European Funds
Knowledge Education Development

Republic of Poland

NAWA
POLISH NATIONAL AGENCY FOR ACADEMIC EXCHANGE

European Union
European Social Fund

Ihor Hroznyi

ARTIFICIAL INTELLIGENCE

Lublin 2021

Artificial intelligence

Ihor Hroznyi

Artificial intelligence

Ihor Hroznyi

Lublin 2021

Reviewer:

- Dr. Volodimir Malyukov

Funding: Research, preparation of materials and preparation of the textbook were carried out under the project – grant no. PPI/KAT/2019/1/00015/U/00001 "Cognitive technologies – second-cycle studies in English" and were carried under the KATAMARAN program Polish National Agency for Academic Exchange (NAWA). The program is co-financed by the European Social Fund under the Knowledge Education Development Operational Program, a non-competition project entitled "Supporting the institutional capacity of Polish universities through the creation and implementation of international study programs" implemented under Measure 3.3. Internationalization of Polish higher education, specified in the application for project funding no. POWR.03.03.00-00-PN 16/18.

The project was carried out in cooperation with the Silesian University of Technology (project leader – Poland) and the Kiev National University of Construction and Architecture (project partner – Ukraine).

Project manager: prof. SUT. Aleksandra Kuzior, PhD., Dsc.
Department of Applied Social Sciences
Faculty of Organization and Management
Silesian University of Technology

Typesetting:
Monika Maciąg

Cover designer:
Marcin Szklarczyk

© Copyright by Politechnika Śląska

ISBN 978-83-66489-58-5

Publisher:
Wydawnictwo Naukowe TYGIEL sp. z o.o.
ul. Głowackiego 35/341,
20-060 Lublin
www.wydawnictwo-tygiel.pl

Table of contents

Artificial Intelligence is rebuilding the Digital Economy and will soon change the economy of the material world.

Klaus Martin Schwab

founder and permanent president

World Economic Forum in Davos

There are a lot of areas in which the level of Artificial Intelligence has already exceeded the level of human. Systems have appeared, which can play logical games and win victories over people.

Nick Bostrom

In 2016, Microsoft, Amazon, Google, Facebook and IBM announced the launch of an AI partnership, which aim is the benefit of the individual and society and the main goal is to improve the understanding of AI by society, as well as the creation of methodological recommendations on problems and prospects in this area.

Satya Nadella

General Director of Microsoft

The intellectualization of information technology, which is characteristic of the modern level of development of information and communication technologies, has drawn attention to such a discipline as Artificial Intelligence.

Nowadays, in the information society, competitive advantages are no longer determined by the size of the country or natural resources. Now everything is decided by the level of education and the amount of knowledge accumulated by society.

In the near future, will prosper states that will be able to surpass others in the creation and development of new knowledge. A special role in this is played by information and communication technologies, in particular methods and means of Artificial Intelligence.

Artificial intelligence (AI) – the science and technology of creating intelligent machines (software) that can take over certain functions of human intellectual activity (for example, to choose and make optimal decisions based on previous experience and rational analysis of external influences).

According to another definition, Artificial Intelligence means a scientific field, the tasks of which is hardware or software modeling of types of human activity that are traditionally considered intellectual, set and solved. In this sense the term of Artificial Intelligence was introduced by John McCarthy in 1956.

The history of AI as a new scientific direction begins in the middle of the 20^{th} century. By this time, many prerequisites for inception had already been formed: among philosophers there for a long time had been debates about the nature of man and the process of understanding the world, neurophysiology, and psychology had developed numerous of theories regarding the robots, the human brain and thinking, economists and mathematicians were asking questions of optimal calculations and ideas about the world in formalized form; finally, the foundation of the mathematical theory of computation – the theory of algorithms – was born and the first computers were created.

The purpose of this manual is to outline the main directions and methods used in AI, as well as to determine the possibility of their use in the context of cognitive technologies.

The manual contains an introduction, three sections, and a list of sources. It addresses the fundamental issues of Artificial Intelligence and the varieties of various Intelligent Systems. It contains information on knowledge representation models, as well as the expert systems and Neural Networks.

The manual contains illustrations, each section contains questions for self-control, which will allow students to test their knowledge of the sections of the textbook.

Chapter 1. Artificial intelligence as a scientific field of cognitive technologies

1.1. The history of the development of AI as a scientific direction

1.2. The structure of AI

1.3. Main strategies, problems and prospects for the development of AI

1.4. Cognitive technologies AI

1.1. THE HISTORY OF THE DEVELOPMENT OF AI AS A SCIENTIFIC DIRECTION

Intelligence is the ability of the brain to think, to operate with knowledge to make certain decisions about a particular task. It is the brain's ability to solve intellectual problems by acquiring memorizing and purposefully transforming knowledge in the process of learning, gaining life experience, and adapting to various external and internal circumstances. The definition of "intelligence" by the term of "knowledge" we mean not only the information that enters the brain through the senses, this information is important but insufficient for intellectual activity.

Person – this is the most complex of objects available for our perception, and the ability to think is the main property. Artificial intelligence is a science that aims at studying and modeling the main property of – human thinking.

The history of attempts to create an Artificial Likeness of the human mind has more than 700 years. The first attempt to create a machine that simulates the human mind is associated with the name of the Spanish knight, poet, philosopher, theologian, alchemist, and inventor Raymund Lullius (Latin Raymundus Lullius), who at the end of XIII invented the logical machine. It was a mechanical expert system, which was endowed with a base knowledge, water-output devices and the natural language of communication. At the age of 80, opponents of Raimund Lullius were beaten by stoned.

In 1910-1913 Bertrand Russell and Alfred North Whitehead have published "Principles of Mathematics", which made a revolution in a formal logic.

Since the invention of the first electronic computers in the 1940s, people have predicted the appearance of a computer, the level of intelligence of which will be comparable to a human. This refers to a reasonable technical system, which is endowed with common sense, possessing the ability to learn and think, able to plan and comprehensively process information collected from a variety of sources – real and theoretical. In 1941 Konrad Zuse prompted the first computer software. Many expected that such machines would become a reality in about twenty years (Armstrong S., Sotala K., 2012). Since then, the time has shifted from one year to another, but futurologists are convinced of the likelihood of creating artificial intelligence and continue to believe that"smart cars" will appear in a couple of decades (Armstrong S., Sotala K., 2012).

Warren Sturgis McCulloch and Walter Pitts in 1943 have published «A Logical Calculus of the Ideas Immanent in Nervous Activity», which laid the foundations of neural networks.

So, AI experts often set themselves not the global goal to create thinking machines, but more specific tasks of finding automatic solutions to some intellectually difficult tasks or modeling aspects of human or animal thinking. Nevertheless, "Artificial Intelligence" has its name from this scientific direction

One of the first works in this direction was the fundamental article of Turing A. (1950) "Computers and Intelligence", published in 1950 in the journal "Mind". And the main topic of this article was the question: "Can a machine think?". Turing's article was most remembered for the reasonableness test proposed in it, called by the author a "game of imitation", but later received the author's name. The Turing test assumes that some person must, while communicating with the interlocutor by the text messages and not seeing him, determine whether it is a computer or a person. That is, a computer should be distinguished from a person only based on how it conducts a conversation and answers certain questions.

The Turing test compares the capabilities of an intelligent machine with those of a human being – the best and only standard of intelligent behavior. In the test, the car and human rival (investigator) are placed in different rooms, separated from the room in which the "simulator" is located. The investigator should not see them or speak directly with them – he communicates with them exclusively using a text device, for example, a computer terminal. An investigator should distinguish a computer from a person solely based on answers to questions through this device. If the investigator cannot distinguish the car from the person, then, according to Turing, the car can be considered reasonable.

By isolating the investigator from the machine and another person, the test eliminates prejudice – the type of machine or its electronic voice will not affect the decision of the investigator. The investigator is free to ask any questions, no matter how devious or indirect, trying to uncover the "identity" of the computer. For example, an investigator may ask both subjects to perform a rather complicated arithmetic calculation, assuming that the computer is more likely to give the correct answer than the person. To deceive this strategy, the computer must know when it should issue an erroneous number to appear human. In order to detect human behavior based on an emotional nature, an investigator may ask both subjects to comment on a poem or picture. In this case, the computer must know about the emotional stock of human beings.

This test has the following important features.

1. Gives an objective concept of intelligence, reactions of a deliberately rational being to a certain set of questions. Thus, a standard is introduced for the definition of intelligence, which prevents the inevitable debate about the "truth" of its nature.
2. It_prevents from being confused: should the computer use any specific internal processes, or should the machine really be aware of its actions.
3. Eliminates bias favoring living things, forcing the interviewer to focus on the content of answers to questions.

Due to these advantages, the Turing test provides a good basis for many circuits that are used in practice to test modern intelligent programs.

Among scientists, a hypothesis that thinking is provided by some intangible substance, was still taken quite seriously. When computers have just appeared, there were extremely bulky and slow, they were used for calculations. In this regard, there were a lot of questions on whether the machine can think and this caused a lot of controversies. In this situation, Turing offered the test not as a specific practical recipe for checking the intelligence of a computer, but for moving from speculative and fruitless discussions about the possibility of machine intelligence to experimental research. After all, it is not so important whether to call a certain machine reasonable or not, it must solve the tasks assigned to it no worse than a person.

The debate on "can a machine think" does not stop until now. One of the positions in this dispute was expressed by Allen Newell and John Simon in 1976, in the hypothesis of a physical symbol system. According to it, to achieve intellectual behavior by a system, it is necessary and sufficient that the physical system performs the conversion of symbolic information. In fact, this hypothesis does not say that artificial intelligence can and should be implemented on the physical embodiment of a universal Turing machine, for example, on a conventional computer.

This hypothesis is heavily criticized. One of the most well-known arguments against it, based on a thought Turing A. (1950), is an experiment called the "Chinese Room". In this experiment, a person who does not know the Chinese language is placed in a room full of baskets with Chinese characters. A person is also provided with a book that describes the formal rules for using these symbols without explaining their meaning. Outside the room, there are people (who understand Chinese) who transmit sets of signs to a person in the room, who performs the actions described in the book, after which he gives out an answer, and this answer corresponds to that which would be given by a person who understands Chinese. The actions of a person in a room are similar to the actions of a computer for understanding symbols. Searle (the author of this experiment) says: the person in the room, working as a symbolic physical system, will pass the Turing test, but he does not understand the meaning of the questions and answers to them. We will not discuss the arguments of two irreconcilable parties in this still ongoing debate, we only note that the lack of a clear answer to the question about the possibility of thinking machines did not prevent AI specialists from creating many useful applications and providing invaluable assistance to psychology, linguistics and other disciplines.

Within the framework of this course, the answer to this question will also not be of particular importance. Here, the main attention will be paid to the methods and technologies developed in the field of AI, which have proved their value in practice when solving problems considering the connives studies. These methods also reveal well-defined aspects of mental activity (which is the reason for their success).

The two objections to the Turing test, first formulated by Ada Lovelace (1815-1852), and second by James Essinger (2014), state that computers can only do what they are told to, and cannot perform reasonable actions. However, expert systems, especially in the field of diagnostics, can formulate conclusions that were not laid down earlier by the developers. Many researchers believe that creativity can be realized programmatically.

Another objection, the "argument of natural behavior", is connected with the impossibility of creating a set of rules that would tell the individual what exactly needs to be done at every possible set of circumstances. Indeed, the flexibility that allows the biological mind to respond to an almost infinite number of different situations in an acceptable, way is a hallmark of intelligent behavior.

Indeed, the control logic used in most traditional computer programs does not show great flexibility or the power of imagination, but it is not true that all programs should be written in this way.

In summer of 1956, ten scientists gathered at Dartmouth College for a two-month seminar, united by a common interest in neural networks, the Automata theory , and the study of intelligence. The Dartmouth seminar is usually considered the starting point of a new field of science – the study of artificial intelligence. Most of its participants will later be recognized as the founders of this area: Allen Newell, Herbert A. Simon, John McCarthy, Marvin Minsky, Nathaniel Rochester and Claude Shannon.

The title "Artificial Intelligence" spread in the 1960s (although it appeared a little earlier). The first industrial robot appeared in 1961, in 1969 the 1st International Joint Conference on Artificial Intelligence was held, which officially consolidated this title. Thus, it is clear that the scientists who determined the face of the new science saw the creation of machine mind as its ultimate goal.

Soon after the recognition of Artificial Intelligence as an independent branch of science, it was divided into two main strategic areas: neurocybernetics and black box cybernetics. The approach to creating an AI system used by neurocybernetics is sometimes called low-level or upstream, and the approach used by black box cybernetics is called high-level or outgoing. The names of these strategic directions correspond to their content, which will become clear in the future.

The basic idea of neurocybernetics, that any thinking device must necessarily be made in the image and likeness of the human brain, reproduces its structure and principle of action. Thus, neurocybernetics is engaged in modeling the structure of the brain and its activity.

The human brain consists of numerous interconnected nerve cells – neurons, so the efforts of neurocybernetics are focused on the development of elements similar to neutrons, and on combining these elements into systems – neural networks and neurocomputers. Based on the development of Warren S. McCulloch and Walter Pitts (1943) of the 40s. The first neural networks and neurocomputers were created in the late 1950s. These were devices simulating the human eye and its interactions with the brain, which were able to recognize the letters of the Latin alphabet.

Unlike neurocybernetics, cybernetics of the "black box" does not attach importance to the structure and principle of action of the thinking device. The main thing is that it adequately deal with the highest level of brain activity – the level of its intellectual functions. This area of artificial intelligence focused on the search for algorithms for solving intellectual problems using existing computers, regardless of their hardware base.

Having set themselves the task of modeling brain functions, representatives of this scientific field faced a serious problem. It turned out that despite the old history of research, none of the existing sciences (philosophy, psychology, linguistics, etc.) was able to offer a concrete algorithm of human thinking, thus programmers had to create their own models of thinking.

The second wave of AI, which began in the late 1960s, was associated with the advent of logical programming (Prolog, 1971). In the 1970s, semantic networks, frame systems, production systems (rule-based systems), and combinations thereof appeared. The first world chess championships, computers held among themselves.

The AI systems of the first and second waves are called "symbolic AI". In general, they were based on formal logic, which is well applicable for formalized problems such as logical games, but it is difficult to imagine real-world systems in it.

Awareness of the problems posed after the Dortmund Conference related to the study of AI came in the mid-1970. The realization of projects, which will never live up to their expectations, led to a period of regression, during which funding was reduced and skepticism grew, and the idea of artificial intelligence ceased to be fashionable.

The rise of interest in AI returned in the early 1980s and continued until the early 90s, when Japan decided to start creating a fifth-generation computer, namely, developing an architecture for parallel computing systems for super-powerful computers with artificial intelligence functions.

Intelligent methods (neural networks and genetic algorithms, although simple models of neural networks were known from the late 1950s). They were distinguished by a better performance because they relied more on natural intelligence, gave impetus to the development of AI and because neural networks were self-learning intelligent systems and they accumulated experience. In their properties and functional similarity to the biological brain, neural networks differed from rigid logic and vulnerability. Based on this, the concept of the connectionist model emerged (connectionism) – models in networks of mental and behavioral phenomena from interconnected simple elements, (the approach is used not only in the field of artificial intelligence but also in cognitivism (Kuzior and Postrzednik-Lotko (2020); Kuzior and Czajkowski (2021). In modern research, neural networks continue to be an important method in machine learning.

Bostrom N. (2012) looks at artificial intelligence as a search for short paths, as a way to approach the Bayesian ideal at an acceptable distance, sacrificing some optimality or versatility, while maintaining a fairly high level of performance in the field of interest to the researcher.

Bayesian networks are a compressed way of representing probabilistic and conditionally independent relationships specific to a particular area. Besides, they have become a significant tool for understanding the concept of causality (Pearl, 2009).

One of the main theoretical achievements of the 90s was a clear understanding of MacKay David, (2003) that apparently dissimilar methods can be considered as for special cases within the framework of a general mathematical model. There are a lot of types of artificial neural networks that can be considered as classifiers that perform certain statistical calculations (maximum likelihood estimation).

The 1990s were marked by the advent of artificial neutron networks in business, they showed their real effectiveness in solving many problems – forecasting exchange rates and predicting the results of the presidential election.

The third revival of interest in AI and difference from the previous ones in both amplitude and scope, because there are not only the necessary technical means for solving AI problems (computer technology based on ultra-large chip-chipsets, ubiquitous wireless networks, and the Internet) but also far-reaching work in this area. The beginning of the third wave was the famous victory (May 1997) in a match of six games of the American program Deep Blue over world chess champion Garry Kasparov.

The era of computers as the engine of the semiconductor industry (microelectronics) is ending. Now all the hopes of investors, manufacturers, and chip developers of AI and robotics will become such an engine. Huge segments have already been formed: industrial, service, and military robotics, unmanned vehicles, medical robotics, etc. AI not only helps substantiate decisions but can also make them independently: according to some forecasts, the function of managing hedge will be transferred to AI in the future Tkachenko, Volodymyr, Kuzior Aleksandra, Kwilinski Aleksy (2019). One investment company has already had an artificial intelligence program on its board of directors.

There are a lot of areas, where the level of artificial intelligence surpasses the level of human. Including in logical game programs, where AI is able not only to conduct logical games, but also win victories over people.

QUESTIONS:

1. What is Artificial Intelligence?
2. What are the main goals of researchers in the field of AI? What are the approaches standing out for these goals?
3. What is the Turing test? What important features of the Turing test need to be considered?
4. Comment on the results obtained in the Turing Chinese room
5. What sections does the AI field consist of? How in general terms can you imagine the structure of this area?
6. What is the essence of Bayesian networks?
7. What are the goals set by AI experts?

