.*

Introduction

The instability of global financial systems with regard to normal and natural disturbances of the modern market and the presence of poorly foreseeable financial crashes indicate, first of all, the crisis of the methodology of modeling, forecasting and interpretation of modern socio-economic realities. The doctrine of the unity of the scientific method states that for the study of events in socio-economic systems, the same methods and criteria as those used in the study of natural phenomena are applicable. Significant success has been achieved within the framework of interdisciplinary approaches and the theory of self-organization – synergetics. The modern paradigm of synergetics is a complex paradigm associated with the

possibility of direct numerical simulation of the processes of complex systems evolution, most of which have a network structure, or one way or another can be reduced to the network. The theory of complex networks studies the characteristics of networks, taking into account not only their topology, but also statistical properties, the distribution of weights of individual nodes and edges, the effects of dissemination of information, robustness, etc. [1-4].

Complex systems are systems consisting of a plurality of interacting agents possessing the ability to generate new qualities at the level of macroscopic collective behavior, the manifestation of which is the spontaneous formation of noticeable temporal, spatial, or functional structures. As simulation processes, the application of quantitative methods involves measurement procedures, where importance is given to complexity measures. I. Prigogine notes that the concepts of simplicity and complexity are relativized in the pluralism of the descriptions of languages, which also determines the plurality of approaches to the quantitative description of the complexity phenomenon [5]. Therefore, we will continue to study Prigogine's manifestations of the system complexity, using the current methods of quantitative analysis to determine the appropriate measures of complexity.

The key idea here is the hypothesis that the complexity of the system before the crashes and the actual periods of crashes must change. This should signal the corresponding degree of complexity if they are able to quantify certain patterns of a complex system. Significant advantage of the introduced measures is their dynamism, that is, the ability to monitor the change in time of the chosen measure and compare it with the corresponding dynamics of the output time series. This allowed us to compare the critical changes in the dynamics of the system, which is described by the time series, with the characteristic changes of concrete measures of complexity. It turned out that quantitative measures of complexity respond to critical changes in the dynamics of a complex system, which allows them to be used in the diagnostic process and prediction of future changes.

Cryptocurrency market is a complex, self-organized system, which in most cases can be considered either as a complex network of market agents, or as an integrated output signal of such a network – a time series, for example, prices of individual cryptocurrency. The research on cryptocurrency price fluctuations being carried out internationally is made more complex by the interplay due to many factors – including market supply and demand, the US dollar exchange rate, stock market state, the influence of crime and the shadow market, and fiat money regulator pressure – that introduces a high level of noise into the cryptocurrency data. Moreover, in the cryptocurrency market, to some extent, the blockchain technology is tested in general. Thus the cryptocurrency prices exhibit such complex volatility characteristics as nonlinearity and uncertainty, which are difficult to forecast and any results obtained are uncertain. Therefore, cryptocurrency price prediction remains a huge challenge.

Unfortunately, the existing nowadays classical econometric [6-8] and modern methods of prediction of crisis phenomena based on machine learning methods [9-16] do not have sufficient accuracy and reliability of prediction.

Thus, lack of reliable models of prediction of time series for the time being will update the construction of at least indicators which warn against possible critical phenomena or trade changes etc. This work is dedicated to the construction of such indicators – precursors based on the theory of complexity.

In this paper, we consider some of the informative measures of complexity and adapt them in order to study the critical and crash phenomena of cryptomarket.

The paper is structured as follows. Section 2 describes previous studies in these fields. Section 3 presents classification of crashes and critical events on the Bitcoin market during the entire period (16.07.2010 – 08.12.2018). In Section 4, new quantum indicators of critical and crash phenomena are introduced using the Heisenberg uncertainty principle and the Random Matrix Theory.

Analysis of previous studies

Throughout the existence of Bitcoin, its complexity became much larger. Crashes and critical events that took place on this market as well as the reasons that led to them, did not go unheeded. We determined that there are a lot of articles and papers on that topic which we will demonstrate.

Donier and Bouchaud [17] found that the market microstructure on Bitcoin exchanges can be used to anticipate illiquidity issues in the market, which lead to abrupt crashes. They investigate Bitcoin liquidity based on order book data and, out of this, accurately predict the size of price crashes.

F. Bariviera [18] demonstrates the dynamics of the intraday price of 12 cryptocurrencies. By using the complexity-entropy causality plane, authors discriminate three different dynamics in the data set. Another paper [19] compares the time-varying weak-form efficiency of Bitcoin prices in terms of US dollars (BTC/USD) and euro (BTC/EUR) at a high-frequency level by using Permutation Entropy. Their research shows that BTC/USD and BTC/EUR markets have been demonstrating more information at the intraday level since the beginning of 2016, and BTC/USD market has been slightly more efficient than BTC/EUR during the same period. And moreover, their research shows that with the higher frequency we have less price efficiency.

Some papers like this one [20] demonstrate how recurrence plots and measures of recurrence quantification analysis can be used to study significant changes in complex dynamical systems due to a change in control parameters, chaos-order as well as chaos-chaos transitions. Tiago Santos et al. [21] discuss how to model activity in online collaboration websites, such as Stock Exchange Question and Answering portals because the success of these websites critically depends on the content contributed by its users. In this paper, they represent user activity as time series and perform an initial analysis of these time series to obtain a better understanding of the underlying mechanisms that govern their creation. For this purpose nonlinear modeling via recurrence plots was used, which gives more granular study and deeper understanding of nonlinear dynamics of governing activity of time series and explaining the activity in online collaboration websites.

Taking to the account studies on network analysis we can notice different papers on this topic [22-24]. Di Francesco Maesa et al. [22] have performed on the users' graph inferred from the Bitcoin blockchain, dumped in December 2015, so after the occurrence of the exponential explosion in the number of transactions. Researchers first present the analysis assessing classical graph properties like densification, distance analysis, degree distribution, clustering coefficient, and several centrality measures. Then, they analyze properties strictly tied to the nature of Bitcoin, like rich-get-richer property, which measures the concentration of richness in the network. Alexandre Bovet et al. [23] analyzed the evolution of the network of Bitcoin transactions among users and built network-based indicators of Bitcoin bubbles.

In this article [24], authors consider the history of Bitcoin and transactions in it. Using this dataset, they reconstruct the transaction network among users and analyze changes in the structure of the subgraph induced by the most active users. Their approach is based on the unsupervised identification of important features of the time variation of the network. Applying the widely used method of principal component analysis to the matrix constructed from snapshots of the network at different times, they show how changes in the network accompany significant changes in the price of Bitcoin.

Separately, it is necessary to highlight the work of Didier Sornette [25, 26], who built a precursor of crashes based on the generation of so-called log-periodic oscillations by the pre-crashing market. However, the actual collapse point is still badly predicted.

Thus, construction of indicators – precursors of critical and crash phenomena in the cryptocurrency market remains relevant.

Data

Bitcoin, despite its uncertain future, continues to attract investors, crypto-enthusiasts, and researchers. Being historically proven, popular and widely used cryptocurrency for the whole existence of cryptocurrencies in general, Bitcoin began to produce a lot of news and speculation, which began to determine its future life. Similar discussions began to lead to different kinds of crashes, critical events, and bubbles, which professional investors and inexperienced users began to fear. Thus, we advanced into action and set the tasks:

(1). Classification of such bubbles, critical events and crashes.

(2). Construction of such indicators that will predict crashes, critical events in order to give investors and ordinary users the opportunity to work in this market.

At the moment, there are various research works on what crises and crashes are and how to classify such interruptions in the market of cryptocurrencies. Taking into account the experience of previous researchers [26-30], we have created our classification of such leaps and falls, relying on Bitcoin time series during the entire period (16.07.2010 – 08.12.2018) of verifiable fixed daily values of the Bitcoin price (BTC) (https://finance.yahoo.com/cryptocurrencies).

For our classification, crashes are short, time-localized drops, with strong losing of price per each day, which are formed as a result of the bubble. Critical events are those falls that could go on for a long period of time, and at the same time, they were not caused by a bubble. The bubble is an increasing in the price of the cryptocurrencies that could be caused by certain speculative moments. Therefore, according to our classification of the event with number (1, 3-6, 9-11, 14, 15) are the crashes that are preceded by the bubbles, all the rest - critical events. More detailed information about crises, crashes and their classification in accordance with these definitions is given in the Table 1.

Accordingly, during this period in the Bitcoin market, many crashes and critical events shook it. Thus, considering them, we emphasize 15 periods on Bitcoin time series, whose falling we predict by our indicators, relying on normalized returns and volatility, where normalized returns are calculated as

$$g(t) = \ln X(t+\Delta t) - \ln X(t) \cong \left[X(t+\Delta t) - X(t)\right] / X(t). \quad (1)$$

and volatility as $V_T(t) = \frac{1}{n}\sum_{t'=t}^{t+n-1} \left|g(t')\right|$. Besides, considering that $g(t)$ should be more than the $\pm 3\sigma$, where sigma is a mean square deviation.

Calculations were carried out within the framework of the algorithm of a moving window. For this purpose, the part of the time series (window), for which there were calculated measures of complexity, was selected, then the window was displaced along the time series in a one-day increment and the procedure repeated until all the studied series had exhausted. Further, comparing the dynamics of the actual time series and the corresponding measures of complexity, we can judge the characteristic changes in the dynamics of the behavior of complexity with changes in the cryptocurrency. If this or that measure of complexity behaves in a definite way for all periods of crashes, for example, decreases or increases during the pre-crashes period, then it can serve as an indicator or precursor of such a crashes phenomenon.

Calculations of measures of complexity were carried out both for the entire time series, and for a fragment of the time series localizing the crash. In the latter case, fragments of time series of the same length with fixed points of the onset of crashes or critical events were selected and the results of calculations of complexity measures were compared to verify the universality of the indicators.

Table 1

BTC Historical Corrections. List of Bitcoin major corrections ≥ 20% since June 2011

№	Name	Days in correction	Bitcoin High Price, $	Bitcoin Low Price, $	Decline, %	Decline, $
1	07.06.2011-10.06.2011	4	29.60	14.65	50	15.05
2	15.01.2012-16.02.2012	33	7.00	4.27	39	2.73
3	15.08.2012-18.08.2012	4	13.50	8.00	40	5.50
4	08.04.2013-15.04.2013	8	230.00	68.36	70	161.64
5	04.12.2013-18.12.2013	15	1237.66	540.97	56	696.69
6	05.02.2014-25.02.2014	21	904.52	135.77	85	768.75
7	12.11.2014-14.01.2015	64	432.02	164.91	62	267.11
8	11.07.2015-23.08.2015	44	310.44	211.42	32	99.02
9	09.11.2015-11.11.2015	3	380.22	304.70	20	75.52
10	18.06.2016-21.06.2016	4	761.03	590.55	22	170.48
11	04.01.2017-11.01.2017	8	1135.41	785.42	30	349.99
12	03.03.2017-24.03.2017	22	1283.30	939.70	27	343.60
13	10.06.2017-15.07.2017	36	2973.44	1914.08	36	1059.36
14	16.12.2017-22.12.2017	7	19345.49	13664.96	29	5680.53
15	13.11.2018-26.11.2018	14	6339.17	3784.59	40	2554.58

In the Figure 1 output Bitcoin time series, normalized returns $g(t)$, and volatility $V_T(t)$ calculated for the window size 100 are presented.

From Figure 1 we can see that during periods of crashes and critical events normalized profitability *g* increases considerably in some cases beyond the limits $\pm 3\sigma$. This indicates about deviation from the normal law of distribution, the presence of the "heavy tails" in the distribution *g*, characteristic of abnormal phenomena in the market. At the same time volatility also grows. These characteristics serve as indicators of critical and collapse phenomena as they react only at the moment of the above mentioned phenomena and don't give an opportunity to identify the corresponding abnormal phenomena in advance. In contrast, the indicators described below respond to critical and collapse phenomena in advance. It enables them to be used as indicators – precursors of such phenomena and in order to prevent them.

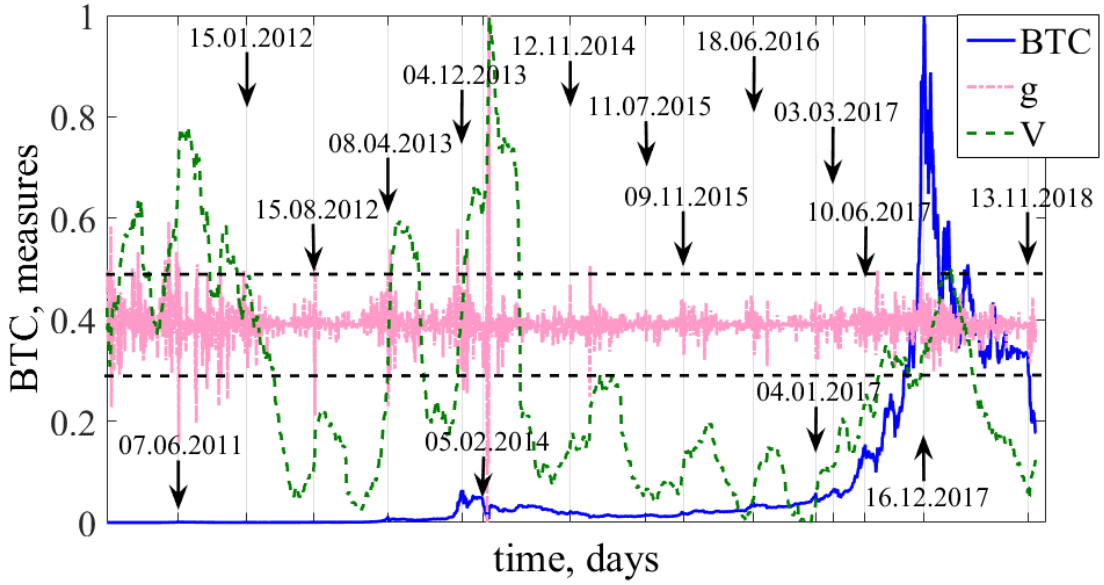


Fig. 1. The standardized dynamics, returns $g(t)$, and volatility $V_T(t)$ of BTC/USD daily values. Horizontal dotted lines indicate the $\pm 3\sigma$ borders. The arrows indicate the beginning of one of the crashes or the critical events

Quantum econophysics indicators

The attempts to create an adequate model of socio-economic critical events, which, as it has been historically proven, are almost permanent, were, are and will always be made. Actually, it is a super task impossible to solve. However, the potentially useful solutions, local in time or other socio-economic logistic coordinates, are possible. In fact, they have to be the object of interest for a real and effective economic science.

Econophysics is a young interdisciplinary scientific field, which developed and acquired its name at the end of the last century [31]. Quantum econophysics, a direction distinguished by the use of mathematical apparatus of quantum mechanics as well as its fundamental conceptual ideas and relativistic aspects, developed within its boundaries just a couple of years later, in the first decade of the 21st century [32-36].

According to classical physics, immediate values of physical quantities, which describe the system status, not only exist, but can also be exactly measured. Although non-relativistic quantum mechanics doesn't reject the existence of immediate values of classic physical quantities, it postulates that not all of them can be measured simultaneously (Heisenberg uncertainty ratio). Relativistic quantum mechanics denies the existence of immediate values for all kinds of physical quantities, and, therefore, the notion of system status seizes to be algoristic.

In this section, we will demonstrate the possibilities of quantum econophysics on the example of the application of the Heisenberg uncertainty principle and the Random Matrices Theory to the actual and debatable now market of cryptocurrencies.

Heisenberg uncertainty principle and economic analogues of basic physical quantities

In our paper [34] we have suggested a new paradigm of complex systems modeling based on the ideas of quantum as well as relativistic mechanics. It has been revealed that the use of quantum-mechanical analogies (such as the uncertainty principle, notion of the operator, and quantum measurement interpretation) can be applied to describing socio-economic processes. Methodological and philosophical analysis of fundamental physical

notions and constants, such as time, space and spatial coordinates, mass, Planck's constant, light velocity from the point of view of modern theoretical physics provides an opportunity to search of adequate and useful analogues in socio-economic phenomena and processes.

The Heisenberg uncertainty principle is one of the cornerstones of quantum mechanics. The modern version of the uncertainty principle, deals not with the precision of a measurement and the disturbance it introduces, but with the intrinsic uncertainty any quantum state must possess, regardless of what measurement is performed [35, 36]. Recently, the study of uncertainty relations in general has been a topic of growing interest, specifically in the setting of quantum information and quantum cryptography, where it is fundamental to the security of certain protocols [37-39].

To demonstrate it, let us use the known Heisenberg's uncertainty ratio which is the fundamental consequence of non-relativistic quantum mechanics axioms and appears to be (e.g. [40]):

$$\Delta x \cdot \Delta v \geq \frac{\hbar}{2m_0}, \tag{2}$$

where Δx and Δv are mean square deviations of x coordinate and velocity v corresponding to the particle with (rest) mass m_0, $\hbar$ – Planck's constant. Considering values Δx и Δv to be measurable when their product reaches its minimum, we derive (from (1)):

$$m_0 = \frac{\hbar}{2 \cdot \Delta x \cdot \Delta v}, \tag{3}$$

i.e. mass of the particle is conveyed via uncertainties of its coordinate and velocity – time derivative of the same coordinate.

Economic measurements are fundamentally relative, are local in time, space and other socio-economic coordinates, and can be carried out via consequent and/or parallel comparisons "here and now", "here and there", "yesterday and today", "a year ago and now" etc.

Due to these reasons constant monitoring, analysis, and time series prediction (time series imply data derived from the dynamics of stock indices, exchange rates, cryptocurrencies prices, spot prices and other socio-economic indicators) becomes relevant for evaluation of the state, tendencies, and perspectives of global, regional, and national economies.

Suppose there is a set of K time series, each of N samples, that correspond to the single distance T, with an equal minimal time step $\Delta t_{\min}$:

$$X_i(t_n),\ t_n = \Delta t_{\min} n;\ n = 0,\ 1,\ 2,\ \ldots N-1;\ i = 1,\ 2,\ \ldots K. \tag{4}$$

To bring all series to the unified and non-dimensional representation, accurate to the additive constant, we normalize them, having taken a natural logarithm of each term of the series. Then consider that every new series $x_i(t_n)$ is a one-dimensional trajectory of a certain fictitious or abstract particle numbered i, while its coordinate is registered after every time span $\Delta t_{\min}$, and evaluate mean square deviations of its coordinate and speed in some time window $\Delta T = \Delta N \cdot \Delta t_{\min} = \Delta N,\ 1 << \Delta N << N$. The «immediate» speed of i particle at the moment t_n is defined by the ratio:

$$v_i(t_n) = \frac{x_i(t_{n+1}) - x_i(t_n)}{\Delta t_{\min}} = \frac{1}{\Delta t_{\min}} \ln \frac{X_i(t_{n+1})}{X_i(t_n)} \tag{5}$$

with variance D_{v_i} and mean square deviation Δv_i.

Keeping an analogy with (1) after some transformations we can write an uncertainty ratio for this trajectory [40]:

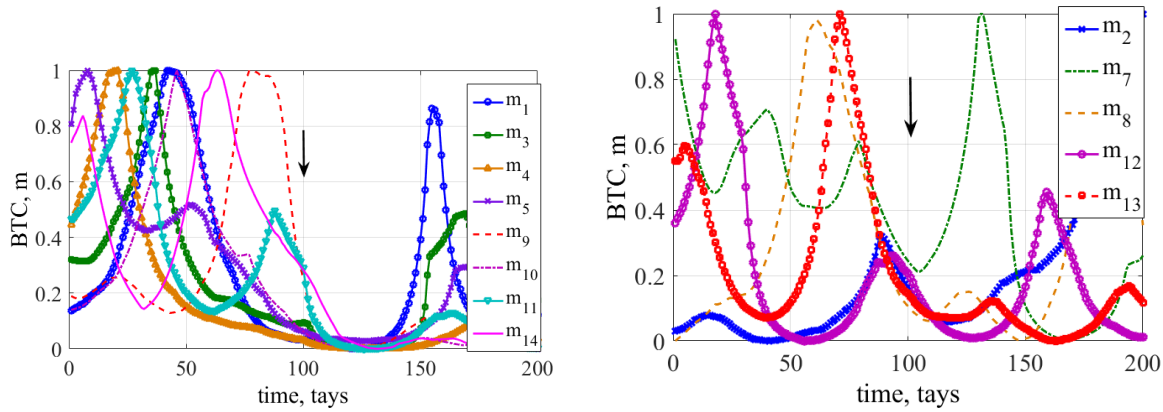
$$\frac{1}{\Delta t_{\min}}\left(<\ln^2\frac{X_i(t_{n+1})}{X_i(t_n)}>_{n,\Delta N}-\left(<\ln\frac{X_i(t_{n+1})}{X_i(t_n)}>_{n,\Delta N}\right)^2\right)\sim\frac{h}{m_i}, \quad (6)$$

where m_i – economic "mass" of an i series, h – value which comes as an economic Planck's constant.

Since the analogy with physical particle trajectory is merely formal, h value, unlike the physical Planck's constant $\hbar$, can, generally speaking, depend on the historical period of time, for which the series are taken, and the length of the averaging interval (e.g. economical processes are different in the time of crisis and recession), on the series number i etc. Whether this analogy is correct or not depends on particular series' properties.

In recent work [40], we tested the economic mass as an indicator of crisis phenomena on stock index data. In this work we will test the model for the cryptocurrency market on the example of the Bitcoin [41].

Obviously, there is a dynamic characteristic values m depending on the internal dynamics of the market. In times of crashes known marked by arrows in the Figures 2(a) and 2(b) mass m is significantly reduced in the pre-crash and pre-critical periods.



a) b)

Fig. 2. Dynamics of measure m for local crashes (a) and critical events (b)

Obviously, that the value of m remains a good indicator-precursor even in this case. Value m is considerably reduced before a special market condition. The market becomes more volatile and prone to changes.

The following method of quantum econophysics is borrowed from nuclear physicists and is called Random Matrix Theory.

Random Matrix Theory and quantum indicators-predictors

Random Matrix Theory (RMT) developed in this context the energy levels of complex nuclei, which the existing models failed to explain (Wigner, Dyson, Mehta, and others [44-46]). Deviations from the universal predictions of RMT identify system specific, nonrandom properties of the system under consideration, providing clues about the underlying interactions.

Unlike most physical systems, where one relates correlations between subunits to basic interactions, the underlying "interactions" for the stock market problem are not known. Here, we analyze cross correlations between stocks by applying concepts and methods of random matrix theory, developed in the context of complex quantum systems where the precise nature of the interactions between subunits are not known.

RMT has been applied extensively in studying multiple financial time series [47-53].

Special databases have been prepared, consisting of cryptocurrency time series for a certain period of time. The largest number of cryptocurrencies 1047 contained a base of 456 days from 31.12.2017 to 15.09.2018, and the smallest (24 cryptocurrencies) contained a base of 1567 days, respectively, from 04.08.2013 to 15.09.2018 (https://coinmarketcap.com/all/views/all/). In order to quantify correlations, we first calculate the logarithmic return (1) of the i cryptocurrencies price series over a time scale $\Delta t = 1$ day. We calculate the pairwise cross-correlation coefficients between any two cryptocurrencies returns time series. for the largest database, a graphical representation of the pair correlation field is shown in the Figure 3a. For comparison, a map of correlations of randomly mixed time series of the same length is shown in Figure 3b.

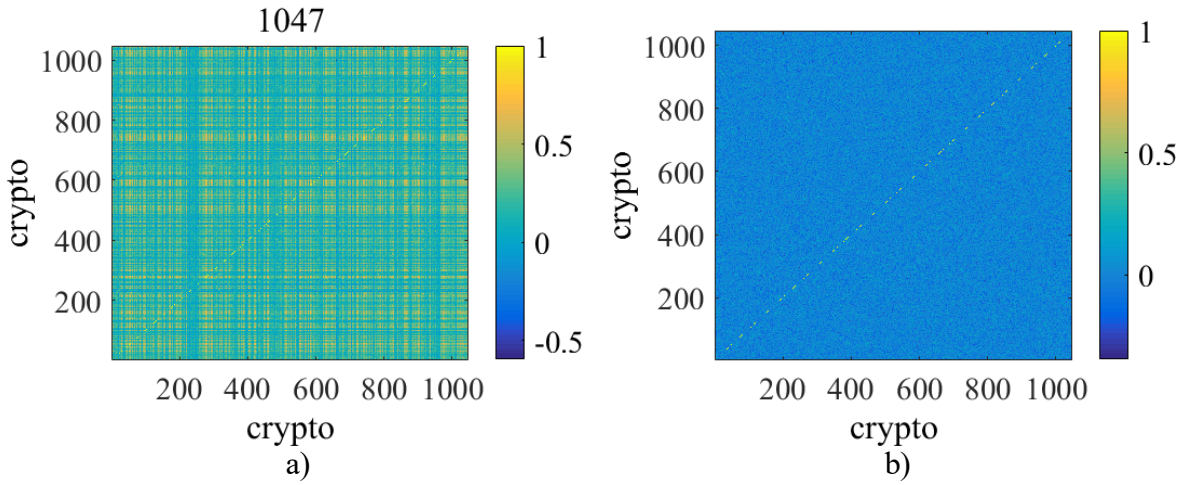


Fig. 3. Visualization of the field of correlations for the initial (a) and mixed (b) time series

For the correlation matrix C we can calculate its eigenvalues, $C = U\Lambda U^T$, where U denotes the eigenvectors, Λ is the eigenvalues of the correlation matrix, whose density $f_c(\lambda)$ is defined as follows, $f_c(\lambda) = \left(1/N\right) dn(\lambda)/d\lambda$. $n(\lambda)$ is the number of eigenvalues of C that are less than λ. In the limit $N \to \infty$, $T \to \infty$ and $Q = T/N \geq 1$ fixed, the probability density function $f_c(\lambda)$ of eigenvalues λ of the random correlation matrix M has a close form:

$$f_c(\lambda) = \frac{Q}{2\pi\sigma^2} \frac{\sqrt{(\lambda_{\max} - \lambda)(\lambda - \lambda_{\min})}}{\lambda} \tag{7}$$

with $\lambda \in [\lambda_{\min}, \lambda_{\max}]$, where $\lambda_{\min}^{\max}$ is given by $\lambda_{\min}^{\max} = \sigma^2\left(1 + 1/Q \pm 2\sqrt{1/Q}\right)$ and σ^2 is equal to the variance of the elements of matrix M.

We compute the eigenvalues of the correlation matrix C, $\lambda_{\max} = \lambda_1 > \lambda_2 > \ldots > \lambda_{15} = \lambda_{\min}$. The probability density functions (pdf) of paired correlation coefficients c_{ij} and eigenvalues λ_i for matrices of 132, 312, and 458 cryptocurrencies are presented in Figure 4. From Fig. 4a, it can be seen that the distribution functions for the paired correlation coefficients of the selected matrices differ significantly from the distribution function. described by the RMT. It can be seen that the crypto market has a significantly correlated, self-organized system (Fig. 4a) and the difference from the RMT of the case, the correlation coefficients exceed the value of 0.6-0.8 on “thick tails”. The distribution of the eigenvalues of the correlation matrix also differs markedly from the case of RMT In our case, only one-third of its own values refer to the RMT region.

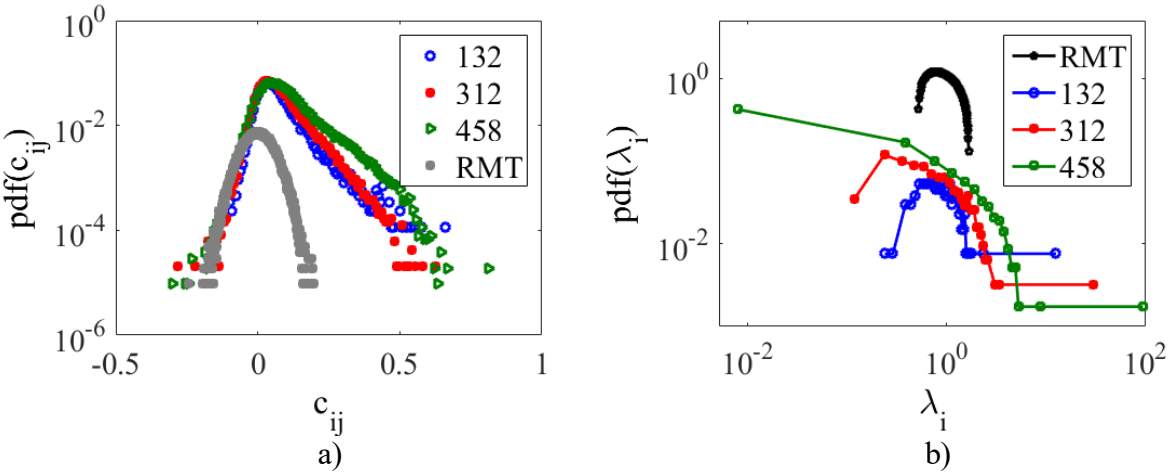


Fig. 4 Comparison of distributions of the pair correlation coefficients (a) and eigenvalues of the correlation matrix (b) with those for RMT

Eigenvectors correspond to the participation ratio PR and its inverse participation ratio IPR $I^k = \sum_{l=1}^{N} \left[u_l^k\right]^4$, where u_l^k, $l = 1,...,N$ are the components of the eigenvector u^k (Fig. 5a). So PR indicates the number of eigenvector components that contribute significantly to that eigenvector. More specifically, a low IPR indicates that cryptocurrency contribute more equally. In contrast, a large IPR would imply that the factor is driven by the dynamics of a small number of cryptocurrencies. The irregularity of the influence of the eigenvalues of the correlation matrix is determined by the absorption ratio (AR), which is a cumulative risk measure $AR_n = \sum_{k=1}^{n} \lambda_k \Big/ \sum_{k=1}^{N} \lambda_k$ and indicates which part of the overall variation is described from the total number N of eigenvalues.

In Figure 5, within the framework of the algorithm of a moving window, comparative calculations of the distribution function of eigenvalues (b), IPR (c) and some measures of complexity (d) are presented. The difference in dynamics is due to the peculiarities of non-random correlations between the time series of individual assets. Under the framework of Random Matrix Theory, if the eigenvalues of the real time series differ from the prediction of RMT, there must exists hidden economic information in those deviating eigenvalues. For cryptocurrencies markets, there are several deviating eigenvalues in which the largest eigenvalue $\lambda_{\max}$ reflects a collective effect of the whole market. As for PR the differences from RMT appear at large and small λ values and are similar to the Anderson quantum effect of localization [54]. Under crashes conditions, the states at the edges of the distributions of eigenvalues are delocalized, thus identifying the beginning of the crash. This is evidenced by the results presented in Figure 5 (c).

We find that both $\lambda_{\max}$ and $PR\lambda_{\max}$ have large values for periods containing the cryptomarket crashes and critical events. At the same time, their growth begins in the pre-crashes periods. Means, as well as the economic mass, they are quantum precursors of crashes and critical events phenomena.

Conclusions

Consequently, in this paper, we have shown that monitoring and prediction of possible critical changes on cryptocurrency is of paramount importance. As it has been shown by us, the theory of complex systems has a powerful toolkit of methods and models for creating effective indicators-precursors of crashes and critical phenomena. In this paper, we have

explored the possibility of using the quantum measures of complexity to detect dynamical changes in a complex time series. We have shown that the measures that have been used can indeed be effectively used to detect abnormal phenomena for the time series of Bitcoin.

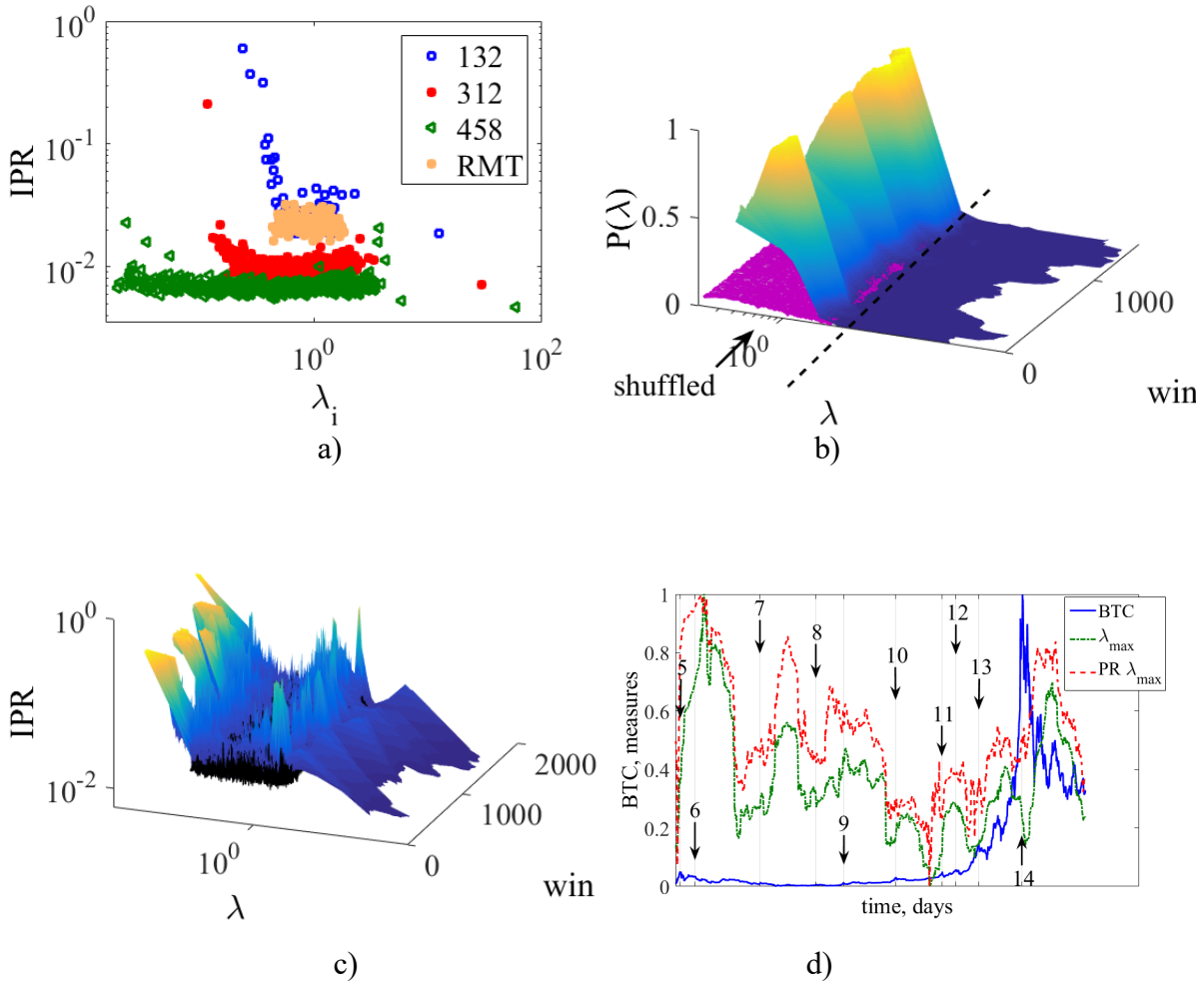


Fig. 5. Inverse participation ratio (a) and moving window dynamics of the eigenvalues distribution (b), IPR for the initial and mixed (or random) matrices (c). (d) quantum measures of complexity λ_{max} and its participation ratio. The numerics in the figure indicate the numbers of crashes and critical events in accordance with the table

We have shown that monitoring and prediction of possible critical changes on cryptocurrency is of paramount importance. As it has been shown by us, the quantum econophysics has a powerful toolkit of methods and models for creating effective indicators-precursors of crisis phenomena. In this paper, we have explored the possibility of using the Heisenberg uncertainty principle and random matrix theory to detect dynamical changes in a complex time series. We have shown that the economic mass m, and the largest eigenvalue λ_{max} may be effectively used to detect crisis phenomena for the cryptocurrencies time series. We have concluded though by emphasizing that the most attractive features of the m, λ_{max} and $PR\lambda_{max}$ namely its conceptual simplicity and computational efficiency make it an excellent candidate for a fast, robust, and useful screener and detector of unusual patterns in complex time series.