1.2. THE STRUCTURE OF ARTIFICIAL INTELLIGENCE

The field of artificial intelligence is extremely heterogeneous. There are various lines of research, which are distinguished either by a task (or subject area) requiring intellectual analysis, or by the tools used, or by the developed model of thinking.

The areas identified based on the problem being solved include:

- machine translation;
- automatic abstracting and information retrieval;
- speech communication systems;
- game intelligence, theorem-proof and research automation;
- computer vision;
- data extraction;
- composition of texts and music, etc.

The listed areas are characterized by the fact that a significant part of the research conducted is devoted not to the processes of thinking, but the subject of intellectual analysis. For example, to compose texts, studying the structure of literary works is almost the same important almost more than studying the writer's mental activity.

Directions in AI, distinguished by the tools developed in them, include:

- artificial neural networks;
- evolutionary computing;
- pattern recognition;
- expert systems;
- heuristic programming;
- multi-agent approach, etc.

This may also include several other areas, the study of more particular groups of methods. The difference between these areas is that they are developing an apparatus for solving a large class of problems. For example, pattern recognition methods can be used to categorize texts when searching for information, computer vision, and in many other directions of the first type. At the same time, different methods can be used to solve one problem. For example, neural networks are able to solve those problems that are solved by pattern recognition methods, and evolutionary computations can replace heuristic programming methods and vice versa. This group of directions is more heterogeneous than the first. It has directions (for example, ANN) a separate approach to artificial intelligence in general. However, during their existence, these approaches have not brought us closer to understanding the process of thinking, so they should be considered precisely as a set of methods combined by a certain general idea or mathematical apparatus.

Unlike the first group of research areas, these areas appeared at different points in time and the moment of their origin is often associated with the release of a particular work.

The directions of the third type include:

- search within solutions;
- presentation of knowledge;
- machine learning.

Each of these areas focuses on one aspect of intelligence. Their origins lie in the field of philosophy. In this regard, the moment of occurrence of each of these directions is difficult to name, but we can distinguish the stage when each of them dominated. Moreover, the change of these stages determines the logic of the development of the field of AI as a whole.

The first researchers in the field of AI were initially forced to borrow ideas from other disciplines of natural intelligence. In psychology in the first half of the 20th century, in the study of animal behavior, an extremely large role of the search was discovered, which begins to be carried out responding to a situation for which there is no ready-made solution. The "labyrinth" hypothesis was developed by the first AI experts, whose subject matter was intellectual games and proof of theorems, where the concept of search also played a key role. Moreover, the essence of intelligence consists in solving problems, and the solution process can be represented as a search for a path from source data to an answer in the space of possible solutions (or as a search for a path from available means to an ultimate goal through achievable sub-goals). However, over time, the limitations of such systems were discovered: for them, a formalized description of the task had to be compiled by man. The problem arose of the formation of the "maze" itself, for the solution of which it was necessary that the machine system could use knowledge about the subject area. Moreover, it was not possible to find a universal algorithm that effectively solves any search problem in an arbitrary "maze".

The problem of knowledge representation has become dominant since the mid-1970s, it was also facilitated by the rapidly developing branch of computer linguistics of that time. Knowledge-based systems are widely used in the form of expert systems, with the help of which almost all areas of AI were identified simultaneously. Using expert knowledge, expert systems were able to construct formal descriptions of problems formulated in a limited natural language for one narrow subject area. The search for a solution to the problem in knowledge-based systems has become the problem of manipulating knowledge, which is now the essence of thinking.

It is important to note that the problem of manipulating the knowledge is narrower and more specific than the problem of search in general. However, while this area was developing, another problem was identified, the problem of automatic acquisition of knowledge. More broadly, this problem has been formulated as a machine learning problem. Although earlier the importance of learning was understood by many scientists, only after research become clear how much information a person receives in the learning process and how difficult it manually to put this knowledge into machine systems. Standing out in an independent direction, machine learning in the 1980s began to attract more attention and, as a result, became central to the field of AI.

Intelligence is no longer understood as a finished product that can be reproduced or as a fixed ability to solve problems or manipulate knowledge. A paradigm shift has led to the formulation of new problems in previously explored areas. In particular, in the search problems, the problem of automatic construction of search heuristics (search optimization) was formulated, which, however, still remains little studied. Machine learning methods in the tasks of acquiring knowledge are applied to vague and contradictory observational data, and therefore the knowledge generated as a result does not have complete reliability. As a result, the problem of representing fuzzy knowledge arises, and the problem of manipulating knowledge turns into a problem of reasoning under conditions of uncertainty.

Another group of problems is related to the application of teaching methods to the training problem, i.e., with meta-training. In particular, if in the tasks of acquiring knowledge it is implied that the representations of knowledge are predetermined a priori, and you only need to build a knowledge system within the framework of these representations, then in the problems of meta-learning the question arises of automatically constructing the representations themselves, the details of which can vary greatly depending on the subject area. Similar problems are very complex and little studied. However, their solution is necessary to remove the following limitation for machine systems – their ability to function only in a narrow subject area.

Thus, research in the field of AI began with the paradigm "thinking as a search" and with the development of methods for solving formally set tasks. A further paradigm shift was associated with an increase in the universality of machine systems, due to a decrease in the amount of information prepared for them by humans. At the first stage of the development of AI, the description of each task was formed by a person. At the second stage, a person set a description of a certain (rather narrow) subject area, including a whole complex of tasks. At the third stage, the machine system got the opportunity, at least in part, to build a description of the subject area independently within the framework of a given representation.

The subsequent development of the field of AI is associated with the further universalization of machine systems and their access to information. The latter may be related to the direction in which the embodied systems are studied, i.e., systems placed in a specific informational, physical, social environment. Thus, according to the position of

Rodney Brooks (1991, 2013), intelligence cannot arise in non-embodied systems, such as traditional systems of proof of theorems or expert systems (the scientist also proposes a specific multilayer architecture consisting of interacting simpler formations and called categorical architecture for controlling robots). This direction is not fundamentally new, as much is borrowed from robotics.

However, in embodied systems, it is understood that the incoming sensory information should serve as the basis for training, as a result of which a knowledge system should be formed to use them for the subsequent solution of the tasks. The problem of learning "from scratch" based on sensory information poses many additional problems, the solution of which remains to be sought in the future. Most likely, the current state of AI can be described as a synthesis stage, at which the methods obtained earlier in the framework of isolated research directions are combined.

Thus, the structure of the AI area can divide into three levels, the structure of the basic level of the field of artificial intelligence in that part that is currently established.

Modern AI programs usually consist of a set of modular components of rules and behavior that are not executed in a strictly specified order but are activated as needed, depending on the structure of a specific task. Coincidence detection systems are allowed applying general rules to a wide range of tasks. These systems are unusually flexible, which allows relatively small programs to exhibit diverse behaviors over a wide range, responding to various tasks and situations.

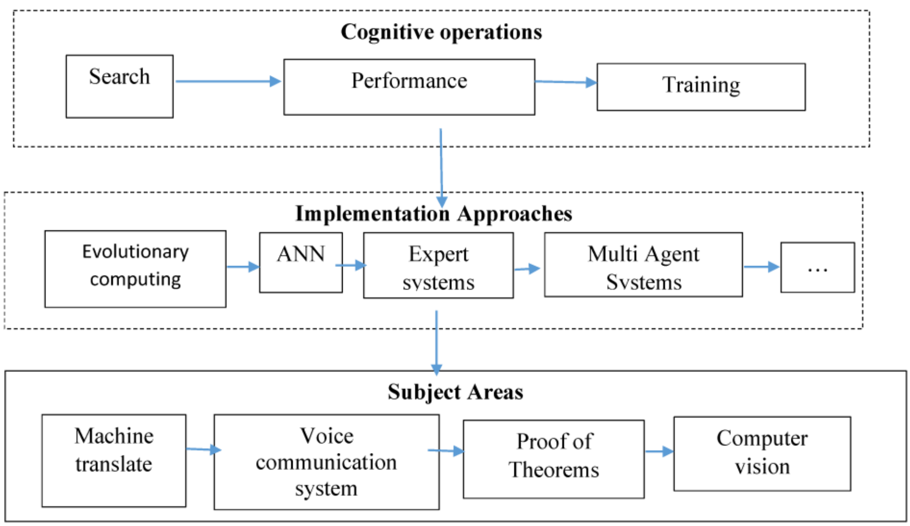

Figure 1.1. General structure of the field of artificial intelligence

Whether or not it is possible to do flexibility of such programs to the level of living organisms, is still a subject of heated debate. Nobel laureate Herbert Simon said that for the most part, the originality, and variability of behavior inherent in living beings arose because of the complexity of their environment more than because of the complexity of their internal "programs" (Simon, 1981).

AI systems are divided into two large groups:

Applied Artificial Intelligence (they also use the term – weak/applied/narrow AI) is an AI designed to solve any intellectual problem or a small number of them. This class includes systems for playing chess, pattern recognition, speech, decision-making, etc.

Universal artificial intelligence (Strong AI/Artificial General Intelligence (AGI) is a hypothetical (for now) AI that can solve any intellectual problem.

In the process of building narrow AI systems, **four main engineering approaches** were formed: logical, structural, evolutionary and imitation.

Logical approach

Most Artificial Intelligence systems are based on a logic of a certain machine for proving theorems. Each machine has a target generation unit, and the output system proves this goal as a theorem. This system is better known as the "computer expert system".

A good example of a diagnostic system is DELTA, commissioned by General Electric (GE) to detect engine failures. The DELTA Knowledge Base contains over 500 rules. The system uses a flexible search engine. It was first developed in LISP, then reprogrammed in FORT.

The IBM Watson system (USA), which was originally developed for diagnosis, has found application not only in medical practice. In April 2015 IBM has announced the launch of the Watson Health Cloud (Watson Health Cloud, a secure and open cloud platform for doctors, researchers, insurance agents, and various companies from the world that specialize in beauty and health solutions).

All expert systems operate in the mode of acquiring knowledge, where the expert communicates with the expert system, and in the consultation mode for ordinary users. A well-built expert system should be able to self-learn the tasks to be solved, automatically replenishing its knowledge base with the results of conclusions and decisions.

Structural approach

The structural approach based on the AI system uses modeling the structure of the human brain. Among the first such attempts, perceptrons should be noted David Rumelhart и Frank Rosenblatt (1958).

Perceptron – a mathematical or computer model of information perception by the brain (cybernetic model of the brain), proposed by Frank Rosenblatt in 1957 and first implemented as a Mark-1 electronic machine in 1960. Perceptron became one of the first models of neural networks, and Mark-1 became the world's first neurocomputer.

McDonald R., Hall K., Mann G. (2010) proposed to apply to large-scale tasks of machine learning in distributed computing, the classic, and most popular option for using neural networks is image processing.

Neural networks are not programmed in the usual sense of the word, they are trained by the method of backpropagation of error (backprop). Learning ability is one of the main advantages of neural networks over traditional algorithms. The more training data that can be fed to the neural network, the more training iterations can be created. Today, deep learning is at the heart of the services of many IT companies. Facebook uses neural networks for automatic tagging algorithms. Google – to search among the user's photos. Amazon for generating product recommendations. Pinterest for personalizing the user's homepage and Instagram for search infrastructure.

The development of deep learning technology was the distributed depth learning (DDL) technology implemented by IBM in the summer of 2017, which can reduce the training time of an Artificial Neural Network by an order of magnitude. The IBM Distributed Deep Learning library (DDL) IBM Watson Studio (April 28, 2020) hooks into popular open-source machine learning frameworks, such as TensorFlow, Caffe, Torch and Chainer and enables these frameworks to scale to multiple GPUs.

From traditional processor systems, neuroprocessors are distinguished by a special architecture. They have a more "homogeneous" structure, consisting of many neurons – the same and relatively simple computational cells with built-in memory. Thus, the architecture of the neuroprocessor, by definition, turns out to be multi-core because each neuron is an independent core.

Combinatorial model of artificial intelligence Kuzior Aleksandra, Kwilinski Aleksy, Tkachenko Volodymyr (2019).

The evolutionary approach

A classic example of an evolutionary algorithm is the genetic algorithm. Evolutionary theory claims that every biological species develops and changes in order to best adapt to the environment. Evolution is a process of optimization of all living organisms.

The founding father of the genetic algorithm came up with the idea of using genetics for own purposes in 1975 by J.H. Holland (1992). The idea of the algorithm is taken from nature: through enumeration and, most importantly, selection, the correct "combination" is obtained.

Genetic algorithms are computer simulations of evolution. The material embodiment of systems constructed in this way has so far been impossible without human intervention.

However, intensive work is underway, the result of which is to reduce the dependence of machine evolution on humans. These works are carried out in two main directions:

1. Natural selection modeled by GA is transferred from the virtual world to the real, for example, experiments on real battles of survival robots.
2. Intelligent systems based on GA design robots, which in principle can be manufactured in automated plants without human intervention.

Genetic algorithms are widely used in order to quickly solve complex optimization problems in business and finance. Numerous variants of genetic algorithms are used in the study of various scientific and technical problems: the creation of jet engines, improving the efficiency of aircraft maintenance by aircraft carriers, etc.

In 1994, Andrew Keene from the University of Southampton used the genetic algorithm in spacecraft design. Based on the model of the support of the space station, designed at NASA. After the change of 15 generations, including 4,500 design options, we got a model that excels in tests the option that NASA engineers developed. John Cosa from Stanford has developed a more sophisticated technique – "genetic programming". The result of "mutations" are whole programs – virtual analogues of real devices. The company Genetic Programming, which hired John Coz, used a makeshift supercomputer consisting of thousands of 350 MHz Pentiums for computing operations using genetic algorithms.

An example of the construction of robots by robots: Golem was created at Brandeis University, which designed robots. The program had a database of details, as well as a mutation mechanism and a suitability function for "sifting out" losers – those who did not learn how to move. After 600 generations, in several days the program received models of three crawling robots. Significantly, the robots turned out to be symmetrical, although the symmetry was not explicitly spelled out in the rules of evolution and initial data. This means that it appeared during the modeling of machine evolution as a useful feature that allows you to move in a straight line.

Simulation approach

The purpose of simulation modeling is to reproduce the behavior of the system under study based on the results of the analysis of the most significant relationships between elements or to develop simulation modeling of the studied subject area for various experiments.

One very interesting idea is associated with this approach. Imagine that we are being watched by some device that monitors what we are doing in what situations, we are talking about. Observation is for the quantities that come to us at the entrance (vision, hearing, taste, tactile, vestibular, etc.), and for the quantities that go out of us (speech, movement, etc.). Further, this device tries to rebuild some models in such a way that, with certain signals at the input, it produces the same data as the person at the output. If this venture is ever realized, then for all outsiders such a model will be the same person as a real person. And after his death, she will express those thoughts by person, who was modeled, would presumably express.

The human brain is not limited to a "logical" component, it is also responsible for issues related to the concepts of personality and philosophical positioning. That’s why they single out Strong AI (General AI), an artificial intelligence that recognizes itself.

Contrasting today’s General AI to Narrow AI, is not the right idea. Narrow AI — are systems designed to perform and automate some kind of cognitive function. And they know how to do it on a human level or even better. General AI – this is an AI that will be able to draw independent conclusions based on the information that comes into it, self-study and at some point in time to become aware of itself. By analogy to a person, such a system can have a multimodal input: image, sound, and so on, and process them together.

AGI – artificial intelligence is capable of performing complex tasks. Its universal applicability can greatly change our reality. Approximate directions of these changes are visible now, with a "simple" AI: from voice assistants and companion programs to self-driving cars and pictures or stories generated by neural networks.

However, the development of General Artificial Intelligence is complicated by both objective factors (the current stage of technological development) and semi-subjective factors (orientation to the applied use of AI). At the same time, the beginning of general-level AI can come sooner than it seems – futurist and inventor Raymond Kurzweil (2012) predicts that this will happen already in 2029.

QUESTIONS:

1. What are the areas of research, which stand out?
2. What are the directions of AI that are distinguished by the tools developed in them?
3. Expand the essence of the "Labyrinth" hypothesis of AI.
4. What are "machine learning methods"?
5. What are the three basic problems (and several derivatives from them) that led to the development of the field of AI?
6. What is the general structure of the field of artificial intelligence?
7. What groups are the AI systems divided into?
8. Uncover the essence of the logical approach in artificial intelligence systems.
9. Uncover the essence of the structural approach in artificial intelligence systems.
10. Uncover the essence of the evolutionary approach in artificial intelligence systems.
11. Expand the essence of the simulation approach in artificial intelligence systems.

1.3. MAIN STRATEGIES, PROBLEMS AND PROSPECTS FOR THE DEVELOPMENT OF AI

Achieving middle-level success in Artificial Intelligence will affect the daily lives of all segments of the world. Until now, only computerized communication networks, such as the mobile telephone network and the Internet, have had such a comprehensive impact on society.

Despite all the advantages and successes, development of Artificial Intelligence and its existence, there are certain problems with possible negative consequences. Modern theoretical problems can be reduced to the following groups:

1. The problem of presenting knowledge.

Neural network problems:

- multicriteria decision-making;
- stochastic decision-making models;
- development of new models of representation for highly specialized Neural network;
- subject areas.

Problems of biomachines, i.e., machines that have their part living beings or structurally imitate man:

- multicriteria decision-making;
- decision-making based on statistical models;
- coordination of several robots;
- problems of improving neural networks.

2. Development of computational linguistics:

- development of new, more reliable programming languages;
- development of robot management language based on natural language.

3. The problem of improving computer logic:

- development of new architectures (parallel machines, research in the field of integrated memory, decentralized machines, modeling of high-speed electrical connections);
- humanoid robots (flexible and portable members of robots; recognition of robots by persons authorized to control robots; development of robot mechanisms; numerical methods for optimizing calculations);
- methods of access to information (multimedia systems; heuristic analysis of texts; automatic extraction of knowledge from the text);
- creation of "intellectual spaces" (intelligent learning environments and shells);
- machine learning (machine reading and comprehension of texts;
- recovery of lost data elements; data cleaning from noise);
- medical vision (automatic analysis of anatomical structures; reading images; machine geometry and spatial scenes);
- mobile works.

4. The problem of improving computational linguistics;

- development of work management language based on natural language;
- creation of models of natural language;
- language comprehension;
- development of programming languages that increase the reliability of the developed software.

Researchers often consider whether Artificial intelligence can be developed. But it is also necessary to analyze the question whether it should still be developed. Stuart J. Russell (2003) believes that if the consequences of the creation of Artificial Intelligence technology are more likely to be negative than positive, people working in this field are morally compliant, which obliges them to direct their research to other areas.

Artificial Intelligence can be a source of some unprecedented problems, namely:

1. As a result of automation, the number of unemployed may increase.
   Until now, automation using artificial intelligence technology has created more jobs than eliminated them, and has led to more interesting and high-paying specialties. After the canonical program of Artificial Intelligence has become an "Intellectual Agent" designed to help people, job loss becomes even less likely.
2. The amount of free time available to people may decrease (or increase). In the information economy, which is characterized by the presence of broadband and the simplification of the replication of intellectual property, the greatest reward comes from the ability to be a little more successful than a competitor, which forces people to work harder.
3. People may lose their sense of uniqueness. With the development of Artificial Intelligence, people will be automata, and this idea leads to a loss of independence or even humanity.
4. People may lose of their rights to privacy. The development of speech recognition technology and the existing threat of terrorism can lead to the widespread use of wiretapping and therefore to the loss of civil liberties.

5. The use of artificial intelligence systems can make people more irresponsible. For example, if expert systems will ever reliably produce more accurate proposals than a person.
6. The success of Artificial Intelligence can be the beginning of the end of the human race. Almost any technology, falling into the wrong hands, reveals the potential for harm. For these reasons, research in the field of Artificial Intelligence cannot be considered in isolation from their ethical implications

The father of Artificial Intelligence, John McCarthy (1996, 2007) said, "If the system starts working normally, they immediately stop calling it Artificial Intelligence".

The development of Artificial Intelligence may entail one of the main existential risks – in our days this state of affairs is perceived as trivial; therefore, the prospects for such growth should be evaluated with extreme seriousness, even if it would have been known (but not so) that the probability of a threat is relatively low. However, pioneers in the field of Artificial Intelligence, despite all the belief in the inevitable appearance of AI, which not inferior to humans, denied the possibility of the appearance of a supermind superior to the human mind. It seems that their imagination – in attempts to comprehend the ultimate possibility of future machines, comparable in their mental abilities to humans – simply dried up, and they easily passed the inevitable conclusion: the next step would be the birth of superintelligent machines.

For example, quantum computing offers an incredible breakthrough in the modeling and optimization of complex systems, which can greatly increase efficiency in various fields – from logistics to the development of new drugs (Schwab K., Davis N., 2018).

Mixed Reality, Artificial Intelligence and Quantum Computing Satya Nadellaq (2017) today are also called separate directions, but they are already merging.

The term "Artificial intelligence" is used to denote a large area of scientific and applied research. It is attached to the trend that most people are more likely associated with intelligent robots or thinking computers, numerous images of which were created in science fiction works. Are Artificial Intelligence experts really setting themselves such ambitious goals? Many of them deny it.

Researchers identify two main goals of the work: (1) the automation of human activity, especially those of its types that have traditionally been considered intellectual, and (2) the creation of computer models that mimic the processes by which people solve certain intellectual problems in order to explain the essence of these processes. Another possible goal, which, however, is often forgotten, is the creation of an intelligence amplifier (AI). The methodology of the AI direction is not much, but still different from the methodology of the AI direction. But what differs more significantly is the predicted social consequences.

It is worth noting that the first two goals correspond to two different AI approaches, which are usually called technical (or heuristic) and bionic. In the framework of the technical approach, the psychophysiological reliability of models of thought processes is sacrificed to the efficiency with which these models solve the tasks assigned to them, and the intelligence of computer programs is determined by how good they get compared to a person. In the bionic approach, on the contrary, the similarity of the processes of solving a certain problem by a computer program and a person is considered necessary at the expense of the quality of the final result. Often these two approaches are contrasted, in which they often go to extremes. So, ardent adherents of the technical approach give such well-beaten analogies as a bird and an airplane or a wheel and legs.

Technically, systems perform the same functions as biological systems, but they do it in a completely different way, for example, the study of the structure of bird feathers hardly helped to create an airplane. Proponents of the bionic approach argue that it makes no sense to try to create AI without knowing the natural, which is the only example of intelligence. And why start from scratching something that can be borrowed from nature? In reality, these two approaches simply pursue two slightly different goals and do not contradict each other.

There is another approach to AI, called evolutionary, which propose to imitate not the thought processes of the already formed intellect of an adult, but the process of formation of this intelligence in ontogenesis and phylogeny from very early stages of evolution.

As was mentioned Artificial Intelligence is a scientific industry engaged in the study and modeling of human natural intelligence. However, the natural human intellect is a very complex object of study, and its modeling is carried out at different levels of abstraction. We can distinguish three levels, which correspond to three main strategic directions (in the future we will call them strategies) of AI:

- technology expert systems (high-level strategy);
- technologies of evolutionary modeling;
- neuroinformation technologies (low-level strategy).

The highest level of abstraction corresponds to the technology of expert systems based on explicit knowledge of the subject area. According to this strategy, knowledge can be obtained from experts or other information sources. It is a human, who is an expert of knowledge of the subject area, and who is able to act at the highest level.

His knowledge is formalized and presented in an easy-to-use form and is downloaded into a software package - an expert system, which then makes logical conclusions based on this knowledge, modeling the intelligence of a human expert. It was this strategy that was first applied in the 13th century by Raimund Lullius.

At the lowest level of abstraction, there are neural network technologies. According to this strategy, a model is built taking into account the structure of the brain, which consists of many neurons connected by nerve fibers. Knowledge enters such a model during its training on specially selected examples characterizing a specific subject area. Like the brain, knowledge is stored here in an implicit form-encoded as a set of synoptic link forces that mimic the electrical conductivity of interneuronal connections.

For half a century, competition has been going on between high- and low-level AI strategies. At the same time, many experts note the recent progress in the integration of these competing strategies. For example, successful attempts to create and use hybrid intelligent systems are known, which combe the ideas of both of these alternative strategies.

The third strategy, which was born in the mid-1970s thanks to the work of a professor at the University of Michigan Holland JH (1992), was inspired by the Darwin theory of the evolution of life on Earth. Evolutionary models begin their work by creating a whole population of individuals – candidates for solving the problem. A separate individual of this population is evaluated according to a certain criterion, which allows selecting one the best, which then interbreed, inherit the positive properties of the parents and mutating, form new generations. This approach involves the search for a source of intelligence in the process of evolution and interaction of individuals. New knowledge in evolutionary models is manifested in the course of competition between individuals.

Summing up the brief image of the existing AI strategies, note that nowadays, the absolute in terms of practical applications are neutron network technologies. Brain modeling at the lowest level of abstraction – at the level of neural structure – leads to the most fruitful technologies for creating intelligent information systems.

National strategies and projects

In October 2016, the United States unveiled the Strategic AI Research Plan (The National Artificial Intelligence Research Strategic Plan). The following areas are identified as priorities in the plan:

- ensuring long-term investments in new generations of AI;
- development of effective methods for the interaction of AI and humans;
- understanding of the ethical, legal and social challenges that AI poses, and the answer to them;
- ensuring the security of AI systems;
- development of publicly accessible databases and a special environment for training and testing AI;
- development of standards and benchmarks for AI;
- determination of the necessary labor resources for developments in the field of AI.

In February 2017, the Chinese Supreme Economic Planning Authority approved a plan to create a national AI laboratory for machine learning research. The project was led by the Chinese search engine Baidu (an analog of Google) in partnership with elite Chinese universities.

The Chinese AI strategy has three phases:

- "AI Competitiveness", 2020. Main focus: Big Data Intelligence, Autonomous Intelligent Systems, Cross-Medium Intelligence, Swarm Intelligence, Hybrid Advanced Intelligence, Fundamental Theories in AI;
- "Breakouts", 2025. AI will be widely used in medicine, urban infrastructure, manufacturing, agriculture, in the construction of national defense systems, in the application of laws and regulations, as well as for assessing security and the possibility of exercising control functions;
- "World Leadership", 2030. Organization of social management mechanisms, including the activities of authorities, building a national defense system, and building global value chains.

In addition to the United States and China, five countries adopted a national Artificial Intelligence strategy by the end of 2017, and another 30 countries followed suit in 2018.

In Russia, the thematic leaders in investment by the state are projects on the militarization of AI and robots. Various International Groups explore the practical and ethical boundaries of such opportunities.

According to analysts at the McKinsey Global Institute (2017), the AI market can reach $ 126 billion by 2025, up to $ 30 billion spent annually by major players. Many investors agree that AI will be the next technological revolution to change everyday life and production.

Recognizing that AI will have a dramatic transformative impact on society, the planet, and the economy, leaders in this area, including Microsoft, Amazon, Facebook, IBM, Google, and DeepMind (DeepMind Ethics & Society homepage. 2017), have created an "AI partnership for the benefit of people and society". The purpose of this partnership is to study and formulate recommendations on artificial intelligence technologies to improve public understanding of AI and create an open platform for discussion and decision-making regarding AI and its impact on people and society.

QUESTIONS:

1. What are the problems of Neural Networks?
2. What are the problems of biomachines, can people be a part of them?
3. What are the problems of side-by-side computer’s logic?
4. What are the problems and companion of computer linguistics?
5. What kind of source of earlier problems can AI become?
6. What are the existing AI strategies?
7. What are the national AI strategies and projects?

1.4. COGNITIVE TECHNOLOGIES AI

Head of the Google corporation, Sundar Pichai, noted that while AI had enormous potential, there were also considerable dangers, such as the misuse of deep fakes, which are computer-generated clips that are designed to look real (BBC News 20 January 2020).

At the same time, the Head of the Chinese company Alibaba, Jack Ma, warned that in case of the absence of proper control, AI could cause the Third World War (BBC News 29 August 2019).

At the VivaTech 2018 event in Paris, IBM CEO Ginni Rometty talked about her vision of the need for ethics and transparency in AI and data management. Rometty invited business leaders to follow IBM’s Principles for Trust and Transparency:

- the purpose of AI is to augment human intelligence;
- data and insights belong to their creators;
- AI systems must be transparent and explainable (IBM July 16, 2018).

Only general artificial intelligence can match human intelligence, and by most estimates, general AI is at least decades away. Powered by deep learning and machine learning, current AI technologies and applications have produced impressive results, and they can often outperform humans in specific tasks such as classifying images or transcribing speech. But the limits and challenges of deep learning prevent it from making trivial abstract decisions that even the simplest human mind can do.

That is why we call it narrow AI, or better yet, "These are technologies to augment human intelligence. Those were the words that Ginni Rometty, the chief executive of IBM, said during an interview with CNN’s Fareed Zakaria on the sidelines of the 2017 World Economic Conference in Davos" (By Ben Dickson – May 8, 2018).

In 2018 World Economic Conference in Davos Ginny Rometti in speech instead of the term "Artificial Intelligence" used the concept of "cognitive", or "augmented intelligence" (augmented intelligence).

Augmented intelligence is an alternative conceptualization of artificial intelligence that focuses on AI's assistive role, emphasizing the fact that cognitive technology is designed to enhance human intelligence rather than replace it.

What is Cognitive Intelligence? How is it different than Artificial Intelligence? Cognitive Intelligence, as well as being a part of Artificial Intelligence, is an area that mainly covers the technology and tools that allow our apps, websites, and bots to see, hear, speak and understand the needs of the user through natural language. They are AI apps that allow machines to learn their users’ language so that the users don’t have to learn the language of machines. AI, for its part, is a much wider concept that includes technology and innovations such as robotics, Machine Learning, Deep Learning, neural networks, NLP, etc. (luca-d3.com, 2019).

Modern AI is gaining more and more cognitive abilities, which we are accustomed to ascribing only to humans, such as general learning ability and high-level intellectual activity. AI was created in response to a request to know the nature of human intelligence and its thought processes. Initially, it was supposed to be done by modeling cognitive processes. Mankind has entered the field of applied research and is solving very specific problems without considering the nature of consciousness, so there are essentially no practical developments in the field of AGI construction.

Artificial general intelligence (AGI) is the intelligence of a machine that could successfully perform any intellectual task that a human being can.

AGI describes research that aims to create machines capable of general intelligent action. The term was used as early as 1997, by Mark Gubrud in a discussion of the implications of fully automated military production and operations. The term was re-introduced and popularized by Shane Legg and Ben Goertzel around 2002 (Source: Wikipedia).

Chinese researchers Feng Liu, Yong Shi and Ying Liu conducted intelligence tests in the summer of 2017 with publicly available and freely accessible weak AI such as Google AI or Apple's Siri and others. At the maximum, these AI reached a value of about 47, which corresponds approximately to a six-year-old child in first grade. An adult comes to about 100 on average. In 2014, similar tests were carried out in which the AI reached a maximum value of 27 (Liu, Feng; Shi, Yong; Liu, Ying, 2017).

A 2017 survey of AGI categorized forty-five known "active R&D projects" that explicitly or implicitly (through published research) research AGI, with the largest three being DeepMind, the Human Brain Project, and OpenAI (Baum, Seth, 2017).

In 2019, video game programmer and aerospace engineer John Carmack announced plans to research AGI (Lawler R., 2019).

Today's AGI is speculated, some argue that intelligence is too complex to be fully reproduced in the near future (Grace K. etc., 2018).

As of March 2020, AGI remains speculative as no such system has been demonstrated yet (Ramamoorthy A., Yampolskiy R., 2018).

Progress in the development of materials and sensor technologies have improved the perception, movement and cognitive abilities of machines. The better artificial intelligence makes decisions, the better the robots controlled by these decisions work with a person, and vice versa.

There have been many AI researchers that debate over the idea of whether machines should be created with emotions. There are no emotions in typical models of AI and some researchers say programming emotions into machines allows them to have a mind of their own (Clocksin W., 2003). Emotion sums up the experiences of humans because it allows them to remember those experiences. David Gelernter (2010) writes, "No computer will be creative unless it can simulate all the nuances of human emotion". This concern about emotion has posed problems for AI researchers, and it connects to the concept of strong AI as its research progresses into the future (Kaplan A., Haenlein, M., 2019).

If we go further into the future, the computer must not only mimic human emotions but also consider the reflex flow in the decision-making process.

Cognitive services, for the most part, aim to imitate rational human processes. These services analyze large amounts of data that is generated by connected systems. They offer tools with diagnostic, predictive, and predictive capabilities that are able to observe, learn and offer Insights, suggestion, and even automatic actions. They are strongly orientated to contextual and human interaction. As such, for experts the challenge of artificial intelligence is to adapt the technology to that people can interact with it in a natural and daily way. They aim to create an application capable of human learning, such as:

- Listening and speaking, or rather the ability to turn audio to text and text to audio.
- Natural Language Processing. Text is not just a combination of keywords, a computer needs to understand grammatical and contextual connections too.
- Understanding emotions and feelings (“sentiment analysis”). To create empathetic systems capable of understanding the emotional state of a person and to make decisions based on this.
- Image recognition. This consists of finding and identifying objects in an image or video sequence. It is a simple task for humans, but a real challenge for machines (luca-d3.com, 2019).

Cognition encompasses many aspects of intellectual functions and processes such as attention, the formation of knowledge, memory and working memory, judgment and evaluation, reasoning and "computation", problem-solving and decision-making, comprehension and production of language. Cognitive processes use existing knowledge and generate new knowledge.

A Cognitive computer or system learns at scale, reasons with purpose and interacts with humans naturally. Rather than being explicitly programmed, these systems learn and reason from their interactions with human beings and their experiences with their environment (Edureka, 2019).

Cognitive systems often make use of a variety of machine-learning techniques, but not exactly a machine-learning method instead it is often a complete architecture of multiple A.I. subsystems that work together.

This is a subset of AI that deals with cognitive behaviors we associate with 'thinking' as opposed to perception and motor control.

With time, cognitive systems learn to refine the way they identify patterns and the way they process data to become capable of anticipating new problems and model possible solutions.

Cognitive systems must be flexible enough to understand the changes in the information. Furthermore, the systems must be able to digest dynamic data in real-time and make adjustments as the data and environment change.

Finally, a cognitive artificial intelligence system has the capability to understand, reason and learn with similar processes that we as humans naturally develop. It is also the product of many other AI-based components which feed the cognitive architecture (Quora.com, 2018).

Cognitive Computing is another subpart of Artificial Intelligence. But still, it is more than AI itself.

Essentially, cognitive computing refers to computing that is focused on reasoning and understanding at a higher level, often in a manner that is analogous to human cognition or at least inspired by human cognition. Typically, it deals with symbolic and conceptual information rather than just pure data or sensor streams, intending to make high-level decisions in complex situations.