Bibliography:

1. Halvin, S., Cohen, R.: Complex networks. Structure, robustness and function. Cambridge University Press, New York (2010).
2. Albert, R., Barabasi, A.-L.: Statistical Mechanics of Complex Networks Rev. Mod. Phys. **74**, 47-97. (2002).
3. Newman, M., Watts D., Barabási A.-L.: The Structure and Dynamics of Networks. Princeton University Press, Princeton and Oxford (2006).
4. Newman, M. E. J.: The structure and function of complex networks. SIAM Reviews. **45**(2), 167-256 (2003)
5. Nikolis, G., Prigogine, I.: Exploring complexity. An introduction. W. H. Freeman and Company, New York (1989).
6. Andrews, B., Calder, M., Davis, R.: Maximumlikelihood estimation for α-stable autoregressive processes. Ann. Stat. **37**, 1946–1982 (2009)
7. Dassios, A., Li, L.: An Economic Bubble Model and Its First Passage Time. arXiv:1803.08160v1 [q-fin.MF] Last accessed 15 Mar 2018
8. Tarnopolski, M.: Modeling the price of Bitcoin with geometric fractional Brownian motion: a Monte Carlo approach. arXiv:1707.03746v3 [q-fin.CP] Last accessed 3 Aug 2017
9. Kodama, O., Pichl, L., Kaizoji, T.: Regime Change and Trend Prediction for Bitcoin Time Series Data. In: CBU International Conference on Innovations in Science and Education. pp 384-388. www.cbuni.cz, www.journals.cz, Prague, (2017). https://doi.org/10.12955/cbup.v5.954.
10. Shah, D., Zhang, K. Bayesian: Regression and Bitcoin. arXiv:1410.1231v1 [cs.AI] Last accessed 6 Oct 2014.
11. Chen, T., Guestrin, C. Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd international conference on knowledge discovery and data mining. pp. 785-794. ACM, San Francisco (2016)
12. Alessandretti, L., ElBahrawy, A., Aiello, L.M., Baronchelli, A.: Machine Learning the Cryptocurrency Market. arXiv:1805.08550v1 [physics.soc-ph] Last accessed 22 May 2018.
13. Guo, T., Antulov-Fantulin, N.: An experimental study of Bitcoin fluctuation using machine learning methods. arXiv:1802.04065v2 [stat.ML] Last accessed 12 Jun 2018.
14. Albuquerque, Y., de Sá, J., Padula, A., Montenegro, M.: The best of two worlds: Forecasting high frequency volatility for cryptocurrencies and traditional currencies with Support Vector Regression. Expert Systems With Applications **97,** 177–192. (2018) https://doi.org/10.1016/j.eswa.2017.12.004.
15. Wang, M., Zhao, L., Du, R., Wang, C., Chen, L., Tian, L., Stanley, H.E.: A novel hybrid method of forecasting crude oil prices using complex network science and artificial intelligence algorithms. Applied Energy **220**, 480-495 (2018). https://doi.org/10.1016/j.apenergy.2018.03.148.
16. Kennis, M.: A Multi-channel online discourse as an indicator for Bitcoin price and volume. arXiv:1811.03146v1 [q-fin.ST] Last accessed 6 Nov 2018.
17. Donier, J., Bouchaud J.P: Why do markets crash? Bitcoin data offers unprecedented insights. PLoS ONE **10**(10), 1-11 (2015) https://doi:10.1371/journal.pone.0139356
18. Bariviera, F. A., Zunino, L., Rosso A. O.: An analysis of high-frequency cryptocurrencies price dynamics using permutation-information-theory quantifiers. Chaos **28**(7), 07551 (2018). https://doi: 10.1063/1.5027153
19. Senroy, A.: The inefficiency of Bitcoin revisited: A high-frequency analysis with alternative currencies. Finance Research Letters (2018). https://doi:10.1016/j.frl.2018.04.002
20. Marwan, N., Schinkel, S., Kurths, J.: Recurrence Plots 25 years later - gaining confidence in dynamical transitions. Europhysics Letters **101**(2) 20007 (2013). https://doi: 10.1209/0295-5075/101/20007
21. Santos, T., Walk, S., Helic, D.: Nonlinear Characterization of Activity Dynamics in Online Collaboration Websites. In: Proceedings of the 26th International Conference on World Wide Web Companion, pp. 1567-1572. WWW '17 Companion, Australia (2017). https://doi:10.1145/3041021.3051117
22. Di Francesco Maesa, D., Marino, A. & Ricci, L.: Int J Data Sci Anal. **6**(1), 63-80 (2018). https://doi: 10.1007/s41060-017-0074-x
23. Bovet, A., Campajola, C., Lazo, J.F. et al.: Network-based indicators of Bitcoin bubbles. arXiv:1805.04460v1 [physics.soc-ph] Last accessed 11 May 2018
24. Kondor, D., Csabai, I., Szüle, J., Pósfai, M., Vattay, G.: Infferring the interplay of network structure and market effects in Bitcoin. New J. Phys. **16**, 125003 (2014). doi:10.1088/1367-2630/16/12/125003
25. Wheatley, S., Sornette, D., Huber, T. et al.: Are Bitcoin Bubbles Predictable? Combining a Generalized Metcalfe's Law and the LPPLS Model. arXiv:1803.05663v1 [econ.EM] Last accessed 15 September 2018.
26. Gerlach, J-C., Demos, G., Sornette, D.: Dissection of Bitcoin's Multiscale Bubble History from January 2012 to February 2018. arXiv:1804.06261v2 [econ.EM] Last accessed 15 September 2018
27. Soloviev, V., Belinskiy, A.: Methods of nonlinear dynamics and the construction of cryptocurrency crisis phenomena precursors. arXiv:1807.05837v1 [q-fin.ST] Last accessed 30 June 2018
28. Casey M. B.: Speculative Bitcoin Adoption/Price Theory, https://medium.com/@mcasey0827/speculative-Bitcoin-adoption-price-theory-2eed48ecf7da. Last accessed 25 September 2018

29. McComb, K.: Bitcoin Crash: Analysis of 8 Historical Crashes and What's Next, https://blog.purse.io/Bitcoin-crash-e112ee42c0b5. Last accessed 25 September 2018
30. Amadeo K.: Stock Market Corrections Versus Crashes And How to Protect Yourself: How You Can Tell If It's a Correction or a Crash, https://www.thebalance.com/stock-market-correction-3305863. Last accessed 25 September 2018
31. Mantegna, R. N., Stanley, H. E.: An Introduction to Econophysics: Correlations and Complexity in Finance. Cambridge Univ. Press, Cambridge UK (2000)
32. Maslov, V.P.: Econophysics and quantum statistics". Mathematical Notes **72**, 811-818 (2002)
33. Hidalgo, E.G.: Quantum Econophysics. arXiv:physics/0609245v1 [physics.soc-ph] Last accessed 15 September 2018.
34. Saptsin, V., Soloviev, V.: Relativistic quantum econophysics - new paradigms in complex systems modelling. arXiv:0907.1142v1 [physics.soc-ph] Last accessed 25 September 2018
35. Colangelo, G., Clurana, F.M., Blanchet, L.C., Sewell, R.J., Mitchell, M.W.: Simultaneous tracking of spin angle and amplitude beyond classical limits. Nature **543**, 525-528 (2017)
36. Rodriguez, E.B., Aguilar, L.M.A.: Disturbance-Disturbance uncertainty relation: The statistical distinguishability of quantum states determines disturbance. Scientific Reports **8**, 1-10 (2018)
37. Rozema, L.A., Darabi, A., Mahler, D.H., Hayat, A., Soudagar, Y., Steinberg, A.M.: Violation of Heisenberg's Measurement-Disturbance Relationship by Weak Measurements. Phys. Rev. Lett. **109**, 100404 (2012)
38. Prevedel, R., Hamel, D. R., Colbeck, R., Fisher, K., Resch, K. J.: Experimental investigation of the uncertainty principle in the presence of quantum memory. Nature Phys. **7**(29), 757-761 (2011).
39. Berta, M., Christandl, M., Colbeck, R., Renes, J., Renner, R.: The Uncertainty Principle in the Presence of Quantum Memory. Nature Phys. **6**(9), 659-662 (2010)
40. Landau, L.D., Lifshitis, E.M.: The classical theory of fields. Course of theoretical physics. Butterworth-Heinemann, Oxford, England (1975)
41. Soloviev, V., Saptsin, V.: Heisenberg uncertainty principle and economic analogues of basic physical quantities, arXiv:1111.5289v1 [physics.gen-ph] Last accessed 15 September 2018
42. Soloviev, V.N., Romanenko, Y.V.: Economic analog of Heisenberg uncertainly principle and financial crisis: In: 20-th International conference SAIT 2017, pp. 32-33. ESC "IASA" NTUU "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine (2017)
43. Soloviev, V.N., Romanenko, Y.V.: Economic analog of Heisenberg uncertainly principle and financial crisis", In: 20-th International conference SAIT 2018, pp. 33-34. ESC "IASA" NTUU "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine (2018)
44. Wigner, E.P.: On a class of analytic functions from the quantum theory of collisions", Ann. Math. **53**, 36-47 (1951).
45. Dyson, F.J.: Statistical Theory of the Energy Levels of Complex Systems. Journal of Mathematical Physics **3**, 140-156 (1962).
46. Mehta, L.M.: Random Matrices, Academic Press, San Diego (1991)
47. Laloux, L., Cizeau, P., Bouchaud, J.-P., Potters, M.: Noise dressing of financial correlation matrices. Phys. Rev. Lett. **83**, 1467–1470 (1999).
48. Plerou, V., Gopikrishnan, P., Rosenow, B., Amaral, L. A. N., Guhr, T., Stanley, H. E:. Random matrix approach to cross correlations in financial data. Phys. Rev. E **65**, 066126 (2002).
49. Shen, J., Zheng, B.: Cross-correlation in financial dynamics, EPL (Europhys. Lett.) **86**, 48005 (2009).
50. Jiang, S., Guo, J., Yang, C., Tian, L.: Random Matrix Analysis of Cross-correlation in Energy Market of Shanxi, China. International Journal of Nonlinear Science **23**(2), 96-101 (2017).
51. Urama, T,C., Ezepue, P.O., Nnanwa, C.P.: Analysis of Cross-Correlations in Emerging Markets Using Random Matrix Theory. Journal of Mathematical Finance **7**, 291-307 (2017).
52. Pharasi, H.K., Sharma, K., Chakraborti, A., Seligman, T.H. Complex market dynamics in the light of random matrix theory, arXiv:1809.07100v2 [q-fin.ST] 24 Sep 2018. Last accessed 15 September 2018
53. Stosic, D., Stosic, D., Ludermir, T.B., Stosic, T. Collective behavior of cryptocurrency price changes. Physica A: Statistical Mechanics and its Applications **507**, 499-509 (2018)
54. Anderson, P., W.: Absence of Diffusion in Certain Random Lattices. Phys. Rev. **109,** 1492 (1958)

СОЛОВЙОВ Володимир Миколайович,
завідувач кафедри інформатики та прикладної математики, Криворізький державний педагогічний університет, м. Кривий Ріг Дніпропетровської області.
СЕРДЮК Олександр Анатолійович,
старший викладач кафедри прикладної математики та інформатики, Черкаський національний університет імені Богдана Хмельницького, м. Черкаси.

ПЕРЕДВІСНИКИ КРАХІВ КРИПТОВАЛЮТ НА ОСНОВІ ПОКАЗНИКІВ КВАНТУМНОЇ ЕКОНОФІЗИКИ

Анотація. Вступ. *Нестабільність глобальних фінансових систем та наявність погано передбачуваних крахів на фінансових ринках свідчать про кризу методології моделювання, прогнозування та інтерпретації сучасних соціально-економічних реалій. Головною причиною цього є висока складність економічних систем, утворених з великої кількості взаємодіючих агентів, здатних генерувати нові властивості на рівні макроскопічної колективної поведінки, проявом якої є мимовільне утворення помітних часових, просторових чи функціональних структур. Одним з підходів до кількісного вимірювання складності є методи, що опираються на аналіз проявів складності системи І. Пригожина. Ключовою ідеєю тут є гіпотеза про зміну складності системи до та у період критичного явища. Ринок криптовалют – приклад економічних систем, складних для прогнозування. Це самоорганізована система, яку у більшості випадків можна розглядати або як складну мережу ринкових агентів, або як інтегрований вихідний сигнал такої мережі – часовий ряд, наприклад, ціни окремої криптовалюти. Дослідження коливань цін на криптовалюту ускладнюється завдяки наявності багатьох чинників – включаючи ринкові попит та пропозицію, обмінний курс долара США, стан фондового ринку, вплив злочинності, тіньовий ринок тощо, які вносять високий рівень шуму у дані. За рахунок цього ціни на криптовалюту демонструють такі складні важко передбачувані характеристики як волатильність, нелінійність та невизначеність.*

Метою статті *є розгляд деяких інформативних мір складності та адаптація їх для вивчення критичних та кризових явищ на ринку криптовалют.*

Проведення дослідження. *Для дослідження використано часовий ряд курсу Біткойна, взятого за період з 16.07.2010 р. по 08.12.2018 р., а також часові ряди курсів 1047 криптовалют, взятих за період з 31.12.2017 р. по 15.09.2018 р. На основі часового ряду Біткойна досліджувалась поведінка показника економічної маси до та під час критичного явища на крипторинку. Система криптовалют досліджувалась з використанням інструментів теорії випадкових матриць. Аналізувалась як безпосередньо сама матриця крос-кореляцій агентів економічної системи, так і її похідні, а саме – розподіл власних значень та власні вектори матриці крос-кореляцій.*

Результати та висновки. *Моніторинг та прогнозування можливих критичних змін на ринку криптовалюти мають першочергове значення. У статті досліджено можливість використання квантових мір складності для виявлення динамічних змін у складному часовому ряді. Показано, що застосовані заходи дійсно можуть бути ефективно використані для виявлення аномальних явищ для часових рядів криптовалют. На основі часового ряду Біткойна показано, що моніторинг та прогнозування можливих критичних змін криптовалют є першочерговим завдання аналізу. Досліджено можливість використання принципу невизначеності Гейзенберга та теорії випадкових матриць для виявлення динамічних змін у складному часовому ряді. Показано, що економічна маса та найбільше власне значення матриці крос-кореляцій можуть ефективно використовуватися для виявлення кризових явищ. Саме економічна маса завдяки концептуальній простоті поняття та обчислювальній ефективності є відмінним кандидатом для швидкого та надійного моніторингу незвичайної поведінки економічної системи.*

Ключові слова: *криптовалюта, біткойн, складна система, міри складності, аварії, критичні події, складні мережі, квантова econофізика, принцип невизначеності Гейзенберга, теорія випадкової матриці, індикатор-попередник.*

Одержано редакцією 12.06.2018
Прийнято до публікації 17.10.2018

УДК 519.6:004.8

DOI 10.31651/2076-5886-2019-1-16-26

ГОЛОВНЯ Борис Петрович,
доктор технических наук, заведующий кафедрой прикладной математики и информатики Черкасского национального

PACS 02.60.-x, 02.60.Pn, 02.70.Wz

университета имени Богдана Хмельницкого
e-mail: bpgolovnya@gmail.com
ORCID 0000-0002-9242-3937

УПРОЩЕННЫЕ МОДЕЛИ ТУРБУЛЕНТНОСТИ ДЛЯ РАСЧЕТА ТУРБУЛЕНТНОГО ТЕПЛО И МАССОПЕРЕНОСА

В работе, на основе модели турбулентности, разработанной автором, строится набор упрощенных моделей ASM типа. Показано, что во многих случаях эти модели можно применять вместо полных моделей турбулентности. Получены ограничения применимости этих моделей.

Ключевые слова: *моделирование турбулентности, энергия турбулентности.*

Постановка задачи

Модели турбулентности, содержащие уравнения переноса всех турбулентных напряжений и тепловых потоков наиболее универсальны но содержат очень много дифференциальных уравнений. В частности, в двумерном случае система уравнений модели содержит 10 дифференциальных уравнений. Естественно, что подобные системы очень неудобны в эксплуатации. Методы упрощения подобных систем известны, результат упрощения называется ASM моделью. К сожалению, ASM модели построенные на основе известных моделей обладают очень слабыми вычислительными возможностями.

В работе рассматриваются возможные упрощения модели турбулентности предложено автором в [1]-[3].

Цель статьи

Получить модель типа ASM, обладающую приемлемыми вычислительными возможностями.

Научная новизна

Научная новизна работы состоит в том, что надежные ASM модели в настоящее время в литературе неизвестны.

Методы решения

Задачу упрощения моделей турбулентности можно поставить двумя способами.

Во первых. Во многих технически важных случаях в течении существует подобие распределения температуры и средней скорости, подобие переноса k_0 и $\overline{t_0^2}$ и т.д. Примером может служить течение в пограничном слое, течение в каналах и трубах и т.д. Но тогда, воспользовавшись эти подобием, можно попытаться упростить основные уравнения переноса, т.е. удалить из уравнений какие-либо слагаемые или даже свести уравнений к простым алгебраическим соотношениям. Общность модельных уравнений при этом снижается, но зато снижаются и трудозатраты на решение конкретных задач. Понятно, что применимость упрощения для решения какой-либо задачи должна предварительно проверяться.

Во вторых. Полученные упрощенные соотношения могут оказаться неприменимыми для использования в расчетной системе уравнений. Но, тем не менее, их вполне можно использовать для проведения оценочных расчетов. Здесь требования, предъявляемые к соотношениям, менее строги, но все равно проверять их применимость хотя бы простейшими расчетами необходимо.

1. Упрощенная модель переноса тепловых потоков при числах Pr≥1

Упрощению будет подвергаться модель (1)–(13) – полная модель переноса турбулентных напряжений и тепловых потоков.

$$\frac{Dk_0}{Dt} = f_0 \frac{\partial}{\partial x_k}\left(\nu + D_k\right)\frac{\partial k_0}{\partial x_k} + f_0 P - \varepsilon_0, \tag{1}$$

$$\frac{D\varepsilon_0}{Dt} = f_0 \frac{\partial}{\partial x_k}\left(\nu + \frac{D_k}{C_\varepsilon}\right)\frac{\partial \varepsilon_0}{\partial x_k} + \frac{\varepsilon_0}{k_0}\left(C_1 f_0 P - C_2 \varepsilon_0\right), \tag{2}$$

$$\frac{D\overline{u_{i0}^2}}{Dt} = f_0 \frac{\partial}{\partial x_k}\left(\nu + D_k\right)\frac{\partial \overline{u_{i0}^2}}{\partial x_k} + f_0\left(P_{ii} - C_a \frac{\varepsilon_0}{k_0}\left(\overline{u_{i0}^2} - \frac{2}{3}k_0\right)\right) - \varepsilon_{ui0}, \tag{3}$$

$$\begin{aligned}\frac{D\varepsilon_{ui0}}{Dt} &= f_0 \frac{\partial}{\partial x_k}\left(\nu + \frac{D_k}{C_\varepsilon}\right)\frac{\partial \varepsilon_{ui0}}{\partial x_k} + \\ &+ \frac{\varepsilon_0}{k_0}\left(C_1 f_0\left(P_{ii} - C_a \frac{\varepsilon_0}{k_0}\left(\overline{u_{i0}^2} - \frac{2}{3}k_0\right)\right) - C_2 \varepsilon_{ui0}\right),\end{aligned} \tag{4}$$

$$\frac{D\overline{u_{i0}u_{j0}}}{Dt} = f_0 \frac{\partial}{\partial x_k}\left(\nu + D_k\right)\frac{\partial \overline{u_{i0}u_{j0}}}{\partial x_k} + f_0 P_{ij} - C_a \frac{\varepsilon_0}{k_0}\overline{u_{i0}u_{j0}}. \tag{5}$$

$$D_k = C_{Diff} f_0 \frac{k_0}{\varepsilon_0}\overline{u_{0k}^2}, \tag{6}$$

$$P_{ij} \equiv -\overline{u_{0i}u_{0k}}\frac{\partial U_j}{\partial x_k} - \overline{u_{0j}u_{0k}}\frac{\partial U_i}{\partial x_k}, \quad P = 0.5\sum P_{ii}.$$

$$f_0 = \left(1 - \exp\left(-\frac{\mathrm{Re}_{y0}}{5.5}\right)\right)\left(1 - \exp\left(-2.4\frac{x_2}{L_{\varepsilon 0}}\right)\right), \tag{7}$$

$$\mathrm{Re}_{y0} = \frac{\sqrt{k_0}x_2}{\nu}, \quad L_{\varepsilon 0} = \frac{k_0^{3/2}}{\varepsilon_0},$$

$$\frac{D\overline{t_0^2}}{Dt} = f_{0-t}\frac{\partial}{\partial x_i}\left(\alpha + D_{tk}\right)\frac{\partial \overline{t_0^2}}{\partial x_k} + 2f_{0-t}P_{t2} - 2\left(\varepsilon_{t0} + \varepsilon_{t0-add}\right), \tag{8}$$

$$\frac{D\varepsilon_{t0}}{Dt} = f_{0-t}\frac{\partial}{\partial x_1}\left(\alpha + \frac{D_{tk}}{C_{\varepsilon t}}\right)\frac{\partial \varepsilon_{t0}}{\partial x_k} + \frac{\varepsilon_{t0} + \varepsilon_{t0-add}}{0.5\overline{t_0^2}}\left(C_{t1}f_{0-t}P_{t2} - C_{t2}\varepsilon_{t0}\right), \tag{9}$$

$$\frac{D\overline{u_{0k}t_0}}{Dt} = f_{0-t}\frac{\partial}{\partial x_i}\left(\frac{\nu + \alpha}{2} + D_{tk}\right)\frac{\partial \overline{u_{0k}t_0}}{\partial x_k} + f_{0-t}P_{uk-t} - C_{at}\frac{\varepsilon_{t0} + \varepsilon_{t0-add}}{0.5\overline{t_0^2}}\overline{u_{0k}t_0} \tag{10}$$

$$D_{tk} = C_{Diff} f_{0-t}\frac{0.5\overline{t_0^2}}{\left(\varepsilon_{t0} + \varepsilon_{t0-add}\right)}\overline{u_{0k}^2}, \tag{11}$$

$$P_{uk-t} = -\overline{u_{0k}u_{0j}}\frac{\partial T}{\partial x_j} - \overline{u_{0j}t_0}\frac{\partial U_i}{\partial x_j} - \beta g_i \overline{t_0^2}, \quad P_{t2} = -\overline{u_{0i}t_0}\frac{\partial T}{\partial x_i},$$

$$f_{0-t} = \left(1 - \exp\left(-R_0\frac{\mathrm{Re}_{y0}}{5.5}\right)\right)\left(1 - \exp\left(-2.4\frac{y}{L_{\varepsilon 0}}\right)\right), \tag{12}$$

$$R_0 = \left(\frac{k_0}{\varepsilon_0}\Big/\frac{0.5\overline{t_0^2}}{\varepsilon_{t0} + \varepsilon_{t0-add}}\right). \tag{13}$$

В случае Pr≥1 уравнения (8) – (9) представляющие собой модель $\overline{t^2}-\varepsilon_t$ типа, допускают значительное упрощение. Смысл его состоит в следующем. Известно, что температурные пульсации создаются в жидкости в результате захвата турбулентными вихрями более нагретых слоев жидкости и переноса их в менее нагретые слои и наоборот. Но, вследствие того, что Pr>1, вихри, как кинематические образования, должны распадаться быстрей, чем температурные пульсации будут устранены теплопроводностью. С точки зрения моделирования турбулентности его можно трактовать следующим образом: турбулентность забирает у осредненного течения тепловой энергии больше, чем может диссипировать, или, другими словами, на поддержание пульсаций используется не вся энергия, описываемая генерационным слагаемым. Эта проблема была разрешена с помощью введения в модель дополнительной диссипации $\varepsilon_{t0\text{-}add}$

$$\varepsilon_{t0-add}=\max\left(0.5\overline{t_0^2}\frac{\varepsilon_0}{k_0}-\varepsilon_{t0},0\right).$$

Из приведенного рассуждения можно заключить, что при числах Pr≥1 мы всегда будем получать

$$0.5\overline{t_0^2}\Big/\left(\varepsilon_{t0}+\varepsilon_{t0-add}\right)=k_0/\varepsilon_0\ . \qquad (14)$$

Из (14) следует, что в этом случае R_0 в выражении (13) равно единице, т.е. вместо функции $f_{0\text{-}t}$ (12) можно использовать функцию f_0 (7). Тогда вместо коэффициента диффузии (11) можно использовать коэффициент (6). В итоге вместо уравнения (10) можно использовать уравнение

$$\frac{D\overline{u_{0k}t_0}}{Dt}=f_0\frac{\partial}{\partial x_i}\left(\frac{\nu+\alpha}{2}+D_k\right)\frac{\partial\overline{u_{0k}t_0}}{\partial x_k}+f_0P_{uk-t}-C_{at}\frac{\varepsilon_0}{k_0}\overline{u_{0k}t_0} \qquad (15)$$

Более того, в случае течения без массовых сил уравнения переноса $\overline{t_0^2}$ и $\boldsymbol{\varepsilon}_{t0}$ из модели могут быть удалены.

1.1. Область применимости упрощений

Модель (1)-(7), (15) проверялась расчетами развитых турбулентных течений в пограничном слое и канале при Pr≥1. На рис. 1 приведены результаты расчетов теплообмена в турбулентном пограничном слое с Q_{wall}=const по упрощенной модели. Сопоставление расчетов с работой Жукаускаса [120] аппроксимацией $0.5C_f/St=0.93+12.5\sqrt{0.5C_f}\left(\mathrm{Pr}^{2/3}-1\right)$ показывает хорошую точность полученных результатов. Отметим также, что хотя приведенное упрощение в полной мере применимо только к диапазону Pr≥1, решение для Pr=0.7 также оказывается вполне удовлетворительным.

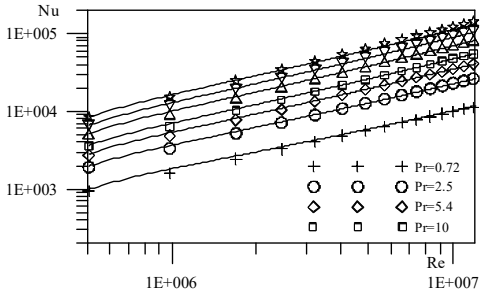


Рис.1. Расчет теплообмена в пограничном слое при вынужденной конвекции по упрощенной модели (символы - Жукаускас).

Упрощенная модель удовлетворительно рассчитывает не только теплообмен, а и осредненные и пульсационные параметры течения.

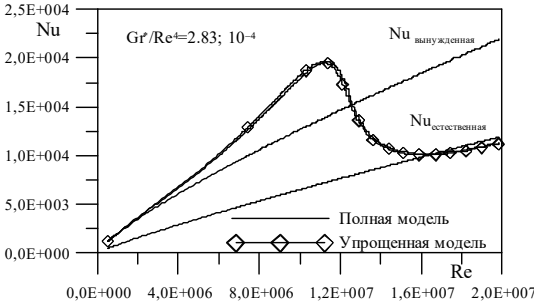


Рис. 2. Расчет смешанной конвекции на вертикальной поверхности по двум моделям.

О неприменимости упрощения. Очевидно, что подобие полей пульсационных параметров может существовать только в случае хоть

какого-то подобия полей средних параметров. Расчеты течения за обратной ступенью показывают, что в застойной области поля средней скорости и температуры отличаются очень значительно. Так в частности в застойной области существуют максимальные/минимальные значения скоростей. В то же время поле температуры в застойной зоне согласно принципу максимума не может иметь экстремальных точек. Расчеты показывают, что в этом случае даже при Pr=1 мы будем иметь $0.5\overline{t_0^2}/\varepsilon_{t0} < k_0/\varepsilon_0$, т.е. соотношение (14) здесь неприменимо.

2. Упрощенные соотношения для расчета $\overline{t_0^2}$, $\overline{u_{0i}t_0}$ и $\overline{u_{0i}u_{0j}}$ (i≠j)

Стандартная методика получения упрощенных соотношений для расчета $\overline{t_0^2}$, $\overline{u_{0i}t_0}$ и $\overline{u_{0i}u_{0j}}$ (i≠j) базируется либо на основе предположения о пропорциональности переноса указанных корреляций переносу энергии турбулентности k, либо на основе гипотезы равновесности. Как показали тестовые расчеты, выражения, построенные на основе гипотезы равновесности и гипотезы пропорциональности переноса корреляций переносу k, имеют примерно одинаковые свойства, но выражения, построенные на гипотезе равновесности, выглядят проще.

Применение гипотезы равновесности к уравнению (8) дает соотношение $f_{0-t}P_{t2} = \left(\varepsilon_{t0} + \varepsilon_{t0-add}\right)$. В разделе 1 показано, что при числах Pr≥1 выполняется соотношение (14) – $0.5\overline{t_0^2}/\left(\varepsilon_{t0} + \varepsilon_{t0-add}\right) = k_0/\varepsilon_0$. Из этого соотношения вычисляем диссипацию. Там же показано, что в этом случае вместо функции $f_{0\text{-}t}$ можно использовать функцию f_0. В итоге получаем алгебраическое соотношение

$$\overline{t_0^2} = 2f_0P_{t2}\frac{k_0}{\varepsilon_0}. \tag{16}$$

Соотношения для расчета $\overline{u_{0i}t_0}$ и $\overline{u_{0i}u_{0j}}$ (i≠j) также получаем на основе гипотезы равновесности. Так, в частности, для $\overline{u_{0i}u_{0j}}$ (i≠j) имеем

$$\overline{u_{0i}u_{0j}} = \frac{f_0}{C_a}\frac{k_0}{\varepsilon_0}P_{ij}. \tag{17}$$

Тестовые расчеты показали, что выражение (17) может приводить к неустойчивости вычислений. В связи с этим соотношение (17) было заменено на обыкновенное дифференциальное уравнение

$$\frac{d\overline{u_{0i}u_{0j}}}{dt} = f_0P_{ij} - C_a\frac{\varepsilon_0}{k_0}\overline{u_{0i}u_{0j}}. \tag{18}$$

Алгоритм решения этого уравнения простейшим методом Эйлера выглядит следующим образом (см. выражение (19))

$$\overline{u_{0i}u_{0j}}^{k+1} = \left(\frac{\overline{u_{0i}u_{0j}}^{k}}{\Delta t} + f_0P_{ij}^k\right)\Bigg/\left(C_a\left(\frac{\varepsilon_0}{k_0}\right)^k + \frac{1}{\Delta t}\right). \tag{19}$$

При расчетах параболизованных течений верхний индекс k в выражении (19) означает величины, вычисленные на k-ом шаге по продольной координате.

Аналогичным образом получаем обыкновенное дифференциальное уравнение для $\overline{u_{0i}t_0}$

$$\frac{d\overline{u_{0i}t_0}}{dt} = -f_0P_{it} - C_a\frac{\varepsilon_0}{k_0}\overline{u_{0i}t_0}. \tag{20}$$

2.1. Область применимости упрощений

Тестовые расчеты показывают, что формулы для $\overline{u_{0i}u_{0j}}$ (i≠j) (18), (19) вполне применимы для расчетов развитых течений без застойных зон, т.к. в застойных зонах на развитие корреляции $\overline{u_{0i}u_{0j}}$ начинает весьма существенно влиять конвекция и диффузия. В итоге гипотеза равновесности оказывается неприменимой. В частности это относится к течению за обратной ступенью.

Выражение для $\overline{u_{0i}t_0}$ (20) применимо в тех условиях при числах Pr≥1.

На рис. 3 показаны результаты расчета теплообмена при смешанной турбулентной конвекции на вертикальной пластине с отброшенными уравнениями переноса $\overline{t_0^2}$ и ε_{t0} и обыкновенными дифференциальными уравнениями для корреляций $\overline{u_0v_0}$, $\overline{v_0t_0}$ и $\overline{u_0t_0}$. Как видно из рисунка, разница между результатами расчетов по упрощенной и полной моделям пренебрежимо мала. Рис. 3 и 4 показывают, что эти соотношения можно использовать и при расчетах естественной конвекции.

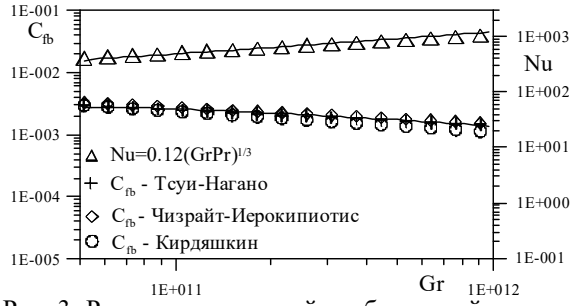


Рис. 3. Расчет естественной турбулентной конвек= ции на вертикальной поверхности по упрощенной модели. Интегральные параметры

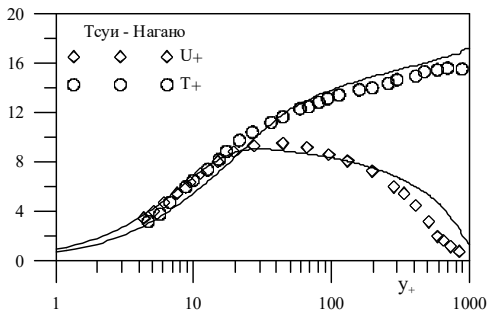


Рис. 4. Расчет естественной турбулентной кон векции на вертикальной поверхности по упро- щенной модели. Скорость и температура.