The technologies behind Cognitive Computing are similar to the technologies behind AI. These include machine learning, deep learning, NLP, neural networks, etc. But they have various differences as well.

Cognitive Computing focuses on mimicking human behavior and reasoning to solve complex problems. AI augments human thinking to solve complex problems (Edureka, 2019).

Microsoft co-founder Paul Allen (2011) believed that such intelligence is unlikely in the 21^{st} century because it would require "unforeseeable and fundamentally unpredictable breakthroughs" and a "scientifically deep understanding of cognition".

While maintaining the current direction of development, the combination of AI and robotics will need to be considered from the perspective of power, responsibility and accountability, which means that comprehensive management will be required.

Most experts tend to agree that we're far from engineering true AI – that is, systems that can independently process, reason and create in the same capacity as the human brain.

While the underlying technologies powering AI and IA are the same, the goals and applications are fundamentally different: AI aims to create systems that run without humans, whereas IA aims to create systems that make humans better. To be clear, this is not a separate category of technology, but simply a different way of thinking about its purpose. Arguably, many AI-branded technologies currently available for businesses can and should be more accurately be described as IA (Aaron Masih, 2019).

Financial institutions integrate IA in fraud detection. Using machine learning, systems can be trained to identify and flag the markers and patterns of fraudulent activity. Employees then use this machine data, and apply their knowledge, judgment and expertise to interpret the data, investigate and make a final call. Aside from saving time, this can also save serious amounts of money – at least $12 billion annually, according to a 2016 Oakhall study.

QUESTIONS:

1. What are the cognitive abilities of machines in the context of AI?
2. How do you see the connection of emotions and the development of AI?
3. What is Cognitive services? Cognitive Computing? Cognitive artificial intelligence system?
4. What is an AGI?

Chapter 2. Expert system

2.1. Features and signs of intelligent information systems

2.2. Appointment of expert systems

2.3. Self-learning systems

2.4. Technology for creating expert systems

2.5. Methods of working with knowledge

2.1. FEATURES AND SIGNS OF INTELLIGENT INFORMATION SYSTEMS

Any information system (IS) performs the following functions: it receives information requests entered by the user and the necessary source data, processes the data entered and stored in the system in accordance with a known algorithm, and generates the required output information. From the point of view of implementing the listed functions, IS can be considered as a structure that produces information, in which the order is an information request, raw materials are the initial data, the product is the required information, and a tool (equipment) is the knowledge by which the data is converted into information.

Knowledge has two components: factual and operational. Actual knowledge – is meaningful and comprehensible data. The data its – is a specially organized character on any medium. Operational knowledge is the general relationship between facts that allows to interpret data or extract information from them. Information is new and useful knowledge for solving any problems.

Actual knowledge is also called extensional (detailed), and operational knowledge is called intensional intensional (generalized). The process of extracting information from data is reduced to an adequate combination of operational and factual knowledge and is performed differently in different types of IS. The easiest way to connect them is within the framework of a single application program.

$$\textit{Program} = \textit{Algorithm (Data Transformation Rules}$$

$$+ \textit{Control Structure}) + \textit{Data Structure}$$

Thus, operational knowledge (algorithm) and factual knowledge (data structure) are inseparable. However, if during the operation of the IS it becomes clear the need to modify one of the two components of the program, then there will be a need to rewrite it. Because only the IS developer has full knowledge of the problem area, and the program serves as an executor. The end-user, due to the procedural and machine-oriented orientation of knowledge representation, understands only the external side of the data processing process and cannot influence it in any way. The consequence of these shortcomings is the poor viability of IS or adaptability to constant changes in information needs. Besides, due to the determinism of the algorithms of the tasks to be solved, the IS is not capable of generating the user's knowledge of actions in incompletely defined situations.

In systems based on database processing (DBMS), the actual and operational knowledge is separated from each other. The first is organized in the form of databases, the second – in the form of programs. Moreover, the program can be automatically generated at the request of the user (for example, the implementation of requests). As an intermediary between the program and the database acts as a software tool for accessing data – database management system.

$$\textit{(DBMS): DBMS} = \textit{Program} \rightarrow \textit{"DBMS"} \rightarrow \textit{Database}$$

The concept of independence of programs from data allows increasing the flexibility of IP to perform arbitrary information requests. However, flexibility, due to the procedural presentation of operational knowledge, has clearly defined boundaries. To formulate an information request, the user must clearly imagine the structure of the database and the algorithm for solving the problem. Therefore, the user should be well versed in the problem area, in the logical structure of the database, and the algorithm of the program. Common shortcomings of traditional information systems, which include the systems of the first two types, are their poor adaptability to changes in the subject area and the information needs of users, the inability to solve poorly formalized tasks that managerial staff constantly deal with. These shortcomings are eliminated in intelligent information systems (IIS). An analysis of the program structure shows the possibility of extracting operational knowledge from the program (data transformation rules) into the so-called knowledge base, which, in a declarative form, stores common knowledge units for various tasks. At the same time, the control structure takes on the character of a universal problem-solving mechanism (inference mechanism), which links knowledge units into executable chains (generated algorithms) depending on the specific problem statement (formulated in the goal's request and initial conditions). Such IP become systems based on the development of knowledge bases or simply knowledge (SBK).

$$SBK = Knowledge\ Base + \text{"}Control\ structure + (output\ mechanism)\text{"} + Database$$

Knowledge-based systems are intelligent IS (IIS) due to the possibility of generating algorithms for solving problems, which are characterized by the following features:

- developed communication skills;
- the ability to solve complex poorly formalized tasks;
- the ability to develop and self-control.

The communicative abilities of IIS characterize by the way end-user interacts (interface) with the system, in particular, the ability to formulate an arbitrary request in a dialogue with IIS in a language that is as close to natural as possible.

Complex poorly formalized tasks are tasks that require the construction of an original solution algorithm depending on a specific situation, for which uncertainty and dynamism of the initial data and knowledge may be characteristic.

The ability to self-study is the ability to automatically extract knowledge to solve problems from the accumulated experience of specific situations. In various IIS, the listed signs of intelligence are developed to a different degree. Conventionally, each of the signs corresponds to its own class of IIS is shown in figure 2.1.

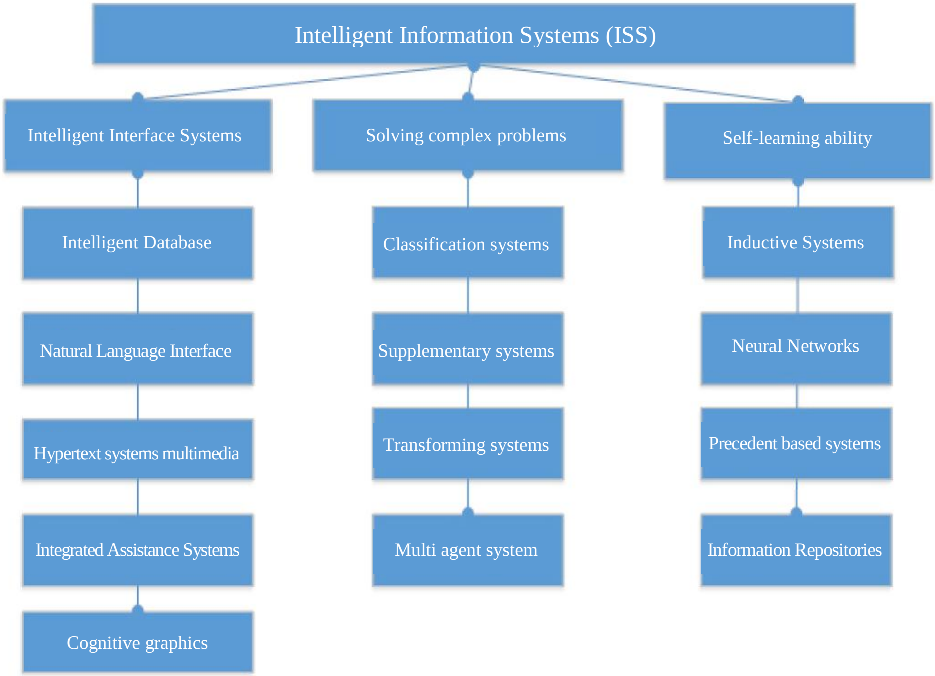

Figure 2.1. Classification of intelligent information systems

Systems with an intelligent interface

Intelligent databases differ from conventional databases by the ability to select, upon request, the necessary information, which may not be explicitly stored, but output from existing in the database. The intelligent user assistance database system on the database structure builds a path to access data files. Formation of the request is carried out in a dialogue with the user, the successive steps of which are performed in the form most convenient for the user. A database query can also be formulated using a natural language interface. The natural-language interface involves the translation of natural-language constructs at the intrafine level of knowledge representation. For this, it is necessary to solve the problems of morphological, syntactic, and semantic analysis and synthesis of sentences in a natural language.

The natural language interface is used for:

- access to intelligent databases;
- contextual search for documentary textual information;
- voice input of commands in the control system;
- machine translation from foreign languages.

Hypertext systems are designed to implement keyword searches in text information databases; search for multimedia information including graphic, audio, and video images in addition to text and digital information.

Context help systems are a partial case of intelligent hypertext and natural language systems. Unlike conventional help systems, the user describes the problem, and the system, with the help of an additional dialogue, specifies it and performs the flow of recommendations related to the situation. Such systems belong to the class of knowledge dissemination systems. Cognitive graphics systems make it possible to implement a user interface with IIS using graphic images that are generated under current events. Such systems are used in monitoring and managing operational processes. For example, the state of a complex controlled object is displayed in the form of a human face, on which each trait is responsible for some parameter, and the general facial expression gives an integrated characterization of the situation. Cognitive graphics systems are also widely used in educational and training systems based on the use of virtual reality principles, when graphic images simulate situations in which the student needs to make decisions and perform certain actions.

QUESTIONS:

1. Define the concept – knowledge and the concept of factual and operational knowledge.
2. What is a database management system?
3. What is a "Management structure"?
4. What characterizes the communicative abilities of IMS?
5. Give a classification of intelligent information systems.
6. What is the natural language interface used for?
7. What are hypertext systems designed for?

2.2. APPOINTMENT OF EXPERT SYSTEMS

In artificial intelligence, an expert system is a computer system that emulates the decision-making ability of a human expert (Jackson P., 1998). Expert systems are designed to solve complex problems by reasoning through bodies of knowledge, represented mainly as if-then rules rather than through conventional procedural code. The first expert systems were created in the 1970s and then proliferated in the 1980s (Leondes C., 2002). Expert systems were among the first truly successful forms of artificial intelligence (AI) software (Russell S., Norvig P., 1995).

An expert system is an example of a knowledge-based system. Expert systems were the first commercial systems to use a knowledge-based architecture. A knowledge-based system is essentially composed of two sub-systems: the knowledge base and the inference engine.

The purpose of expert systems is to solve problems that are quite difficult for experts based on the accumulated knowledge base. It reflects the experience of experts in this problem area. The advantage of using expert systems is the ability to make decisions in unique situations for which the algorithm is not known in advance and is formed according to the initial data in the form of a chain of reasoning (decision-making rules) from the knowledge base. Moreover, the solution of tasks is supposed to be carried out under conditions of incompleteness, inaccuracy, ambiguity of the initial information, and qualitative assessments of the processes.

The expert system is a tool that enhances the intellectual features of the expert, and can perform the following roles:

- a consultant for inexperienced or unprofessional users;
- an assistant in connection with the need for an expert to analyze various decision-making options;
- partner of an expert on issues related to sources of knowledge from related fields of activity;
- knowledge base (storage of knowledge units);
- a software tool for access and processing of knowledge, consisting of mechanisms for exporting conclusions, acquiring knowledge, explaining the results and an intelligent interface.

A knowledge base – is a set of knowledge units that are a formalized description of objects of a problem area and relationships, actions on objects and, possibly, the uncertainties with which these actions are carried out, using a method of representing knowledge. As methods for representing knowledge, either rules or objects (frames), or a combination of them, are most often used.

The knowledge base represents facts about the world. In early expert systems such as Mycin and Dendral, these facts were represented mainly as flat assertions about variables. In later expert systems developed with commercial shells, the knowledge base took on more structure and used concepts from object-oriented programming. The world was represented as classes, subclasses, as well as instances and assertions were replaced by values of object instances. The rules worked by querying and asserting values of the objects.

Rules are constructions:

If <conditions>, then <conclusion> CF (certainty factor) <value>.

Objects are a collection of attributes that describe properties and relationships with other objects.

Intelligent interface. The data exchange between the end-user of the IIS is carried out by an intelligent interface program that accepts user messages and converts them into a knowledge base representation form and transfers the internal representation of the processing result to the user format and issues a message to the required medium.

Output mechanism. This software tool receives a request transformed into an internal representation from an intelligent interface, generates a specific algorithm for solving a problem from the knowledge base, executes an algorithm, and the result is provided to an intelligent interface for issuing a response to a user request. The inference engine is an automated reasoning system that evaluates the current state of the knowledge-base, applies relevant rules, and then asserts new knowledge into the knowledge base.

There are mainly two modes for an inference engine: forward chaining and backward chaining. The different approaches are dictated by whether the inference engine is being driven by the antecedent (left-hand side) or the consequent (right-hand side) of the rule. In forward chaining it is an antecedent who fires and asserts the consequent.

Backward chaining is a bit less straightforward. In backward chaining, the system looks at possible conclusions and works backward to see if they might be true. One of the early innovations of expert systems shells was to integrate inference engines with a user interface. This could be especially powerful with backward chaining. If the system needs to know a particular fact but does not, then it can simply generate an input screen and ask the user if the information is known.

As expert systems evolved, many new techniques were incorporated into various types of inference engines (Mettrey W., 1987). Some of the most important of these were:

- truth maintenance. These systems record the dependencies in a knowledge-base so that when facts are altered, dependent knowledge can be altered accordingly;
- hypothetical reasoning. In this, the knowledge base can be divided up into many possible views, a.k.a. worlds. This allows the inference engine to explore multiple possibilities in parallel;
- uncertainty systems. One of the first extensions of simply using rules to represent knowledge was also to associate a probability with each rule;
- ontology classification. With the addition of object classes to the knowledge base, a new type of reasoning was possible. Along with reasoning simply about object values, the system could also reason about object structures. These types of special-purpose inference engines are termed classifiers. Although they were not highly used in expert systems, classifiers are very powerful for unstructured volatile domains and are a key technology for the Internet and the emerging Semantic Web (MacGregor R., 1991; Berners-Lee T., Hendler J., Lassila O., 2001).

The mechanism of explanation. In the process based on the results of solving the problem, the user can request an explanation or update of the solution code. The IS should provide an appropriate mechanism of explanation. The explanatory abilities of IS are determined by the ability of the inference mechanism to remember the way to solve the problem.

Knowledge acquisition mechanism. The knowledge base reflects the knowledge of experts in this problem area about actions in various situations or processes for solving characteristic problems. Identification of such knowledge and their subsequent presentation in the knowledge base is carried out by specialist knowledge engineers. To enter knowledge into the database there is a mechanism for acquiring knowledge. In the simplest case, this is an intelligent editor that allows you to enter units of knowledge into the database and bring their synthetic and semantic control.

The inference engine may also include abilities for explanation, so that it can explain to a user the chain of reasoning used to arrive at a particular conclusion by tracing back over the firing of rules that resulted in the assertion (Hayes-Roth F., Waterman D., Lenat D., 1983).

An expert system is divided into two subsystems: the inference engine and the knowledge base. The knowledge base represents facts and rules. The inference engine applies the rules to the known facts to deduce new facts. Inference engines can also include explanation and debugging abilities.

Expert systems were formally introduced around 1965 by the Stanford Heuristic Programming Project led by Edward Feigenbaum, who is sometimes termed the "father of expert systems"; other key early contributors were Bruce Buchanan and Randall Davis. The Stanford researchers tried to identify domains where expertise was highly valued and complex, such as diagnosing infectious diseases (Mycin) and identifying unknown organic molecules (Dendral). The idea that "intelligent systems derive their power from the knowledge they possess rather than from the specific formalisms and inference schemes they use"– as Feigenbaum E. (2003) said – was at the time a significant step forward. Expert systems became some of the first truly successful forms of artificial intelligence (AI) software (Russell S., Norvig P., 1995).

Classes of expert systems are subdivided according to the method of forming the solution (analytical – the choice of a solution from many known alternatives and synthetic – the generation of unknown solutions), according to the method of taking into account the time attribute (static – solving problems with constant data and dynamic – changing data is allowed), by the type of data use and knowledge (with deterministic knowledge and with indefinite knowledge), according to the number of sources of knowledge used (one and many).

Expert systems solve situations recognition problems are called classifying because they determine whether the analyzed situation belongs to a certain class. The method of logical deductive inference from general to particular is used as the main method of forming decisions. Redefining expert systems solve more complex analytical problems based on uncertain input data and knowledge. An expert system identifies missing knowledge.

The use of classifying and additional determining IS:

- data interpretation – the choice of a solution from many alternatives (direction, analysis of the financial condition of the enterprise);
- diagnostics – identifying the causes that led to situations. A preliminary interpretation of the situation is required, followed by verification of additional facts;
- correction – diagnostics, supplemented by the ability to evaluate and recommend actions to correct deviations from the normal state of the situations under consideration.