3. Упрощенные соотношения для расчета $\overline{u_{0i}^2}$

Рассмотрим уравнение переноса $\overline{u_{0i}^2}$. Простейшее из упрощений этого уравнения состоит в замене диссипации ε_{ui0} одним из выражений

$$\varepsilon_{ui0} = 2/3\varepsilon_0, \tag{21}$$

$$\varepsilon_{ui0} = \varepsilon_0 \overline{u_{0i}^2}/k. \tag{22}$$

Для проверки применимости соотношений (21)-(22) была проведена серия тестовых расчетов. Расчеты проводились по модели переноса турбулентных напряжений, но вместо дифференциальных уравнений переноса скорости диссипации ε_{ui0} использовались соотношения (21) и (22). Результаты показали, что такое упрощение модели неправомочно даже при решении простейших задач расчета вынужденной конвекции на плоской пластине и смешанной конвекции на вертикальной поверхности. Подчеркнем, что преобразование системы дифференциальных уравнений в частных производных, описывающей перенос $\overline{u_{i0}^2}$ и ε_{ui0}, в обыкновенное дифференциальное уравнение или алгебраическое соотношение для $\overline{u_{i0}^2}$ возможно только при использовании какой-либо аппроксимации для ε_{ui0}.

В то же время оценочное соотношение для $\overline{u_{i0}^2}$ ASM типа несложно получить из полного уравнения переноса (3) используя для расчета ε_{ui0} выражение $\varepsilon_{ui0} = \overline{u_{0i}^2}/k\,\varepsilon_0$ и

полагая течение равновесным, т.е. считая, что диффузионный перенос равен конвективному.

$$\overline{u_{i0}^2} = \frac{k_0}{\varepsilon_0}\frac{f_0}{(C_a f_0 + 1)}\left(P_i + \frac{2}{3}C_a\varepsilon_0\right). \qquad (23)$$

3.1. Область применимости упрощений

Как уже говорилось, непосредственное использование выражений (21) – (23) в вычислениях ведет к неверным результатам даже при решении простейших задач. В то же время использование формул (23) для оценочных расчетов оказывается вполне удовлетворительным. В качестве подтверждения на рис. 5 и 6 показаны расчеты компонент пульсаций скорости по полной и упрощенной моделям при течениях в пограничном слое и за обратной ступенью. Расхождение оказывается заметным только в пристенных областях.

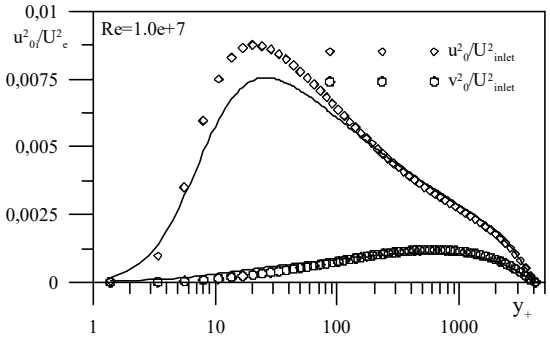


Рис. 5. Сопоставление расчетов пульсаций скорости в пограничном слое по модели переноса турбулентных напряжений (сплошные линии) с расчетами по ASM выражению (23) (символы).

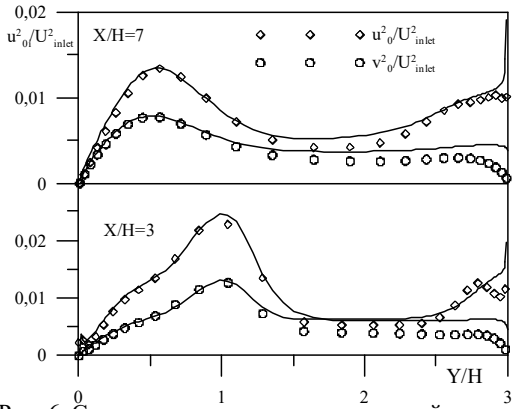


Рис. 6. Сопоставление расчетов пульсаций скорости в течении за обратной ступенью по модели переноса турбулентных напряжений (сплошные линии) с расчетами по ASM выражению (23) (символы).

4. Упрощение модели типа k-ε-$\overline{t'^2}$ -ε_t

Результаты предыдущего раздела показывают, что при расчетах течений с подъемными силами уравнение, описывающее перенос пульсационной компоненты скорости нормальной к направлению подъемных сил в модели должно присутствовать без каких-либо упрощений. Отсюда следует, что течения с подъемными силами должны рассчитываться как минимум по упрощенной модели переноса турбулентных тепловых потоков и напряжений, построенной в разделе 1. В то же время, течения без подъемных сил вполне доступны расчетам по модели типа k-ε-$\overline{t'^2}$ -ε_t. Поэтому рассмотрим, как методика упрощения моделей может быть применена к модели типа k-ε-$\overline{t'^2}$ -ε_t.

В данной модели для корреляций $\overline{u_0 v_0}$ и $\overline{v_0 t_0}$ используются соотношения

$$\nu_t = C_\nu F_\nu \frac{k_0^2}{\varepsilon_0}, \qquad -\overline{u_0 v_0} = \nu_t \frac{\partial U}{\partial y},$$
$$F_\nu = \left(1 - \exp\left(-\frac{\mathrm{Re}_y}{45}\right)\right)\left(1 - \exp\left(-2.4\frac{y}{L_\varepsilon}\right)\right), \qquad (24)$$

$$\alpha_t = C_\lambda F_\lambda k_0 \frac{0.5\overline{t_0^2}}{\varepsilon_{t0} + \varepsilon_{t0-add}}, \qquad -\overline{v_0 t_0} = \alpha_t \frac{\partial T}{\partial y} \qquad (25)$$

,

$$F_\lambda = \left(1-\exp\left(-R_0\frac{\mathrm{Re}_y}{45}\right)\right)\left(1-\exp\left(-2.4\frac{y}{L_\varepsilon}\right)\right),\quad R_0 = \left(\frac{k_0}{\varepsilon_0}\Big/\frac{0.5\overline{t_0^2}}{\varepsilon_{t0}+\varepsilon_{t0-add}}\right)$$

Рассуждая по аналогии со случаем модели переноса турбулентных тепловых потоков, в случае чисел Pr≥0.7 можно положить $0.5\overline{t_0^2}\big/\left(\varepsilon_{t0}+\varepsilon_{t0-add}\right)=k_0/\varepsilon_0$. Отсюда следует, что

$$\alpha_t = C_\lambda F_\lambda \frac{k_0^2}{\varepsilon_0},\quad -\overline{v_0 t_0} = \alpha_t\frac{\partial T}{\partial y}$$
$$F_\lambda = \left(1-\exp\left(-\frac{\mathrm{Re}_y}{45}\right)\right)\left(1-\exp\left(-2.4\frac{y}{L_\varepsilon}\right)\right). \qquad (26)$$

Как видно из выражений (26) теперь F_λ полностью совпадает с F_ν. Но, так как уравнение переноса $\overline{t_0^2}$, а значит и уравнение переноса ε_{t0}, необходимы только для расчета корреляции $\overline{v_0 t_0}$, то эти уравнения вполне можно удалить из модели.

Отметим, что соотношения (24) и (26) показывают, что

$$\alpha_t = \frac{C_\lambda}{C_\nu}\nu_t\,. \qquad (27)$$

Выражение (27) полностью соответствует гипотезе о существовании турбулентного числа Прандтля $\mathrm{Pr_t}$, но получено оно на основе прозрачных физических соображений.

4.1. Область применимости упрощений

Из способа получения упрощенных соотношений (26) следует, что они применимы в расчетах параболических течений в отсутствие массовых сил и при нулевых краевых условиях на стенках.

На рис. 7 оказаны результаты расчетов теплообмена в турбулентном пограничном слое при вынужденной конвекции. Соответствие с экспериментальными данными очень хорошее.

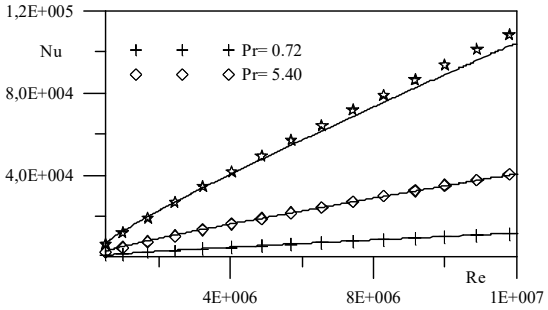


Рис. 7. Расчет теплообмена в пограничном слое при вынужденной конвекции по упрощенной мо дели (символы - Жукаускас).

Символами обозначены расчеты числа Нуссельта по аппроксимации $0.5C_f/St = 0.93+12.5\sqrt{0.5C_f}\left(\mathrm{Pr}^{2/3}-1\right)$, взятой в работе Жукаускаса [4]. Еще раз отметим, что жидкие металлы в данной работе не рассматривались.

На рис. 8 показан расчет средней температуры при течении на начальном участке трубы по полной модели **k-ε-**$\overline{t'^2}$ **-ε$_t$** и с использованием упрощения (26). Как видно из рисунка – соответствие практически точное.

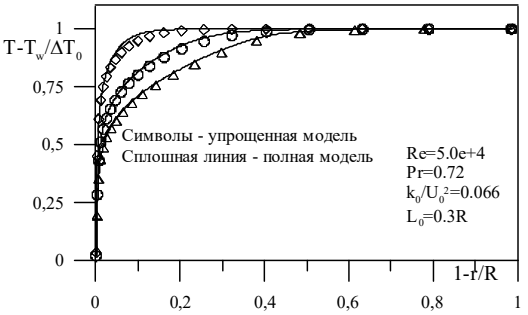


Рис. 8. Расчет течения на начальном участке в трубе по полной модели типа k-ε-t^2-ε$_t$ и по упрощенной модели.

5. Заключительные рекомендации по выбору наименее трудоемкой модели

Прежде, чем переходить к конкретным рекомендациям, необходимо сказать

следующее. В предложенной в данной работе модели k-ε типа для течений в пограничном слое корреляция $\overline{u_0 v_0}$ рассчитывается по выражению

$$\overline{u_0 v_0} = -C_\nu F_\nu \frac{k_0^2}{\varepsilon_0} \frac{\partial U}{\partial y}. \qquad (27)$$

Предполагая слой ненагреваемым и сопоставляя (27) и (17) замечаем, что из этих соотношений следует, что

$$\overline{v_0^2} = C_\nu C_a \frac{F_\nu}{f_0} k_0. \qquad (28)$$

Отметим, что здесь не учтена разница в константах k-ε модели и модели переноса турбулентных напряжений. Из выражения (28) следует, что модель k-ε типа работоспособна только при наличии в течении связи между $\overline{v_0^2}$ и k_0, описанной соотношением (28). Если же эта связь по каким-либо причинам нарушается, то гарантировать адекватность решения нельзя. Причины нарушения могут быть самыми разными, например – сложная геометрия области, наличие вдува или отсоса на границе, наличие массовых сил и т.д.

Таким образом, мы имеем целую иерархию моделей турбулентности разной сложности. На основании проведенных рассуждений можно предложить следующие рекомендации по их использованию.

Случай вынужденной конвекции без учета массовых сил. В любом случае попытку расчета следует начинать с простейшей модели k-ε типа. Если решение проводится в какой-либо сложной области или в задаче присутствуют необычные краевые условия, причем полученный результат по каким-либо критериям не удовлетворяет исследователя, то в модель можно добавить систему уравнений, описывающую перенос корреляции $\overline{v_0^2}$. В этом случае вместо соотношения (27) для расчета турбулентного трения используется решение обыкновенного дифференциального уравнения (19) или алгебраическое соотношение вида (17). Если же и теперь расчеты получаются неудовлетворительными, то от упрощенного соотношения для корреляции $\overline{u_0 v_0}$ следует переходить к полному уравнению вида (5).

Если течение происходит в присутствии массовых сил, то почти наверняка модель должна включать четыре дифференциальных уравнения в частных производных, а именно уравнения модели k-ε типа плюс уравнения переноса $\overline{v_0^2}$ и ε_v.

При расчете теплообмена для чисел $\mathrm{Pr} \geq 0.72$ в параболизованных течениях по всей видимости почти всегда можно использовать либо соотношение (20), либо соотношение (27). Выбор соотношения зависит от модели расчета кинематических пульсаций. Распределение температурных пульсаций при необходимости можно найти из простого алгебраического соотношения (16). При расчетах течений в средах с малыми числами Pr необходимо в модель включать уравнения переноса $\overline{t^2}$ -ε_t.

Результаты исследования и выводы

Как уже говорилось, предложенная модель для расчета турбулентных тепловых потоков и полного тензора турбулентных напряжений, как, впрочем, и любая другая модель этого класса, очень сложна в эксплуатации. По этой причине подобные модели, несмотря на все свои положительные качества, вряд ли могут быть рекомендованы к использованию в инженерной практике. В связи с этим автором было проведено систематическое упрощение исходной полной модели. Все варианты упрощения проверялись тестовыми расчетами решаемых по полной модели задач. Упрощение

получилось очень значительным. Так, в частности, упрощенная версия модели для расчета смешанной конвекции на вертикальной поверхности содержит всего четыре параболических дифференциальных уравнения и три обыкновенных. Расчеты трения и теплообмена полностью соответствуют расчетам по полной модели. Обыкновенные дифференциальные уравнения в данном случае решаются простейшим методом ломаных и, фактически, сводятся к алгебраическим соотношениям. Отметим, что полная модель содержит 11 параболических дифференциальных уравнений.

Список использованной литературы:

1. Golovnya B.P. Modelling of the fluctuating component in the form of the sum of an infnite number of random quantities/ B.P.Golovnya // Int.J. of Heat and Mass Transf. - V 52 - 2009 - pp.5218-5228
2. Golovnya B.P. Modelling of the fluctuating component in the form of the sum of an infnite number of random quantities. Part 2. Model of transfer of turbulent stresses and turbulent heat fluxes/ B.P.Golovnya // Int.J. of Heat and Mass Transf. - V. 52 – 2009 - pp. 5229-5240
3. Головня, Б. П. Модель переноса турбулентных напряжений и турбулентных тепловых потоков/ Б.П. Головня // Холодильна техніка і технологія. – 2007. –– Т.110, № 6. - С.49-54.
4. Жукаускас А.А. Конвективный перенос в теплообменниках/ А.А. Жукаускас. - Москва: - Наука - 1982.
5. Tsuji T., Nagano Y. Turbulence Measurements in a Natural Convection Boundary Layer Along a Vertical Flat Plate. // Int. J. Heat and Mass Transfer – 1988 - V.31 - P. 2101-2111.
6. Cheeswright R., Ierokipiotis,E. Velocity Measurements in a Turbulent Natural Convection Boundary Layer // Proc 7th Int. Heat Transfer Conference, Munich, F.R.G. – 1982 - V.2 - P.305-309.
7. Кирдяшкин А.Г. Структура термогравитационных потоков около поверхности теплообмена. Дисс. докт. т. н., Новосибирск: ИТФ СОАН СССР, 1975.

Bibliography:

1. Golovnya B.P. Modelling of the fluctuating component in the form of the sum of an infnite number of random quantities// Int.J. of Heat and Mass Transf. - V 52 - 2009 - pp.5218-5228
2. Golovnya B.P. Modelling of the fluctuating component in the form of the sum of an infnite number of random quantities. Part 2. Model of transfer of turbulent stresses and turbulent heat fluxes// Int.J. of Heat and Mass Transf. - V. 52 – 2009 - pp. 5229-5240
3. Golovnya B.P. Model of turbulent stresses and turbulent heat fluxes transfer // Refrigeration technology and technology.. – 2007. –– T.110, № 6. - C.49-54.
4. Žukauskas A.A. Convective transfer in heat exchangers. M.: - Nauka - 1982.
5. Tsuji T., Nagano Y. Turbulence Measurements in a Natural Convection Boundary Layer Along a Vertical Flat Plate. // Int. J. Heat and Mass Transfer – 1988 - V.31 - P. 2101-2111.
6. Cheeswright R., Ierokipiotis,E. Velocity Measurements in a Turbulent Natural Convection Boundary Layer // Proc 7th Int. Heat Transfer Conference, Munich, F.R.G. – 1982 - V.2 - P.305-309.
7. Kirdyashkin A.G. The structure of thermogravitational flows near the surface of heat transfer. Diss. Doct.Tech. Sci., Novosibirsk: ITF SBAS USSR, 1975.

GOLOVNYA Boris,
Doctor of Science, Chair of Department of Applied Mathematics and Informatics, The Bohdan Khmelnytsky National University of Cherkasy

SIMPLIFIED MODELS OF TURBULENCE FOR SIMULATION CALCULATION OF TURBULENT HEAT AND MASS TRANSFER

Summary. Introduction. *The problem of simplifying turbulence models can be put in two ways. At first. In many technically important cases, there is a similarity in the flow of temperature distribution and mean velocity, a similarity in the transfer of k0 and etc. An example is the current in the boundary layer, the flow in channels and pipes, etc. But then, using this similarity, one can try to simplify the basic transport equations, i.e. remove from the equations some summands or even reduce the equations to simple algebraic relations. The generality of the model equations is reduced, but the labor costs for solving specific problems are also reduced. It is clear that the applicability of simplification for the solution of a problem should be checked beforehand.Second. Obtained simplified relations may not be applicable for use in the computational system of equations. But, nevertheless, they can be fully used for making estimates. Here the requirements for the relations are less strict, but it is still necessary to verify their applicability at least by the simplest calculations.*

Results and conclusion. *As already mentioned, the proposed model for calculating turbulent heat fluxes and the total turbulent stress tensor, as, indeed, any other model of this class, is very difficult to operate. For this reason, such models, despite all their positive qualities, can hardly be recommended for use in engineering practice. In connection with this, the author made a systematic simplification of the original complete model. All variants of simplification were verified by test calculations of the problems solved by the full model. Simplification was very significant. Thus, in particular, a simplified version of the model for calculating mixed convection on a vertical surface contains only four parabolic differential equations and three ordinary ones. Calculations of friction and heat transfer fully correspond to the calculations for the full model. Ordinary differential equations in this case are solved by the simplest method of broken lines and, in fact, reduce to algebraic relations. We note that the complete model contains 11 parabolic differential equations.*

Одержано редакцією 30.08.2018 р.
Прийнято до публікації 21.11.2018 р.

УДК 519.6:004.8

DOI 10.31651/2076-5886-2019-1-26-33

PACS 02.60.-x, 02.60.Pn, 02.70.Wz

ІЛЬЯХОВА Наталія Олександрівна
студентка Черкаського національного університету імені Богдана Хмельницького
e-mail: ilyahova.nata@gmail.com
ORCID 0000-0003-0737-514X

КРАСНОШЛИК Наталія Олександрівна
кандидат технічних наук, доцент,
доцент кафедри прикладної математики та інформатики Черкаського національного університету імені Богдана Хмельницького
e-mail: wlik007@ukr.net
ORCID 0000-0003-4661-6997

БАГАТОРОЙОВИЙ АЛГОРИТМ MULTI-SWARM OPTIMIZATION ТА ЙОГО ВИКОРИСТАННЯ ПРИ РОЗВ'ЯЗУВАННІ ЗАДАЧ БІНАРНОЇ КЛАСИФІКАЦІЇ

У роботі розглянуто багаторойовий алгоритм оптимізації частинок (Multi-Swarm Optimization Algorithm) для розв'язування задач глобальної оптимізації, який відноситься до метаевристичних методів. Метою даної роботи є реалізація та дослідження даного алгоритму при розв'язувані задач оптимізації, а також його застосування до розв'язування задач бінарної класифікації. Проведено порівняльний аналіз та досліджено ефективність алгоритму при знаходженні глобального мінімуму деяких тестових функцій. Описано постановку задачі бінарної лінійної класифікації. Для мінімізації функціоналу похибки при побудові класифікатора використано багаторойовий алгоритм MSO та метод стохастичного градієнтного спуску.

Ключові слова: *багаторойовий алгоритм оптимізації частинок, алгоритм MSO, задача оптимізації, задача бінарної класифікації*

Постановка проблеми

У сучасних умовах в науці і техніці існує стійка тенденція, пов'язана з необхідністю розв'язання широкого спектру практичних задач в оптимізаційній постановці. Для цього застосовуються досить різноманітні методи, але далеко не всі з поставлених задач можуть бути розв'язані з використанням традиційних підходів. Умовно існуючі методи розв'язання задач глобальної оптимізації можна розділити на

детерміновані та стохастичні, тобто такі, що характеризуються організацією пошуку з відсутністю або наявністю псевдовипадкових елементів. Особливе місце серед стохастичних методів посідають методи, які ґрунтуються на імітації природних процесів та реалізують адаптивний випадковий пошук.

Метод багаторойової є варіацією методу оптимізації роєм частинок (PSO). При цьому метод MSO розширює можливості оптимізації роєм частинок, використовуючи кілька роїв імітованих частинок, а не один рій. MSO є метаевристичним методом, що означає, що ця методика є набором принципів проектування, які можуть бути використані для побудови конкретного алгоритму для розв'язання конкретної задачі оптимізації.

Виклад основного матеріалу

1. Класичний багаторойовий алгоритм оптимізації MSO

Задача безумовної глобальної оптимізації формулюється як задача мінімізації цільової функції $f(x)$ в просторі пошуку D:

$$f(x) \to \min, X \in D = \{x \in R^d\}, \tag{1}$$

де область D – дійсний гіперкуб з розмірністю d, X – векторний аргумент цільової функції $f(x)$, а її глобальний мінімум досягається у точці X^*.

У методі MSO рій частинок являє собою сукупність точок-розв'язків, які переміщаються у просторі в пошуках глобального мінімуму. При переміщенні частинки намагаються покращити знайдений ними раніше розв'язок і при цьому обмінюються інформацією зі своїми сусідами.

З роєм частинок також асоціюється множина векторів їх швидкостей

$$V = \{v_1, v_2, ..., v_s\}, \tag{2}$$

Ключова операція в MSO – обчислення нової швидкості для частинки. На нову швидкість для даної частинки впливає поточна швидкість, поточний стан, найкраще відоме положення частинки, найкраще відоме положення будь-якої частинки в тому ж рої, що і частинка, і найкраще відоме положення будь-якої частинки в будь-якому рої.

$$\begin{aligned} v_{ij}^{'} &= wv_{ij} + c_1 r_1 \left(p_{ij} - x_{ij} \right) + c_2 r_2 \left(g_j - x_{ij} \right), \\ x_{ij}^{'} &= x_{ij} + v_{ij,}^{'} \end{aligned} \tag{3}$$

де $v_i^{'}$ і $x_i^{'}$ – нові швидкість і положення i-ої частинки; p_i – найкраще положення, знайдене цією частинкою раніше (personal best); s – найкраще положення будь-якої частинки в рої; g – найкраще положення, знайдене всім роєм (global best); c_1, c_2, c_3 – відповідно когнітивний, соціальний та глобальний коефіцієнти; r_1, r_2, r_3 – рівномірно розподілені випадкові числа з інтервалу [0, 1]. У формулі (3) передбачається, що час між оновленнями стану частинок роя $\Delta t = 1$.

Значення коефіцієнта w відіграє важливу роль і визначає компроміс між дослідженням і локалізацією в просторі пошуку. При $w \geq 1$ швидкість частинки збільшується і рій «розходиться». При $w < 1$ частинки сповільнюються до тих пір, поки швидкість не стане рівною нулю. Таким чином, великі значення w сприяють дослідженню простору пошуку, а малі – локалізації розв'язку. Однак, занадто малі значення w позбавляють рій здатності досліджувати простір пошуку. Чим менше w, тим більший вплив когнітивної та соціальної компоненти. Як і інші параметри, оптимальне значення w проблемно-орієнтоване, тобто обирається для конкретної задачі. Якщо в процесі оптимізації частинка виходить за межі простору пошуку,

відбувається обнулення відповідних компонент її швидкості, а сама частинка повертається до найближчої границі.

Величини вагових коефіцієнтів у (3) обираються наступні

$$1 < c_1, c_2, c_3 \le 2, \tag{4}$$

що забезпечує збіжність методу.

Для підтримання балансу між локальним і глобальним пошуком, чисельні значення коефіцієнтів c_1 і c_2, як правило, обираються рівними. У більшості випадків найкращі результати і стабільність пошуку отримують при $c = 0.5 + \ln 2 \approx 1.19$. У MSO мало досліджень константи c_3, тому зазвичай використовують значення 0.36.

Рій володіє пам'яттю про найкращі розв'язки, знайдені його окремими частинками і всім роєм в цілому. Під час ініціалізації початкові позиції частинок вважаються найкращими. На кожній наступній ітерації алгоритму MSO індивідуальні кращі позиції кожної частинки і найкраще положення g, знайдене всім роєм оновлюються за правилами

$$\begin{cases} p_i = x_i, якщо\ f(x_i) < f(p_i), \\ g = p_i, якщо\ f(p_i) < f(g). \end{cases} \tag{5}$$

2. Модифікації алгоритму MSO

Існує кілька способів зміни базового алгоритму MSO. Один із можливих варіантів полягає у тому, щоб час від часу знищувати випадково вибрану частинку, а потім народжувати нову частинку замість неї. Для цього можна згенерувати випадкове значення між 0 і 1, і зберегти його у q. Якщо дане випадкове значення менше 0.005, то поточна частинка повторно створюється, знищуючи поточну частинку і народжуючи нову частинку у випадковому місці.

Інший варіант модифікації MSO – моделювати імміграцію, періодично беручи дві частинки у різних роях і обмінюючи їх. Одна частинка ефективно проникає в поточний рій, а інша частка емігрує з рою.

3. Реалізація та дослідження алгоритму MSO

Багаторойовий алгоритм MSO був реалізований у середовищі MATLAB R2012b. Проведено дослідження бгаторойового алгоритму MSO для розв'язання задачі (1) з використанням різних значень параметрів N (кількість ітерацій) і k – кількість роїв, розмірність простору пошуку d = 2. При цьому алгоритм застосовувався до кожної функції 10 разів, а отримане значення усереднювалося. Одержані результати наведено у табл. 1.

Було проведено порівняння класичного алгоритму оптимізації роєм частинок та досліджуваного багаторойового алгоритму оптимізації частинок. Отримані результати свідчать, що у більшості випадків найкраще метод MSO виконує пошук глобального мінімуму при виконанні великої кількості ітерацій та невеликій розмірності простору пошуку. Метод MSO показав кращі результати для всіх тестових функцій, крім функцій Растрігіна і Швефела, оскільки пошуковий алгоритм має тенденцію застрявати в одному із локальних оптимумів та збігатися в неправильному напрямку. Порівняльний аналіз результатів роботи класичного алгоритму PSO та MSO показав, що алгоритм MSO показав кращі результати на більшості функцій.

4. Розв'язання задачі бінарної класифікації з використанням алгоритму MSO

Розглянемо задачу бінарної класифікації, коли множина можливих значень відповідей складається з двох елементів: $Y = \{-1, +1\}$. Сукупність об'єктів, які мають

однакову відповідь, називають класом. Потрібно встановити належність об'єктів вибірки до одного з двох класів, тобто класифікувати їх.

Щоб побудувати лінійний класифікатор необхідно:

- обрати функціонал похибки, тобто задати спосіб визначення якості роботи того чи іншого алгоритму на навчальній вибірці;
- побудувати сімейство алгоритмів, тобто множину алгоритмів, з якої потім буде обиратися найкращий з точки зору певного функціоналу похибки;
- задати метод навчання, тобто визначити спосіб вибору кращого алгоритму з побудованого сімейства алгоритмів.

Таблиця 1

Результати обчислювальних експериментів

Тестова функція, точний розв'язок	N = 50			N = 100		
	k = 2	k = 3	k = 5	k = 2	k = 3	k = 5
1. Сферична функція $f(x)=\sum_{i=1}^{d} x_i^2,\ f(X^*)=0$	6.79e-18	8.74e-20	3.79e-20	4.89e-20	8.96e-21	4.22e-20
2. Функція Швефела $f(x)=\sum_{i=1}^{d}\left(\sum_{j=1}^{i} x_j\right)^2,\ f(X^*)=0$	1.13e-18	5.50e-21	4.73e-20	0.0244	3.54e-06	1.28e-10
3. Функція Розенброка $f(x)=\sum_{i=1}^{d-1}[100(x_{i+1}-x_i^2)^2+(1-x_j)^2]$, $f(X^*)=1$	0.0366	5.95e-08	1.45e-07	1.5687	1.0032	2.89e-14
4. Функція Растрігіна $f(x)=\sum_{i=1}^{d}[x_i^2-10\cos(2\pi x_i)+10]$, $f(X^*)=0$	0	0	0	0.8791	0.0257	3.52e-04
5. Функція Алпайна $f(x)=\sum_{i=1}^{d}\lvert x_i \sin x_i+0.1x_i\rvert$, $f(X^*)=0$	3.49e-10	2.35e-10	5.96e-11	0.0019	1.85e-02	3.51e-06

Лінійні класифікатори повинні повертати бінарне значення, а отже у якості результату можна обирати знак наступного виразу:

$$a(x) = sign(w_0 + \sum_{j=1}^{d} w_j x^j), \qquad (6)$$

де w_0 – вільний коефіцієнт, x^j – ознака об'єкта, w_j – ваговий коефіцієнт для відповідної ознаки, d – кількість об'єктів.

Вираз $\langle w, x\rangle = 0$ є рівнянням деякої площини у просторі ознак. При цьому для точок по одну сторону від цієї площини скалярний добуток $\langle w, x\rangle$ буде додатнім, а з

іншого – від'ємним. Таким чином, лінійний класифікатор проводить площину у просторі ознак і відносить об'єкти по різні боки від неї до різних класів (рис. 1).

Для задачі лінійної класифікації природний спосіб визначити якість того чи іншого алгоритму полягає в обчислені для об'єктів навчальної вибірки частки неправильних відповідей:

$$Q(a,x)=\frac{1}{l}\sum_{i=1}^{l}[a(x_i)\neq y_i], \tag{7}$$

де l – кількість об'єктів, y_i – відповідь для даного об'єкта.

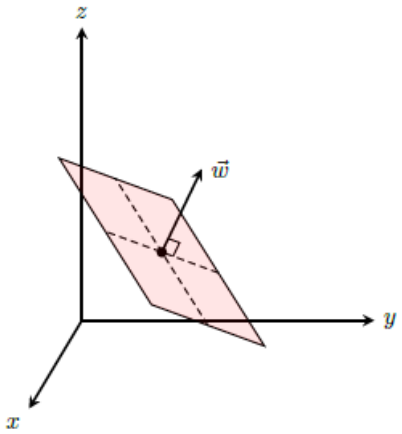


Рис. 1 Геометричний зміст лінійного класифікатору [4]

Вираз (7) можна переписати для випадку лінійної класифікації у наступному вигляді:

$$Q(a,x)=\frac{1}{l}\sum_{i=1}^{l}[y_i\langle w,x_i\rangle<0]=\sum_{i=1}^{l}[M_i<0]. \tag{8}$$

Функція, що стоїть під знаком суми, називається пороговою функцією втрат [4]. Використовуючи будь-яку гладку оцінку порогової функції:

$$[M_i<0]\leq\tilde{L}(M_i)$$

можна побудувати оцінку $\tilde{Q}(a,x)$ для функціонала похибки $Q(a,x)$:

$$Q(a,x)\leq\tilde{Q}(a,x)=\frac{1}{l}\sum_{i=1}^{l}\tilde{L}(M_i)\to\min_a. \tag{9}$$

У роботі для побудови лінійного класифікатора будемо використовувати логістичну функцію втрат:

$$\tilde{L}(M)=\log_2(1+\exp(-M)). \tag{10}$$

В цілому існує декілька основних алгоритмів оптимізації, що застосовуються в машинному навчанні для мінімізації функціонала похибки. У даній роботі застосуємо розглянутий алгоритм багаторойової оптимізації MSO. У MSO кожна частинка має позицію, яка відповідає набору значень *w*-вагів, а рій – це набір частинок, які рухаються так, як визначає групова поведінка, наприклад, зграї птахів. MSO підтримує декілька роїв одночасно, що взаємодіють один з одним, на відміну від PSO, де використовується один рій.

Для задачі бінарної класифікації використовували наступні набори даних:

– Задача про виживання Хабермана. Дана вибірка містить результати дослідження про виживання пацієнтів, які перенесли операцію з видалення пухлини молочної залози. Вона складається з 200 об'єктів з 4 ознаками: вік пацієнта в момент операції, рік

проведення операції, кількість виявлених пухлин та ознаку виживання (1 : пацієнт прожив 5 років і більше, –1 : пацієнт помер протягом 5 років).

– Задача ідентифікації скла. Дана вибірка містить 71 об'єкт з 10 ознаками: показник заломлення, вміст натрію, магнію, алюмінію, кремнію, калію, кальцію, барію, заліза, та ознаку типу скла (1 : термополіроване скло, –1 : не термополіроване скло).

– Задача визначення виду квітів ірису. Дана вибірка містить 98 об'єктів з 5 ознаками, серед яких довжина та ширина чашолистка, довжина та ширина пелюстки, і вид відповідної квітки (1 : ірис Сетоза, –1 : ірис Версиколор).

Кожен набір даних розбивався на навчальну та тестову вибірки у співвідношенні 75% : 25%. На навчальній виборці знайдемо вагові коефіцієнти w шляхом мінімізації функції втрат. Для навчання класифікатора застосуємо метод MSO, який використовує 10 роїв, кожний з 30 частинками. При цьому отримано наступні значення вагових коефіцієнтів: задача виживання Хабермана w =(-0.5530; -0.0133; 0.0334; -0.0463); задача ідентифікації скла w =(-0.8577; -2.2800; -0.4369; 1.1402; -1.8115; 0.0359; -0.6912; 0.7864; 1.9194; -1.2079); задача визначення виду квітів ірису w =(5.0000; -0.5710; 5.0000; 5.0000; -5.0000).

Також для навчання бінарного класифікатора використовували класичний метод PSO та метод стохастичного градієнтного спуску. Для оцінки якості класифікатора обраховували долю правильних відповідей на тестовій виборці. Отримані результати представлено у табл. 2.

Таблиця 2

Доля правильних відповідей на тестовій виборці

Метод для навчання	Виживання Хабермана (4 ознаки)	Ідентифікація скла (10 ознак)	Іриси (5 ознак)
1. MSO	72%	62%	98%
2. Класичний PSO	72%	50%	97%
3. Стохастичний градієнтний спуск	70%	50%	98%

Результати обчислювальних експериментів підтверджують ефективність та доцільність використання алгоритму MSO для мінімізації функціоналу похибки при побудові бінарного лінійного класифікатора по логістичній регресії.

Висновки

У роботі розглянуто багаторойовий алгоритм оптимізації частинок та його модифікації для розв'язування задач глобальної оптимізації. Досліджено залежність ефективності даного алгоритму від кількості ітерацій, розмірності простору пошуку та кількості роїв. Проведено порівняльний аналіз багаторойового алгоритму MSO при знаходженні глобального мінімуму тестових функцій.

Розглянутий алгоритм застосовано для мінімізації функціоналу похибки при побудові бінарного лінійного класифікатора. Отримані результати свідчать про ефективність застосування багаторойового алгоритму MSO для даної задачі. Таким чином, доцільно застосовувати алгоритм MSO до задач машинного навчання та здійснювати його подальше вдосконалення.

Список використаної літератури:

1. Гальченко В. Я. Популяционные метаэвристические алгоритмы оптимизации роем частиц: Учебное

пособие / В. Я. Гальченко, А. Н. Якимов. – Черкассы: ФЛП Третяков А. Н., 2015. – 160 с.
2. Скобцов Ю. А. Метаэвристики : монография / Ю. А. Скобцов, Е. Е. Федоров. – Донецк: Изд-во «Ноулидж» (Донецкое отделение), 2013. – 426 с.
3. Матренин П. В. Методы стохастической оптимизации: учебное пособие / П. В. Матренин, М.Г. Гриф, В. Г. Секаев. – Новосибирск: Изд-во НГТУ, 2016. – 67 с.
4. Конспект лекции «Линейные модели: статистический взгляд» [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – [Курс «Обучение на размеченных данных»]. – Режим доступу: https://www.coursera.org/learn/supervised-learning/supplement/Dw3Ws/konspiekt (дата звернення 01.03.2017) – Назва з екрана.
5. Карпенко А. П. Обзор методов роя частиц для задачи глобальной оптимизации (Particle Swarm Optimization) / Карпенко А.П., Селиверстов Е.Ю.-Сетевое издание «Наука и образование: научное издание МГТУ им. Н.Э. Баумана»
6. Презентация «Машинное обучение (Machine Learning). Регрессионные модели», Уткин Л.В., Санкт-Петербургский политехнический университет Петра Великого.
7. Методы стохастической оптимизации: учебное пособие / П. В .Матренин, М. Г. Гриф, В. Г. Секаев. – Новосибирск: Изд-во НГТУ, 2016. – 67 с.
8. James McCaffrey Test Run – Multi-Swarm Optimization [Електронний ресурс] : [Веб-сайт]. Режим доступу: https://msdn.microsoft.com/ru-ru/magazine/dn385711
9. Классификация по логистической регрессии с помощью оптимизации на основе нескольких роев частиц [Електронний ресурс] : [Веб-сайт]. Режим доступу: https://msdn.microsoft.com/ru-ru/magazine/dn890377.aspx
10. Machine Learning Repository [Електронний ресурс] : [Веб-сайт]. Режим доступу: https://archive.ics.uci.edu/ml/datasets.html

Bibliography:

1. Galchenko V. Ya. Population metaheuristic algorithms for optimizing the swarm of particles: Textbook / V. Ya. Galchenko, A. N. Yakimov. – Cherkasy: FLP Tretyakov A.N., 2015. – 160 p.
2. Skobtsov Yu. A. Metaheuristics: monograph / Yu. A. Skobtsov, EE Fedorov. – Donetsk: Publishing house "Knolidzh" (Donetsk branch), 2013. – 426 p.
3. Matrenin P. V. Methods of stochastic optimization: textbook / P.V. Matrenin, M.G. Grif, V.G. Sekaev. – Novosibirsk: Publishing house of NSTU, 2016. – 67 p.
4. Lecture notes: "Linear models: statistical view" [Electronic resource]: [Web site]. – Electronic Data. – [Course "Training on marked data"]. – Access mode: https://www.coursera.org/learn/supervised-learning/supplement/Dw3Ws/konspiekt (the date of the zombie date is 03/01/2017) – The name of the screen.
5. Review of particle swarm methods for the problem of global optimization (Particle Swarm Optimization) / Karpenko A. P., Seliverstov E. Yu. – Network publication "Science and Education: a scientific publication of the MSTU. N.E. Bauman»
6. Presentation «Machine Learning (Machine Learning). Regression models ", L.V. Utkin, St. Petersburg Polytechnic University of Peter the Great.
7. Methods of stochastic optimization: a tutorial / P.V. Matrenin, M.G. Grief, V.G. Sekaev. – Novosibirsk: Publishing house of the NSTU, 2016. – 67 p.
8. James McCaffrey Test Run – Multi-Swarm Optimization: [Web site]. Access mode: https://msdn.microsoft.com/en-us/magazine/dn385711
9. Classification by logistic regression using optimization based on multiple particle swarms [Electronic resource]: [Web site]. Access mode: https://msdn.microsoft.com/en-us/magazine/dn890377.aspx
10. Machine Learning Repository: [Web site]. Access mode: https://archive.ics.uci.edu/ml/datasets.html

ILYAKHOVA Nataliya,
student, The Bohdan Khmelnytsky National University of Cherkasy
KRASNOSHLYK Nataliya,
PhD, Senior Lecturer, The Bohdan Khmelnytsky National University of Cherkasy

MULTI-SWARM OPTIMIZATION ALGORITHM AND ITS APPLICATION FOR SOLVING BINARY CLASSIFICATION

Summary. Introduction. *In modern conditions in science and technology, there is a steady trend associated with the need to solve a wide range of practical problems in an optimization setting. To do this, there are quite a variety of methods used, but not all of the tasks can be solved using traditional approaches. Conditionally existing methods for solving global optimization problems can*

be divided into deterministic and stochastic ones, that is, those characterized by an organization of search with the absence or presence of pseudorandom elements. A special place among stochastic methods is the methods based on the simulation of natural processes that implement an adaptive random search.

Multiplayer Optimization (MSO) is a method for evaluating solutions to complex or impossible numeric tasks. This is a variation of particle swarm optimization. Regular models of particle swarm optimization, such as in groups of birds and flocks of fish. MSO expands the optimization of the swarm of particles using several swarms of simulated particles, and not one swarm.

MSO can be applied to several machine learning scenarios, such as estimating the values of weight and displacement for an artificial neural network, or assessing the weight of weak students in ensemble classification and prediction. MSO is meta-verbal, which means that this technique is a set of design principles that can be used to construct a specific algorithm for solving a specific optimization problem.

Purpose. *Comparative analysis of MSO algorithm and its modifications for solving optimization problems and linear classification problem.*

Results*. The paper deals with a multi-swarm algorithm for particle optimization and its modification for solving global optimization problems. The dependence of the efficiency of this algorithm on the number of iterations, the size of the search space and the number of swarms are investigated. A comparative analysis of the multi-swarm MSO algorithm has been carried out in finding the global minimum of test functions.*

The presented methods are used to minimize the error functionality when constructing a binary linear classifier. The obtained results testify to the efficiency of the application of the multi-swarm MSO algorithm for this task. Therefore, it is advisable to apply the MSO algorithm for machine learning tasks and to further improve it.

Conclusion. *This article examines the multi-swarm particle optimization algorithm and its modifications for solving global optimization related to metaheuristic methods, and applications for solving binary linear classification problems is considered. The purpose of this work is to conduct a comparative analysis of the multi-swarm algorithm of MSO and its modifications for solving optimization problems. A comparative analysis was carried out and the efficiency of the algorithm was investigated in finding the global minimum of some test functions. The formulation of the problem of binary linear classification is described. To minimize the functional error in constructing classifiers, a multi-swarm MSO algorithm with modifications and stochastic gradient takeoff method was used.*

Keywords: *multi-swarm optimization algorithm, MSO algorithm, optimization problem, the problem of binary classification.*

Одержано редакцією 24.05.2018 р.
Прийнято до публікації 19.09.2018 р.

УДК 519.6:323.2

DOI 10.31651/2076-5886-2019-1-33-41

МОРГУН Микола Геннадійович
аспірант кафедри прикладної математики та інформатики Черкаського національного університету імені Богдана Хмельницького
e-mail: mykolamorhun4edu@gmail.com
ORCID 0000-0002-5520-3302

ІЄРАРХІЧНА САМОПОДІБНА МОДЕЛЬ ПЕРЕРОЗПОДІЛУ ВЛАДИ В СУЧАСНОМУ СУСПІЛЬСТВІ

У статті пропонується самоподібна модель сучасного суспільства для прогнозування перерозподілу влади в результаті виборів. Наведений алгоритм роботи моделі для подальшого комп'ютерного аналізу.

Ключові слова: *модель, влада, вибори, суспільство.*

Постановка проблеми

В наш час все більше уваги приділяється питанням пов'язаним із суспільними процесами. І це не дивно, адже такі знання дозволяють прогнозувати подальший розвиток і реакцію суспільства на той чи інший фактор. Складно знайти сферу для якої така інформація не була б вагомою. А для того, щоб передбачити подальший розвиток подій, спочатку необхідно побудувати модель, яка б відображала цей розвиток. У другій половині XX століття почали з'являтися моделі, завданням яких було описати суспільство чи процеси у ньому. Більшість з них описують взаємозв'язки компонентів або подають у їх вигляді діаграм [1-2]. Суттєві корективи у методи моделювання були внесені у зв'язку з розвитком комп'ютерної техніки, що дозволяло обробляти все більші об'єми даних. На даний час важливим є не тільки знання взаємозв'язків між компонентами, але й можливість розрахунку їх подальшого розвитку, де, звісно, не обійтися без комп'ютерного моделювання, що в свою чергу означає необхідність переходу від описових та діаграмних моделей до більш формальних, які можуть бути переведені на мову алгоритмів та оброблені за допомогою комп'ютера.

Досить відомою у сфері математичного моделювання пов'язаного з владою є модель "влада-суспільство" [3] А. П. Михайлова, яка, мабуть, стала класичною і отримала багато варіацій та доповнень від різних авторів, наприклад [4-7].

Модель "влада-суспільство" приділяє більше уваги зміні кількості влади у вертикалі, а також взаємодія із громадянським суспільством не є обмеженою, тобто немає ліміту на кількість влади, яка передається до/від владних структур. Моделі [8-10] використовують клітинний автомат, але вони є значно обмеженими через використання квадратного поля та перпендикулярної сітки.

Мета статті

Метою статті є побудова моделі перерозподілу влади між політичними силами у сучасному суспільстві під час виборів, та алгоритму її комп'ютерної реалізації.

Виклад основного матеріалу

Існування в суспільстві вносить деякі обмеження у життя людини, її поведінку, реакції на різні чинники. У суспільстві кожен повинен підпорядковуватися правилам, які встановлені у цьому суспільстві, або ж непокора останнім викличе реакцію з боку суспільства і потягне за собою наслідки для індивіда чи групи людей. Це спричинено тим, що суспільство має деяку владу над людиною. Якщо ж розглянути людину поза суспільством, вона вільна від будь яких обмежень які вносить суспільство, тобто повністю віддається своїй волі. Влада суспільства черпається з його членів, тобто при створенні будь-якого суспільства кожен його учасник надає частину своєї влади цьому суспільству, а деяка частина влади залишається у кожного індивіда (не обов'язково однакова). З розвитком суспільства кількість влади, якою володіє кожен індивід може змінюватися. Цей процес може бути ініційований як самими людьми так і владними структурами, але важливою особливістю цієї зміни є те, що вона відбувається стрибкоподібно при створенні суспільства, виборах, революціях, зміні устрою та/чи законів. Розглянемо приклад. Якщо владні структури в суспільстві приймають і впроваджують рішення які викликають незадоволення у значної частини суспільства це знижує довіру суспільства до керуючої частини, але не змінює розподілу влади. Останній же відбувається на виборах (і тоді члени суспільства віддають свою владу іншій політичний силі) або ж становище стає настільки критичним, що цей перерозподіл відбувається достроково.

Під час проведення виборів до конкретного індивіда повертається не вся його влада, а лише її частина (вибори не можуть змінити усе в державі: устрій, закони тощо),

яку він знову зобов'язаний віддати тій чи іншій політичній силі. На виборах, окремий індивід намагається віддати частину своєї влади тому лідеру чи політичні силі, яка пропонує план дій, що найбільше збігається з поглядами самого виборця. Якщо таких не існує, то або індивід віддасть свою частину влади випадковій політичній силі, або проігнорує самі вибори і тоді його частина влади буде пропорційно розділена між усіма кандидатами. Тобто для того щоб передбачити якій політичній силі надасть свою владу виборець необхідно передбачити його погляди. Для побудови моделі пропонується ввести систему поглядів людини як набір числових значень в межах від -1 до 1 (рис. 1), яка відображатиме ставлення індивіда до суспільно важливих питань.

При обміні поглядами необхідно враховувати їх різницю. Якщо один індивід має досить сильне переконання щодо деякого питання, а позиція іншого близька до нейтральної, то, з великою ймовірністю, після їх взаємодії ставлення другого буде суттєво змінено в бік позиції першого. У випадку обміну думками по питанню на яке в усіх учасників немає сильної позиції, вплив буде незначний.

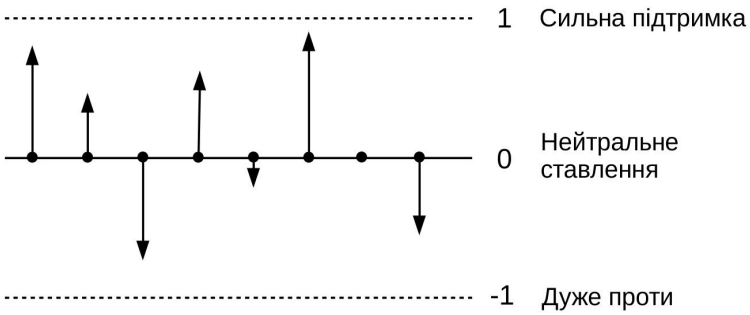


Рис. 1. Приклад системи поглядів індивіда

Для відображення структури та ієрархії суспільства в моделі, розділимо суспільство на рівні та структурні одиниці в них. Покладемо, що структурна одиниця містить в собі групу одиниць нижчого рівня (за винятком найнижчого), та має зв'язки як у своїй вертикалі так і у горизонталі, причому зв'язки між рівнями можуть проходити лише в рамках вертикалі об'єкта на рівень вище та нижче. Аналізуючи місце та зв'язки об'єкта у загальній системі вищеописаних рівнів можна побачити, що вони однакові для різних рівнів, тобто запропонована модель є самоподібною (рис. 2). Крім того, кожне суспільство може перебувати у зв'язках з іншими, подібними собі, структурами, що вписується в загальну картину моделі.

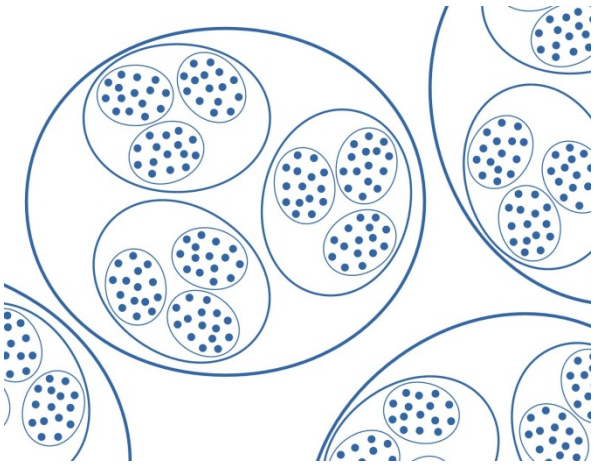


Рис. 2. Структура моделі

Означимо основні частини структурної одиниці. Як зазначалося вище, вона має зв'язки з собі подібними і з вищою одиницею, а це значить що, представляючи нижчі рівні, вона проводить зовнішню політику. Зв'язки зі своїми складовими є її внутрішньою політикою. Незалежно від рівня, кожна структурна одиниця керується одним або групою індивідів. А це означає, що вона має особистісну частину (рис 3).

Визначимо детальну структуру кожної з цих частин (рис. 4).

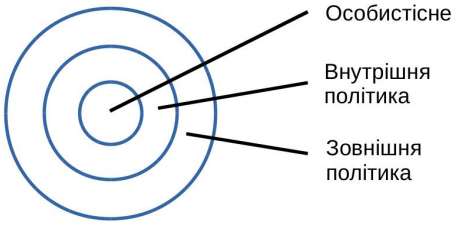


Рис. 3. Основні частини структурної одиниці

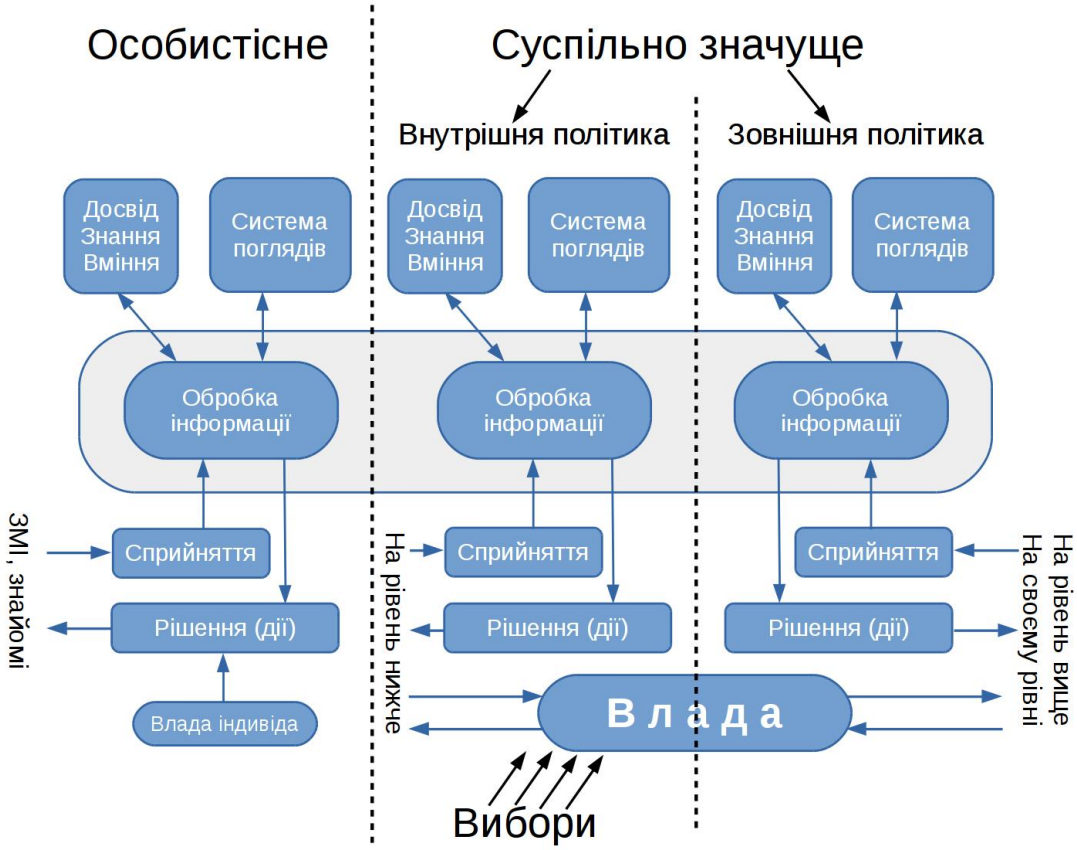


Рис. 4. Детальна схема структурної одиниці

Почнемо з особистісної. Очевидно, що індивід не відокремлений від інших членів суспільства і комунікує з ними. В процесі комунікації індивід сприймає інформацію з кола друзів та знайомих, а також засобів масової інформації, аналізує її та відповідає деякою реакцією, використовуючи залишок своєї влади, тим самим впливаючи на своє оточення. Для цього він використовує свої знання, досвід та уміння, а також керується своєю системою поглядів. Цей процес впливає на досвід, знання і вміння індивіда, а

також може змінювати його систему поглядів. Подібна ситуація прослідковується і з внутрішньою та зовнішньою політикою. Різниця полягає в джерелі сприйняття та напрямку дії. Так для проведення внутрішньої політики аналізуються сигнали з складових частин структурної одиниці, і дії, в загальному випадку, направлені в їхній бік. Для зовнішньої політики це також справедливо, але для структурних одиниць цього ж рівня в межах вищої структурної одиниці, та її самої. Наприклад, в результаті стихійного лиха, в сусідній області пошкоджено значну частину ліній електропередач. Управління, аналізуючи ситуацію, вирішує допомогти, та, використовуючи свою владу, діє, відправляючи на допомогу техніку та особовий склад.

Важливим аспектом є те, що внутрішню та зовнішню політику від імені всіх учасників рівня проводить індивід або група індивідів, які мають свій досвід, знання та вміння ведення внутрішньої чи/та зовнішньої політики, а також систему поглядів, яка використовується для прийняття рішень і використання влади. Слід зазначити, що ці системи поглядів та досвід, знання і вміння, у загальному випадку, є різними, не дивлячись на те що вони поєднуються в одному індивіді чи їх групі. Тобто, у вільний від роботи час, цей індивід (який є чиновником) керується своєю особистою системою поглядів, а коли він приймає політичні рішення, то використовується вже зовсім інша система поглядів, яка, в загальному випадку, не збігається з його особистими інтересами. На практиці така розбіжність може стати причиною корупції — використання наданої суспільством влади в особистих цілях та інтересах.

Підсумуємо:

- Суспільство має багаторівневу структуру.
- Будова структурної одиниці однакова для всіх рівнів (самоподібна).
- Об’єктом найменшого рівня є людина, обмежень на кількість рівнів немає.
- Структурна одиниця може взаємодіяти з іншими об’єктами на своєму рівні та на ± 1 рівень у своїй піраміді.
- Структурна одиниця проводить зовнішню і внутрішню політику:
  - В межах зовнішньої вона комунікує з одиницями свого рівня, які належать до тієї самої вищої структурної одиниці, а також із нею самою.
  - В межах внутрішньої політики комунікує з своїми складовими.

  Винятком є людина (перший рівень) в якої відсутня внутрішня політика, а також найвищий рівень, зовнішня політика якого або відсутня (якщо суспільство ізольоване), або зводиться до комунікації з собі подібними.
- Владу людині не наддають вона їй надана апріорі.
- Якщо людина є членом суспільства вона віддає частину своєї влади суспільству (керівництву суспільства). Якщо рівень один — суспільство відсутнє і влада індивіда не віддається на рівень вище, а залишається в нього самого, тобто він є одинаком.
- Перерозподіл влади відбувається періодично, стрибком, під час перевиборів або революцій.
- При виборах частина наданої влади суспільству індивідом повертається знову до нього і він зобов’язаний її віддати одній з політичних сил яка обирається. У випадку ігнорування виборів його влада розподіляється пропорційно між усіма кандидатами.
- Виборець віддає свою владу тій політичній силі, система поглядів якої найбільше збігається з його системою поглядів, або ж ігнорує вибори, якщо всі політичні сили мають систему поглядів досить розбіжну з його особистою.

– Якщо присутня корупція, то при прийнятті владних рішень система особистих поглядів чиновника накладається на суспільну значущу систему поглядів структурної одиниці.

Опис алгоритму моделі

Введемо деякі спрощення:

– Немає нестатку в матеріальному плані. Матеріальних благ вистачає на всіх і на все.
– Громадяни поосібно не перевищують своєї влади.
– Влада прислухається до думки громадян.
– Кандидати на виборах чесно проголошують свої плани та намагаються їх дотримуватися.

Ініціалізація моделі:

– Визначимо структуру суспільства: кількість рівнів та складові кожного з них.
– Для кожної групи, введемо відстані між її членами. Як одне з найпростіших наближень, можна використати циклічні індекси.
– Задаємо питання системи поглядів.
– Визначаємо кількість політичних сил та їх погляди.

Взаємодія між об'єктами в суспільстві:

– Етап перший: взаємодія між об'єктами однакових рівнів.
  – Проводимо обмін поглядами між членами кожної групи. Ймовірність взаємодії залежить від відстані між об'єктами. Для розрахунку ймовірностей пропонується взяти розподіл Гауса. Причому спочатку відбувається взаємодія з ближчими об'єктами, а потім із більш віддаленими.
    Якщо ж взаємодія відбувається, то вона проходить в обох напрямках і відбувається наступним чином:
      Якщо один об'єкт діє на інший, то взаємодія відбувається по кількох питаннях, які є найбільш вагомими для нього, тобто мають найбільше значення погляду по модулю. При взаємодії по конкретному погляду використовуємо наступну формулу впливу:

$$\underset{x \to y}{influence} = \frac{1}{2}\left|x_{p_i} - y_{p_i}\right| \cdot \underset{x \to y}{k_{infl}}, \quad (1)$$

      де x_{p_i} та y_{p_i} – значення і-го погляду для об'єкта x та y відповідно,
      $\underset{x \to y}{k_{infl}} = \frac{\cos\left(\pi \cdot y_{p_i}\right) + 1}{2} \cdot \Delta$, Δ – максимально можлива зміна погляду за ітерацію.
  – Проводимо взаємодію між об'єктами одного рівня, які не знаходяться в одній групі. Для цього випадковим чином обираємо деяку кількість пар об'єктів, та проводимо взаємодію між ними як було описано вище.
– Етап другий: взаємодія між об'єктами сусідніх рівнів.
  – Складові об'єкта діють системою поглядів сформованою як середнє арифметичне їх поглядів на цей об'єкт за формулою (1).
  – Взаємодія із складовими нижчого рівня проводиться з кожним об'єктом без виключення. Вплив відбувається за формулою (1).
– Етап третій: вплив ЗМІ.
  Вплив відбувається односторонньо.

– Для кожної політичної сили вибираємо об’єкт другого або вищого рівня і діємо з деякою ймовірністю на всі елементи в середині вибраного об’єкта за формулою (1).
– Для політичної сили, яка знаходиться при владі береться найвищий об’єкт.

Процес виборів:

Для кожного виборця знаходимо найближчу політичну силу. Відстань розраховуємо за формулою:

$$d(x,y)=\sum_{i}^{n_p}\begin{cases} sgn\left(x_{p_i}\right)+sgn\left(y_{p_i}\right)\neq 0, \left|x_{p_i}-y_{p_i}\right| \\ sgn\left(x_{p_i}\right)+sgn\left(y_{p_i}\right)=0, \left|x_{p_i}-2\cdot y_{p_i}\right| \end{cases}$$

де n_p – кількість значень у системі поглядів, sng – сигнум функція, x_{p_i} та y_{p_i} – значення і-го погляду для виборця та політичної сили відповідно.

Якщо відстань не перевищує деякого порогового значення, зараховуємо голос цій політичній силі.

Висновки

У статті запропонована ієрархічна самоподібна модель сучасного суспільства для прогнозування перерозподілу влади у ньому між політичними силами під час виборів. Модель має покроковий опис алгоритму, що робить досить простим завдання її комп’ютерної реалізації та дозволить проводити чисельні експерименти з нею за допомогою комп’ютера.

Побудована модель може бути використана для спроби прогнозування результатів виборів у сучасному суспільстві. Більшість вхідних даних можна взяти з статистичних даних (структуру рівнів, кількість виборців) та соціальних опитувань (питання, які найбільш важливі для суспільства).

Наведена модель була розроблена з урахуванням можливості комп’ютерного моделювання і аналізу результатів, що робить її більш привабливою та затребуваною. Для того, щоб модель була більше наближеною до реальності було б добре враховувати корупцію. Однак корупція здебільшого зумовлена бажанням покращити матеріальний стан в той чи інший спосіб. Для цього у майбутніх дослідженнях можна ввести економічну складову, що суттєво ускладнить модель.

Список використаної літератури:

1. Easton D. An Approach to the Analysis of Political Systems // Political System and Change. Princeton, N. J., 1986. P. 24.
2. Almond Gabriel A. The Political of Developing Areas / Gabriel A. Almond and James Coleman, Princeton, NJ.: Princeton University Press, 1960. P. 7.
3. Михайлов А. П. Моделирование системы «власть-общество». — М.: Физмат-лит, 2006. - 144 с.
4. М. Г. Дмитриев, А. А. Павлов, А. П. Петров, Модель «власть-общество-экономика» для случая слабокоррумпированной дискретной иерархии, Матем. моделирование, 2012, том 24, номер 2, 120–128
5. Прончева О.Г. Модель системы 'Власть–Информация–Общество' // Препринты ИПМ им. М.В.Келдыша. 2018. № 11. 15 с.
6. Дмитриев М.Г., Петров А.П. Математическая модель «Власть-общество-инфраструктура-производство» // В сб. «Тезисы докладов и выступлений Всероссийского социологического конгресса: Глобализация и социальные изменения в современной России, т.11, 3-5 октября 2006 года». М.: Альфа-М, 2006, с.123-124.
7. Дмитриев М.Г. Макаров Д.А. Павлов А.А. Кафарова М.В. Стабилизация в нелинейной модели «власть – общество – экономика». // Математическое моделирование и информатика социальных процессов, 2016, 18. 54-67
8. Петров А.П., Степанцов М.Е. Дискретная распределенная модификация модели «власть–общество» на основе клеточного автомата // Препринты ИПМ им. М.В.Келдыша. 2014. No 100. 19 с.

9. Степанцов М.Е. Учет экономических факторов, коррупции, и транспортных связей в модели "власть-общество" на основе клеточного автомата// математическое моделирование социальных процессов: сборник трудов, выпуск 19. — М.: ИПМ им. М.В.Келдыша, 2017. — 140 с.
10. Степанцов М.Е. Моделирование системы «власть-общество-экономика» на основе клеточного автомата // Компьютерные исследования и моделирование, 2016, т. 8, № 3, с. 561-572

Bibliography:

1. Easton D. (1986) An Approach to the Analysis of Political Systems. Political System and Change. Princeton, p. 24
2. Almond A., Coleman J., (1960) The Political of Developing Areas, Princeton University Press, p. 7
3. Mikhailov A. P. (2006) Modellirovaniie sistiemy "Vlast-Obshchiestvo" ["Power-Society" model]. Phizmat-lit – PhysMath-lit [In Russian]
4. Dmitriev M.G., Pavlov A.A., Petrov A.P. Model "Vlast- Obshchiestvo-Ekonomika" dlia sluchaia slabokorrumpirovannoi diskrietnoi iierarkhii [The "Power-Society-Economy" Model With A Slightly Corrupt Discrete Hierarchy]. Matem. modellirovaniie – Math. Modeling, vol. 24 (2), 120–128 [In Russian]
5. Proncheva O.G. (2018) "Vlast-informatsiia-obshchestvo" model ["Power-Information-Society" model]. Preprynti IPM im. M.V. Keldesha – Preprints of KIAM named after M.V. Keldysh. 11, p. 15 [In Russian]
6. Dmitriev M.G., Petrov A.P. (2006, Oct. 3-5) Matiematicheskaia model "Vlast- Obshchiestvo-Infrastruktura-Proizvodstvo" [«Power-Society-Infrastructure-Production » mathematical model]. V sbornikie "Tezisy dokladov i vystuplenii Vserosiiskogo sotsiologicheskogo kongressa: Globalizatsiia i sotsialniie izmenieniia v sovremmenoi Rossii" – In compilation "Theses of reports and speeches of the All-Russian Sociological Congress. Globalization and social changes in modern Russia", vol.11, 123-124 [In Russian]
7. Dmitriev M.G. Makarov D.A. Pavlov A.A. Kafarova M.V. (2016)Stabilizatsiia v nelineinoi modeli "Vlast-Obshchiestvo- Ekonomika" [Stabilization in a nonlinear model « Power-Society-Economy»]. Mattiematicheskoie moddelirovaniie i informatyka sotsialnikh protsesov – Mathematical modeling and informatics of social processes, 18. 54-67 [In Russian]
8. Petrov A.P., Stepantsov M.E. (2014) Diskrietnaia raspriedieliennaia model "Vlast- Obshchiestvo" na osnovie kletochnogo avtomata [Discrete distributed modification of the "Power-Society" model based on a cellular automaton]. Preprynti IPM im. M.V. Keldesha – KIAM Preprints M.V. Keldysh., 100, p.19. [In Russian]
9. Stepantsov M.E. (2017) Uchiet ekonomicheskikh faktorov, korruptsii i transportnikh sviazei v modeli "Vlast-Obshchiestvo" na osnovie kletochnogo avtomata [Consideration of economic factors, corruption, and transport ties in the "Power-Society" model based on a cellular automaton]. Metiematicheskoie modelirovaniie sotsialnikh protsessov – Mathematical Modeling of Social Processes: Proceedings, Issue 19. KIAM Preprints M.V. Keldysh,, p. 140 [In Russian]
10. Stepantsov M.E. (2016) Modelirovaniie systemy "Vlast- Obshchiestvo-Ekonomika" na osnovie klietochnogo avtomata [Modeling of the "Power-Society-Economy" system based on a cellular automaton].Kompiuternoie issliedovanie i modelirovaniie – Computer Research and Modeling, vol. 8 (3), 561-572 [In Russian]

MORHUN Mykola,
PhD student, The Bohdan Khmelnytsky National University of Cherkasy

HIERARCHICAL SELF-SIMILAR MODEL OF POWER REDISTRIBUTION IN MODERN SOCIETY

Summary. Introduction. *Nowadays more and more interest is being paid to social processes. In the second part of XX century some models related to society began to emerge. The most famous are model of a political system by D. Easton and structural and functional model by G. Almond. With significant technical progress, computing resources grew significantly and this caused demand to move from descriptive and diagram style models to more strict and formal to be able to use computer simulation. One of such models is "power-society" model by A. Mikhailov. That model considers distribution of power between hierarchical levels of a government branch and between people and government itself. But one of the very important questions in modern society is distribution of power between political forces and election result forecasting, on which this article is focusing.*

Purpose. *The aim of this article is to create a model of power redistribution between political forces in modern society.*

The article discusses the nature of power which a society has over its members and explains some aspects of its moving.

In the article is suggested a model of modern society, using which it is possible to simulate power redistribution via election, i.e. try to predict elections result. The model considers not only citizens, but also governmental structures, their political position and influence to citizens and vice versa.

In the article discussed structure of a model unit including its internal blocks and connections as well as external interactions and political views. With introduced structure of a model unit, corruption mechanism might be explained. The suggested model unit structure is self-similar that simplifies the model representation for computer simulation.