Transforming IS belong to synthesizing dynamic systems. As methods for solving problems are used:

- generation and testing, when hypotheses are generated according to the initial data, and then verification of the formulated hypotheses for confirmation by incoming facts;
- the method of sentences and defaults, when, based on incomplete data, knowledge about similar classes of objects is selected, which in the future will dynamically adapt to a specific situation depending on its development;
- the use of general laws (meta-management in the case of unknown situations that allow the generation of missing knowledge).

Multi-agent systems. Such dynamic systems are characterized by integration in the knowledge base, exchanging with each other the obtained results on a dynamic basis, for example, through a "message board".

Features:

- conducting alternative considerations based on the use of various sources of knowledge with a mechanism for eliminating contradictions;
- a distributed solution to problems, which divided into simultaneously solved sub-problems, corresponding to an independent source of knowledge, etc.

Application:

- design (enterprise budget, investment portfolio);
- forecasting;
- dispatching – work distribution, scheduling;
- planning;
- monitoring;
- management.

Hayes-Roth etc. (1983) divides expert systems applications into 10 categories space of expert systems applications, they are not rigid categories, and, in some cases, an application may show traits of more than one category.

Table 2.1. Applications (table)

Category	Problem addressed	Examples
Interpretation	Inferring situation descriptions from sensor data	Hearsay (speech recognition), PROSPECTOR
Prediction	Inferring likely consequences of given situations	Preterm Birth Risk Assessment
Diagnosis	Inferring system malfunctions from observables	CADUCEUS, MYCIN, PUFF, Mistral, Eydenet, Kaleidos
Design	Configuring objects under constraints	Dendral, Mortgage Loan Advisor, R1 (DEC VAX Configuration), SID (DEC VAX 9000 CPU)
Planning	Designing actions	Mission Planning for Autonomous Underwater Vehicle
Monitoring	Comparing observations to plan vulnerabilities	REACTOR
Debugging	Providing incremental solutions for complex problems	SAINT, MATHLAB, MACSYMA
Repair	Executing a plan to administer a prescribed remedy	Toxic Spill Crisis Management
Instruction	Diagnosing, assessing, and repairing student behavior	SMH.PAL Intelligent Clinical Training STEAME
Control	Interpreting, predicting, repairing, and monitoring system behaviors	Real Time Process Control, Space Shuttle Mission Contro

About 30% of expert systems serve for diagnostics, 15% for interpretation, 15% for recommendations, 12% for planning, 9% for monitoring, and 8% for management. A smaller share is occupied by the tasks of modeling, selection.

Research on expert systems was also active in France. While in the US the focus tended to be on rules-based systems, first on systems hardcoded on top of LISP programming environments and then on expert system shells developed by vendors such as Intellicorp, in France research focused more on systems developed in Prolog. The advantage of expert system shells was that they were somewhat easier for nonprogrammers to use. The advantage of Prolog environments was that they were not focused only on if-then rules; Prolog environments provided a much better realization of a complete first-order logic environment (George F. Luger, William A. Stubblefield, 1989).

The first expert system to be used in a design capacity for a large-scale product was the SID (Synthesis of Integral Design) software program, developed in 1982. Written in LISP, SID generated 93% of the VAX 9000 CPU logic gates (Carl S. Gibson et al., 1990).

Many of the leading major business application suite vendors (such as SAP, Siebel, and Oracle) integrated expert system abilities into their suite of products as a way of specifying business logic – rule engines are no longer simply for defining the rules an expert would use but for any type of complex, volatile, and critical business logic.

Modern systems can incorporate new knowledge more easily and thus update themselves easily. Such systems can generalize from existing knowledge better and deal with vast amounts of complex data. Related is the subject of big data here. Sometimes these types of expert systems are called "intelligent systems" (Yanase J., Triantaphyllou E., 2019).

Advantages

The goal of knowledge-based systems is to make the critical information required for the system to work explicit rather than implicit (Hayes-Roth F., Waterman D., Lenat, D. 1983). In a traditional computer program, the logic is embedded in code that can typically only be reviewed by an IT specialist. With an expert system, the goal was to specify the rules in a format that was intuitive and easily understood, reviewed, and even edited by domain experts rather than IT experts. The benefits of this explicit knowledge representation were rapid development and ease of maintenance.

Ease of maintenance is the most obvious benefit. This was achieved in two ways. First, by removing the need to write conventional code, many of the ordinary problems that can be caused by even small changes to a system could be avoided with expert systems. Essentially, the logical flow of the program (at least at the highest level) was simply given for the system, simply invoked the inference engine. This also was a reason for the second benefit: rapid prototyping. With an expert system Shell, it was possible to enter a few rules and have a prototype developed in days rather than the months or years typically associated with complex IT projects.

Disadvantages

The most common disadvantage cited for expert systems in the academic literature is the knowledge acquisition problem. However, when looking at the life-cycle of expert systems in actual use, other problems – essentially the same problems as those of any other large system – seem at least as critical as knowledge acquisition: integration, access to large databases, and performance (Kendal S.L., Creen M., 2007).

Performance, system and database integration. These issues were resolved mainly by the client-server paradigm shift, as PCs were gradually accepted in the IT environment as a legitimate platform for serious business system development and as affordable mini-computer servers provided the processing power needed for AI applications (Wong Bo, Monaco J., 1995).

Another major challenge of expert systems emerges when the size of the knowledge base increases. This causes the processing complexity to increase. An inference engine would have to be able to process huge numbers of rules to reach a decision.

When there are too many rules, such a problem leads to a satisfiability (SAT) formulation (Bezem M., 1988). This is a well-known NP-complete problem Boolean satisfiability problem, space can grow exponentially.

There are also questions on how to prioritize the use of the rules to operate more efficiently, or how to resolve ambiguities, and so on (Mak B., Schmitt B.H., Lyytinen K., 1997).

Other problems are related to the overfitting and overgeneralization effects when using known facts and trying to generalize to other cases not described explicitly in the knowledge base. Such problems exist with methods that employ machine learning approaches too (Pham H.N., Triantaphyllou E., 2008).

Another problem related to the knowledge base is how to make updates of its knowledge quickly and effectively (Coats P.K., 1988; Shan N, Ziarko W., 1995). Furthermore, how to add a new piece of knowledge (i.e., where to add it among many rules) is challenging.

Because of the above challenges, it became clear that new approaches to AI were required instead of rule-based technologies. These new approaches are based on the use of machine learning techniques, along with the use of feedback mechanisms (Yanase J., Triantaphyllou E., 2019).

QUESTIONS:

1. What roles can an expert system play?
2. Give the architecture of the expert system.
3. What is an Intelligent Interface?
4. What is a withdrawal mechanism?
5. What is an explanation mechanism?
6. What is a knowledge acquisition mechanism?
7. How are the classes of expert systems divided?
8. What are the expert systems that solve situation recognition problems?
9. What are multi-agent systems?

2.3. SELF-LEARNING SYSTEMS

Self-learning systems are based on the methods of automatic classification of examples of real-life situations (learning by examples). Examples of real situations accumulate over a certain historical period and constitute a training sample. These examples are described by many classification features. Moreover, the training sample may be:

- "with a teacher", when for each example the value of the attribute of his belonging to a certain class of situations (class-forming attribute) is set explicitly;
- "without a teacher", when the system itself identifies classes of situations according to the degree of closeness of the values of the signs of classification. As a result of training the system, generalized rules or functions are automatically constructed that determine whether the situations belong to the classes that the trained system uses to interpret new situations that arise. Thus, a knowledge base is automatically generated that is used to solve classification and forecasting problems. This knowledge base is periodically automatically adjusted as experience accumulates in real situations, which reduces the cost of creation and updating.

Common disadvantages of all self-learning systems are as follows:

- incompleteness and/or noisiness (redundancy) of the training sample and, as a consequence, the relative adequacy of the knowledge base to emerging problems are possible;
- there are problems associated with poor semantic clarity of feature dependencies and, as a result, inability to explain the results to users;
- restrictions in the dimension of the attribute space cause a shallow description of the problem area and a narrow focus of the application.

Inductive Systems

The generalization of examples on a principle from particular to general comes down to identifying subsets of examples belonging to the same subclasses and identifying significant features for them. The classification process for examples is as follows:

1. A classification criterion selected from the set of specified ones (either sequentially or by some rule, for example, under the maximum number of obtained subsets of examples).
2. By the value of the selected attribute, many examples are divided into subsets.
3. A check is made to see if each formed a subset of the examples of one subclass.
4. If a subset of examples belongs to one subclass, i.e. for all examples of the subset, the value of the class-forming feature coincides, then the classification process ends (while the remaining classification features are not considered).
5. For subsets of examples with a mismatching value of the class-forming feature, the process of classifying features continues with a sequential classification, the values of the attribute of belonging to a particular class are determined. An example of constructing a decision tree based on a fragment of an example iы shown in table 2.2. and in figure 2.2.

The analysis of a typical situation is reduced to the choice of a tree branch, which completely defines this situation. The search for a solution is carried out as a result of a consistent check of classification features. Each tree branch corresponds to one decision rule:

If Demand = "low" and Costs = "small", then Price = "low".

Examples of tools that support inductive knowledge output are 1-st Class (Programs in Motion), Rulcmaster (Radian Corp.), ILIS (ArgusSoft), KAD (IPS Pereyaslavl-Zalessky).

Table 2.2. Table of Examples of Solutions

Class-forming feature	Signs of classification			
Price	Demand	Competition	Costs	Amount
low	low	small	small	low
high	low	small	big	high
high	high	small	big	low
high	high	small	small	high
high	high	small	small	low
high	high	small	big	high

[demand] – high – [high price]

low [costs] – large – [high price]

– small – [low price]

Figure 2.2. A fragment of the decision tree

Use-Case Systems (Case-Based Reasoning (CBR))

In these systems, the knowledge base contains descriptions of not generalized situations, but situations or precedents themselves. Then the search for a solution to the problem comes down to a search by analogy (abductive conclusion from particular to particular):

1. Getting detailed information about the current problem.
2. Comparison of the received information with the values of the signs of precedents from the knowledge base.
3. The choice of a precedent from the knowledge base closest to the problem under consideration.
4. If necessary, the selected precedent is adapted to the current problem.
5. Validation of each decision received.
6. Entering detailed information about the solution into the knowledge base.

As well as for inductive systems, precedents are described by a set of signs by which quick search indices are built. But unlike inductive systems, a fuzzy search is allowed with the receipt of a set of valid alternatives, each of which is evaluated by a certain confidence coefficient. Further, the most suitable solutions are adapted according to special algorithms to real situations. System training comes down to remembering each new processed situation with the decisions made in the use case database.

Use-case systems are used as knowledge dissemination systems with advanced features or as context-sensitive help systems.

Dialog example with CBR system

Description of the situation (problem)

Does not print the printer

Questions

Is it included? Yes

Did the test pass? Yes

Is paper jammed? Yes

Is a driver connected? I do not know

Actions

Make paper-free confidence 80

Download Driver confidence 50

Call staff confidence 10

As an example of a tool for supporting precedent knowledge bases distributed in Russia, the CBR-Express system was named (Inference, distributor, Metatechnology company). Information Warehouse. Unlike an intelligent database, an information storage is a repository of extracted significant information from an operational database, which is intended for operational data analysis (implementing OLAP technology) – Knowledge is extracted from databases regularly, for example, daily. Typical tasks of operational situational analysis are:

- determining the profile of consumers of a particular product;
- prediction of changes in the market situation;
- analysis of the dependencies of the signs of situations (correlation analysis), etc.

Special methods (Data Mining or Knowledge Discovery) are used to extract meaningful information from databases, based either on the use of multidimensional statistical tables, or inductive methods for constructing decision trees, or Neural Networks. The request is formulated as a result of the use of an intelligent interface that allows the dialog to flexibly determine the significant signs of analysis. The use of information storages in practice increasingly demonstrates the need to integrate intelligent and traditional information technologies, the integration of various methods for presenting and outputting knowledge, and the complexity of the architecture of information systems. The development and dissemination of information repositories are currently being done by computer companies such as IBM (Intelligent Miner), Silicon Graphics (MineSet), Intersolv (DataDirect, SmartData), Oracle (Express), SAS Institute (SAS/Assist) etc.

QUESTIONS:

1. What are the self-learning systems?
2. What are the disadvantages inherent in all self-learning systems?
3. Explain the essence of inductive systems.
4. What is the point of Case-based reasoning?

2.4. TECHNOLOGY FOR CREATING EXPERT SYSTEMS

At the initial stages of identification and conceptualization associated with determining the contours of a future system, a knowledge engineer acts as a student, and an expert as a teacher, master. At the final stages of implementation and testing, the knowledge engineer demonstrates the development results, the adequacy of which the problem area is assessed by the expert. At the testing stage, these may be completely different experts. At the testing stage, the created expert systems are evaluated from the position of two main groups of criteria: accuracy and utility. In general terms, the following stages of creating expert systems are distinguished:

- identification of the problem area;
- conceptualization of the problem area;
- formalization of the knowledge base;
- database implementation;
- testing the knowledge base;
- trial operation.

The stage of identifying a problem area includes determining the purpose and scope of the expert system, selecting experts and a group of knowledge engineers, allocating resources, setting and parameterizing the tasks to be solved.

The purpose of the expert system is related to one of the following areas:

- training and consultation of inexperienced users;
- dissemination and use of the unique experience of experts;
- automation of the work of decision-making experts;
- optimization of problem-solving, advancing, and testing hypotheses.

The limiting factors for the development of the expert system are the allotted timelines, financial resources, as well as the software and hardware environment. The quantitative composition of groups of knowledge engineers and experts, the depth of issues being worked out, the adequacy and effectiveness of problem-solving depend on these limitations. Usually, there are three strategies for developing expert systems:

- a wide range of tasks, each of which is focused on a narrow problem area;
- a concentrated set of tasks that defines the main directions of increasing the efficiency of the economic object;
- a comprehensive set of tasks that determines the organization of all activities of an economic object.

The main parameters of the problem area include the following:

- the class of tasks to be solved (interpretation, diagnostics, correction, forecasting, planning, design, monitoring, management);
- criteria for the effectiveness of the results of solving problems (minimizing the use of resources, improving the quality of products and services, accelerating the turnover of capital, etc.);
- criteria for the effectiveness of the process of solving problems (improving the accuracy of decisions, taking into account a greater number of alternative options,

adaptability to changes in the problem area of the information needs of users, reducing the time for decision-making);

- goals of the tasks to be solved (selection from alternatives, for example, budget allocation by articles);
- subgoals – dividing a task into subproblems, each of which has its own goal),
- source data (set of factors used);
- features of the knowledge used (determinism, uncertainty, static/dynamic, single-purpose and multi-purpose orientation, uniqueness/multiplicity of sources of knowledge).

At the stage of constructing the conceptual model, a holistic and systematic description of the knowledge used is created that reflects the essence of the functioning of the problem area.

The result of the conceptualization of the problem area is usually fixed in the form of visual graphic diagrams at the object, functional and behavioral levels of modeling:

- the object model describes the structure of the subject area as a set of inter-connected objects;
- the functional model reflects actions and transformations on objects;
- the behavioral model considers the interactions of objects in the temporal aspect.

The first two models describe the static aspects of the functioning of the problem area, and the third model describes the dynamics of its states. The object model reflects factual knowledge about the composition of objects, their properties, and relationships. An elementary unit of structural knowledge is a fact that describes one property or one relationship of an object, which is presented in the form of a triple: a predicate (object, value). A class of objects refers to a collection of objects with the same set of predicates (properties and relationships). A class of objects is often described as an *n*-ary relational relation.

If the objects have a partially overlapping set of predicates, then a complex classification of objects is carried out: the class of objects according to the values of a property (attribute) is divided into subclasses in such a way that the class of objects contains properties and relationships common to subclasses, and each of the subclasses reflects specific properties and communication. In this case, subclasses of objects automatically inherit the common properties and relationships of higher classes, and the set of classes of objects interconnected with respect to the generalization forms an inheritance hierarchy of properties.

Usually, object knowledge is represented graphically by means ER- models (the "Entity – Communication" model). The functional model describes the transformation of facts, their relationships, which show how facts are formed from others.

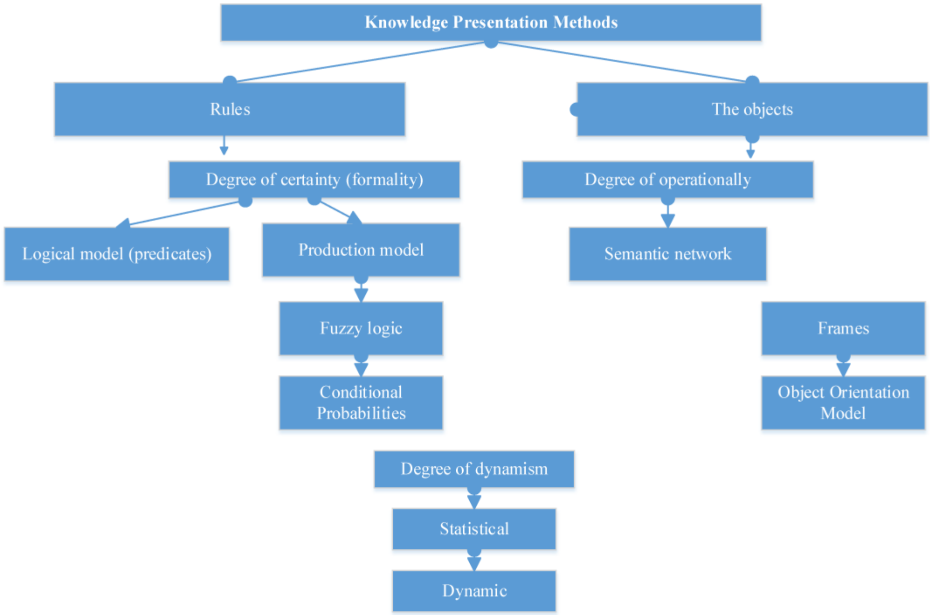

Figure 2.3. Methods of knowledge representation in expert systems

The functional model is built by sequentially composing the goals, namely: for the goal, subgoals are determined, for which subgoals are set in turn, and so on, until the initial facts appear as subgoals (the process of composing "from above" – "down"). Each goal (sub-goal) corresponds to a certain task (sub-task), which cannot be solved until its sub-goals (the sub-tasks) are achieved. Thus, the functional model reflects in a generalized form the process of solving its characteristic problems.