The approach used in the model could be used to represent a country or its subpart as well as relations between different countries, i.e. a world might be simulated using this model.

A more formal description is given to simplify computer implementation of the model.

Conclusions. *Model of power redistribution between political forces by election has been developed. The model is hierarchical and self-similar. Given algorithm of computer implementation of the model.*

Keywords: *model, power, elections, society.*

Одержано редакцією 21.06.2018 р.
Прийнято до публікації 19.09.2018 р.

СЕКЦІЯ «ІНФОРМАТИКА»

УДК 004.85:519.6

DOI 10.31651/2076-5886-2019-1-42-53

PACS 02.70.Wz, 07.05.Kf, 07.05.Mh, 07.05.Tp

ПІСКУН Олександр Варфоломійович,
к. т. н., доцент, доцент кафедри автоматизації та комп'ютерно-інтегрованих технологій, Черкаський національний університет імені Богдана Хмельницького
e-mail: piskun@ukr.net
ORCID 0000-0001-5334-6337

ЗАСТОСУВАННЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ ДЛЯ ПОБУДОВИ МОДЕЛІ РІШЕННЯ ЗАДАЧІ КЛАСИФІКАЦІЇ

У роботі наведено алгоритм побудови моделі рішення задачі класифікації щодо визначення наявності або відсутності захворювань серця у людини на основі методів машинного навчання. Попередня обробка та аналіз даних дали можливість виявити нелінійну залежність цільової змінної, а також підготувати дані для ефективного моделювання. Для побудови моделі класифікатора застосований метод SVM з rbf ядром. Оптимізація параметрів моделі проведена, використовуючи пошук по сітці з крос-валідацією кожної комбінації параметрів.

Ключові слова: *машинне навчання, класифікація, метод опорних векторів, інтелектуальна обробка даних, t-SNE візуалізація.*

Постановка проблеми

Задача класифікації - одна з найбільш поширених задач в аналізі даних і розпізнаванні образів. Вона має широку область практичних застосувань в автоматиці, управлінні, економіці, соціології, медицині, геології, астрономії, ядерній фізиці і т.д.

Класифікація – це один із розділів машинного навчання, що вирішує наступну задачу: нехай маємо множину об'єктів, що розділені на класи за деякою або рядом ознак. Окрім цього, задана скінчена множина об'єктів, щодо яких відомо, яким саме класам вони належать. Цю множину називають навчальною вибіркою. До якого класу відносяться решта об'єктів – невідомо. Необхідно побудувати алгоритм, що буде здатний класифікувати будь-який з об'єктів вихідної множини. Під класифікацією об'єкта розуміють відповідь алгоритму, що вказує номер (або назву) класу, до якого він відноситься [1].

Виклад основного матеріалу

Розглянемо найбільш поширені алгоритми класифікації даних [2].

Дерево рішень. Принцип даного методу полягає в рекурсивному розбитті множини елементів на підмножини, що містять елементи, які належать одному класу. Різні різновиди дерев рішень відрізняються знаходженням цільової змінної та її атрибутів для побудови нового вузла.

Найбільш поширеним і успішним алгоритмом є алгоритм, в якому вибір наступної змінної та її атрибутів ґрунтується на визначенні взаємозв'язку між ентропією та класифікованої інформацією. Вибір зупиняється на елементах, з мінімальним значенням ентропії і максимальним значенням інформативності відповідно.

Байєсова класифікація. Алгоритм базується на теоремі Байеса та передбачає незалежність розглянутих ознак. Метод використовує ймовірності, отримані в ході навчання за тестовою вибіркою, а саме апріорні ймовірності належності будь-якого з розглянутих об'єктів даного класу та умовні ймовірності того, що об'єкт, який належить до даного класу, має даний набір ознак. Потім визначається ймовірність належності даного об'єкту з тестового набору до кожного з класів. Вибирається змінна з найвищою ймовірністю.

Метод k найближчих сусідів. Метод передбачає, що черговий об'єкт з тестового набору належить тому класу, до якого належать більшість його найближчих «сусідів» з навчальної вибірки. Сусідами, в даному випадку, називають найбільш близькі до тестового об'єкту спостереження. Таким чином, новий об'єкт поміщається в клас, що містить найбільшу кількість найближчої схожості. Параметр k, що задається, визначає кількість розглянутих найближчих сусідів, тобто визначає скільки сусідів буде впливати на класифікацію. Близькість визначається «відстанню» між об'єктами.

Метод опорних векторів. Даний метод передбачає використання гиперплощини для поділу простору на підпростори, що характеризують окремі класи. Об'єкти, які лежать при цьому на кордоні освічених областей, називають опорними векторами. Метод намагається провести дане розбиття таким чином, щоб відстань між елементами різних класів була максимальною. Щодо вибору розділяючої гиперплощини, це означає її максимальне віддалення щодо найближчих точок обох класів. У разі виникнення проблеми лінійної нероздільності, використовується перетворення ядра.

Метою даної роботи є покроковий виклад та застосування на практиці алгоритму побудови моделі для вирішення задачі класифікації.

Види статистичних даних [3]

Статистичні дані можуть бути представлені як кількісними (числовими неперервними або дискретними), так і якісними (категоріальними порядковими або номінальними) змінними.

Кількісні (числові) дані – це дані, які приймають деякі числові значення. Вони, в свою чергу, поділяються на: дискретні дані, які можуть приймати строго визначені значення, та неперервні, які можуть бути представлені будь-якими значеннями.

Категоріальні дані застосовуються для опису стану об'єкта шляхом присвоєння йому номера, відповідного до категорії, куди цей об'єкт належить. Важливою умовою для застосування категоріальних даних є приналежність одного об'єкта дослідження тільки до однієї можливої категорії для одного критерію.

Якісні номінальні дані використовуються в тому випадку, якщо категорії не впорядковані. Числа в даному випадку є лише позначенням для стану об'єкта і не впорядковують цей стан. Наприклад, по статі: 1 – чоловіча, 2 – жіноча.

Якісні порядкові (рангові, ординарні) дані – це дані, для яких категорії можуть бути впорядковані. Наприклад, від поганого самопочуття до гарного: 1 – гарне, 2 – задовільне, 3 – погане.

Опис набору даних

В якості експериментальних даних для дослідження використовується набір даних (датасет) з серцевих захворювань Statlog (Heart) з репозиторію UCI ML [4]. Дані містять 13 атрибутів (табл. 1) і 180 спостережень.

Змінна для прогнозування **y** може приймати два значення: 0 – відсутність, 1 – наявність серцевих захворювань.

Дослідження проводимо з використанням бібліотеки машинного навчання Scikit-Learn, в середовищі Python 3.

Таблиця 1

Атрибути експериментальних даних з серцевих захворювань

№	Назва	Опис	
1	age	вік в роках	
2	sex	стать	1 = чоловіча
			0 = жіноча
3	chest pain type (4 values)	тип болю в грудях (4 значення: 1, 2, 3, 4)	
4	resting blood pressure	артеріальний тиск у спокої (мм рт. ст.)	
5	serum cholestoral in mg/dl	рівень холестерину в крові в мг/дл	
6	fasting blood sugar > 120 mg/dl	рівень цукру в крові > 120 мг/дл	1 = так
			0 = ні
7	resting electrocardiographic results (values 0, 1, 2)	електрокардіографічні результати в стані спокою (значення 0, 1, 2)	
8	maximum heart rate achieved	максимальна частота серцевих скорочень	
9	exercise induced angina	стенокардія, викликана фізичними вправами	1 = так
			0 = ні
10	oldpeak = ST depression induced by exercise relative to rest	пониження сегменту ST на ЕКГ, викликане фізичними вправами відносно спокою	
11	slope of the peak exercise ST segment	нахил пікового сегмента ST	1 = зростає
			2 = фіксований
			3 = убуває
12	number of major vessels (0-3) colored by fluoroscopy	кількість основних судин (0-3), забарвлених при флюороскопії	
13	thal: 3 = normal; 6 = fixed defect; 7 = reversable defect	тип дефекту	3 = в межах норми
			6 = фіксований дефект
			7 = оборотний дефект

Підготовка та аналіз даних

Завантажуємо дані про захворювання серця. Дані мають наступний вигляд (перші 5 записів):

```
In [123]: X.head()
Out[123]:
           slope_of_peak_exercise_st_segment              thal  \
patient_id
0z64un                                     1            normal
ryoo3j                                     2            normal
yt1s1x                                     1            normal
l2xjde                                     1  reversible_defect
oyt4ek                                     3  reversible_defect
           resting_blood_pressure  chest_pain_type  num_major_vessels  \
patient_id
0z64un                        128                2                  0
ryoo3j                        110                3                  0
yt1s1x                        125                4                  3
l2xjde                        152                4                  0
oyt4ek                        178                1                  0
           fasting_blood_sugar_gt_120_mg_per_dl  resting_ekg_results  \
patient_id
0z64un                                        0                    2
ryoo3j                                        0                    0
yt1s1x                                        0                    2
l2xjde                                        0                    0
oyt4ek                                        0                    2
```

```
           serum_cholesterol_mg_per_dl  oldpeak_eq_st_depression  sex  age  \
patient_id
0z64un                             308                       0.0    1   45
ryoo3j                             214                       1.6    0   54
yt1s1x                             304                       0.0    1   77
l2xjde                             223                       0.0    1   40
oyt4ek                             270                       4.2    1   59
           max_heart_rate_achieved  exercise_induced_angina
patient_id
0z64un                         170                        0
ryoo3j                         158                        0
yt1s1x                         162                        1
l2xjde                         181                        0
oyt4ek                         145                        0
```

Перевіряємо наші дані на наявність пропущених значень:

```
In [125]: X.isnull().sum()
Out[125]:
slope_of_peak_exercise_st_segment     0
thal                                  0
resting_blood_pressure                0
chest_pain_type                       0
num_major_vessels                     0
fasting_blood_sugar_gt_120_mg_per_dl  0
resting_ekg_results                   0
serum_cholesterol_mg_per_dl           0
oldpeak_eq_st_depression              0
sex                                   0
age                                   0
max_heart_rate_achieved               0
exercise_induced_angina               0
dtype: int64
```

Пропусків немає.

Проводимо перевірку наявності дисбалансу цільових класів (функція countplot, рис. 1).

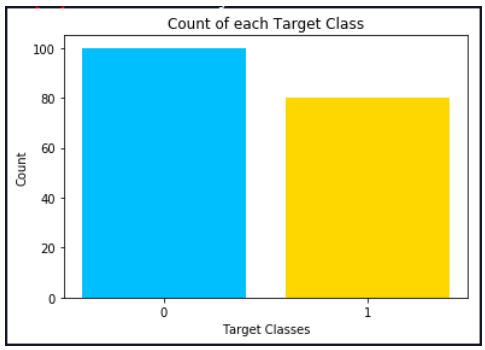


Рис. 1

Класи можна вважати збалансованими (об’єктів класу «0» - 100, «1» - 80).

Розглянемо, наскільки відрізняються значення атрибутів для цільових класів. Для цього побудуємо діаграму розсіювання для кожного з атрибутів за допомогою функції swarmplot, в якій точки коригуються уздовж категорійної осі так, щоб вони не перекривалися. Це дає кращу картину розподілу значень:

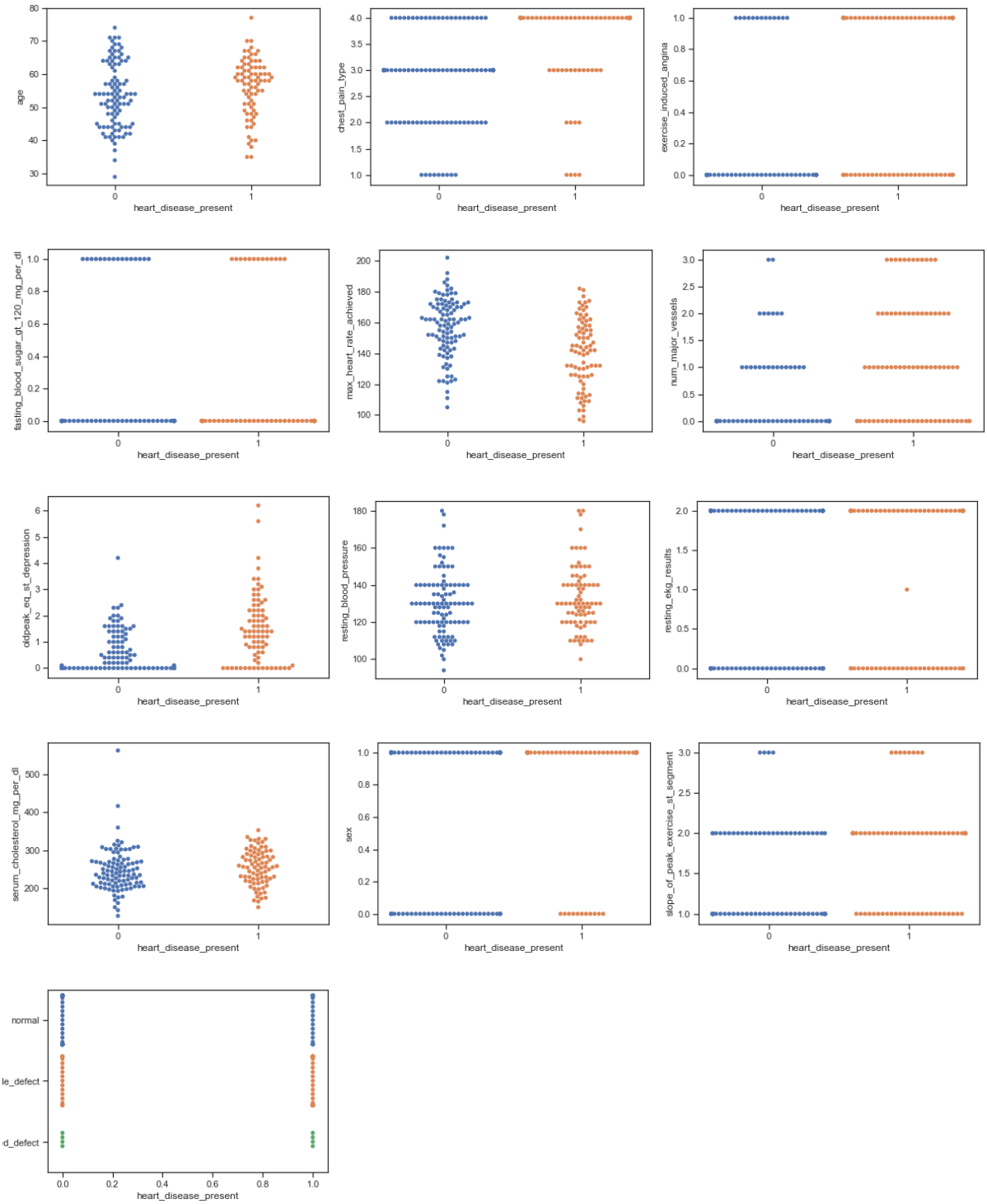


Рис. 2

Як видно з діаграм, незважаючи на присутність деяких відмінностей в значеннях атрибутів у різних цільових груп, ми не можемо чітко їх розмежувати. Це вказує на необхідність застосування класифікатора для побудови моделі.

Виконаємо обробку даних: перетворимо категоріальні дані у бінарні, нормалізуємо усі числові до одного діапазону [0..1].

```
# Data Preprocessing

## Handle Categorical features
def Preprocessing_categorical(X):

    import pandas as pd

    trainThal = pd.get_dummies(X['thal'])
    trainSlope = pd.get_dummies(X['slope_of_peak_exercise_st_segment'], prefix='slope')
    trainChestPain = pd.get_dummies(X['chest_pain_type'], prefix='chestPain')
    trainResting = pd.get_dummies(X['resting_ekg_results'], prefix='restingEkg')
    trainnum = pd.get_dummies(X['num_major_vessels'], prefix='num_major_vessels')

    X.drop(['thal', 'slope_of_peak_exercise_st_segment','chest_pain_type','resting_ekg_results', 'num_major_vessels'], axis=1, inplace=True)
    X = X.join([trainThal, trainSlope, trainChestPain, trainResting, trainnum])

    del trainThal, trainSlope, trainChestPain, trainResting, trainnum

    return (X)


## Normalization of numerical features
def Preprocessing_numerical(X):

    from sklearn.preprocessing import StandardScaler

    col_names = ['resting_blood_pressure', 'serum_cholesterol_mg_per_dl', 'oldpeak_eq_st_depression', 'age', 'max_heart_rate_achieved']
    features = X[col_names]
    scaler = StandardScaler().fit(features.values)
    X[col_names] = scaler.transform(features.values)

    return (X)
```

Проведемо t-SNE візуалізацію змінної **y**. T-Distributed Stochastic Neighbor Embedding (t-SNE) - це метод нелінійного зменшення розмірності, який добре підходить для візуалізації багатовимірних наборів даних [5]. Метод моделює кожен об'єкт простору високої розмірності дво- або тривимірною точкою таким чином, що близькі за характеристиками елементи даних в багатовимірному просторі (наприклад, датасета з великим числом стовпців) проектуються в сусідні точки, а різнорідні об'єкти з великою ймовірністю моделюються точками, які стоять далеко одна від одної.

```
def t_SNE_visualization(X,y):

    from sklearn.manifold import TSNE
    import matplotlib.pyplot as plt

    model = TSNE(learning_rate=50)  #50-200
    transformed = model.fit_transform(X)
    xs = transformed[:,0]
    ys = transformed[:,1]
    plt.scatter(xs, ys, c=y, alpha=0.5,cmap='viridis')

    return plt.show()
```

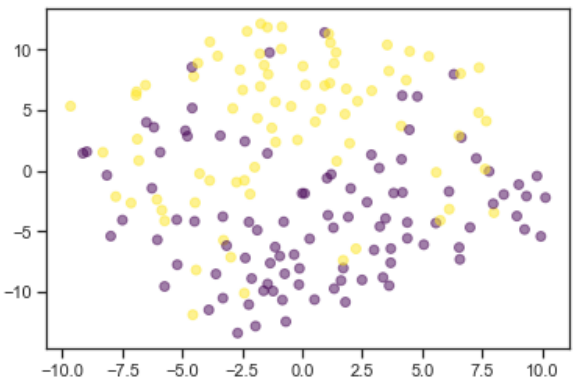


Рис. 3

Отримані результати показують, що ми маємо лінійно нероздільну вибірку.

Для побудови моделі класифікатора використаємо метод опорних векторів (SVM) с rbf (радіальна базова функція) ядром [6, 7].

Побудова моделі класифікатора

Розбиваємо вибірку на тренувальну та тестову (75% / 25%):

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=109)

Навчання моделі класифікатора проведемо на основі методу опорних векторів. Створимо об'єкт класифікатора, використовуючи параметри за замовченням. Проведемо навчання моделі на даних тренувальної вибірки. Використовуючи навчену модель, виконаємо передбачення класу даних тестової вибірки та визначимо базову оцінку якості моделі класифікатора:

```
from sklearn.svm import SVC
clf = SVC(kernel="rbf", probability=True)
clf.fit(X_train, y_train)

print('Train Accuracy:', clf.score(X_train, y_train))
print('Test Accuracy:', clf.score(X_test, y_test))
```

Ми отримали точність на тренувальному наборі 0,89 та 0,8 на тестовому. Для підвищення якості моделі необхідна оптимізація параметрів.

Побудуємо криві навчання.

```
## Learning curves
Model_Development_Functions.Learning_curves(clf,X_train,y_train)
```

```
def Learning_curves(estimator,X_train,y_train):

    import numpy as np
    from sklearn.svm import SVC
    from sklearn.model_selection import learning_curve
    import matplotlib.pyplot as plt

    train_sizes, train_scores, test_scores = \
        learning_curve(estimator=estimator,
                       X=X_train,
                       y=y_train,
                       cv=10,
                       n_jobs=-1,
                       train_sizes=np.linspace(0.1, 1.0, 10))

    train_scores_mean = np.mean(train_scores, axis=1)
    train_scores_std = np.std(train_scores, axis=1)
    test_scores_mean = np.mean(test_scores, axis=1)
    test_scores_std = np.std(test_scores, axis=1)

    plt.plot(train_sizes, train_scores_mean, 'o-', color="r", label="Training score")
    plt.fill_between(train_sizes, train_scores_mean - train_scores_std,
                     train_scores_mean + train_scores_std, alpha=0.1,
                     color="r")

    plt.plot(train_sizes, test_scores_mean, 'o-', color="g", label="Cross-validation score")
    plt.fill_between(train_sizes, test_scores_mean - test_scores_std,
                     test_scores_mean + test_scores_std, alpha=0.1, color="g")

    plt.xlabel('Training examples')
    plt.ylabel('Score')
    plt.legend(loc="best")
    plt.ylim([0.4, 1.0])
    plt.grid()

    return plt.show()
```

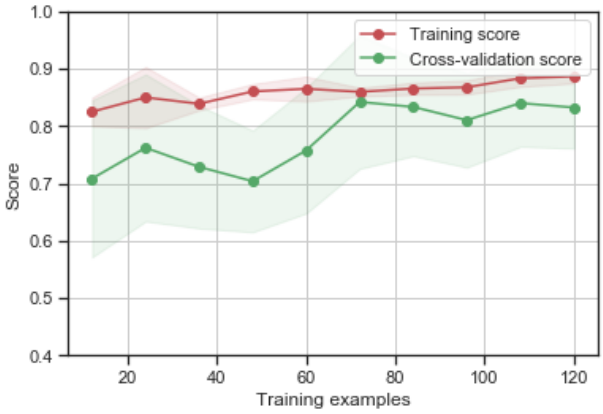


Рис. 4

Отримані криві навчання показують, що збільшення об’єму тренувальної вибірки може покращити якість моделі.

Побудуємо криві валідації для параметру C:

```
## Validation curves
param_name= 'C'
param_range = [0.001, 0.01, 0.1, 0.25, 0.5, 0.75, 1.0, 10.0, 100.0]
Model_Development_Functions.Validation_curves(clf, param_range, param_name, X_train, y_train)
```

```
def Validation_curves(estimator, param_range, param_name, X_train, y_train):

    import numpy as np
    from sklearn.svm import SVC
    import matplotlib.pyplot as plt
    from sklearn.model_selection import validation_curve

    train_scores, test_scores = validation_curve(
            estimator=estimator,
            X=X_train,
            y=y_train,
            param_name=param_name,
            param_range=param_range,
            cv=10,
            n_jobs=-1)

    train_mean = np.mean(train_scores, axis=1)
    train_std = np.std(train_scores, axis=1)
    test_mean = np.mean(test_scores, axis=1)
    test_std = np.std(test_scores, axis=1)

    plt.plot(param_range, train_mean, 'o-', color="r", label="Training score")
    plt.fill_between(param_range, train_mean - train_std,
                         train_mean + train_std, alpha=0.1,
                         color="r")

    plt.plot(param_range, test_mean, 'o-', color="g", label="Cross-validation score")
    plt.fill_between(param_range, test_mean - test_std,
                         test_mean + test_std, alpha=0.1, color="g")

    plt.xscale('log')
    plt.xlabel('Parameter ' + param_name)
    plt.ylabel('Score')
    plt.legend(loc="best")
    plt.ylim([0.4, 1.0])
    plt.grid()

    return plt.show()
```

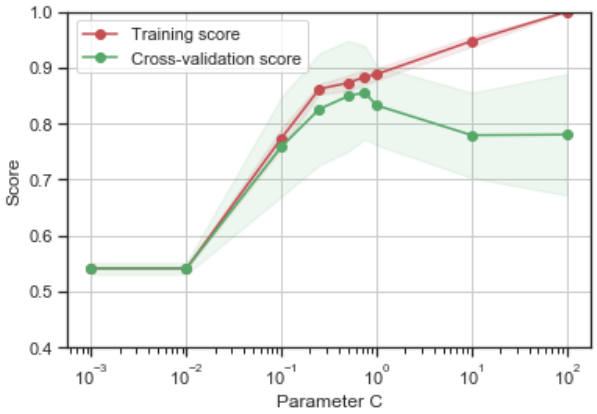


Рис. 5

Ми бачимо, що точність на тренувальній вибірці зростає зі збільшення параметра C, але на тестовому наборі вона починає падати з C = 0,75.

Побудуємо криві валідації для параметру gamma (γ) при C = 0,75:

```
clf = SVC(kernel="rbf", probability=True, C=0.75)
param_name= 'gamma'
param_range = np.linspace(0.0001, 10, 100)
Model_Development_Functions.Validation_curves(clf, param_range, param_name, X_train, y_train)
```

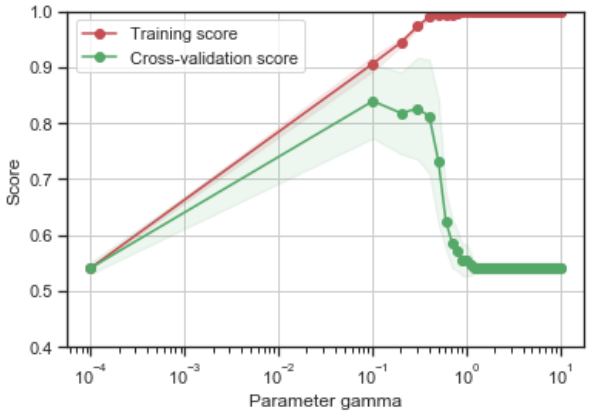


Рис. 6

Виходячи з отриманих кривих, оптимальним значенням gamma є 0,1.

За допомогою кривих валідації ми отримали, в першому наближенні, область оптимальних значень параметрів. Далі, для визначення найкращої комбінації C і γ використаємо пошук по сітці. Для цього згенеруємо по 100 значень параметрів C і γ. Таким чином, ми отримаємо сітку 100x100 комбінацій C і γ для нашої моделі. Кожна комбінація перевіряється з використанням крос-валідації, і вибирається та, яка проявила себе краще за всіх інших. Фінальна модель, яка використовується для тестування та класифікації нових даних, навчається потім на всій множині з використанням обраних параметрів.

```
## Grid_Search_CV Hyperparameter Tuning
param_grid = {"C": np.linspace(0.1, 1, 100), "gamma": np.linspace(0.005, 0.4, 100)}
best_clf=Model_Development_Functions.Grid_Search_CV(clf, param_grid, X_train, y_train, X_test, y_test)
```

```
def Grid_Search_CV(estimator, param_grid, X_train, y_train, X_test, y_test):

    import time
    from sklearn.svm import SVC
    from sklearn.model_selection import GridSearchCV

    searchCV = GridSearchCV(estimator, param_grid, cv=10, scoring='accuracy', n_jobs = 3, verbose = 0)
    start = time.time()
    searchCV.fit(X_train, y_train)
    end = time.time()

    print("Elapsed time: " + str(round((end-start)/60)) + " min.")

# Examine the best model
    print('Examine the best model:')
    print('Best Score: ' + str(searchCV.best_score_))
    print('Best Parameters: ' + str(searchCV.best_params_))
    print('Best Estimator: ' + str(searchCV.best_estimator_))

    best_svm=searchCV.best_estimator_
    best_svm.fit(X_train, y_train)

    print('Training Accuracy:', best_svm.score(X_train, y_train))
    print('Testing Accuracy:', best_svm.score(X_test, y_test))

    return best_svm
```

Результати оптимізації:

```
In [136]: best_clf=Model_Development_Functions.Grid_Search_CV(clf, param_grid, X_train, y_train, X_test, y_test)
Elapsed time: 4 min.
Examine the best model:
Best Score: 0.8592592592592593
Best Parameters: {'C': 0.5272727272727272, 'gamma': 0.04090909090909091}
Best Estimator: SVC(C=0.5272727272727272, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.04090909090909091,
  kernel='rbf', max_iter=-1, probability=True, random_state=None,
  shrinking=True, tol=0.001, verbose=False)
Training Accuracy: 0.8740740740740741
Testing Accuracy: 0.8222222222222222
```

Таким чином, оптимальними значеннями параметрів C і γ є:
best_clf =SVC(C=0.527, gamma=0.041, kernel='rbf')
При цьому точність моделі на тестовому наборі збільшилась до 82,2%.

Висновки

Для побудови моделі, насамперед, необхідно провести аналіз наявних даних, що дасть можливість визначити складність поставленого завдання та звузить коло відповідних методів. Далі необхідно провести чистку і обробку даних, тобто підготовку до моделювання. Потім застосовується обрана модель, в нашому випадку класифікації, з її подальшою оптимізацією. При підборі параметрів необхідно використовувати крос-валідацію як на тренувальній та тестовій вибірках, так і всередині тренувального набору даних. Таким чином можна отримати високу точність моделі при її робастності.

Список використаної літератури:

1. Классификация [Електронний ресурс]. – Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Классификация.
2. Машинное обучение (курс лекций, К. В. Воронцов) [Електронний ресурс]. – Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Машинное_обучение_%28курс_лекций%2C_К.В.Воронцов%29.
3. Малик І.В. Методи машинного навчання для статистичної обробки медичних даних / І.В. Малик, Т.В. Кнігніцька // Науковий вісник Чернівецького національного університету. Серія: Комп'ютерні системи та компоненти. – 2017. – Том 8, випуск 2. – С. 77 – 85.

4. Statlog (Heart) Data Set [Електронний ресурс]. – Режим доступу: http://archive.ics.uci.edu/ml/datasets/statlog+(heart).
5. L.J.P. van der Maaten. Visualizing High-Dimensional Data Using t-SNE / L.J.P. van der Maaten, G.E. Hinton // Journal of Machine Learning Research. – 2008. – № 9. – P. 2579 – 2605.
6. Cortes C. Support-vector networks / Cortes C., Vapnik V. // Machine Learning. – 1995. – № 20 (3). – P. 273–297.
7. Cristianini N. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods / N. Cristianini, J. Shawe-Taylor. – Cambridge, U.K.: Cambridge University Press, 2000. – 189 p.

Bibliography:

1. Klassifikatsiya [Classification]. (2011). Retrieved from http://www.machinelearning.ru/wiki/index.php?title= Classification [in Russian]
2. Mashinnoe obuchenie (kurs lektsiy, K. V. Vorontsov) [Machine learning (lecture course, K.V. Vorontsov)]. (2019). Retrieved from http://www.machinelearning.ru/wiki/index.php?title=Machine_learning_%28course_lecture%2C_K.V.Vorontsov%29 [in Russian]
3. Malyk I.V., Knizhnitskaya T.V. (2017). Metody mashynnoho navchannia dlia statystychnoi obrobky medychnykh danykh [Methods of machine learning for statistical processing of medical data]. Naukovy Visnyk Chernivetskogo Natsionalnogo Universitetu. Seriia: Kompiuterni systemy ta komponenty, – Scientific Bulletin of Chernivtsi National University. Series: Computer Systems and Components , v. 8, is. 2, 77–85 [in Ukrainian].
4. Statlog (Heart) Data Set. (n.d.). Retrieved from http://archive.ics.uci.edu/ml/datasets/statlog+(heart).
5. L.J.P. van der Maaten and G.E. Hinton. (2008) Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research, 9(Nov), 2579-2605.
6. Cortes C., Vapnik V. (1995) Support-vector networks. Machine Learning, 20 (3), 273–297.
7. Nello Cristianini, John Shawe-Taylor. (2000) An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge, U.K.: Cambridge University Press.

PISKUN Oleksandr,
PhD, Associate Professor of Department of Automation and Computer-integrated Technologies, The Bohdan Khmelnytsky National University of Cherkasy

CLASSIFICATION MODEL BUILDING USING MACHINE LEARNING METHODS

Summary. Introduction. *Classification is one of the machine learning fields that solves the following problem: let there be a multitude of objects separated into classes according to one or several attributes. A finite set of objects is given for which it is known which classes they belong to. What class the rest of the objects belong to is unknown. It is necessary to build an algorithm capable of classifying an arbitrary object of the original set.*

Let's consider the most common data classification algorithms.

Decision tree. *The principle of this method is to recursively split the set of elements into subsets containing elements that belong to the same class. Different types of decision trees differ in finding the target variable and its attributes to build a new node.*

Bayesian classification. *Bayesian classification belongs to statistical classification methods that predict an object's class with the help of the probability theory. The algorithm is based on the Bayes theorem and assumes the independence of the considered attributes.*

K-nearest neighbors. *The method assumes that the next object from the test set belongs to the class to which the majority of its closest "neighbors" from the training set belong. In this case, the closest observations to the test object defined as neighbors. Thus, the new object is placed in the class containing the largest number of closest similarities.*

Support Vector Machine. *This method uses a hyperplane to divide space into subspaces that characterize individual classes. The objects lying on the border of the formed regions are called support vectors. In the case of a linear inseparability problem kernel transformation is used.*

Purpose. *The purpose of this paper is a step-by-step presentation and practical application of the algorithm for the classification model building.*

Results. *Application of the provided algorithm for the classification model building to determine the presence or absence of heart disease based on the real data set made it possible to achieve high efficiency and robustness of the model.*

Conclusion. *The paper presents an algorithm for building a model of the classification problem solving to determine the presence or absence of heart disease based on machine learning methods.*

Data pre-processing and analysis enabled identification of the non-linear dependencies of the target variable and prepared the data for effective modeling. SVM method with Rbf Kernel was applied for the classification model building. The model parameters were optimized using a grid search with cross-validation of each combination of the parameters.

Building a model first of all it is necessary to analyze the available data in order to reveal the complexity of the given problem and narrow down the range of suitable methods. Next, it is needed to clean and pre-process the data – prepare it for modeling. Then the selected method should be applied and optimization of the parameters should be provided. By the parameter selection, it is crucial to use cross-validation both on the training and test samples, and within the training data set itself. Thus, it is possible to achieve high accuracy of the model with a good degree of its robustness.

Keywords: *machine learning, classification, support vector machine, data mining, t-SNE visualization.*

Одержано редакцією 06.09.2018 р.
Прийнято до публікації 21.11.2018 р.

УДК 004.032.26

DOI 10.31651/2076-5886-2019-1-53-60

PACS 07.05.Mh, 07.05.Kf, 07.05.Pj

КРАСНОШЛИК Наталія Олександрівна
кандидат технічних наук, доцент,
доцент кафедри прикладної математики та інформатики Черкаського національного університету імені Богдана Хмельницького
e-mail: wlik007@ukr.net
ORCID 0000-0003-4661-6997

СЕРДЮК Марина Олександрівна
студентка 4 курсу спеціальності «Прикладна математика», Національний університет «Львівська політехніка», м. Львів
e-mail: mary_serdyuk@ukr.net
ORCID 0000-0003-4661-1234

ЗАСТОСУВАННЯ АНСАМБЛІВ НЕЙРОННИХ МЕРЕЖ ДЛЯ РОЗВ'ЯЗАННЯ ЗАДАЧІ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ

У роботі описано існуючі підходи до побудови ансамблів моделей у машинному навчанні. Наведено найбільш популярні системи для навчання нейронних мереж. У якості базової моделі обрано двошарову нейрону мережу прямого розповсюдження, що має один прихований шар. Розглянуто два підходи до побудови ансамблю нейронних мереж, такі як усереднюючий ансамбль та ансамбль з керівником. Вони були реалізовані за допомогою бібліотек Keras і TensorFlow. Проведено дослідження ефективності застосування ансамблів до розв'язання задач класифікації зображень. Для тестування обрано набір даних MNIST для класифікації рукописних цифр. Досліджено ефективність використання ансамблів різної структури з 3-9 нейронних мереж.

Ключові слова: *машинне навчання, задача класифікації, нейронна мережа, ансамбль моделей.*

Постановка проблеми

Останнім часом машинне навчання є однією з передових технологій сучасності. Машинне навчання як область практичної діяльності, і як сфера наукових досліджень алгоритмів, полягає у отримані знань з даних шляхом виявлення прихованих закономірностей в них. Перетворення даних у знання є особливо актуальною і цікавою

задачею, оскільки наразі людством накопичені великі об'єми цифрових даних. Замість того, щоб у ручному режимі виявляти правила і будувати моделі, методологія машинного навчання пропонує більш дієву альтернативу – автоматизоване поступове поліпшення якості прогнозних моделей за рахунок узагальнення відомих прикладів із опрацьованих ними масивів даних. Окрім того, що машинне навчання набуває все більшої вагомості у наукових дослідженнях в області інформатики, воно відіграє все більш значиму роль у повсякденному житті. Завдяки алгоритмам машинного навчання досягнуто значних успіхів у розпізнаванні мови і рукописного тексту, класифікації зображень, розроблено надійні поштові спам-фільтри, якісні пошукові системи, програми для виявлення шахрайської діяльності по кредитним картах та ін. [1].

Особливим розділом машинного навчання є глибоке навчання. Це інший підхід до пошуку представлення даних, який робить акцент на вивченні послідовних шарів все більш значущих представлень. Сучасне глибоке навчання часто залучає до процесу десятки і навіть сотні послідовних шарів представлення даних. Типовим прикладом моделі глибокого навчання, за допомогою яких вивчаються такі багатошарові представлення, є нейронні мережі [2].

Наразі існує багато типів нейронних мереж, які відрізняються одна від одної архітектурою, видом нейронів, структурою зв'язків, алгоритмами навчання. Досить часто одну і ту саму задачу можна розв'язати за допомогою різних нейронних мереж, при цьому вибір кращої з них не визначається строгими правилами. Якість розв'язання конкретної поставленої задачі може бути суттєво підвищена за допомогою ансамблів нейронних мереж, в яких одні і ті самі данні паралельно опрацьовують декілька нейронних мереж, вихідні сигнали яких деяким чином комбінуються в узагальнюючий результат [3].

Таким чином, об'єднання сукупності нейронних мереж у єдину модель є перспективним напрямом досліджень у глибокому навчанні. Передбачається, що нейронні мережі, об'єднані у ансамбль будуть мати кращу узагальнюючу здатність і дозволять розв'язувати більш складні задачі.

Більшість робіт, присвячених вивченню властивостей і можливостей ансамблевих структур з нейронних мереж, направлені на дослідження методів отримання (генерації) спільного кінцевого результату ансамблю. Підходи до об'єднання нейронних мереж у ансамблі розглянуто у [3-4], особливості об'єднання (створення ансамблю) для задач прогнозування часових рядів у [5-6], для задачі класифікації у [7].

Метою статті є дослідження ефективності застосування ансамблів нейронних мереж до розв'язання задач класифікації зображень.

Виклад основного матеріалу

1. Нейронні мережі та програмні засоби для їх реалізації

Прикладом моделі глибокого навчання є нейронна мережа прямого розповсюдження, або багатошаровий перцептрон. Багатошаровий перцептрон – це математична функція, що відображає множину вхідних значень на множину вихідних. Ця функція являє собою композицією декількох більш простих функцій. Таку композицію функцій можна представити у вигляді ланцюга, довжина якого і буде визначати глибину нейронної мережі. Сучасні архітектури нейронних часто мають досить складну структуру і складаються з десятків шарів.

На початковому етапі реалізація глибоких нейронних мереж потребувала значних зусиль від дослідників, але наразі ситуація змінилася. На сьогодні існує ряд зручних програмних засобів, які дозволяють побудувати модель нейронної мережі, сформувати

граф обчислень і виконати процес навчання, використовуючи при цьому різні рівні абстракції.

Найбільш популярними системами для навчання нейронних мереж є TensorFlow, Theano, Caffe, Torch і CNTK. Бібліотека TensorFlow розроблена у 2015 році компанією Google і побудована на графах операцій, які оперують з тензорами ы потоковою обробкою данних на графі. Бібліотека Theano розробляється з 2007 року, головним чином, групою MILA з Університету Монреаля. Основними принципами є: інтеграція з numpy, прозоре використання різних обчислювальних пристроїв (GPU/CPU), динамічна генерація оптимізованого коду на C. Theano, як і TensorFlow, використовує символьний підхід до обчислень. Бібліотека Caffe – одна з найперших популярних систем глибокого навчання, яка була розроблена у центрі комп'ютерного зору і навчання у Берклі. Caffe підтримує багато різних типів архітектур глибинного навчання, орієнтованих в першу чергу на класифікацію та сегментування зображень. Також вона містить найбільшу кількість готових до використання попередньо навчених моделей. Бібліотека Torch розроблена на мові Lua і надає зручний високорівневий інтерфейс для створення програм машинного навчання, аналогічний MATLAB. Висока продуктивність забезпечується, так само як і у Theano, за рахунок інтеграції з мовою C. Бібліотека CNTK (Cognitive Toolkit) розроблена компанією Microsoft, вона орієнтована саме на роботу з різними типами нейронних мереж. Набір засобів бібліотеки представляє нейронні мережі як порядок обчислювальних кроків через орієнтований граф [8]. Окремо варто зазначити бібліотеку Keras, яка не є самостійною системою, а являє собою надбудову над Theano, TensorFlow або CNTK, та надає зручний і простий інтерфейс для навчання глибоких нейронних мереж.

2. Ансаблі моделей у машинному навчанні

Більшість моделей машинного навчання є чутливими до вибору конкретних параметрів та гіперпараметрів моделі при навчанні, тобто дають різні по точності результати. Тому для підвищення точності отриманих результатів у теорії машинного навчання широко застосовують ансамблі моделей. Ансамбль являє собою сукупність декількох моделей, об'єднаних тим чи іншим способом, які використовуються для розв'язання однієї спільної задачі. Теоретично обґрунтовано, що використання ансамблю моделей є більш ефективним, у порівняні з результатами окремих самостійних моделей. При цьому рекомендується створювати ансамблі на базі простих моделей, що не корелюють між собою.

Існують різні підходи до об'єднання моделей у ансамблі. Якщо у ансамблі використовують однакові моделі, які навчаються паралельно на різних підмножинах тренувальних даних з подальшим усередненням результату, то такий підхід називають беггінгом. Іншим важливим класом ансамблевих моделей є бустінг. У методах бустінгу однакові моделі навчають послідовно таким чином, щоб навчання моделі на даному етапі залежало від навчання моделей на попередніх етапах, тим самим підсилюючи попередній результат. Ще один підхід до побудови ансамблю – це стекінг. В цьому випадку паралельно навчають різні типи моделей, результати яких або усереднюють, або використовують для навчання узагальнюючої моделі для отримання остаточного результату.

В цілому усереднення моделей є потужним і дієвим методом зменшення похибки. Досить часто у змаганнях з машинного навчання перемагають моделі, які здійснюють усереднення по десяткам моделей.

Останнім часом спостерігається тенденція до більшого практичного застосування ансамблів з нейронних мереж. Незважаючи на високу точність результатів, ансамблі

нейронних мереж застосовуються не так широко, як ансамблі з класичних моделей машинного навчання, оскільки вимагають значних обчислювальних ресурсів [7].

Ансамблі утворюють із слабких та простих моделей. Нейронні мережі можуть вважатись такими, якщо використовувати невелику кількість епох навчання. Окрім того методи навчання нейронних мереж передбачають випадкове початкове визначення значень вагових коефіцієнтів, тому моделі, що навчались на одних і тих самих даних будуть відрізнятися. Важливою складавою ансамблю є метод отримання остаточного спільного прогнозу. Досить часто у якості остаточного результату обирають усереднене значення прогнозів всіх нейронних мереж, що входять до ансамблю. Дану ідею можна удосконалити. Наприклад, у роботі [5] пропонується навчити, ще одну нейронну мережу, вхідними даними для якої будуть прогнози всіх нейронних мереж ансамблю, а виходом – результуючий прогноз ансамблю.

3. Побудова і дослідження ансамблю нейронних мереж

У даній роботі в якості базової моделі оберемо двошарову нейрону мережу прямого розповсюдження, що має один прихований шар. Для реалізації нейронної мережі використаємо бібліотеку Keras, а також TensorFlow у якості обчислювального бекенду. Основними параметрами такої нейронної мережі виступатимуть кількість нейроні у вхідному та прихованому шарах, кількість епох навчання, розмір міні-вибірок, метод оптимізації.

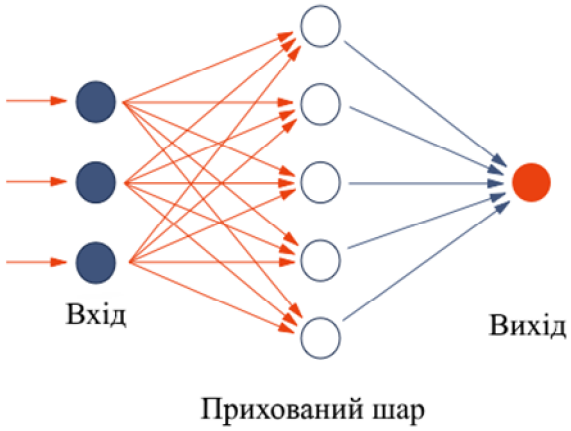


Рис. 1 Двошарова нейрона мережа прямого розповсюдження

Розглянемо два підходи до побудови ансамблю нейронних мереж для задачі класифікації, коли необхідно спрогнозувати ймовірність віднесення зображення до одного з класів:

1) Усереднюючий ансамбль. Окремо навчаємо декілька нейронних мереж, а у якості остаточного результату такої моделі обираємо середне арифметичне значення ймовірностей, отриманих всіма нейронними мережами.
2) Ансамбль з керівником. Спочатку окремо навчаємо декілька нейронних мереж. Потім навчаємо нейрону мережу («керівника ансамблю»), вхідними даної для якої будуть результати прогнозів ймовірностей залучених нейронних мереж. У якості остаточного результату обираємо прогнози ймовірностей «керівника ансамблю».

Спочатку потрібно обрати базові нейронні мережі для ансамблів. Для всіх мереж задамо кількість епох навчання – 25, розмір міні-вибірок – 100 і у якості оптимізатора – метод стохастичного градієнтного спуску. Перевіримо всі варіанти нейронних мереж,

коли вхідний та прихований шари містять 50, 100, 200, 300, 400, 500, 600, 700 та 800 шарів.

Для тестування розглянемо класичний набір даних: MNIST (класифікація рукописних цифр). Даний набір містить 60 000 зображень розміром 28×28 пікселів для навчання, і 10 000 зображень для тестування. Точність моделі визначали як долю вірно класифікованих об'єктів.

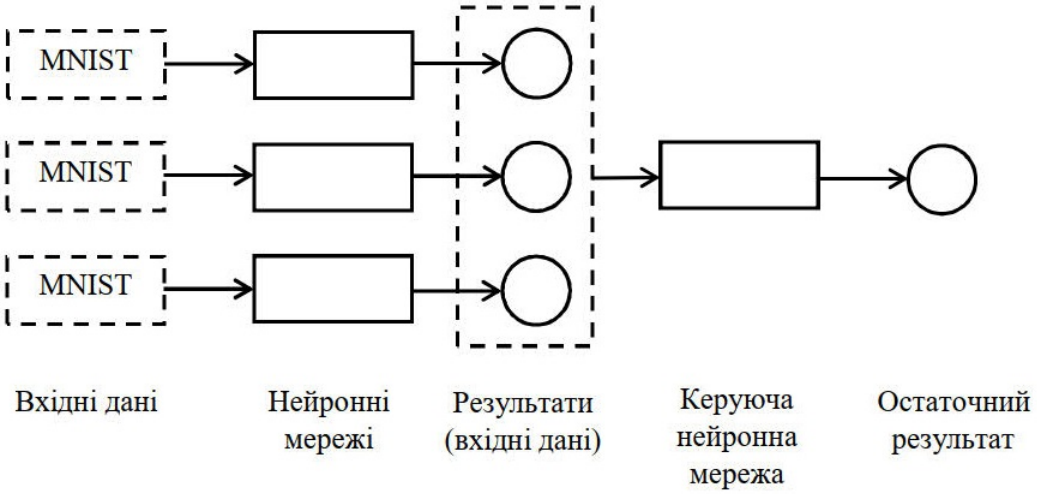


Рис. 2 Структура ансамблю з керівником

Було обрано наступні нейронні мережі, для яких зазначено кількість нейронів у вхідному, прихованому і вихідному шарі та отриману точність:

1) нейронна мережа №1 (NN1): 400×600×10, 97.3%;
2) нейронна мережа №2 (NN2): 400×800×10, 97.5%;
3) нейронна мережа №3 (NN3): 700×300×10, 97.4%;
4) нейронна мережа №4 (NN4): 300×600×10, 97.39%;
5) нейронна мережа №5 (NN5): 600×200×10, 97.52%.

Побудовано два усереднюючих ансамблі та два ансамблі з керівником, в які увійшли відповідно нейронні мережі №1-3 і №1-5. Архітектура керуючої нейронної мережі була наступною: 400×600×10.

Результати отриманих обчислювальних експериментів наведено у табл. 1.

Таблиця 1

Точність класифікації для ансамблів нейронних мереж

Ансамбль нейронних мереж	Отримана точність
Усереднюючий ансамбль з 3-ох нейронних мереж	97.90%
Усереднюючий ансамбль з 5-ти нейронних мереж	98.08%
Ансамбль з керівником з 3-ох нейронних мереж	97.42%
Ансамбль з керівником з 5-ти нейронних мереж	97.43%

Як видно з результатів проведених досліджень, усереднюючі ансамблі демонструють більшу точність, у порівнянні з ансамблями з керівником. Спробуємо

збільшити кількість нейронних мереж в усереднюючому ансамблі. Додамо наступні двошарові та тришарові моделі нейроні мережі та отриману для кожної з них точність:

6) нейронна мережа №6 (NN6): 300×200×10, 97.16%;
7) нейронна мережа №7 (NN7): 600×600×10, 97.43%;
8) нейронна мережа №8 (NN8): 200×50×40×10, 96.46%;
9) нейронна мережа №9 (NN9): 200×100×300×10, 96.69%.

Результати отриманих обчислювальних експериментів наведено у табл. 2.

Таблиця 2

Точність класифікації для усереднюючих ансамблів нейронних мереж

Усереднюючий ансамбль нейронних мереж	Отримана точність
Ансамбль з 7-ми нейронних мереж (NN1-NN7)	98.11%
Ансамбль з 7-ми нейронних мереж (NN1-NN5, NN8, NN9)	98.10%
Ансамбль з 7-ми нейронних мереж (NN3-NN9)	97.98%
Ансамбль з 9-ти нейронних мереж (NN1-NN9)	98.15%

Збільшення кількості нейроних мереж приводить до покращення результатів класифікації. Також варто зазначити, що, оскільки ансамблі складаються з простих моделей, то загальний час їх навчання менший, ніж час навчання згорткових нейронних мереж, які часто використовуються для класифікації зображень. Загальний час навчання ансамблю з 7-ми нейронних мереж (NN1-NN7) складає 14 хв. 24 сек., а ансамблю з 9-ти нейронних мереж (NN1-NN9) – 16 хв. 32 сек.

Розглянемо для прикладу згорткову нейронну мережу (CNN1) з двома шарами згортки і підвибірки та одним повнозв'язним шаром, детальну архітектуру якої представлено на рис. 3.

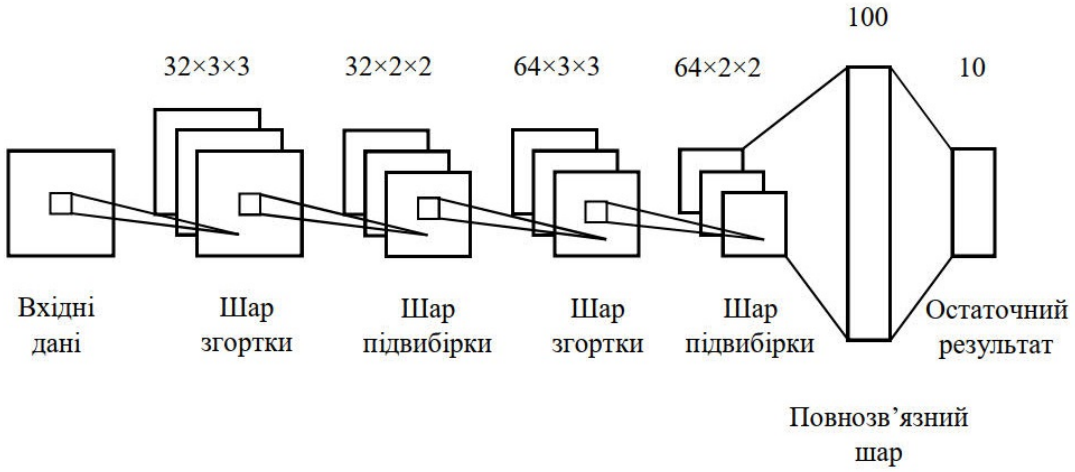


Рис. 3 Згорткова нейронна мережа

Точність класифікації за допомогою згорткової нейронної мережі CNN1 становить 99.02%, при цьому час навчання мережі з 10 епохами склав 2 год. 8 хв. 7 сек. Отримані результати продемонструвати, що невеликі ансамблі з двошарових нейроних мереж навчаються значно швидше, ніж згорткові нейронні мережі. Крім того, з

практичної точки зору може виявитися простіше реалізувати ансамбль із простих моделей, ніж розробити архітектуру згорткової нейронної мережі.

Висновки

У статті описано розробку ансамблів з двошарових нейронних мереж прямого розповсюдження для розв'язання задачі класифікації зображень з набору даних MNIST. Було розглянуто два способи побудови ансамблю: усереднюючий ансамбль і ансамбль з керівником. Для реалізації описаних моделей використано бібліотеки Keras і TensorFlow.

Кращі результати продемонстрували усереднюючі ансамблі. Точність класифікації такого ансамблю з 5-9 нейронних мереж у порівнянні з результати окремої нейронної мережі збільшується в межах 1%. Таким чином, усереднення результатів виявляється вигідним, оскільки члени ансамблю допускають різні помилки при класифікації, хоч і навчались на одним даних. Це обумовлено і випадковою ініціалізацією, і випадковим вибором міні-пакетів, та загалом результатами недермінованої реалізації нейронной мережі.

Було встановлено, що загальний час навчання ансамблів з двошарових нейронних мереж значно менший, ніж час навчання згорткових нейронних мереж, які часто використовуються для класифікації зображень.

Таким чином можемо зробити висновок, що використання усереднюючих ансамблів з простих нейронних мереж для класифікації зображень є досить ефективним, оскільки вони мають більшу прогнозуючу здатність та порівняно невеликий час навчання.

Список використаної літератури:

1. Рашка С. Python и машинное обучение / Рашка Себастьян. – М.: ДМК Пресс, 2017. – 614 с.
2. Шолле Ф. Глубокое обучение на Python / Франсуа Шолле. – СПб.: Питер, 2018. – 400 с.
3. Бодянский Е.В. Искусственные нейронные сети: архитектуры, обучение, применения / Е.В. Бодянский, О.Г. Руденко. – Харьков: ТЕЛЕТЕХ, 2004. – 369 с.
4. Гольцев А.Д. Нейронные сети с ансамблевой организацией / А.Д. Гольцев. – К.: Наукова думка, 2005. – 200 с.
5. Каширина И.Л. О методах формирования нейросетевых ансамблей в задачах прогнозирования финансовых временных рядов / И.Л. Каширина // Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии, 2009. – № 2. – С. 116-119.
6. Брюхнова В.О. Ансамбли нейронных сетей при прогнозировании объемов продаж в торговой сети / В. О. Брюхнова, Н. И. Цуканова // Вестник Рязанского государственного радиотехнического университета, 2018. – №66, Часть 1. – С. 90-98.
7. Кривохата А. Г. Застосування ансамблевого навчання в задачах класифікації акустичних даних / Кривохата А. Г., Кудін О. В., Давидовський М. В., Лісняк А. О. // Вісник Запорізького національного університету. Фізико-математичні науки, 2018. – №1. – С. 48-60.
8. Созыкин А.В. Обзор методов обучения глубоких нейронных сетей / А.В. Созыкин // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика, 2017. – Т. 6. – №3. С. 28-59.

Bibliography:

1. Raschka S. (2017) Python and machine learning. M .: DMK Press. 614. (in Rus.).
2. Chollet F. (2018) Deep Learning in Python. St. Petersburg: Peter. 400. (in Rus.).
3. Bodyansky E.V., Rudenko O.G. (2004) Artificial neural networks: architectures, training, applications. Kharkov: TELETEKH. 369. (in Rus.).
4. Goltsev A.D. (2005) Neural networks with ensemble organization. Naukova Dumka. 200. (in Rus.).
5. Kashirina I.L. (2009) About the methods of forming neural network ensembles in the tasks of forecasting financial time series. Vestnik Voronezhskogo gosudarstvennogo universiteta. Seriya: Sistemnyj analiz i informacionnye tekhnologii [Bulletin of Voronezh State University. Series: System Analysis and Information Technologies], 2, 116-119. (in Rus.).
6. Bryukhnova V.O., Tsukanova N.I. (2018) Ensembles of neural networks in predicting sales volumes in a distribution network. Vestnik Ryazanskogo gosudarstvennogo radiotekhnicheskogo universiteta [Bulletin of the Ryazan State Radio Engineering University], 66(1), 90-98. (in Rus.).

7. Krivokhata A. G., Kudin O. V., Davidovskiy M. V., Lisnyak A. O. (2018) Application of ensemble training in problems of classification of acoustic data. Visnik Zaporiz'kogo nacional'nogo universitetu. Fiziko-matematichni nauki [Bulletin of the Zaporizhzhya National University. Physical and Mathematical Sciences], 1, 48-60. (in Ukr.).
8. Sozykin A.V. (2017) A review of teaching methods of deep neural networks. Vestnik YuUrGU. Seriya: Vychislitel'naya matematika i informatika [Bulletin of SUSU. Series: Computational Mathematics and Computer Science], 6(3), 28-59. (in Rus.).

KRASNOSHLYK Nataliya,
PhD, Senior Lecturer, The Bohdan Khmelnytsky National University of Cherkasy
SERDIUK Maryna,
student of applied mathematics speciality, Lviv Polytechnic National University

APPLICATION OF NEURAL NETWORK ENSEMBLES TO SOLVE THE PROBLEM OF CLASSIFICATION OF IMAGES

Summary. Introduction. *Recently, machine learning is one of the leading technologies of today. Machine learning algorithms have made significant progress in language and handwriting recognition, image classification, robust email spam filters, high-quality search engines, credit card fraud detection programs, and more.*

A special section of machine learning is deep learning. A typical example of a deep learning model is neural networks. Currently, there are many types of neural networks that differ in architecture, type of neurons, communication structure, learning algorithms. Often, the same task can be solved using different neural networks, and choosing the best one is not determined by strict rules. The quality of solving a particular task can be significantly enhanced by neural network ensembles. The ensembles share the same data in parallel with several neural networks, whose output signals are somehow combined to summarize the result.

Thus, combining a set of neural networks into a single model is a promising area of research in deep learning. It is anticipated that neural network ensembles will have better generalizability and enable more complex tasks.

Purpose. *The purpose of this paper is to investigate the effectiveness of using neural network ensembles to solve image classification problems.*

Results*. The paper describes the development of ensembles of two-layer direct-propagation neural networks to solve the problem of image classification from the MNIST dataset. Two ways of constructing an ensemble were considered: averaging ensemble and an ensemble with a leader. The described models were implemented using the Keras and TensorFlow libraries.*

The best results were demonstrated by the averaging ensembles. The classification accuracy of such an ensemble from 5-9 neural networks in comparison with the results of a single neural network increases within 1%. Thus, the averaging of the results is advantageous, since the members of the ensemble make different classification errors, although they have been trained on the same data. This is due to both the random initialization, and the random selection of mini-packets, and generally the results of the non-term implementation of the neural network.

It has been found that the total learning time of ensembles from two-layer neural networks is significantly less than the learning time of convolutional neural networks, which are often used to classify images.

Conclusion. *This paper describes the existing approaches to building model ensembles in machine learning. The most popular systems for learning neural networks are given. The base model is a two-layer direct propagation neural network with one hidden layer. Two approaches to building a neural network ensemble are considered and implemented. The effectiveness of ensembles in solving image classification problems has been investigated.*

We can conclude that the use of averaging ensembles of simple neural networks for image classification is quite effective, since they have greater predictive power and relatively little training time.

Keywords: *machine learning, classification problem, neural network, ensemble of models.*

Одержано редакцією 05.06.2018 р.
Прийнято до публікації 17.10.2018 р.

УДК 519.688

DOI 10.31651/2076-5886-2019-1-61-75

PACS 02.60.-x, 02.60.Pn

КУЧЕР Олександр Іванович,
аспірант кафедри прикладної математики
та інформатики Черкаського національного
університету імені Богдана Хмельницького
e-mail: olexandr.kucher@gmail.com
ORCID 0000-0002-6902-5068

ОСОБЛИВОСТІ ЗАСТОСУВАННЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ЗАДАЧ ОБРОБКИ ЗОБРАЖЕНЬ

У статті дано базове поняття та характеристики згорткових нейронних мереж, описана структура та математичні підходи до реалізації нейронної мережі цього типу. Наданий опис основних шарів згорткових нейронних мереж, наведені параметри, завдяки яким можна змінювати процес навчання мережі. Детально розглянутий процес навчання кожного з шарів нейронної мережі. Додатково описані техніки, що дають можливість збільшити набір даних та покращити процес навчання мережі за допомого кількох перетворень

Ключові слова: *глибоке навчання, машинне бачення, нейронні мережі, розпізнавання облич, згорткові нейронні мережі, рецептивні поля, глибоке навчання, нейронні мережі, згорткові нейронні мережі, класифікація, локалізація, виявлення, сегментація, випрямлені лінійні одиниці.*

Постановка проблеми

Згорткові нейронні мережі (ЗНМ, Convolutional Neural Networks, CNN) – одні з найвпливовіших інновацій в області комп'ютерного зору. Вперше нейронні мережі привернули до себе увагу в 2012 році на конкурсі ImageNet. За допомого ЗНМ було встановлено новий рекорд помилок при класифікації – 15% (попереднє значення було 26%) [1]. На сьогоднішній день глибоке навчання (deep learning) використовують безліч відомих компаній:

- Facebook – використовує для алгоритмів автоматичного виставлення тегів
- Google – для пошуків по фотографіях
- Amazon – для генерації рекомендації товарів

Але класичний і найбільш популярний спосіб застосування нейронних мереж це обробка зображень. Розглянемо як ЗНМ використовуються для класифікації зображень.

Виклад основного матеріалу

Задача класифікації зображень – це обробка зображення і визначення класу до якого воно належить (автомобіль, тварина і т. д.) або групи класів, які найкраще його характеризують.

Вхідні і вихідні дані

Коли комп'ютер отримує на вхід оцифроване зображення, він працює з масивом пікселів. Залежно від розширення зображення розміри масиву можуть бути, наприклад, 64x64x3 (де 3 – це значення кожного з каналів RGB). Для кращого розуміння уявимо, що в нас є кольорове зображення в форматі JPG розміром 360x360 пікселів. Воно буде представлено масивом з розмірами 360x360x3. Кожне із значень масиву набуває значення від 0 до 255, що описує інтенсивність пікселя. Для людини ці значення не мають сенсу, але це єдиний спосіб представити зображення в комп'ютері. Ідея полягає в тому, щоб надати програмі (алгоритму) цю матрицю і отримати в результаті значення

що описують ймовірність відношення цього зображення до певного класу (0.8 – автомобіль, 0.15 – тварина, 0.05 - людина).

Рис. 1. Що бачить людина та що «бачить» комп'ютер

Зв'язок з біологією

ЗНМ – це прототип зорової кори головного мозку. Зорова кора має невеликі ділянки клітин, що є чутливими до певних областей поля зору. Цю ідею детально розглянули за допомогою експерименту Г'юбел і Візел в 1962 році в якому показали, що окремі нервові клітини реагували лише при візуальному сприйнятті меж певної орієнтації. Наприклад, деякі нейрони активувалися, коли сприймали вертикальні границі, а деякі – горизонтальні або діагональні. Г'юбел і Візел з'ясували, що всі нейрони розташовані в виді стержневої архітектури і разом формують візуальне сприйняття. Цю ідею спеціалізованих компонентів, що розв'язують якусь задачу і використовують машини, і ця ідея – основа ЗНМ. [7]

Структура

Зображення пропускається через серію згорткових, нелінійних шарів, шарів об'єднання і повнозв'язних шарів в результаті чого генеруються вихідні дані. Вихідними даними може бути клас або ймовірність класів, які найкраще описують зображення.

Перший шар – математична частина

Перший шар в ЗНМ завжди згортковий. Найлегше зрозуміти, що таке згортковий шар, допоможе представлення його у вигляді ліхтарика, який світить на верхню ліву частину зображення. Припустимо, що світло з ліхтарика покриває площу 5x5 пікселів. В термінах комп'ютерного навчання цей ліхтарик – це фільтр, а область на яку він світить називається рецептивним полем. Глибина фільтра повинна бути такою ж як глибина вхідного зображення (5x5x3). Оскільки фільтр здійснює згортку, тобто рухається по зображенню, він перемножує значення фільтра на вихідні значення пікселів зображення (поелементно). В результаті отримуємо одне число. Тепер необхідно повторити цей процес в кожній позиції (Наступний крок – це переміщення фільтра вправо на 1 піксель). Кожна унікальна позиція зображення генерує одне число. Після проходження фільтра по всіх позиціях (для зображення 32x32x3) отримуємо матрицю 28x28x1, яку називають функцією активації або картою ознак. Матриця 28x28 отримується тому, що є 784 різні позиції, які можуть пройти через фільтр 5x5. Ці 784 числа перетворюються в матрицю 28x28. [6]

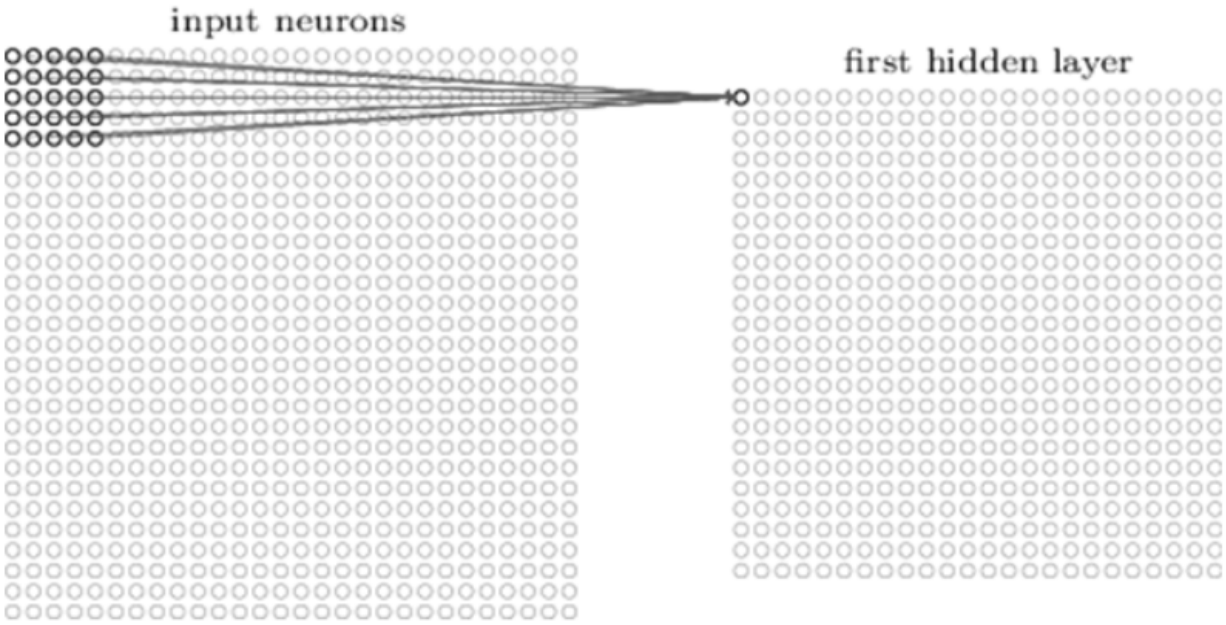


Рис. 2. Візуалізація фільтра 5x5, що створює карту активації з вхідних даних

Якщо ми використовуватимемо два 5x5x3 фільтри замість одного, то отримаємо матрицю 28x28x2.

Перший шар

Кожен фільтр можна розглядати як ідентифікатор властивості. Під властивостями маються на увазі прямі межі, прості кольори і криві. Припустимо, що наш перший фільтр 7x7x3 і він буде детектором кривих. (Для простоти розглянемо лише верхні шари фільтра і зображення). Фільтр має піксельну структуру, у якої числові значення більші вздовж області, що визначає форму кривої.

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

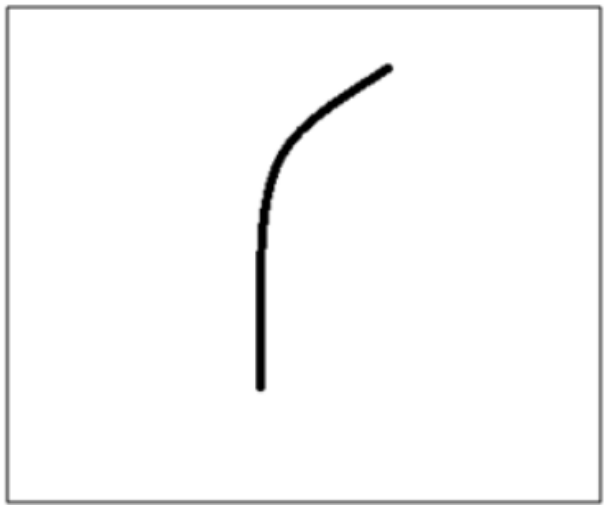


Рис. 3. Цифрове та візуальне представлення фільтра що знаходить криві

Повернемося до математичної візуалізації. Коли в верхньому лівому куті вхідного зображення є фільтр, він здійснює множення значень фільтра на значення пікселів області. Розглянемо приклад зображення, яке потрібно класифікувати, і встановимо фільтр в верхньому лівому куті.