Typically, the functional dependencies of facts are represented graphically in the form of goal trees or graphs. The behavioral model reflects a change in the state of objects as a result of the occurrence of certain events that entail the execution of certain actions (procedures). The state of an object is the time-varying values of a property. A set of actions associated with a certain event constitutes the behavior of an object, which is expressed in the form of rules or procedures.

At the stage of formalization of the knowledge base, a method of representing knowledge is selected. Within the framework of the chosen formalism, the design of the logical structure of the knowledge base is carried out.

QUESTIONS:

1. List the stages of creating expert systems.
2. What are the limiting factors for the development of an expert system?
3. What are the strategies for developing expert systems?
4. How is the result of the conceptualization of the problem area recorded?
5. How is the functional model built?

2.5. METHODS OF WORKING WITH KNOWLEDGE

Knowledge is a familiarity, awareness, or understanding of someone or something, which can be facts, information, descriptions, or skills, which is acquired through experience or education by perceiving, discovering, or learning.

Knowledge can refer to a theoretical or practical understanding of a subject. It can be implicit (as with practical skill or expertise) or explicit (as with the theoretical understanding of a subject); it can be more or less formal or systematic. In philosophy, the study of knowledge is called epistemology; the philosopher Plato famously defined knowledge as "justified true belief", though this definition is now thought by some analytic philosophers[citation needed] to be problematic because of the Gettier problems, while others defend the Platonic definition (Boghossian P., 2007), However, several definitions of knowledge and theories to explain it exist.

Knowledge acquisition involves complex cognitive processes: perception, communication, and reasoning; while knowledge is also said to be related to the capacity of acknowldgement in human beings (Stanley C., 2002).

The acquisition of knowledge is the identification of knowledge from sources and their transformation into the desired form, as well as the transfer to the knowledge base of IS. Sources of knowledge can be books, archival documents, the contents of other knowledge bases, etc., i.e., some objectified knowledge translated into a form that makes them accessible to the consumer. Another type of knowledge is expert knowledge, which is available to specialists but is not fixed in storage facilities external to it. Expert knowledge is subjective. Empirical knowledge is another type of subjective knowledge. Such knowledge can be gained by IP by observing the environment (if IS has the means of observation).

Some methods of generating knowledge, such as trial and error, or learning from experience, tend to create highly situational knowledge. Situational knowledge is often embedded in language, culture, or traditions. This integration of situational knowledge is an allusion to the community, and its attempts at collecting subjective perspectives into an embodiment "of views from somewhere" (Haraway D., 1988).

Entering objectified knowledge into the knowledge base is not a particular problem; identifying and entering subjective and especially expert knowledge is quite difficult. To develop a methodology for the acquisition of subjective knowledge obtained from an expert, it is necessary to clearly distinguish between two forms of knowledge representation. One form is related to how and in which models this knowledge is stored by a human expert. Moreover, the expert does not always fully realize how his knowledge is represented. Another form is related to how a knowledge engineer who designs IS is going to describe and represent them. The efficiency of the knowledge engineer's work depends on the degree of consistency of these two forms of representation among themselves.

In cognitive psychology, the forms of knowledge representation (cognitive structures of knowledge) characteristic of a person are studied. Examples: representing a class of concepts through its elements (for example, the concept of a bird being represented next to a seagull, sparrow, starling...) representing class concepts using a basic prototype that reflects the most typical properties of class objects (for example, the concept of a bird something with wings, beak, flies is represented by the prototype) representation thanks to signs (for the concept of "bird", for example, the presence of wings, beak, two legs, feathers).

In addition to concepts, relations between them are represented. As a rule, relations between concepts are determined in a procedural way, and relations between components of concepts (determining the structure of a concept) in a declarative way. The presence of two types of descriptions makes knowledge models simultaneously have both components, for example, a semantic network and a production system, as presented in the cognitive model.

Acquiring knowledge, an important role is played by the so-called field of knowledge, which contains the basic concepts, which are used in the description of the subject area, and the properties of all relations used to establish connections between concepts. The field of knowledge is associated with the conceptual model of the problem area, in which the limitations that inevitably arise during the formal representation of knowledge in the knowledge base have not yet been considered. The transition from the description of a certain field in the knowledge field to the description in the knowledge base is similar to the transition from the conceptual model of the database to its logical scheme, when the database management system is already fixed. It is important to note that the transition directly to formal representations in the knowledge base without the conceptual description stage in the knowledge field leads to numerous errors, which slows down the process of forming the IS knowledge base.

Three modes of interaction between a knowledge engineer and an expert specialist are possible: protocol analysis, interviews, and game simulations of professional activities. Protocol analysis consists in fixing (for example, by recording on magnetic tape) the "thoughts out loud" of the expert during the solution of the problem and in the subsequent analysis of the information received. In the interview mode, the knowledge engineer conducts an active dialogue with the expert, directing him in the right direction. In the game simulation, the expert is placed in situations similar to those in which his professional activity takes place. Watching his actions in various situations, the knowledge engineer forms his thoughts on expert knowledge, which can subsequently be clarified with the expert in an interview mode. The principles of game simulation have found application in a variety of business games, special simulators.

Each of the mentioned methods for extracting knowledge has advantages and disadvantages. Thus, when analyzing protocols, it is not easy for a knowledge engineer to separate concepts that are important for inclusion in a domain dictionary from those that appear "randomly" when they occur aloud. In addition, gaps are detected in the protocols when the expert’s reasoning is interrupted and continues based on the omitted output steps. Filling such gaps is only possible in interview mode. Thus, in all three approaches to extracting knowledge from experts, an interview stage is necessary, which makes it one of the most important methods for acquiring knowledge.

There are at least two dozen interviewing strategies. Three are best known: tiling, repertory grid, and affirmation of similarity,

When dividing into steps, the expert is invited to name the most important, in his opinion, concepts of the subject area and indicate between them the relations of structuring, that is, relations of the type "genus-species", "element class", "whole-part", etc. These concepts are used in the next step of the survey as basic. The strategy is aimed at creating a hierarchy of concepts of the subject area, the allocation in concepts of closely related groups of hacks (clusters).

The strategy of the repertory grid is aimed at identifying the characteristic properties of concepts that allow one to separate one concept from another. The methodology consists in presenting to the expert triples of concepts with a proposal to name signs for every two concepts that would separate them from the third. Since each concept is included in several triples, based on such a procedure, the volumes of concepts are refined and "symptom complexes" of concepts are formed with the help of which these concepts can be identified in the knowledge base.

The strategy for confirming the similarity is that the expert is invited to establish that each pair of concepts from the subject area belongs to a certain similarity (tolerance) ratio. For this, the expert is asked a sequence of fairly simple questions, the purpose of which is to clarify the understanding of the similarities that the expert puts into the statement about the similarity of the two concepts of the subject area.

The process of interaction of a knowledge engineer (analytics) with an expert specialist includes three main stages.

1. The preparatory phase. For communication success, both participants must carefully prepare for a dialogue or a game. It is desirable that the expert was not only a competent specialistbut also an interested (morally or financially) person in achieving the ultimate goal of building IS. He should be friendly to the analyst and be able to explain his knowledge (the best case is when the expert has experience in teaching).

Analysts need to: deeply familiarize themselves with the special literature on the subject field "in order not to ask very" stupid "questions (just" stupid "questions are extremely useful), as well as increase the number of" packages of expectations"; be able to listen and correctly ask questions; tune in to the role of a "student" rather than an "examiner"; to understand models of cognitive psychology, as well as models of knowledge representation, in order to distinguish clear structures from expert knowledge.

In any joint activity, the psychological qualities of researchers, such as personality, manner of behavior, and the style of scientific thinking, are of great importance. There are various classifications of scientists. As an example, we cite the following: initiator – quickly responds to promising problems, that is, one of the first to feel the need to solve the problem with elements of uncertainty; the diagnostician is capable of quickly assessing the strengths, and weaknesses of solving the problem, the erudite is endowed with exceptional memory, is distinguished by increased attention to detail and a desire for ordering; artisan – able to realize the poorly designed ideas of others; esthetician – seeks to investigate problems leading to elegant solutions, is not prone to painstaking work; the methodologist is interested in the methodological aspects of research; independent – strives for an individual solution to problems; fanatic – selflessly passionate about his scientific problem, it requires the same from others.

The belonging of a scientist to one or another type is determined using indirect methods (tests of personality, intelligence, cognitive styles, design techniques). Automation of the survey and obtaining a psychological portrait of the subject is implemented, for example, in the AUTANTEST system.

For the role of an expert, the initiator of the scholar, the diagnostician, and the artisan (paired with the analyst-scholar) are most preferred, and for the role of the analyst-diagnostician, the methodologist, the scholar, the initiator. At the same time, the best combination is given by combinations of different types. Due to differences in approaches to solving the problem, in terms of viewpoint, style of thinking, perception, memory, etc., participants in such a pair approach the goal from different angles, as a result, the total number of hypotheses, ideas, alternative options increases, and therefore the field of knowledge is enriched. However, not all combinations, even from acceptable types, improve interaction, and some types (e.g., fanatic, aesthete, independent, artisan) are often poorly adapted for creative interaction, which leads to the emergence of hidden and overt conflicts that complicate the process of productive communication.

Pair leadership is also important. During any dialogue, one side usually takes the leading position, more often the interviewer, that is, the analyst, takes this role. The role of the leader in the dialogue allows the analyst to direct and systematize the process of creating the field of knowledge, preventing the expert from “blurring” or unnecessarily detailing the process. On the other hand, dogmatism and perseverance can lead to an inadequate field. There is also a "facade" effect, that is, the expert’s desire not to face “in the dirt” in front of the analyst, and hence the generation of unconfirmed hypotheses.

2. Establishment of a "common code". To create a linguistic alliance of interaction, the participants in the interaction should try to reduce the distance between the object (i.e., the subject area being studied) and the analyst. It is necessary to determine the main concepts, that is, develop a vocabulary base for the knowledge base; level of detail; relationships between concepts.
3. The epistemological stage. At this stage, the patterns inherent in the subject area, the conditions of the reliability and truth of the statements are clarified, structured by introducing relationships, etc. This stage is crucial in the interaction of the analyst and expert. In the process of analyzing a game or dialogue, the expert’s knowledge is verbalized and formalized, and often new knowledge is generated for him.

The representation of the external world in his memory receives a material embodiment in the form of a field of knowledge.

In the process of extracting knowledge, it is first desirable to obtain superficial knowledge from the expert (such as, for example, the representation of features), gradually moving to deeper structures and more abstract concepts (such as, for example, prototypes).

When forming the field of knowledge, the features of empirical knowledge are taken into account: modality, inconsistency, incompleteness, etc.,

The analyst must always see the general behind the quotient, that is, build the chains of 1 fact – a generalized fact – an empirical law – a theoretical law. "The central link in the chain is the formalization of empiricism. In this case, sometimes the main thing at the formalization stage is not to extract "blind" incomprehensible connections, and understanding the internal structural connection of the concepts of the subject area. The art of the analyst consists in the desire to create a clear and understandable model of the problem area.

It should be borne that experts in the problem area do not always rely on logical reasoning. In their ideas about the problem area and methods for solving problems characteristic of it, associative reasoning and likelihood reasoning are widely used. We describe an approximate methodology for working with an expert on the formation of a field of knowledge.

Preparatory stage

A clear definition of the tasks of the designed system (narrowing the field of knowledge): determining what is at the input and output; determination of the mode of work, consultations, training, etc.

Selection of experts: determining the number of experts; the choice of the level of competence (it is not always good to choose the highest level immediately); identification of ways and opportunities to interest experts in the work; testing experts.

Acquaintance of the analyst with special literature in the subject area.

Acquaintance of the analyst and experts (in the future, for simplicity, we assume that there is only one expert).

Acquaintance of the expert with the popular literature on artificial intelligence (desirable, but not necessary).

An analyst’s attempt to create a field of knowledge of the first approximation to a priori knowledge from literature (prototype of the field of knowledge).

Main stage

"Pumping" knowledge fields:

a) depending on the subject area, the choice of method of interviewing;
b) logging thoughts aloud or recording the expert’s reasoning on a tape recorder (the analyst should not interfere in the reasoning if possible).

"Homework". The analyst’s attempt to highlight some causal relationships in the expert’s reasoning; building a domain dictionary (possibly on cards) and preparing questions for an expert.

"Swap" of the field of view. Discussion with an expert of the prototype of the field of knowledge and homework, as well as answers to questions from the analyst.

Formalization of the conceptual model.

Building a field of knowledge of the second approximation.

Expertise acquisition systems

Some psychologists associate the problems that arise when extracting expert knowledge with the so-called cognitive defense. A well-known theory of human cognition is based on the concept of "personal constructs" that a person creates and tries to adapt to the realities of the world. The theory of personal constructs is also known, which was used to create a system for extracting expert knowledge and has shown its ability to successfully overcome cognitive protection, i.e., the reluctance of experts to achieve a clear and conscious interpretation of the basic concepts, relations between concepts and methods of solving problems in the knowledge engineer of interest to the problem area.

Interviewing expert methods for a subject domain using several different strategies have been applied to create the TEIRESIAS system. There is an opportunity to describe the interview methodology, aimed at clarifying the essence, hypotheses, symptoms, conditions, relationships and ways. A hypothesis is an event whose identification results in a diagnosis. A symptom is an event resulting from the existence of a hypothesis, the observation of which brings closer the subsequent acceptance of the hypothesis. A condition is an event or some set of events that is not directly symptomatic of any hypothesis, but which may have diagnostic value for some other events. Relationships are compounds of entities (including other relationships). A path is a distinguished type of connection that connects hypotheses with symptoms. In accordance with this, the following interview strategies are used: differentiation of hypotheses, distinction of symptoms, symptomatic conditioning, division of the path, etc.

Differentiation of hypotheses is aimed at finding symptoms that provide a more accurate distinction between hypotheses. The most powerful in this sense are those symptoms that come from one diagnosed event. The distinction of symptoms reveals the specific characteristics of a symptom, which, on the one hand, identify it as a consequence of a certain hypothesis, and on the other hand, contrast it with others.

Symptomatic conditionality is aimed at identifying negative symptoms, i.e., symptoms, the absence of which has a greater diagnostic weight than their presence. The division of the path ensures the finding of symptomatic events that lie on the path to the symptom already found. If such a symptom exists, then it has a greater diagnostic value than already found.

In the KRITON system, two sources are used to acquire knowledge: an expert with his knowledge gained in practice (this knowledge is usually incomplete, fragmentary, poorly structured); book knowledge, documents, instruction descriptions (this knowledge is well-structured and fixed by traditional means). To extract knowledge from the first source, KRITON employs an interview technique that uses the strategies of the repertory lattice and tiling. In this case, the technique of switching strategies is applied: if, upon presenting a trio of semantically related concepts, an expert is not able to name a feature that distinguishes two of them from a third, the system starts a strategy of dividing into steps and attempts to determine the taxonomic structure of these concepts to identify features that distinguish them.

To identify the expert's procedural knowledge, KRITON uses the protocol analysis method. It is carried out in five steps. At the first step, the protocol is divided into segments based on the pauses that the expert makes during the recording process. The second step is the semantic analysis of segments, the formation of statements for each segment. In the third step, operators and arguments are highlighted from the text. Next, an attempt is made to search by the model in the knowledge base for detecting variables in statements (a variable is inserted into the statement if the corresponding link is not found in the text). At the last step, the statements are ordered according to their appearance in the protocol.

Text analysis is used at KRITON to identify well-structured knowledge from books, documents, descriptions, and instructions.

There is a method for identifying a domain model. The first phase is the formation by the engineer of knowledge of a rough model of the subject area by defining the predicates and varieties of their possible arguments and communicating to the system of facts about the domain expressed by these predicates. The system reveals the properties of predicates and establishes relationships between them, thus structuring the subject area. In the second phase, thanks to metacognition (general structures) that reflect the characteristics of human thinking, the facts are checked for predicates, the rules are inductively derived from facts, and the rules are derived from other rules.

In SIMER systems, the main method of acquiring knowledge is automated expert interviewing, which is controlled by the knowledge acquired by the system. In SIMER systems, a preliminary region model is not detected. All objects (events) and their attributes are determined in the direct interview mode of the expert. It is only assumed that a set of objects can be assigned several relations from a known (finite) set: "element-set", "part-whole", "example-prototype", relations of structural similarity of objects, structural hierarchy, and some others. All relations are pairwise distinguished by formal properties. So, the relationship of structural similarity is not transitive, but symmetrical. The relation of the structural hierarchy, on the contrary, does not have symmetry, but it is transitive. The interview is aimed at clarifying these and several other properties of relations and objects.