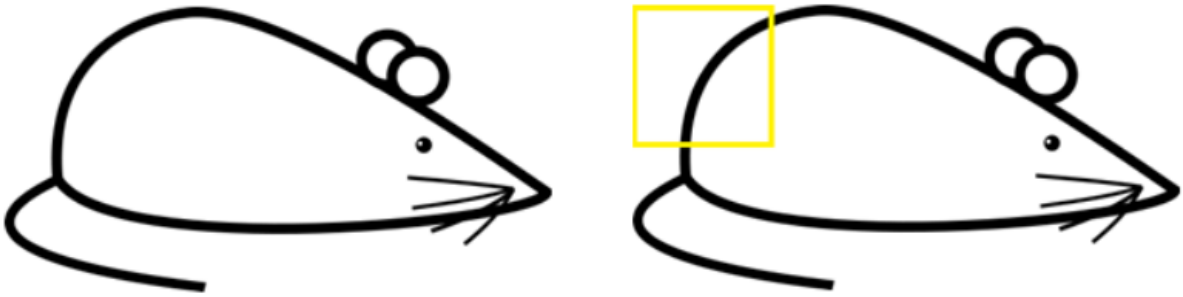


Рис. 4. Візуалізація фільтра на зображенні

Все що необхідно зробити – це перемножити значення фільтра на значення пікселів зображення.

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Рис. 5. Застосування фільтра в верхньому лівому куті зображення

Multiplication and Summation =

$$(50\times 30)+(50\times 30)+(50\times 30)+(20\times 30)+(50\times 30)=6600.$$

Якщо на зображенні є форма, що схожа на криву, яку представляє фільтр, то всі перемножені і додані значення утворюють значення на кілька порядків більше за 0 (порогове значення встановляється програмістом).

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Рис. 6. Застосування фільтра в верхньому правому куті зображення

***Multiplication and Summation* = 0**

Коли ми перемістимо фільтр, то отримаємо значення близьке до нуля. Причиною є те, що в цій області зображення немає нічого, що фільтр кривої міг виявити. Пам'ятайте – результат цього згорткового шару – це карта властивостей. В найпростішому випадку, при наявності одного фільтра згортки, карта властивостей виявить області, в яких з більшою ймовірністю є криві (якщо фільтр – детектор кривих). В цьому прикладі значення карти властивостей у верхньому лівому куті буде 6600. Це значення показує, що, можливо, щось схоже на криву є на зображенні, і така ймовірність активувала фільтр. В правому верхньому куті значення карти властивостей

буде 0, тому що на зображенні не було нічого, що могло б активувати фільтр. Ці значення є характеристиками лише одного фільтра, фільтра що знаходить криві зі згином назовні. Можуть бути й інші фільтри. Чим більше фільтрів, тим більша глибина карти властивостей і більше інформації про зображення можна отримати. [8]

Глибше через мережу

Зараз в традиційній згортковій мережі існують й інші шари, що перемножуються з згортковими. Класична архітектура ЗНМ матиме наступний вигляд:

Input -> Conv -> ReLU -> Conv -> ReLU -> Pool -> ReLU -> Conv -> ReLU -> Pool -> Fully Connected

де ***Conv*** – згортковий шар, ***ReLU*** – функція активації, ***Pool*** – об'єднуючий шар, ***Fully Connected*** – повнозв'язний шар.

Ми розглянули фільтри першого шару. Вони виявляють властивості базового рівня, такі як межі та криві. Для того щоб розпізнавати типи об'єктів на зображеннях, нам потрібна мережа, що здатна розпізнавати об'єкти більш високого рівня, наприклад руки, лапи або вуха. Результат обробки зображення першим фільтром має розміри 28x28x3 (за умови що використовувався фільтр 5x5x3). Коли зображення проходить через один згортковий шар, результат першого шару стає вхідними даними другого. Коли ми описували перший шар, вхідними даними було лише початкове зображення. Але коли ми перейшли до другого шару, вхідними значеннями для нього стала одна або декілька карт властивостей – результат обробки попереднього шару. Кожен набір даних описує позиції, де на початковому зображенні знаходяться певні базові ознаки. [1]

Тепер, коли застосовується набір фільтрів поверх цього (зображення пропускається через другий згортковий шар), в результаті будуть активовані фільтри, які представляють властивості більш високого рівня. Типами цих властивостей можуть бути півкільця або квадрати. Чим більше згорткових шарів застосовується до зображення, тим більш складні характеристики проявляються в картах активації. В кінці мережі можуть бути фільтри, що активуються при наявності рукописного тексту на зображенні, при наявності зелених об'єктів і т. д.

При русі в глибину мережі, фільтри працюють з усе більшим полем сприйняття, а значить, вони в змозі опрацьовувати інформацію з більшої площі початкового зображення (вони краще адаптуються до обробки більшої області піксельного простору).

Повнозв'язні шари

Тепер коли можливо виявити високорівневі властивості, можна прикріпити повнозв'язний шар вкінці мережі. Цей шар бере вхідні дані і виводить N-мірний вектор, де N – число класів, серед яких програма обирає потрібний. Наприклад, якщо ви працюєте над програмою розпізнавання цифр, N буде рівне 10. Кожне значення в цьому векторі представлятиме собою ймовірність конкретного класу. Наприклад, якщо результуючий вектор для програми розпізнавання цифр це: ***[0 0.1 0.1 0.75 0 0 0 0 0 0.05]*****,** значить існує ймовірність 10% що на зображенні «1», ймовірність 10%, що на зображенні «2», ймовірність 75%, що на зображенні «3», і ймовірність 5%, що на зображенні «9».

Спосіб, за допомогою якого працює повнозв'язний шар – це взаємодія з вихідними даними попереднього шару (який повинен виводити високорівневі карти властивостей) і визначення властивостей, які більш зв'язані з певним класом. Повнозв'язний шар виявляє, що функції високого рівня сильно зв'язані з певним класом і мають певні коефіцієнти ваги, тобто, коли відбувається обрахунок ваги з попереднім шаром, отримуються правильні ймовірності для різних класів.

Навчання (або що змушує це працювати)

Спосіб за допомогою якого комп'ютер здатний корегувати значення фільтра (або коефіцієнтів ваги) – це процес навчання, який називається методом зворотного розповсюдження помилки.

Що ж нейронній мережі необхідно для роботи. До моменту побудови мережі, ваги або значення фільтра випадкові. Фільтри не вміють шукати межі і криві. Фільтри верхніх шарів не вміють шукати руки, лапи і вуха. Для навчання мережі застосовується механізм надання їй зображення і ярлика чи класу, що йому відповідає.

Метод зворотного розповсюдження помилки можна розділити на 4 окремих блоки: пряме розповсюдження, функція втрати, зворотне розповсюдження і оновлення ваги. Під час прямого розповсюдження, береться тренувальне зображення (це матриця 32x32x3) і пропускається через всю мережу. В першому прикладі, оскільки всі ваги або значення фільтрів були ініційовані випадковим чином, вихідним значенням буде щось на зразок ***[0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1]***, тобто таке значення, яке не надає перевагу жодному з наявних класів. Мережа з такими властивостями не здатна знайти властивості базового рівня і не здатна аргументовано визначити клас зображення. Це веде до функції втрати. На цьому етапі використовуються дані для навчання. У таких даних є зображення і ярлик. Якщо перше зображення для навчання це цифра 3, то ярликом буде ***[0 0 0 1 0 0 0 0 0 0]***. Функція втрати може бути виражена по різному, але часто використовується середньоквадратична помилка.

$$E_{total} = \sum \frac{1}{2}(target - output)^2 . \quad (1)$$

Нехай це значення буде **L.** Втрата буде дуже великою для перших двох зображень. Для того щоб прогнозований ярлик був таким же, як ярлик зображення для навчання, потрібно звести до мінімуму кількість втрат, які в нас є. Розглядаючи цю задачу як задачу оптимізації з математичного аналізу, необхідно з'ясувати, які вихідні дані сприяли втратам (помилкам) мережі.

Один зі способів візуалізувати ідею мінімізації втрат – це 3-D графік, де ваги нейронної мережі (їх більше 2-х, але тут приклад спрощений) це незалежні змінні, а залежна змінна – це втрата. Задача мінімізації втрат – відрегулювати ваги таким чином, щоб знизити втрату. Візуально необхідно наблизитися до найнижчої точки чашоподібного об'єкту. Щоб зробити це, необхідно знайти похідну втрати (в рамках зображеного графіка – розрахувати кутовий коефіцієнт в кожному з напрямків) з врахуванням ваг.

Це математичний еквівалент ***dL/dW***, де W – ваги одного шару. Далі необхідно виконати **зворотне розповсюдження** через мережу, яке визначає, які ваги здійснили більший вплив на втрати, і знайти способи, як їх налаштувати, щоб зменшити втрати. Після знаходження похідної, переходимо до останнього етапу – оновлення ваг. Необхідно взяти всі ваги фільтрів і оновити їх так, щоб вони змінювалися в напрямку градієнту.

$$w = w_i - n\frac{dL}{dW}, \quad (2)$$

де w - вага, w_i - початкова вага, n - швидкість навчання.

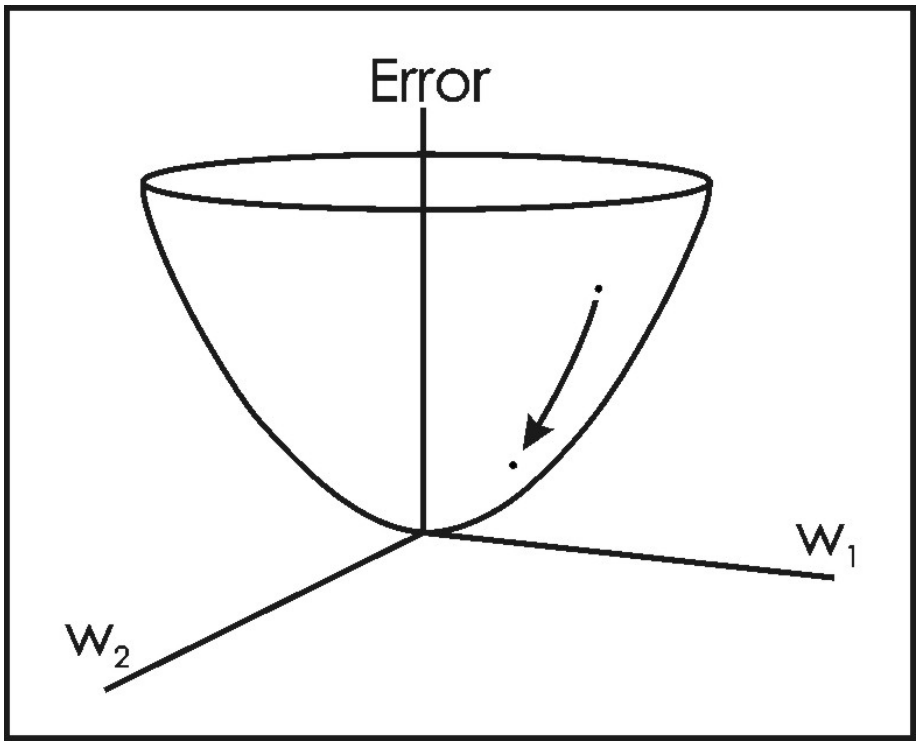


Рис. 7. Візуалізація задачі мінімізації

Швидкість навчання – це параметр, який обирається програмістом. Висока швидкість навчання значить, що в нових коефіцієнтах ваги були зроблені більші кроки, завдяки чому може знадобитися менше часу, щоб отримати оптимальний набір ваг. Але занадто велика швидкість навчання може привести до дуже великих і не достатньо точних стрибків, які завадять досягненню оптимальних показників.

Процес прямого розповсюдження, функцію втрати, зворотне розповсюдження і оновлення ваг, зазвичай називають одним періодом дискретизації (або epoch - епохою). Програма буде повторювати цей процес фіксовану кількість періодів для кожного тренувального зображення. Після того, як оновлення параметрів завершиться на останньому тренувальному зразку, мережа повинна бути достатньо добре навчена і ваги шарів повинні бути налаштовані правильно.

Тестування

Нарешті, щоб побачити, чи працює ЗНМ, ми беремо інший набір зображень і ярликів і пропускаємо зображення через мережу. Порівнюємо результати з реальними даними і перевіряємо чи працює наша мережа.

Крок і відступ

При роботі з згортковими нейронними мережами (ЗНМ) ми можемо змінювати 2 параметри, для того, щоб змінити поведінку кожного з шарів. Після обранні розміру фільтра, ми можемо також обрати **крок** і **відступ**.

Крок – це параметр, що контролює процес згортки вхідного зображення за допомогою фільтра. На Рис. 8 зображено приклад згортки вхідних даних зі зміщенням на одну комірку за раз. Кількість комірок, на які зміщується фільтр на кожному кроці, називається кроком. На Рис. 8 розмір кроку – 1. За крок завжди беруть ціле число.

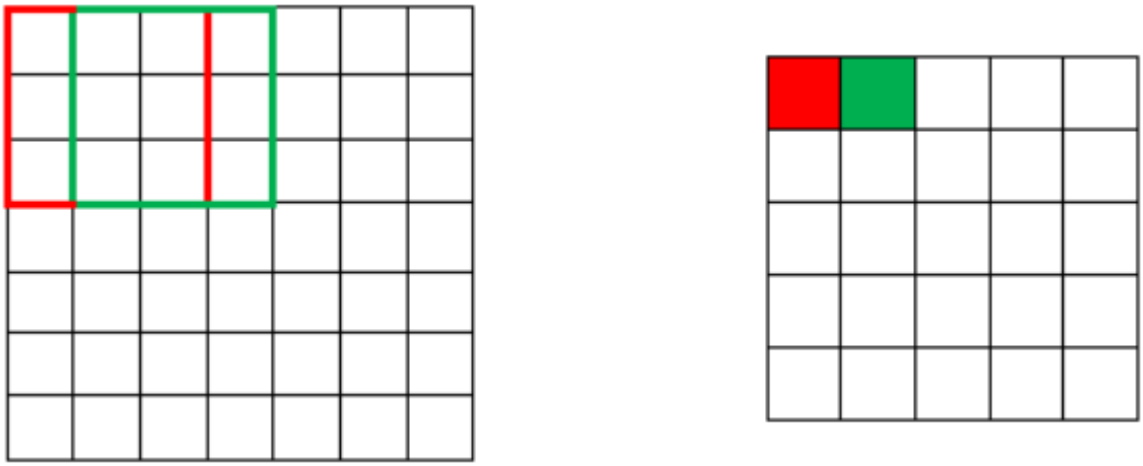


Рис. 8. Вхідні і вихідні дані фільтра з кроком 1

На Рис. 9 зображено фільтр з кроком 2, використання якого призводить до зменшення даних на виході. Як правило, крок фільтру збільшують для зменшення перекриття рецептивних полів та зменшення кількості даних на виході фільтра.

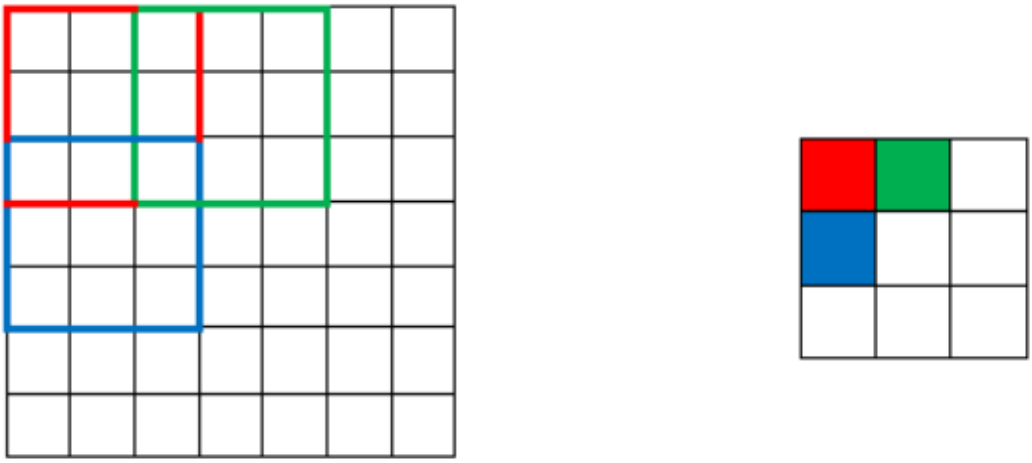


Рис. 9. Вхідні і вихідні дані фільтра з кроком 2

Тепер давайте поглянемо на відступ. Що відбудеться, якщо застосувати три фільтри 5x5x3 до вхідних даних розміром 32x32x3? Розмір вихідних буде 28x28x3. Оскільки ЗНМ побудована на застосуванні ряду згорткових шарів, то розмір вихідних даних після застосування кожного з них зменшуватиметься швидше, ніж хотілося б. У ранніх шарах мережі ми хочемо зберегти максимальну кількість інформації про вхідні дані. Скажімо, ми хочемо застосувати один і той же згортковий шар, але хочемо, щоб обсяг вихідних даних залишався 32x32x3. Для цього ми можемо застосувати нульовий відступ розміру 2 до цього шару. Нульовий відступ збільшує розмір матриці вхідних даних за допомогою додавання нулів «навколо».

На Рис. 10 зображено вхідні дані, представлені матрицею з розміром 32x32x3. Якщо додати нулі «навколо», ми отримаємо матрицю з розміром 36x36x3. Після застосування згорткового шара з 3-ма фільтрами 5x5x3 і кроком 1, отримаємо вихідну матрицю все того ж розміру – 32x32x3. [3]

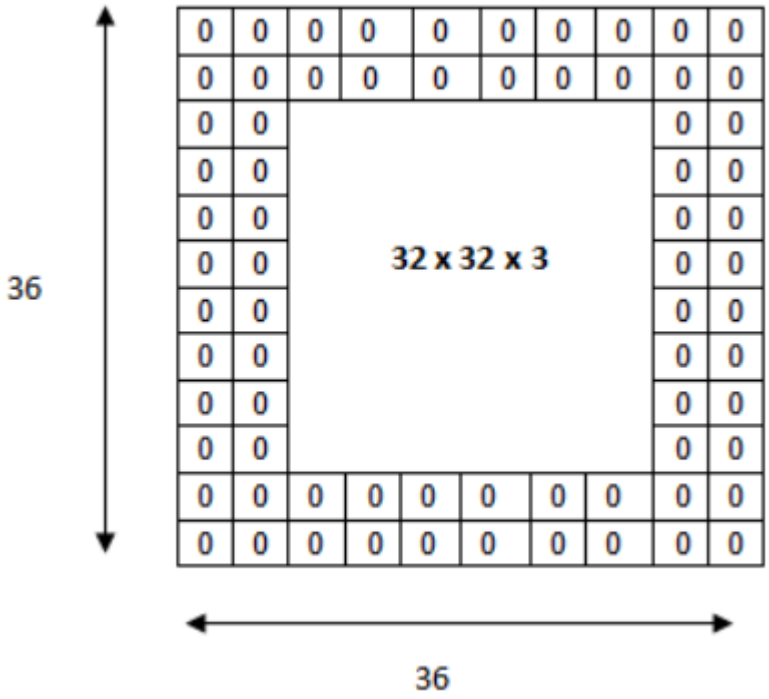


Рис. 10. Вхідні дані розміром 32x32x3 з відступом – 2

Якщо розмір кроку – 1 і розмір відступу задати за формулою:

$$P = \frac{K-1}{2}, \tag{3}$$

де K - розмір фільтру. Розмір матриці вхідних даних завжди дорівнювати розміру матриці вихідних даних.

Формула для обрахунку розмірів вихідних даних для будь-якого згорткового шару має наступний вигляд:

$$O = \frac{W - K + 2 \cdot P}{S} + 1, \tag{4}$$

де O – вихідна висота або ширина, W – вхідна висота або ширина, K – розмір фільтра, P – відступ, S – крок.

Вибір параметрів навчання

Як визначити, скільки шарів слід використовувати, скільки згорткових шарів, які розміри фільтрів та значення для кроку і відступу? Це не тривіальні питання і немає встановленого стандарту, який використовують усі дослідники. Це пояснюється тим, що мережа значною мірою залежить від типу даних, з якими вона працює. Дані можуть відрізнятися за розміром, складністю зображення, типом завдання обробки зображень тощо. Один із способів обрати параметри – це знайти правильну комбінацію, яка створює абстракції зображення у належному масштабі.

ReLU (Rectified Linear Units) шари

Після кожного згорткового шару прийнято застосовувати нелінійний шар (або шар активації). Метою цього шару є введення нелінійності в систему, яка в основному просто обраховує лінійні операції під час застосування згорткових шарів (множення і додавання елементів). В минулому застосовувалися нелінійні функції, такі як tanh і sigmoid, але дослідники з'ясували що ReLU шари працюють значно краще, то му що мережа має змогу навчатися значно швидше (через обчислювальну ефективність) без істотної різниці в точності. Це також допомагає полегшити проблему зникаючого градієнта, що є проблемою, коли нижні шари мережі навчаються дуже повільно, через градієнт що експоненційно зменшується. Шар ReLU застосовує функцію $f(x) = \max(0, x)$ до всіх вхідних значень. Коротко кажучи, ця функція просто замінює всі негативні значення на 0. ReLU шар збільшує нелінійні властивості моделі та мережі в загальному, не впливаючи на рецептивні поля згорткового шару. [5]

Шари об'єднання (Pooling layers)

Після деяких шарів ReLU розробники можуть застосувати шар об'єднання. Його також називають шаром зменшення. В цій категорії існує також кілька варіантів, найпопулярнішим є макспулінг (maxpooling, рис. 11). Такий підхід застосовує фільтр (зазвичай розміром 2x2) з кроком такого ж розміру. Фільтр застосовується до вхідних даних і залишає лише максимальне значення в кожній з областей згорнутих фільтром.

Інші підходи до об'єднання – це середнє значення та ***L2-norm***, яка визначається за формулою:

$$|x| = \sqrt{\sum_{k=1}^{n} |x_k|^2} \,. \qquad (5)$$

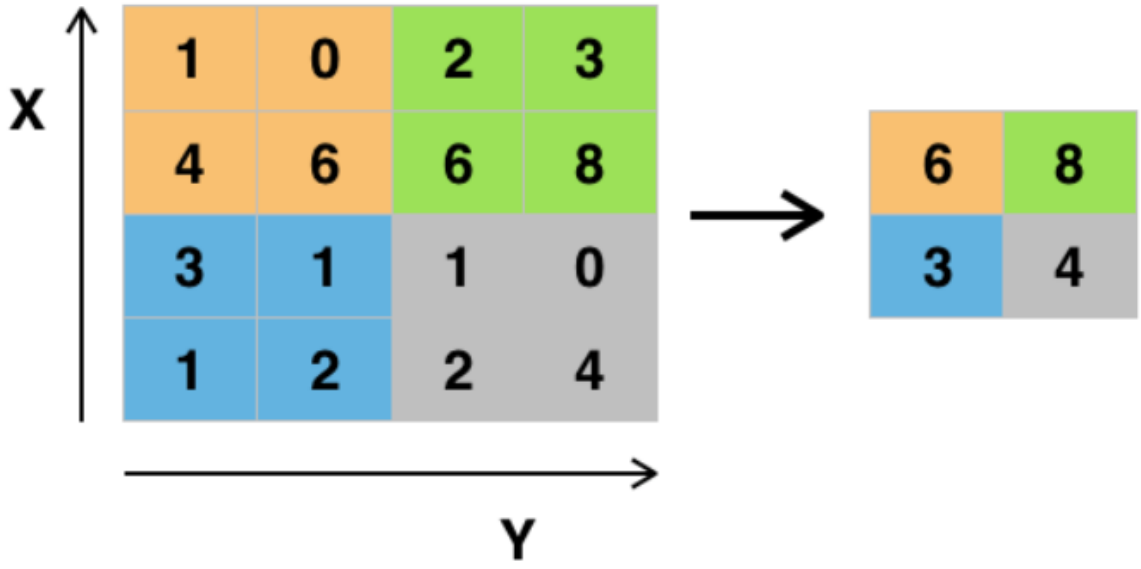


Рис. 11. Макспулінг з фільтром розміром 2x2 і кроком 2

Коли ми дізнаємося, що специфічна характеристика є у вхідних даних (в цій області буде високе значення активації), її точне розташування не так важливе як відносне розташування відносно інших характеристик. ReLU шар дуже швидко зменшує розміри (довжину і ширину, але не глибину) вхідних даних. Такі

трансформації мають 2 основні цілі. Перша – зменшення кількості параметрів або ваг на 75%, що зменшує складність обчислень. Друга – контроль ***перенавчання (overfitting).*** Це поняття описує процес при якому модель настільки налаштована на навчальні приклади, що не в змозі узагальнити дані для перевірки на тестових даних. Симптомом перенавчання є наявність моделі, яка видає 100% або 99% результат на навчальних даних, але лише 50% результат на тестових даних.

Шари відсівання (Dropout Layers)

Шари відсівання мають дуже специфічну функцію в нейронних мережах. Цей шар «відсіває» випадкові набори активацій, встановлюючи їх значення рівним нулю. Які ж переваги такого простого і, здавалося б, непотрібного і неінтуїтивного процесу? Такий підхід змушує мережу забезпечувати правильну класифікацію результату для конкретного прикладу, навіть якщо деякі з активацій відкинуті. Це гарантує, що мережа не стане занадто «пристосованою» до навчальних даних, а отже, знижує ймовірність виникнення проблеми перенавчання (перенасичення). Варто зазначити, цей шар використовується лише під час навчання, але не під час тестування. [11]

Шар мережі в мережі (Network in Network Layers)

Шар мережі в мережі посилається на згортковий шар з фільтром розміру 1x1. Виникає питання, чому цей шар є корисним, оскільки рецептивні поля зазвичай більші за простір, на який вони відображаються. Однак, згортка розміром 1x1 охоплює певну глибину, тому таку згортку необхідно розглядати як згортку розміром 1x1xN, де N – кількість фільтрів застосованих в цьому шарі. Фактично цей шар виконує N-D елементарне множення, де N – глибина вхідних даних шару. [10]

Класифікація, локалізація, виявлення, сегментація

Класифікація зображень - це процес отримання вхідного зображення і виведення номеру або назви класу з набору категорій. Але для задачі **локалізації** об’єктів потрібно не лише визначити клас, а і створити обмежувальну рамку, яка описує, де об’єкт знаходиться на зображенні.

Також є завдання **виявлення об’єктів**, де локалізація повинна здійснюватися на всіх об’єктах зображення (рис. 12, 13). Таким чином з’являється кілька обмежувальних полів і декілька класів.

Рис. 12. Класифікація – визначення що зображення це собака

Рис. 13. Локалізація – визначення класу та знаходження об’єкта на зображенні

Трансферне навчання (TransferLearning)

Поширена помилка у спільноті машинного навчання полягає в тому, що без величезних об’ємів даних (таких як в Google), ми не маємо можливості створити ефективні моделі глибокого навчання. Хоча дані є важливою частиною створення мережі, ідея трансферного навчання допомогла зменшити вимоги до даних. **Трансферне навчання** – це процес взяття попередньо навченої моделі (ваги та параметри мережі, яка була навчена на великій кількості даних кимось іншим) і «тонкого налаштування» моделі з нашим власним набором даних. Ідея полягає в тому, що ця попередньо навчена модель буде виступати в якості екстрактора ознак. Ми вилучаємо останній шар мережі і заміняємо його власним класифікатором (залежно від області задачі). Далі ми заморожуємо ваги всіх інших шарів і навчаємо мережу стандартним способом (замороження означає заборону змін ваг під час оптимізації).

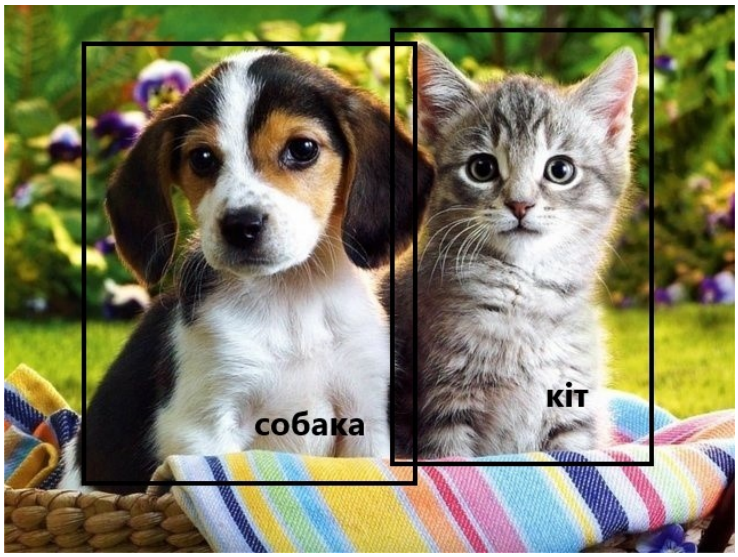


Рис. 14. Виявлення об’єктів – локалізація кількох об’єктів

Завдання **сегментації об’єктів** – це визначення класу та контуру кожного об’єкта у вхідному зображенні (рис. 15). [4]

Рис. 15. Сегментація об’єктів – визначення класу об’єкта та його виділення на вхідному зображенні

Розглянемо, чому це працює. Припустимо, попередньо підготовлена модель була навчена на ImageNet (ImageNet – набір даних, що містить 14 мільйонів зображень з більш ніж 1000 класами) [2]. Коли ми говоримо про низькорівневі шари мережі, ми знаємо, що вони будуть виявляти такі риси, як ребра та криві. Тепер, якщо в нас не дуже унікальний простір проблем і набір даних, мережі потрібно буде виявити криві і ребра. Замість того, щоб навчати мережу через ініціалізацію ваг випадковими значеннями, ми можемо використати ваги вже навченої моделі і зосередитися на більш важливих для навчання шарах. Якщо ж наш набір даних абсолютно інший ніж той що в ImageNet, тоді необхідно збільшити кількість шарів і заморозити лише кілька нижніх. [9]

Методи збільшення даних

Розглянемо, як можна збільшити існуючий набір даних лише за допомогою кількох легких перетворень. Як вже згадувалося раніше, коли комп’ютер отримує на вхід зображення, він отримує його у виді масиву чисел, що представляють пікселі. Припустимо, що все зображення зміщується вліво на один піксель. Для людини ця зміна непомітна. Однак для комп’ютера цей зсув може бути досить значним, оскільки клас зображення не змінюється, а масив пікселів – так. Підходи, що змінюють навчальні дані способами, які модифікують подання масиву, зберігаючи клас зображення, відомі як **методи збільшення даних.** Вони є способом штучного розширення набору даних. Також популярними підходами є використання чорно-білих зображень, випадкові обрізання, обертання та багато іншого. Застосувавши лише кілька цих перетворень до навчальних даних, можна легко подвоїти або потроїти кількість навчальних прикладів.

Висновки

У статті розглянуто поняття згорткових нейронних мереж, їх зв'язок з біологією та структуру. Далі більш детально розглянуто основні шари, що є в ЗНМ, описано їх роботу за допомогою математики. Кожен з шарів розглянуто окремо та у взаємодії з іншими шарами нейронної мережі.

Показано як нейронні мережі працюють, та як їх правильно навчати. Розглянуто кілька підходів до навчання нейронних мереж та прості способи вирішення поширених проблем, що виникають при навчанні: збільшення об’єму даних для навчання та

використання вже навчених мереж як бази для вирішення вузькоспеціалізованих задач. Наведено параметри, завдяки яким можна змінювати процес навчання мережі.

Розглянуто кілька найпоширеніших задач, які розв'язують за допомогою згорткових нейронних мереж: класифікація, локалізація, виявлення, сегментація.

Список використаної літератури:

1. C. Zhang, and Z. Zhang, "Improving multiview face detection with multi-task deep convolutional neural networks," IEEE Winter Conference on Applications of Computer Vision, 2014, pp. 1036-1041.
2. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In CVPR09.
3. Dumitru Erhan, Christian Szegedy, Alexander Toshev, Dragomir Anguelov, Scalable Object Detection using Deep Neural Networks, arXiv preprint arXiv: arXiv:1312.4400, 2014.
4. E. Borenstein; J. Malik, "Shape Guided Object Segmentation" in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006.
5. Geoffrey E. Hinton, Vinod Nair. "Rectified Linear Units Improve Restricted Boltzmann Machines", Proceedings of International Conference on Machine Learning (ICML), 2010, pp 807-814.
6. H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5325-5334.
7. Hubel, D. H., & Wiesel, T N. (1979). Brain mechanisms of vision. Scientific American, 241, 150–162.
8. J. Yan, Z. Lei, L. Wen, and S. Li, "The fastest deformable part model for object detection," in IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 2497-2504.
9. Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson, How transferable are features in deep neural networks?, arXiv preprint arXiv: arXiv: 1411.1792v1, 2014.
10. Min Lin, Qiang Chen, Shuicheng Yan, Network In Network, arXiv preprint arXiv: arXiv: 1312.2249, 2013.
11. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning Research 15, 2014, pp. 1929-1958.

Bibliography:

1. C. Zhang, and Z. Zhang, "Improving multiview face detection with multi-task deep convolutional neural networks," IEEE Winter Conference on Applications of Computer Vision, 2014, pp. 1036-1041.
2. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In CVPR09.
3. Dumitru Erhan, Christian Szegedy, Alexander Toshev, Dragomir Anguelov, Scalable Object Detection using Deep Neural Networks, arXiv preprint arXiv: arXiv:1312.4400, 2014.
4. E. Borenstein; J. Malik, "Shape Guided Object Segmentation" in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006.
5. Geoffrey E. Hinton, Vinod Nair. "Rectified Linear Units Improve Restricted Boltzmann Machines", Proceedings of International Conference on Machine Learning (ICML), 2010, pp 807-814.
6. H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5325-5334.
7. Hubel, D. H., & Wiesel, T N. (1979). Brain mechanisms of vision. Scientific American, 241, 150–162.
8. J. Yan, Z. Lei, L. Wen, and S. Li, "The fastest deformable part model for object detection," in IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 2497-2504.
9. Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson, How transferable are features in deep neural networks?, arXiv preprint arXiv: arXiv: 1411.1792v1, 2014.
10. Min Lin, Qiang Chen, Shuicheng Yan, Network In Network, arXiv preprint arXiv: arXiv: 1312.2249, 2013.
11. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning Research 15, 2014, pp. 1929-1958.