In particular, to establish structural similarities in the first phase of the interview, for each newly introduced concept, the expert is invited to indicate (using the menu) those domain concepts with which this can be associated (without specification of the relationship). Then, during the interview for each pair of concepts (from those identified in the first phase), the relationship is specified, the properties and type of relationship are established, the number of elements of which includes the pair under study. So, to include a certain pair of concepts X and Y, about which the expert said that X affects Y (for example, X increases the possibility of Y), among the elements of a certain relation I, which has symmetry among other properties, it is necessary to ask the expert a question: "Increases Is Y an Opportunity for X? " If the answer to this question is positive (and if other properties are already established and satisfy the definition of the relation), the pair (X, Y) is included in R. To establish structural similarity and the structural hierarchy of concepts, strategies for confirming similarity and splitting into steps are used.

The model has meta-procedures and meta-rules that verify the correctness of the model, use the formal properties of relations to supplement the model, and generate rules.

The main stages of the implementation of the knowledge acquisition system:

1. Interviews to determine the actual area in which the process of solving the problem of interest occurs, and its division into autonomous areas.
2. Automated interviews to identify and generate declarative domain model.

3. Protocol analysis to the concepts and relations of the subject area identified at the previous stage for replenishing the model with procedural knowledge (stages 2 and 3 can be used interchangeably until the model reaches the desired completeness).
4. Protocol analysis to supplement the declarative knowledge of the model.
5. Checking the completeness of the model. Typically, protocol analysis reveals voids in a model. This refers to the case when the concepts used in "thinking out loud" are not adequately described. In this case, the interview and protocol analysis are repeated.

Formalization of quality knowledge

In formalizing qualitative knowledge, the theory of fuzzy sets can be used, especially those aspects of it that are associated with the linguistic uncertainty that most often occurs when working with experts in natural language. Linguistic uncertainty does not mean the polymorphism of words in a natural language, which can be overcome at the level of understanding the meaning of sentences in the Bayesian model, but the qualitative estimates of the natural language for length, time, intensity, for inference, decision-making, planning.

Linguistic uncertainty in knowledge representation systems is defined using linguistic models based on the theory of linguistic variables and the theory of approximate reasoning. These theories are based on the concept of a fuzzy set, a system of operations on fuzzy sets, and methods for constructing membership functions.

One of the basic concepts used in linguistic models is the concept of a linguistic variable. The meanings of linguistic variables are not numbers, but words or sentences of some artificial or natural language. For example, the numerical variable "age" takes discrete values between zero and one hundred, and an integer is the value of the variable. The linguistic variable age can take on the values: young, old, fairly old, very young, etc. These terms are the linguistic values of the variable. This set (as well as numbers) is also subject to restrictions. The set of valid values of a linguistic variable is called a term set.

When entering information about linguistic variables and a term set into a computer, it must be presented in a form suitable for working on a computer. A linguistic variable is defined by a set of five components:

$$<L, T(L), U, V, M>,$$

where L – name of a linguistic variable;

T(L) – term set;

U – area where linguistic variable values are defined.

And it describes operations to generate derived values of a linguistic variable based on those values that are included in the term set. Using the rules from 0, we can expand the number of values of a linguistic variable, that is, expand its term set. Each value A of the linguistic variable A corresponds to a fuzzy set, which is a subset of V. By analogy with formal systems, the rules are often called syntactic. Finally, component M forms a set of semantic rules. With their help, the linguistic variable values are displayed.

In the fuzzy sets, *Xa* and the inverse transforms are performed. It is these rules that ensure the formalization of high-quality statements by experts in the formation of a problem area in the memory of IS. To move from qualitative descriptions to formalized ones, it is necessary to construct mappings included in *M*, i.e., to construct membership functions.

When receiving information from experts about the type of membership functions, it is necessary to take into account the nature of measurements (primary and derivative measurements) and the type of scale on which measurements are projected, and on which membership functions will be determined. On this scale, the form of admissible operators and operations is defined, i.e., some algebra for membership functions. In addition, it is necessary to distinguish between characteristics that can be measured directly and characteristics that are qualitative and require pairwise comparison of objects with these characteristics in order to determine their relation to the concept under study.

Two groups of methods for constructing membership functions can be distinguished: direct and indirect. In direct methods, the expert directly sets the rules for determining the values of the membership function.

In indirect methods, the values of the membership function are chosen in such a way that predefined conditions are satisfied. Expert information is only the source for further processing. Additional conditions may be imposed both on the type of information received and on the processing procedure. Examples of additional conditions include the following: membership function should reflect proximity to a pre-allocated standard, set objects are points in parametric space; the result of the processing procedure should be a membership function that satisfies the conditions of the interval scale; in a pairwise comparison of objects, if one object is estimated *k* times stronger than another, then the second object is estimated *1/k* times stronger than the first object, etc.

As a rule, direct methods are used to describe concepts that are characterized by measurable features (height, height, mass, volume).

In this case, it is convenient to directly specify the membership function. Direct methods include methods based on the probabilistic interpretation of membership functions: $a(u) == P(a / u)$, i.e., the probability that the object will belong to the set. Since people often distort estimates, for example, shift them towards the ends of the rating scale, direct measurements based on the direct determination of membership function values can be used only when such distortions are insignificant or unlikely. Indirect methods are more time-consuming than direct methods, but they are resistant to distortions in the answer. The result of using indirect methods is the interval scale. Limitations of the use of indirect methods are known (the condition of an unconditional extremum): when determining the degree of membership, the set of studied objects should contain at least two objects, the numerical representations of which on the interval [0; 1] - 0 and 1, respectively.

Membership functions can reflect both the opinion of a certain group of experts and the opinion of one unique expert. Combining the possible two methods of constructing membership functions with two types of experts (collective and unique), you can get four types of expertise.

When analyzing the situation, the expert reasons in the semantic space (the space of scales), in which the evaluated image corresponds to the situation. The semantic space is similar to the subjective space of sensations, in which an internal image of external signals is formed and subjective relationships arise between properties (signs, parameters). Depending on the individual perception, the same sign value can be evaluated in different ways. However, for a particular individual, the assessed situation is invariant regarding a certain class of situations. Therefore, when identifying the real values of attributes with a semantic image, the form of fuzzy mapping of the attribute space into the semantic space is essential.

The mapping of any situation to a unit interval occurs in such a way that the point of the interval characterizes the degree of manifestation of a certain property (0 corresponds to the absence of a property, 1 is the maximum manifestation of a property that interests us). When constructing the membership function, a measurement model is used, which is determined by two parameters: the type of membership scale on which information from the expert is displayed and the type of measurement (direct or indirect).

The process of formalizing the knowledge obtained from an expert consists of the following steps: choosing a method for measuring fuzziness, obtaining initial data by interviewing an expert, implementing an algorithm for constructing a membership function.

In the process of implementing the method, the following characteristics are used: type of measurement method (P – direct, K – indirect); interpretation of affiliation (FB – frequency probability, SP – subjective probability, B – possibility, D – determined); the procedure for obtaining initial data (PF – determination of the membership function in the form of formulas, 03-assignment of membership values, YN – yes-no assessment; EPO – evaluation of pairs of objects; R – ranking, AR-arranging pairs of objects, CP – comparison in pairs); measurements (F – fundamental, D – derivative); type of scale (N – nominal, P – ordinal, I – interval, O – relations, A – absolute).

QUESTIONS:

1. What are the forms of representation of knowledge.
2. What is the "field of knowledge"?
3. What are the modes of the interaction of a knowledge engineer with an expert specialist.
4. List interviewing strategies.
5. The essence of the strategy of the repertory grid.
6. The essence of the affinity confirmation strategy.
7. What are the stages of interaction of a knowledge engineer (analyst) with an expert specialist.
8. What is needed to create a linguistic interaction alliance?
9. Describe the methodology of working with an expert in the formation of a field of knowledge.

Chapter 3. Models and methods for solving problems

3.1. Classification of Representation of Tasks

Logical models

The formulation and solution of any problem are always associated with its "immersion" in a suitable subject area. Thus, solving the problem of buying products at the enterprise involves in the subject area such objects as the cash register, seller, time intervals, and general concepts of "seller", "purchase", "counter", etc.

All objects and events that form the basis of a common understanding of the information necessary to solve the problem are called the subject area. Mentally, the subject area appears to be composed of real or abstract objects called entities.

Entities of the subject area are in certain relations to each other (associations), which can be considered as entities and included in the subject area. Various similarities are observed between entities. The totality of such entities makes up the entity class, which is the new entity of the subject area.

Relations between entities are expressed through judgment.

Judgment is a mentally possible situation, which may or may not occur for the entities presented. In language (formal or natural) propositions correspond to judgments. Judgments and suggestions can be considered as entities and included in the subject area.

Languages designed to describe subject areas are called knowledge representation languages. The universal language for representing knowledge is natural language. However, the use of natural language in machine-based knowledge representation systems encounters great difficulties due to irregularities, ambiguities, etc. But the main obstacle is the lack of formal semantics of the natural language that would have sufficiently effective operational support.

Logical formalisms have long been used to represent mathematical knowledge in mathematical logic – mainly predicate calculus, which has clear formal semantics and operational support in the sense that inference mechanisms have been developed for it. Therefore, predicate calculus was the first logical language, which was used to formally describe subject areas related to solving problems.

Domain descriptions, which were made in logical languages, are called (formal) logical models.

A logical data model or logical schema is a data model of a specific problem domain expressed independently of a particular database management product or storage technology (physical data model) but in terms of data structures such as relational tables and columns, object-oriented classes, or XML tags. This is in opposition to a conceptual data model, which describes the semantics of an organization without reference to technology.

Logical data models represent the abstract structure of a domain of information. They are often diagrammatic in nature and are most typically used in business processes that seek to capture things of importance to an organization and how they relate to one another. Once validated and approved, the logical data model can become the basis of a physical data model and form the design of a database.

Logical data models should be based on the structures identified in a preceding conceptual data model, since this describes the semantics of the information context, which the logical model should also reflect. Even so, since the logical data model anticipates implementation on a specific computing system, the content of the logical data model is adjusted to achieve certain efficiencies.

The term 'Logical Data Model' is sometimes used as a synonym of 'domain model' or as an alternative to the domain model. While the two concepts are closely related and have overlapping goals, a domain model is more focused on capturing the concepts in the problem domain rather than the structure of the data associated with that domain.

A logical data model is sometimes incorrectly called a physical data model, which is not what the ANSI people had in mind. The physical design of a database involves deep use of particular database management technology.

Conceptual, logical, and physical data models are very different in their objectives, goals, and content (West M., Fowler J., 1999). Key differences noted below (tab. 3.1).

Table 3.1. The main differences from logical data models from conceptual and physical

Conceptual Data Model (CDM)	Logical Data Model (LDM)	Physical Data Model (PDM)
Includes high-level data constructs	Includes entities (tables), attributes (columns/fields) and relationships (keys)	Includes tables, columns, keys, data types, validation rules, database triggers, stored procedures, domains, and access constraints
Non-technical names, so that executives and managers at all levels can understand the data basis of Architectural Description	Uses business names for entities & attributes	Uses more defined and less generic specific names for tables and columns, such as abbreviated column names, limited by the database management system (DBMS) and any company defined standards
Uses general high-level data constructs from which Architectural Descriptions are created in non-technical terms	Is independent of technology (platform, DBMS)	Includes primary keys and indices for fast data access.
Represented in the DIV-1 Viewpoint (DoDAF V2.0)	Represented in the DIV-2 Viewpoint (DoDAF V2.0), and OV-7 View (DoDAF V1.5)	Represented in the DIV-3 Viewpoint (DoDAF V2.0), and SV-11 View (DoDAF V1.5)

When ANSI first laid out the idea of a logical schema in 1975 (American National Standards Institute, 1975), the choices were hierarchical and network. The relational model – where data is described in terms of tables and columns – had just been recognized as a data organization theory but no software existed to support that approach. Since that time, an object-oriented approach to data modeling – where data is described in terms of classes, attributes, and associations – has also been introduced.

Network models

The network model is a database model conceived as a flexible way of representing objects and their relationships. Its distinguishing feature is that the schema, viewed as a graph in which object types are nodes and relationship types are arcs, is not restricted to being a hierarchy or lattice.

The network model's original inventor was Charles W. Bachman (1973), and it was developed into a standard specification published in 1969 by the Conference on Data Systems Languages (CODASYL) Consortium. This was followed by a second publication in 1971, which became the basis for most implementations. Bachman's influence is recognized in the term Bachman diagram, a diagrammatic notation that represents a database schema expressed using the network model. In a Bachman diagram, named rectangles represent record types, and arrows represent one-to-many relationship types between records (CODASYL set types).

While the hierarchical database model structures data as a tree of records, with each record having one parent record and many children, the network model allows each record to have multiple parent and child records, forming a generalized graph structure. This property applies at two levels: the schema is a generalized graph of record types connected by relationship types (called "set types" in CODASYL), and the database itself is a generalized graph of record occurrences connected by relationships (CODASYL "sets"). Cycles are permitted at both levels.

The chief argument in favor of the network model, in comparison to the hierarchical model, was that it allowed more natural modeling of relationships between entities. Although the model was widely implemented and used, it failed to become dominant for two main reasons. Firstly, IBM chose to stick to the hierarchical model with semi-network extensions in their established products such as IMS and DL/I. Secondly, it was eventually displaced by the relational model, which offered a higher-level, more declarative interface. Until the early 1980s, the performance benefits of the low-level navigational interfaces offered by hierarchical and network databases were persuasive for many large-scale applications, but as hardware became faster, the extra productivity and flexibility of the relational model led to the gradual obsolescence of the network model in a corporate enterprise usage.

We introduce numerous definitions. By essence, we mean an object of arbitrary nature. This object can exist in the real world.

In this case, it will be called the P-entity. In the knowledge base, a certain description corresponds to it, the completeness of which is determined by the information that it has about the P-essence of IS. Such a representation in the knowledge base is called an M-entity. M-entities may exist for which there are no corresponding P-entities in the world surrounding IS. Such M-entities are abstract objects obtained as a result of operations such as generalization within the knowledge base.

In network models, the division allows the use of ideas first formulated in the theory of semiotic models and situational management based on them. Semiotic models of problem areas will be understood as a set of procedures which allow displaying P-entities and their relationships, recorded in the problem area by the knowledge engineer, in the aggregate of interconnected M-entities in the knowledge base. The method of interpreting interconnected M-entities will be called denotative semantics, and the method of interpreting interconnected M-entities will be called connotative semantics. The P-entity regarding to the corresponding M-entity is called the denotate or referent of this M-entity, and the M-entity regarding to the original P-entity is called its designate, name, label, identifier, etc. The designate is the simplest element in the network model. It is included in the class of terminal objects of the network model. A terminal object is called an M-entity, which cannot be decomposed into simpler entities. The remaining M-entities are called derived objects or derived M-entities.

The list of terminal objects, which form classes or types is specified during the design of IS. They can be integer real numbers, identifiers, strings, lists, etc. The semantics of terminal objects are determined by the set of valid procedures that operate with them. For example: arithmetic operations on numbers, comparing strings or identifiers among themselves, input-output operations, including the necessary transformation of representations, etc.

Product Models

Products and frames are the most popular means of representing knowledge in IS. Products, on the one hand, are close to logical models, which makes it possible to organize effective output procedures on them. On the other hand, reflect knowledge more clearly than classical logical models. They do not have stringent restrictions characteristic of logical calculi, which makes it possible to change the interpretation of production elements.

In general terms, production means an expression of the following form:

$$(i); Q; P; A => B; N.$$

Here is i-name of the product with which this product stands out from the entire set of products. The name may be some token that reflects the essence of the product (for example, "buying a book" or "dialing a lock code"), or the serial number of products in their set stored in the system’s memory.

Element Q is characterized by the scope of the product. Such spheres are easily distinguished in the cognitive structures of a human. Our knowledge is sort of "laid out on the shelves." On one "shelf" knowledge is stored on how to cook food, on the other – how to get to work, etc.

Dividing knowledge into separate areas saves time on finding the right knowledge. The same division into spheres in the IS knowledge base is also advisable when using production models to represent knowledge.

The main element of the production is core: A => B. The interpretation of the core of the product may be different and depends on what is left and right of the sequence sign =>. A typical reading of the product core looks like: IF A, THEN B, more complex core designs allow an alternative choice on the right side, for example, IF A, THEN B1, ELSE B2. The sequence can be interpreted in the usual logical sense as a sign of the logical following of B from true A (if A is not a true expression, then nothing can be said about B). Other interpretations of the core of the product are possible. For example, A describes some condition, which is necessary for the action B.

Element P is the condition for the applicability of the core of the product. Usually, P is a logical expression (usually a predicate). When P takes on the value "true", the core of the product is activated. If P is false, then the core of the product cannot be used. For example, if in the product «AVAILABLE MONEY; IF YOU WANT TO PURCHASE THING X, PAY THE COST AND GIVE THE SELLER INSPECTION» the condition of applicability of the core of the product is false, that is, there is no money, then it is impossible to apply the core of the product.

Element N describes post-production conditions. They are updated only if the core product is sold. Postconditions of products describe the actions and procedures that must be performed after the implementation of B. For example, after buying a certain item in a store, it is necessary to reduce the number of items of this type by one in the inventory of goods available in this store. N execution may not occur immediately after the implementation of the core products.

A manufacturer usually provides an identifier for each particular type of product they make, known as a model, model variant, or model number (often abbreviated as MN, M/N or model no.). For example, Dyson Ltd, a manufacturer of appliances (mainly vacuum cleaners), requires customers to identify their model in the support section of the website. Brand and model can be used together to identify products in the market. The model number is not necessarily the same as the manufacturer part number (MPN).