KUCHER Oleksandr,
PhD-student, The Bohdan Khmelnytsky National University of Cherkasy

FEATURES OF APPLICATION OF CONVERSIONAL NEURAL NETWORKS FOR IMAGE PROCESSING PROBLEMS

Summary. Introduction. *Convolutional Neural Networks (CNN) are one of the most influential innovations in the field of computer vision. For the first time neural networks attracted attention in 2012 at the ImageNet competition. With the help of CNN, a new record of classification errors was set*

- 15% (the previous value was 26%). Today, many well-known companies use deep learning. But the classic and most popular way to use neural networks is through image processing. Consider how CNNs are used to classify images.

Purpose. *The task of image classification is to process the image and determine the class to which it belongs (car, animal, etc.) or the group of classes that best characterize it.*

Results. *The article gives the basic concept and characteristics of convolutional neural networks, describes the structure and mathematical approaches to the implementation of this type of neural network. Provided description of the main layers of convolutional neural networks, mentioned parameters that allow changing the process of training the network. The process of training each of the neural network's layer is considered in detail. In addition, described techniques that allow you to increase the data set and improve the training process of the network only by several transformations.*

Conclusion. *The concept of convolutional neural networks, their connection with biology and structure has been considered in the article. The main layers that are found in CNN are discussed in detail, described their work through mathematics. Each of the layers is considered individually and in conjunction with the other layers of the neural network.*

Described how neural networks work and how to train them properly. Several approaches to learning neural networks are discussed, as well as simple ways to solve common learning problems: increasing the amount of data you need to train and using already trained networks as a base for solving specialized tasks. Here are some options for changing your network's learning process.

The article discusses some of the most common tasks that are solved using convolutional neural networks: classification, localization, detection, and segmentation.

Keywords: *deep learning, machine vision, neural networks, face recognition, convolutional neural networks, receptive fields, deep learning, neural networks, convolutional neural networks, classification, localization, detection, segmentation, rectified linear units.*

Одержано редакцією 11.04.2018 р.
Прийнято до публікації 19.09.2018 р.

УДК 519.688

DOI 10.31651/2076-5886-2019-1-75-85

PACS 02.60.-x, 02.60.Pn

КУЦЕНКО Олександр Анатолійович
студент спеціальності «Прикладна математика» Черкаського національного університету імені Богдана Хмельницького
e-mail: alexkutsenko1995@gmail.com
ORCID 0000-0002-6220-6034

СЕРДЮК Олександр Анатолійович
кандидат економічних наук, старший викладач кафедри прикладної математики та інформатики Черкаського національного університету ім. Б. Хмельницького
e-mail: serdyuk4labs@ukr.net
ORCID 0000-0002-3919-4661

ОГЛЯД АЛГОРИТМІВ ПОШУКУ ПЛАГІАТУ У ПРОГРАМНОМУ КОДІ

У статті розглянуто поняття плагіату, його класифікацію та загальновживані алгоритми пошуку плагіату, а саме: метод ідентифікаційних міток, алгоритм Хескела, метод вирівнювання рядків та метод жадібного рядкового заміщення. З розвитком технологій та можливістю використання інтернету студент, не прикладаючи багато зусиль, може видавати матеріали іншої персони за свої. Тому дані методи та алгоритми часто

використовуються для пошуку плагіату у програмному коді студентів. Метою даної роботи є визначення поняття плагіату, описання класифікації та розгляд методів ідентифікаційних міток, вирівнювання рядків, жадібного рядкового заміщення та алгоритму Хескела. Також розглянуто програмне забезпечення та сайти пошуку плагіату.

Ключові слова: *метод ідентифікаційних міток, алгоритм Хескела, метод вирівнювання рядків, метод жадібного рядкового заміщення, задача пошуку плагіату.*

Постановка проблеми

Нині при швидкому розвитку інформаційних технологій інтелектуальна власність стає цінішою, аніж раніше. Беручи до уваги швидке зростання обсягів цього виду власності, виникає потреба у захисті авторських прав для перевірки авторства та пошуку плагіату. Задача пошуку та виявлення фрагментів програмного коду, що був запозичений у іншої людини, залишається однією з найбільш актуальних, складних та важливих проблем для викладачів, що навчають програмуванню.

Якщо дві програми мають істотну загальну частину (на рівні мови програмування), то можна вважати, що в одній з них міститься плагіат; причому, плагіатор може змінити оригінальну програму вставкою додаткових операторів (синонімізація та виконання несуттєвих дій), перейменуванням змінних, зміною порядку виконання незалежних операторів, розбиттям деяких функцій на дві і так далі.

Об'єктом дослідження у роботі виступать алгоритми та методи пошуку плагіату.

Метою статті є розгляд методів ідентифікаційних міток, вирівнювання рядків, жадібного рядкового заміщення алгоритм Хескела.

Виклад основного матеріалу

1. Плагіат та класифікація його типів та видів.

Плагіат - це протизаконне використання роботи іншої людини та оприлюднення її під грифом «власної» праці. Плагіатор – людина, яка привласнює чужі праці, роботи або частину рооти під власним іменем, присвоюючи собі їх авторство.

Відповідно до [4], плагіат – це оприлюднення (опублікування) повністю або частково чужого твору під іменем особи, яка не є автором.

Найчастіше плагіат зустрічається у наукових чи творчих околах, де відбувається присвоєння чужих наукових, творчих, художніх творів тощо.

Метою плагіату чи плагіатора є, переважно, бажання стати відомим та/або отримати певну грошову винагороду за рахунок іншої особи.

Згідно з [7], мета плагіату наукового твору – виправдання витрачених бюджетних коштів або коштів замовника шляхом привласнення результатів чужої інтелектуальної праці.

Відповідно до історичних даних, плагіат літературних творів існував ще в давні часи.

У сьогоденні, розвиваючи інформаційні технології, людина створює усі умови для швидкого пошуку плагіату. Раніше автори не мали можливості дізнатися про те, що хтось використовує їх роботу, але зараз приховати це – нелегка задача: написавши у пошуковому рядку деякі ключові слова з роботи, яку автор підозрює у запозиченнях, можна знайти відповідності зі своєю працею та визначити ступінь запозичень.

З розвитком інформаційних технологій до візуального пошуку плагіату шляхом порівняння двох творів додалися й технічні засоби, що полягають у автоматичному порівнянні тексту з іншими [7].

Існує багато різних типів і видів плагіату. Найбільш поширені з них наведено у таблиці 1 (відповідно з [5]).

2. Загальновживані алгоритми пошуку плагіату

Наразі існує досить багато алгоритмів пошуку плагіату, тому описати всі неможливо. Однак, декілька з них наведені нижче.

Таблиця 1

За наявністю умислу:	
Ненавмисний плагіат – у разі незнання вимог, яким повинна відповідати робота.	*Навмисний плагіат.*
За формою відтворення:	
Прямий (відкритий) плагіат – пряме відтворення (відображення) чужого твору або його частини під своїм іменем.	*Завуальований плагіат* – за умов, якщо текст твору зазнає несуттєвих змін шляхом заміни окремих слів та виразів їх синонімічними аналогами. При цьому форма в цілому не змінюється.
За наявністю вказівки джерела:	
Запозичення без вказівки джерела: *"примарний автор"* – автор видає виконану іншою людиною роботу за свою, не змінюючи її зміст; *"фотокопія"* – автор копіює значну частину тексту (але не весь текст) з одного джерела, не вносячи до нього змін; *"натрапив на влучний матеріал"* – робиться спроба приховати плагіат шляхом копіювання з декількох різних джерел, текст яких не змінюється, але автор пише свої перехідні фрази між частинами тесту; *"погане маскування"* – залишається зміст тексту джерела, але деякі формулювання замінюються; *"праця ліні"* – трудомісткий процес, у результаті якого інформація з різних джерел практично повністю перефразовується, пишеться в одному стилі; *"вкрав у себе"* – передбачає запозичення тексту з власних, більш ранніх робіт.	***Запозичення з вказівкою джерела:*** *"забуте посилання"* та *"дезінформатор"* – пов'язані з неправильним або помилковим оформленням посилань на джерело; *"занадто ідеальне перефразування"* – якщо дослівна цитата не взята в лапки. Таким чином, у читача створюється невірне враження про те, що автор навів свою оригінальну інтерпретацію поглядів, викладених у джерелі; *"ідеальний злочин"* – вчиняється, коли автор правильно наводить деякі цитати, а решту перефразує. У результаті читач помилково думає, що перефразований текст є авторським аналізом цитованим думок; *"рясне цитування"* або *компіляція* – відбувається з дотриманням усіх правил цитування та перефразування, але робота практично не містить оригінальних результатів авторського дослідження.

Джерело: відповідно до [5]

Нижче розглянуто окремі, найбільш поширені, алгоритми пошуку плагіату.

Метод жадібного рядкового заміщення

Відповідно до [3], приймає на вході два рядки, а на виході повертає набір їх загальних непересічних підрядків.

Нехай P і T – токенізоване представлення порівнюваних програм.

Перша фаза методу. Шукаємо максимальні спільні підрядки P і T, які не відмічені (спочатку алгоритму всі елементи непомічені). Для цього використовуються три вкладених цикли: перший пробігає по P_p, другий по T_t, а третій знаходить максимальний префікс в P_p і T_t.

Потім відбувається сортування:

- якщо *maxmatch* менше – видаляється зі списку загальних підрядків *matches* все до цього додане і поміщається туди знайдений префікс;
- якщо *maxmatch* більше – нічого не змінюється;
- якщо рівні – додається найбільший префікс P_p і T_t до списку *matches*.

Друга фаза методу. Йдемо по списку, якщо поточний елемент списку – підрядок, який не містить помічених елементів, то записуємо у кінцевий tiles та помічаємо всі елементи розглянутого рядка, що входять до *P* і *T*. Якщо ж довжина рядків у *maxmatch* більша за *MinimumMatchLength*, то повертаємося до першої фази.

Псевдокод алгоритму наведено далі [3]:

```
Greedy-String-Tiling(String P, String T) {
  tiles = {};
  do {
    maxmatch = MinimumMatchLength;
    Forall unmarked tokens Pp in P {
      Forall unmarked tokens Tt in T {
        j = 0;
        while (Pp+j == Tt+j) && unmarked(Pp+j) && unmarked(Tt+j)
          j++;
        if (j == maxmatch) {
          matches = matches ∪ match(p, t, j);
        } else if (j > maxmatch) {
          matches = {match(p, t, j)};
          maxmatch = j;
        }
      }
    }
    Forall match(p, t, maxmatch) ∈ matches {
      if (not occluded) {
        for j = 0...(maxmatch − 1) {
          mark(Pp+j);
          mark(Tt+j);
        }
        tiles = tiles ∪ matches(a, b, maxmatch);
      }
    }
  } while (maxmatch > MinimumMatchLength);
  return tiles;
}
```

У даному методі використовується кілька евристик:

- довші послідовні збіги кращі, ніж набір менших і непослідовних, незалежно від суми довжини останнього;
- ігноруються збіги, довжини яких менші за певне значення порогу.

Ці евристики сприяють втраті оптимального заміщення, проте результати виконання роботи методу дуже близькі до нього для виявлення плагіату.

Асимптотика у найгіршого випадку – $O(n^3)$, але на практиці найчастіше – $O(n^2)$, де n – довжина рядка, у яку переводять програму, використовуючи токенізоване подання.

Метод вирівнювання рядків

Нехай є два файли програмного коду, представлених у вигляді токенізованих рядків s і t відповідно (можливо, різної довжини). Тепер можна використати метод локального вирівнювання рядків, розроблений для визначення подібності ДНК. Вирівнювання виконується за допомогою вставки пропусків так, щоб довжини рядків стали рівними. Існує багато варіантів вирівнювань двох рядків. Наприклад, для рядків "masters" і "stars" цілком допустиме наступне вирівнювання:

masters masters
sta rs stars

Тепер нехай, s і t після вирівнювання, відповідно s' і t'. Розглянемо s_i і t_i: вартість їх збігу – m, вартість пропуску – g, вартість розбіжності – d, де m, d і g – довільні числа. Ціна вирівнювання – це сума вартостей усіх пар s'_i і t'_i, максимальне значення цієї цільової функції для всіх i, j $(i \le j \le |s'| = |t'|)$ у рядках $s'[i..j]$ і $t'[i..j]$ – розмір вирівнювання. Якщо $m = 1$, $d = -1$ і $g = -2$, тоді "rs/rs" та "sters/stars" – фрагменти рядків з максимальним значенням цільової функції. Тоді ціна їх вирівнювань -2 і 3. Вона відповідає редакційній відстані і використовується для визначення відстані між схожими об'єктами, а також успішно використовується для визначення неточності ДНК у обчислювальній біології .

Оптимальне вирівнювання – це таке максимальне значення цільової функції серед усіх вирівнювань, яке можна обрахувати за допомогою динамічного програмування. Нехай s та t – рядки, $D(i, j)$ – оптимальне вирівнювання $s[1..i]$ і $t[1..j]$. Необхідно знайти $max_{1 \le i \le |s|, 1 \le j \le |t|}(D(i, j))$. Визначимо

$$score(s[i], t[j]) = \begin{cases} m, if\ s[i] = t[j] \\ d, else \end{cases}$$

Наступне рекурентне співвідношення дозволяє нам знайти необхідне оптимальне вирівнювання

$$D(i,j) = \begin{cases} D(i-1, j-1) + score(s[i], t[j]) \\ D(i-1, j) + g \\ D(i, j-1) + g \\ 0 \end{cases}$$

Граничні умови задаються співвідношеннями $D(0, i) = 0$ і $D(j, 0) = 0$. Інші елементи матриці D обчислюються при проході зліва направо і згори донизу. Це можливо, оскільки $D(i, j)$ залежить від $D(i-1, j-1)$, $D(i-1, j)$ і $D(i, j-1)$. Час обчислення оптимального вирівнювання – $O(|s||t|)$; витрати пам'яті – $O(max(|s|, |t|))$, тому що для обчислення необхідні лише два рядки.

Використання методу виглядає приблизно так: отримуємо токенізоване представлення p_1 і p_2 двох програм, ділимо другий рядок p_2 на підрядки, кожен з яких представляє модуль вихідної програми. Для кожного підрядка і p_1 отримуємо значення оптимального локального вирівнювання. Це дозволяє методу коректно обробляти перестановки модулів вихідної програми.

Переваги та недоліки

Якщо розглядати реалізацію алгоритму, використовувану детектором SIM, – ніяких покращень порівняно з іншими алгоритмами не отримується, зокрема, на базі цього алгоритму не можна організувати базу даних, яка прискорює перевірку *один-проти-всіх*. Однак, застосування алгоритму до токенізації програмного коду у нашому розумінні, можливо, дозволить отримати хороші результати, що нами буде проводитись у подальшому.

Метод ідентифікаційних міток

Згідно [9], під час перевірки на плагіат необхідно знайти копії чи фрагменти копій у текстовій базі. У такому випадку безпосереднє порівняння файлів є неефективним.

Метод ідентифікаційних міток дозволяє перетворити файл у більш коротку послідовність: файл зіставляється з набом ідентифікаційних міток.

Нехай дано довільний текст:

abracadabra (складається з 11 символів; $m = 11$).

Розглянемо набір міток для файлу на даному прикладі.

k-грамом називається група будь-яких *k* символів розташованих підряд. Побудуємо *k*-грами для прикладу при, наприклад, $k = 3$:

abr, bra, rak, aka, kad, ada, dab, abr, bra

Кількість *k*-грамів, які можна побудувати для тексту довжини *m*, позначимо *n*:

$n = (m - (до - 1))$ (тут $n = 9$).

Проведемо хешування усіх *k*-грамів. Для даного прикладу послідовність хеш-значень буде такою: 12, 35, 78, 3, 26, 48, 55, 12, 35.

Використання усіх значень нераціональне, тому вибирається невелика їх кількість. Обрані хеш-значення стають меншими файлами. Окрім мітки зберігається також інформація про те, до якого файлу вона належить. Якщо хеш-функція дає малу ймовірність колізій, то однакові мітки у наборах двох файлів свідчать про те, що у них є спільний фрагмент, а за кількістю спільних міток можна говорити про схожість файлів.

Переваги методу:

- можна створити базу даних для перевірку *один-проти-всіх*;
- плюси токенізованого представлення;
- спільні підрядки, які менші певного порогу, ігноруються;
- метод нечутливий до переміщення фрагментів коду.

Недоліки:

- можливість збігу токенізованого представлення програм, але відсутність збігу у початкових кодах програм.

Алгоритм Хескела

Розглянемо алгоритм, описаний у [9].

Нехай маємо два програмних коди, які подані у вигляді списку токенів *a* і *b* відповідно. Одним з критеріїв подібності вважається довжина *найбільшого спільного*

підрядка. Ми завжди маємо змогу знайти такий елемент рядка a_i, що НЗП (найбільша загальна (спільна) підпослідовність) рядків $a_0=a_{|a|}a_{|a|}$ *-1 ...* a_ia_1 *...* a_{i-1} і b буде меншою (щонайбільше – у 2 рази), аніж НЗП (a, b) (якщо НЗП (a, b) > 1). Щоб уникнути цього, скористаємося алгоритмом Хескела, який вимагає кількох проходів по циклах і при цьому виконується за лінійний час. Розкладемо a і b на k-грами. Знайдемо в a і b ті k-грами, які зустрічаються лише один раз. Для кожної пари перевіряємо, чи елементи рядків схожі з тими, що знаходяться до них; якщо так, то повторюємо ті ж дії для них і так далі, поки не знайдеться розбіжність. Аналогічно для рядків, які лежать далі. У результаті, отримується набір спільних підрядків a і b, що не перетинаються. Їх загальна довжина буде інтерпретуватись як подібність програм, що відповідають a і b.

Беручи до уваги подане у [2], можна навести наступні приклади переваг і недоліків цього методу.

Переваги:

– лінійна залежність роботи методу від часу.

Недоліки:

– можливість збігу токенізованого представлення програм, але відсутність збігу у початкових кодах програм;
– мала кількість унікальних k-грамів у великих файлах;
– вставка в блок або зміна на еквівалентний оператор у більшості випадків буде приводити до пропуску цієї частини блоку;
– не можна створити базу даних для перевірку *один-проти-всіх*.

3. Огляд програмного забезпечення та сайтів пошуку плагіату

Якщо розглядати сайти, які призначені для пошуку плагіату, можна виділити наступні.

Turnitin (http://submit.ac.uk/) – сайт перевіряє роботу на унікальність, використовуючи базу даних різних робіт (авторських, дипломних тощо). Якщо ж такої роботи не було знайдено у базі, то система доповнює свою базу цим файлом. [5]

AntiPlagiat.ru (http://www.antiplagiat.ru) – чимось схожий на *Turnitinш*, але на цьому ресурсі зареєструватись може як викладач, так і студент (щоб користуватися всіми можливостями, необхідно купити платний аккаунт).

Unplag – пошук плагіату можливий як у режимі реального часу, так і по базі даних, у якій збережено велику кількість документів; працює з Canvas, Sakai, Moodle.

www.miratools.ru – дозволяє здійснювати онлайн-перевірку тексту на плагіат. Система використовує результати видачі пошукових систем.

www.istio.com – здійснює перевірку тексту на унікальність з використанням систем Яндекс.XML і Yahoo.com.

Plagiatinform – перевіряє тексти на унікальність у власній базі та у Інтернеті. Сайт може знаходити не унікальні фрагменти тексту у вигляді текстів, де «перемішані» фрагменти тексту кількох джерел.

Copyscape – дозволяє шукати копії сайтів у мережі Інтернет; повертає список сайтів, де можливий збіг*;* використовує системи Google і Yahoo!

http://plagiarisma.net/ – веб-сайт, за допомогою якого може проводитись перевірка тексту курсових, кваліфікаційних та магістерських робіт; використовує системи Google та Bing.

Якщо ж, брати до уваги програмне забезпечення, призначене для перевірки тексту на унікальність і доступне як викладачам, так і студентам, то можна вказати наступні [5]:

Advego Plagiatus (http://advego.ru/plagiatus/);
Etxt Антиплагиат (http://www.etxt.ru/antiplagiat/);
Anti-Plagiarism (http://ikc2.tup.km.ua/index_ru.shtml).

Розглянемо кожну програму окремо.

1. *Advego Plagiatus* – програма, яка шукає в Інтернеті фрагменти або повні копії тексту; має зручний та простий інтерфейс. Результат роботи – відсоток унікальності тексту та відсоток збігу тексту. Також надає список джерел тексту, де були взяті фрагменти (рис. 1). [5]

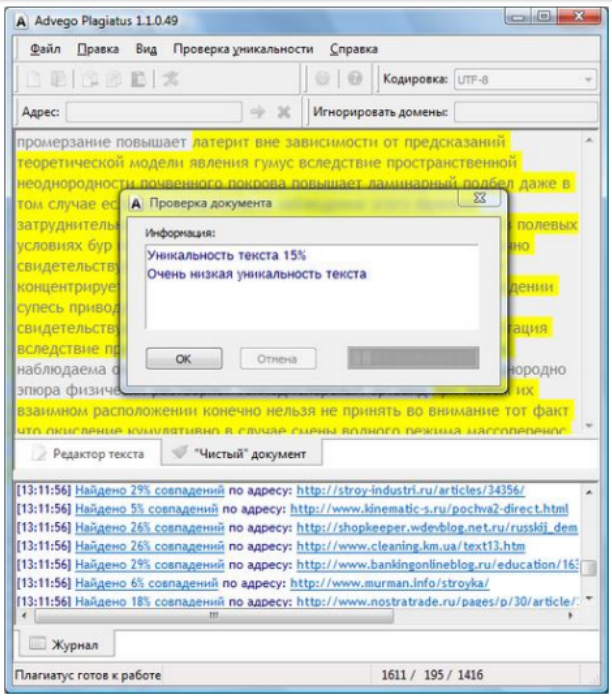


Рис. 1. Програма Advego Plagiatus

2. *Etxt Антиплагіат* – програма, призначена для перевірки тексту чи документу на унікальність. Надає можливість провести порівняльний аналіз фрагментів власного тексту на знайдених джерелах (рис. 2). [8]

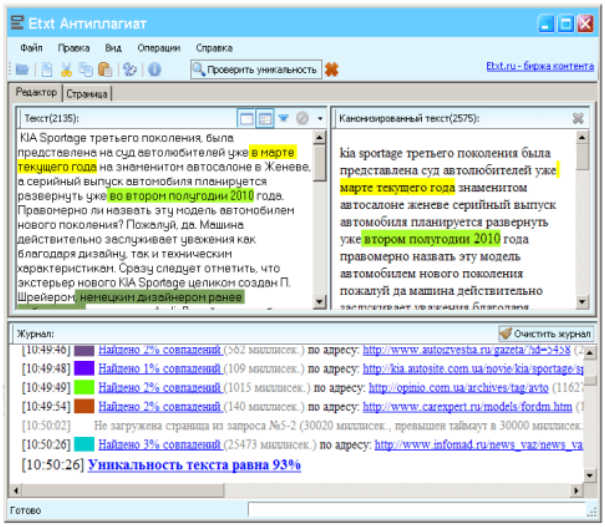


Рис. 2. Програма Etxt Антиплагіат

Anti-Plagiarism – програма, призначена для виявлення і запобігання плагіату. Це ефективний інструмент для боротьби з копіюванням інформації з вебу (рис. 3). [5]

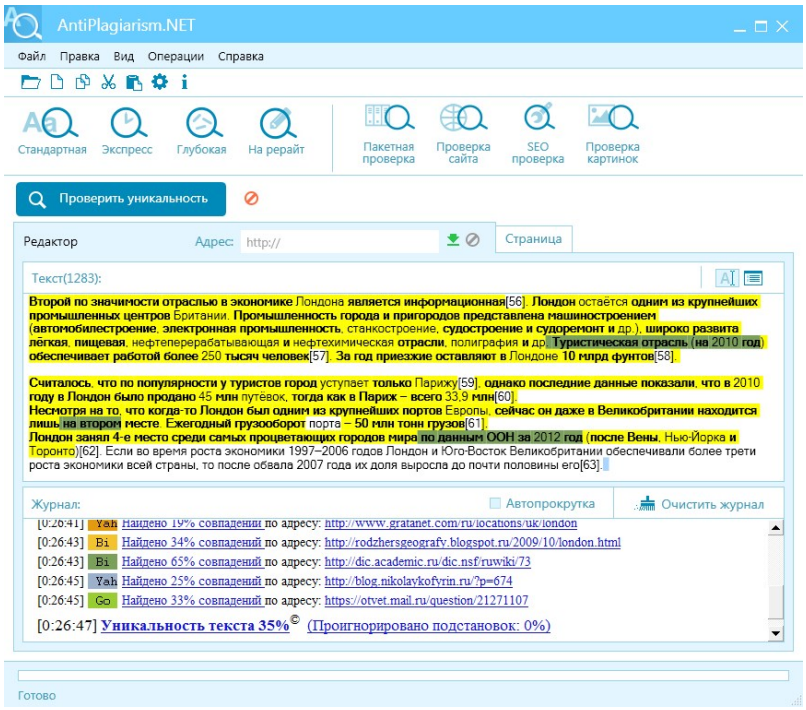


Рис. 3. Програма Anti-Plagiarism

Висновки

У статті розглянуто окремі методи та алгоритми пошуку плагіату, а саме: метод ідентифікаційних міток, алгоритм Хескела, метод вирівнювання рядків, метод жадібного рядкового заміщення. Представлено загальний опис кожного з методів та алгоритмів. Кожен з них потенційно може бути використаний для виявлення плагіату у програмному коді. Однак, при використанні токенізованого подання коду програми усі розглянуті алгоритми мають головний недолік: можливі збіги токенізованого представлення програм при відсутності збігу у початкових програмах.

На основі поданого у статті можна стверджувати, що наразі фактично немає якогось одного загальноприйнятого універсального способу, який можна було б використовувати для розв'язання задачі пошуку плагіату у програмному коді. Тому у подальшому нами буде проводитись робота по модифікації існуючих алгоритмів для адаптації їх до розв'язання задачі аналізу вихідного коду програм на наявність запозичень.

Список використаної літератури:

1. Michael J. Wise. String similarity via greedy string tiling and running KarpRabin matching. Dept. of CS, University of Sydney, December 1993.
2. Амонс А.А. Выявление плагиата в программном коде C# / Амонс А.А., Зайцев С.Ю., Киричек А.А. // Вестник НТУУ «КПИ» Информатика, управление и вычислительная техника № 53. – с.170-179.
3. Евтифеева О.А. АНАЛИЗ АЛГОРИТМОВ ПОИСКА ПЛАГИАТА В ИСХОДНЫХ КОДАХ ПРОГРАММ / Евтифеева О.А., Красс А.Л., Лакунин М.А., Лысенко Е.А., Счастливцев Р.Р. // Санкт-Петербургский государственный университет – С. 188-196.
4. Закон України Про авторське право і суміжні права // Відомості Верховної Ради України (ВВР). – 1994. – № 13. – С. 64 .
5. Лисиченко М. Л. Методичні рекомендації щодо механізму перевірки письмових робіт на плагіат / М. Л. Лисиченко, В. І. Жила, А. В. Левкін. – Х: ХНТУСГ, 2017. – 28 с.
6. Лупаренко Л. А. Інструментарій виявлення плагіату в наукових роботах: аналіз програмних рішень / Л. А. Лупаренко // Інформаційні технології і засоби навчання. – 2014. – Том 40, № 2. – С. 151-169.

7. Плагіат авторських творів [Електронний ресурс] – Режим доступу до ресурсу: https://pidruchniki.com/1834071963579/pravo/plagiat_avtorskih_tvoriv.
8. Плагіат у студентський роботах: методи виявлення та запобігання / [Н. В. Стукало, К. В. Ковальчук, М. В. Литвин та ін.]. – Дніпропетрівськ: ДНУ ім.О. Гончара, кафедра міжнародної економіки і світових фінансів, 2013. – 44 с.
9. Чижова А. А. АЛГОРИТМИ ПОШУКУ ПЛАГІАТУ / А. А. Чижова // Восточно-Европейский журнал передовых технологий. – Х., 2010. - № 2(4). – с. 13-16.
10. Що таке плагіат? [Електронний ресурс] – Режим доступу до ресурсу: https://uk.kagouletheband.com/ucheba/28889-chto-takoe-plagiat.html.

Bibliography:

1. Michael J. Wise. String similarity via greedy string tiling and running KarpRabin matching. Dept. of CS, University of Sydney, December 1993.
2. Amons A.A. Vyiyavlenie plagiata v programmnom kode C# / Amons A.A., Zaytsev S.Yu., Kirichek A.A. // Vestnik NTUU «KPI» Informatika, upravlenie i vyichislitelnaya tehnika # 53. – s.170-179.
3. Evtifeeva O.A. ANALIZ ALGORITMOV POISKA PLAGIATA V ISHODNYiH KODAH PROGRAMM / Evtifeeva O.A., Krass A.L., Lakunin M.A., Lyisenko E.A., Schastlivtsev R.R. // Sankt-Peterburgskiy gosudarstvennyiy universitet – S. 188-196.
4. Zakon Ukrainy Pro avtorske pravo i sumizhni prava // Vidomosti Verkhovnoi Rady Ukrainy (VVR). – 1994. – № 13. – S. 64 .
5. Lysychenko M. L. Metodychni rekomendatsii shchodo mekhanizmu perevirky pysmovykh robit na plahiat / M. L. Lysychenko, V. I. Zhyla, A. V. Levkin. – Kh: KhNTUSH, 2017. – 28 s.
6. Luparenko L. A. Instrumentarii vyiavlennia plahiatu v naukovykh robotakh: analiz prohramnykh rishen / L. A. Luparenko // Informatsiini tekhnolohii i zasoby navchannia. – 2014. – Tom 40, № 2. – S. 151-169.
7. Plahiat avtorskykh tvoriv. Retrieved from: https://pidruchniki.com/1834071963579/pravo/plagiat_avtorskih_tvoriv.
8. Plahiat u studentskyi robotakh: metody vyiavlennia ta zapobihannia / [N. V. Stukalo, K. V. Kovalchuk, M. V. Lytvyn ta in.]. – Dnipropetrivsk: DNU im.O. Honchara, kafedra mizhnarodnoi ekonomiky i svitovykh finansiv, 2013. – 44 s.
9. Chizhova A. A. ALGORITMI POShUKU PLAGIATU / A. A. Chizhova // Vostochno-Evropeyskiy zhurnal peredovyih tehnologiy. – H., 2010. - # 2(4). – s. 13-16.
10. Shcho take plahiat? Retrieved from: https://uk.kagouletheband.com/ucheba/28889-chto-takoe-plagiat.html.

KUTSENKO Oleksandr,
student, The Bohdan Khmelnytsky National University of Cherkasy
SERDIUK Oleksandr,
PhD, Senior Lecturer, The Bohdan Khmelnytsky National University of Cherkasy

PECULIARITIES OF WORK OF PLAGIATE SEARCH ALGORITHMS IN SOFTWARE

Summary. Introduction. *In recent years, students are increasingly practicing the assignment of other people's works and giving them for their own. All this is due to the development of computer technology and the possibility of using the Internet. The student without putting a lot of effort copies someone else's program code, and brings it to the teacher as his own.*

There are quite a few algorithms to detect plagiarism in the works of students.

The most common algorithms and methods that can cope with this task are the methods of identification labels, string alignment, greedy string substitution, Heskel algorithm. Each of these algorithms has its advantages and disadvantages. The main advantages of these algorithms are that they all have a linear dependence of the method on time and work with tokenized representation of the program code, which makes the task of checking for plagiarism much easier.

Purpose. *The purpose of this paper is to define what plagiarism is and its classification, review existing software and sites that are capable of checking text for uniqueness and review methods of identification labels, string alignment, greedy string substitution Heskel algorithm*

Results. *At the beginning of the analysis of the topic of this article, there was a problem, and all-so what can be considered as plagiarism of software code. Then there are fair questions:*

- *how different programs must be for one of them to be considered plagiarism;*
- *what can be cited as evidence to be sufficient for plagiarism.*

The paper considers various methods and algorithms for plagiarism search in the software code, namely the method of identification labels, the Heskel algorithm, the method of string alignment,

the method of greedy string tiling. A general description of each of the methods and algorithms is presented. Each of them is able to detect plagiarism in the program code, but because of their tokenized representation, they all have one common disadvantage: the main drawback of all these methods is that the tokenized representation of programs may match, but there is no coincidence in the initial codes of programs.

Conclusion. *Taking into account what was said above about the algorithms and methods of plagiarism detection, we can conclude that now there is actually no one universally accepted universal way that could cope with the task of finding plagiarism in the software code, a way that would not be subject to time. In the future, we will develop a method for searching for plagiarism in program code based on tokenization of source programs.*

Keywords*: identification tag method, line alignment method, greedy string tiling method, the Heskel algorithm, plagiarism search problem.*

Одержано редакцією 27.09.2018 р.
Прийнято до публікації 19.12.2018 р.

ЗМІСТ

СЕКЦІЯ «ПРИКЛАДНА МАТЕМАТИКА»

СЕКЦІЯ «ІНФОРМАТИКА»

CONTENTS

APPLIED MATHEMATICS SECTION

INFORMATICS SECTION

ВІСНИК ЧЕРКАСЬКОГО УНИВЕРСИТЕТУ

Серія Прикладна математика. Інформатика
№1.2019

Відповідальний за випуск
Головня Б.П.

Відповідальний секретар
Гришко Л.В.

Комп'ютерне верстання
Гришко Л.В.

Підписано до друку 07.02.2019.
Формат 60x84/8. Папір офсет. Гарнітура Times.
Ум. др. арк 10,23. Наклад 300 прим.

Це видання надруковано на папері із деревини відповідної нормам екологічного лісовикористання

Надруковано ФОП Гордієнко Є.І.
Свідоцтво про внесення суб'єкта видавничої справи до Державного реєстру видавців, виготовників і розповсюджувачів видавничої продукції
Серія ДК № 4518 від 04.04.2013 р.
Україна, 18000, м. Черкаси
тел./факс: (0472) 56-56-12, (067) 444-28-94
e-mail: book.druk@gmail.com