Because of the huge amount of similar products in the automotive industry, there is a special kind of defining a car with options (marks, attributes), that represent the characteristics features of the vehicle. A model of a car is defined by some basic options like body, engine, gearbox and axles. The variants of a model are built by some additional options like color, seats, wheels, mirrors, trims, entertainment and assistant systems, etc. Options, that exclude each other (pairwise) build an option-family. That means, that you can choose only one option from each family, and you have to choose exactly one option. This kind of product definition fulfills the requirements of an ideal Boolean Algebra and can be helpful to construct a product configurator. Occasionally, a set of options (car features) are combined to an automotive package and are offered at a lower price. A consistent car definition is essential for production planning and control in the automotive industry, to generate a master production schedule (Herlyn W., 2012), which is fundamental for enterprise resource planning.

Besides, a specific unit of a product is usually (and has to be) identified by a serial number, which is necessary to distinguish products with the same product definition. In case of automotive products, it is called the Vehicle Identification Number VIN, an international standardized format.

If a certain set of products is stored in the system's memory, then they form a system of products. In the product system, special product management procedures must be defined, with the help products are updated and the products selected from the list of actualized products are to be executed.

Numerous IS use a combination of network and production knowledge representation models. In such models, declarative knowledge is described in the network component of the model, and procedural knowledge is described in production. In this case, they talk about the work of the production system on the semantic network.

Scenarios

A special role in knowledge representation systems is played by stereotyped knowledge describing well-known standard situations of the real world. Such knowledge makes it possible to recover information missing in the description of the situation, to predict the emergence of new facts, which can be expected in this situation, to establish the meaning of the origin of the situation from the point of view of a more general situational context.

Various models are used to describe stereotypical knowledge. Among them, the most common are scripts. A scenario is a formalized description of a standard sequence of interrelated facts that determine a typical situation in a subject area. This can be a sequence of actions or procedures that describe how to achieve the goals of the characters in the script (for example, lunch at a restaurant, business trip, airplane flight, admission to college). In IS, scenarios are used in procedures for understanding natural language texts, planning behavior, learning, making decisions, managing environmental changes, etc.

Intelligence interface

We suppose that the input of the IS is a text and IS understands the text, if it gives answers that are correct from the point of view of a person to any questions related to what the text says.

By "human" we mean a specific expert person who is entrusted with evaluating the ability of the system to understand. This contributes to the share of subjectivity because different people can understand the same texts in different ways.

QUESTIONS:

1. Describe the essence of logical models.
2. What are the differences between logical data models and conceptual and physical?
3. Describe the essence of network models.
4. Describe the essence of production models.
5. What is a stereotyped knowledge?

3.2. CLASSIFICATION OF UNDERSTANDING LEVELS

Understanding is a psychological process related to an abstract or physical object, such as a person, situation, or message whereby one is able to think about it and use concepts to deal adequately with that object. Understanding is a relation between the knower and an object of understanding. Understanding implies abilities and dispositions regarding an object of knowledge that is sufficient to support intelligent behavior.

Someone who has a more sophisticated understanding, more predictively accurate understanding, and/or an understanding that allows them to make explanations that others commonly judge to be better, of something, is said to understand that thing "deeply".

Conversely, someone who has a more limited understanding of a thing is said to have a "shallow" understanding. However, the depth of understanding required to usefully participate in an occupation or activity may vary greatly.

Gregory Chaitin (2006), a noted computer scientist, propounds a view that comprehension is a kind of data compression/ In his essay "The Limits of Reason", he argues that understanding something means being able to figure out a simple set of rules that explains it. For example, we understand why day and night exist because we have a simple model – the rotation of the earth – that explains a tremendous amount of data – changes in brightness, temperature, and atmospheric composition of the earth. We have compressed a large amount of information by using a simple model that predicts it. Similarly, we understand the number 0.33333... by thinking of it as one-third. The first way of representing the number requires five concepts ("0", "decimal point", "3", "infinity", "infinity of 3"); but the second way can produce all the data of the first representation, but uses only three concepts ("1", "division", "3"). Chaitin argues that comprehension is this ability to compress data.

It is possible for a person, or a piece of "intelligent" software, that in reality only has a shallow understanding of a topic, to appear to have a deeper understanding than they actually do, when the right questions are asked of it. The most obvious way this can happen is by memorization of correct answers to known questions, but there are other, more subtle ways that a person or computer can (intentionally or otherwise) deceive somebody about their level of understanding, too. This is particularly a risk with artificial intelligence, in which the ability of a piece of artificial intelligence software to rapidly try out millions of possibilities (attempted solutions, theories, etc.) could create a misleading impression of the real depth of its understanding. Supposed that AI software could, in fact, come up with impressive answers to questions that were difficult for unaided humans to answer without really understanding the concepts at all, it could do this simply by dumbly applying rules very quickly. (However, see the Chinese room argument for a controversial philosophical extension of this argument.)

In existing IS, five main levels of understanding and two levels of meta-understanding can be distinguished.

The first level is characterized by a diagram showing the system forms any answers to questions based on direct content entered from the text. If, for example, the text is entered into the system: "At eight in the morning, after breakfast, Petya left for school. At two in the afternoon he returned home. After lunch, he went for a walk", then at the first level of understanding, the system must be able to answer correctly questions like: "When did Petya go to school?" or "What did Petya do after dinner?" In the linguistic processor, a morphological, syntactic, and semantic analysis of the text and questions related to it takes place. At the output of the linguistic processor, an internal representation of the text and questions the output unit can work with is obtained. Using special procedures, this block generates responses. In other words, understanding at the first level requires certain means from the IS to present data and output to this data.

At the **second level**, inference tools are added based on the information contained in the text. These are various text logics (temporal, spatial, causal, etc.). It can generate information that is clearly absent in the text. For example, at the second level, it is possible to form the correct answers to questions like: "What happened before: Petya's departure to school or his lunch?" or "Did Petya go for a walkPetya after returning from school?" Only by building a temporary text structure IS can answer questions. The IS scheme, in which a second level of understanding can be implemented, has another knowledge base. It stores the laws related to the time structure of events, their possible spatial organization, causal dependence, etc., and the logical unit has all the necessary tools for working with pseudophysical logics.

Third level. The rules of replenishing the text with knowledge of the system about the environment are added to the second-level tools. This knowledge in IS, as a rule, is logical and is recorded in the form of scenarios or procedures of a different type. At the third level of understanding, IS should give the right answers to questions like: "Where was Petya at ten in the morning?" or "where".

Petya returned at two in the afternoon?" For this, you need to know what the process of "staying at school" means and, in particular, that this process is continuous and that the subject participating in it is always at school.

The IS scheme, which implements the understanding of the third level, does not look different from the scheme of the second level. However, in the logical unit, means should be provided not only for purely deductive inference but also for inference according to scripts.

The three listed levels of understanding are implemented in all practically working IS. The first level and partially the second are included in a variety of natural language communication systems.

The following two levels of understanding are only partially implemented in existing IPs.

Fourth level. Instead of text, it uses extended text, which is generated only if there are two channels for obtaining information. First – when the text is transferred to the system, and the second – additional information that is not in the text. In human communication, the role of the second channel, as a rule, is played by vision. More than one communication channel has intelligent robots with vision.

The visual communication channel allows to record the state of the environment "here and now" and enter the observed information into the text. The system becomes capable of understanding texts in which words are entered that are directly related to the situation in which the text is generated. At lower levels of understanding it is impossible to understand, for example, the text: "Look what Petya did! He should not have taken it!" In the presence of the visual channel, the process of understanding becomes possible.

Within the fourth level of understanding, IP can answer questions like: "Why shouldn't Petya have to take this?" or "What did Petya do?" If the question received by the system corresponds to the third level, then the system gives the desired answer. If for the answer it is necessary to attract additional ("exegetical") information, then the internal presentation of the text and the question is transmitted to the block that correlates the text with the actual situation of its generation, which is available to the IS through the visual or some other channel of fixing the situation of the outside world.

Fifth level. In order to answer at this level, IS uses, in addition to text, information about a specific subject, which is the source of the text, and general information related to communication (knowledge about the organization of communication, about the goals of participants in communication, and norms for participation in the communication) stored in the system's memory. The theory corresponding to the fifth level is the so-called theory of speech acts.

Attention was drawn to the fact that any phrase not only indicates a certain phenomenon of reality but also combines three actions: locution, illocution and perlocution. Location is speaking as such, that is, those actions that the speaker performed to express his thought. Illocution is an act of speaking: question, motivation (order or request), and affirmation. Perlocution is the action by which the speaker tries to exert some influence on the listener: "flatter", "surprise", "persuade", etc. A speech act can be defined as the minimum meaningful (or appropriate) unit of speech behavior. Each speech act consists of a locative, illocutionary, and perlocutionary act.

For the fourth and fifth levels of understanding, the results on non-verbal (non-verbal) communication components and psychological principles underlying communication are interesting. Moreover, the rules of replenishment of the text include the rules of inference, based on knowledge about this particular subject of communication, if the system has such knowledge. For example, a system can trust a given subject, believing that the text it generates is true. But he may not trust him and understand the text, adjusting it under his knowledge of the subject that generated the text.

Knowledge of this type should be based on psychological theories of communication, which are not sufficiently developed yet.

For example, the system receives the text: Nina promised to come soon. "If the system does not have any information about Nina, it can turn to the knowledge base and use some normative information to evaluate the time index "soon". From this information, with a great deal of confidence, "soon" does not exceed half an hour. But the system may have special information about Nina, which was discussed in the input text. In this case, the system, having received the necessary information from the knowledge base, can prepare, for example, for the fact that Nina will most likely arrive no earlier than in an hour.

The first metalevel. At this level, the content of the knowledge base changes. It is supplemented by facts known to the system and contained in those texts that are entered into the system. Different IPs differ from each other in the nature of the rules for generating facts from knowledge.

For example, in systems intended for examination in the field of pharmacology, these rules are based on methods of inductive inference and pattern recognition. The rules can be based on the principles of probabilities, vague conclusions, etc. But in all cases, the knowledge base is a priori incomplete and in such information systems it is difficult to find answers to queries. In particular, a nonmonotonic conclusion becomes necessary in knowledge bases.

The second metalevel. At this level, metaphorical knowledge is generated. The rules for generating metaphorical knowledge used for purposes are special procedures based on the conclusion by analogy and association. Presently known analogy inference schemes use, as a rule, the Leibniz diagram, which reflects only a special case of reasoning by analogy. Associative reasoning schemes are even poorer.

If we consider the levels and metalevels of understanding from the point of view of the architecture of IS, can observe the sequential build up of new blocks and the complexity of the procedures they implement.

At the first level, a linguistic processor with a knowledge base related only to the text itself is sufficient.

At the second level, a logical inference procedure arises in this processor.

At the third level. The emergence of a new channel of information that works independently of the source characterizes the fourth level. In addition to the procedures associated with the operation of the channel, there are procedures that link the results of the two channels, integrating the information received for each of them.

At the fifth level, various ways of deriving knowledge and data are developed. At this level, models of individual and group behavior become important.

At the metalevels, new procedures are emerging for manipulating knowledge, which were not present at lower levels of understanding. And this process is open. Understanding in full is some seemingly unattainable dream. But understanding at the level of "everyday understanding" of people in IS is quite achievable.

There are other interpretations of the phenomenon of understanding. You can, for example, evaluate the level of understanding by the ability of the system to explain the result. Here, not only the level of explanation is possible when the system explains what it has done, for example, based on the text entered into it, but also the level of justification (argumentation), when the system justifies its result, showing that it does not contradict that system of knowledge and data, which she has. In contrast to the explanation, justification is always associated with the sum of facts and knowledge, which are determined by the current moment of the existence of the system. And the text introduced for understanding in some states can be perceived by the system as true, and in others – as false.

The constructive justification provides the linking of theoretical schemes to experiment, and hence the connection with the experience of the physical quantities of the mathematical apparatus of the theory. It is thanks to the procedures of constructive justification that the theory of conformity appears.

In addition to explanation and justification, another function related to the understanding of texts is possible – justification. The theory of justification is a part of epistemology that attempts to understand the justification of propositions and beliefs.

The subject of justification has played a major role in the value of knowledge as "justified true belief". Some contemporary epistemologists, such as Jonathan Kvanvig assert that justification is not necessary for getting to the truth and avoiding errors. Kvanvig attempts to show that knowledge is no more valuable than true belief, and in the process dismissed the necessity of justification due to justification not being connected to the truth.

Different theories of justification require different amounts and types of evidence before a belief can be considered justified. Theories of justification generally include other aspects of epistemology, such as knowledge.

Justifying something means that the statements made do not contradict the system of norms and values that are embedded in IP. Existing IS such as expert systems are able to provide explanations and only partially justifications. The full substantiation and justification procedures have not been implemented yet.

QUESTIONS:

1. What are the five main levels of understanding?
2. What are the levels of metaunderstanding?
3. Describe the essence of the first level of understanding.
4. Describe the essence of the second level of understanding.
5. Describe the essence of the third level of understanding.
6. Describe the essence of the fourth level of understanding.
7. Describe the essence of the fifth level of understanding.
8. How can one assess the level of understanding?

3.3. TASK PLANNING

Automated planning and scheduling, sometimes denoted as simply AI planning (Ghallab M., Nau D., Traverso P., 2004), is a branch of artificial intelligence that concerns the realization of strategies or action sequences, typically for execution by intelligent agents, autonomous robots and unmanned vehicles. Unlike classical control and classification problems, the solutions are complex and must be discovered and optimized in multidimensional space. Planning is also related to decision theory.

In known environments with available models, planning can be done offline. Solutions can be found and evaluated prior to execution. In dynamically unknown environments, the strategy often needs to be revised online. Models and policies must be adapted. Solutions usually resort to iterative trial and error processes commonly seen in artificial intelligence. These include dynamic programming, reinforcement learning, and combinatorial optimization. Languages used to describe planning and scheduling are often called action languages.

The functioning of IS is focused. A typical act of such functioning is to solve the problem of planning the path to achieve the desired goal from a fixed initial situation. The result of solving the problem should be an action plan - a partially-ordered set of actions. Such a plan resembles a scenario in which relations of the vertices are relations of the type: "goal-sub-goal", "goal-action", "action-result", etc. Any path in this scenario leading from the vertex corresponding to the current situation, any of the target vertices determines the action plan.

Given a description of the possible initial states of the world, a description of the desired goals, and a description of a set of possible actions, the planning problem is to synthesize a plan that is guaranteed (when applied to any of the initial states) to generate a state which contains the desired goals (such a state is called a goal state).

The difficulty of planning is dependent on the simplifying assumptions employed.

The search for an action plan arises in the IS only when it is faced with a non-standard situation, for which there is no previously known set of actions leading to the desired goal.

In AI planning, planners typically input a domain model (a description of a set of possible actions which model the domain) as well as the specific problem to be solved specified by the initial state and goal, in contrast to those in which there is no input domain specified. Such planners are called "domain independent" to emphasize the fact that they can solve planning problems from a wide range of domains. Typical examples of domains are block-stacking, logistics, workflow management, and robot task planning. Hence, a single domain-independent planner can be used to solve planning problems in all these various domains. On the other hand, a route planner is typical of a domain-specific planner.

All tasks of constructing an action plan can be divided into two types, which correspond to different models: planning in the state space (SS-problem) and planning in task spaces (PR-problem).

In the first case, a certain space of situations is given. Description of situations includes the state of the outside world and the state of IS, characterized by several parameters. Situations form some generalized states and IS actions or changes in the external environment lead to a change in the states actualized at the moment.

Among the generalized states, the initial states (usually one) and the final (target) states are distinguished. The SS-problem is finding a path leading from the initial state to one of the final ones. If the IS is intended for playing chess, then the generalized states will be the positions that develop on the chessboard. As the initial state, a position that is fixed at the moment of the game can be considered, and as a target position – a lot of draw positions. It should be noted that in the case of chess, a direct transfer of target positions is impossible. Matte and draw positions are described in a language different from the language for describing states characterized by the arrangement of figures on the board's margins. This is what makes it difficult to find an action plan in a chess game.

When planning in the task space, the situation is somewhat different.

The space is formed as a result of introducing relationships on the set of tasks such as: "part-whole", "task-subproblem", "general case-special case", etc. In other words, the task space reflects the decomposition of tasks into subproblems (goals on subgoals). The PR problem consists in finding a decomposition of the original problem into subproblems, leading to problems whose solution the system knows. For example, IS know how the values of *sinx & cosx* are calculated for any value of the argument and how the division operation is performed.

If the IS needs to calculate *tgx,* then the solution to the PR-problem will be the representation of this problem in the form of decomposition *tgx=a=sinx/cosx (exept of π:=π/2+kπ).*

Let us classify the methods used to solve SS- and PR-problems.

1. State planning

Representation of tasks in the state space involves specifying several descriptions: states, the set of operators, and their effects on transitions between states, target states. State descriptions can be strings of characters, vectors, two-dimensional arrays, trees, lists, etc.

Operators translate one state into another. Occasionally they are presented in the form of products A => B, meaning that state A is converted to state B.

The state space can be represented as a graph whose vertices are labeled by states, and arcs by operators. If some arc is directed from the vertex *ni*, to the vertex *nj*, when *ni*- is called the child and *nj*-parent vertices.

A sequence of vertices *ni1, ni2… nik,* in which each *ni* - is the daughter vertex for the vertex *ni-1, i = 2, ..., k* - is called the path of length *k* from the vertex *ni1* to the vertex *nik.*

Thus, the problem of finding a solution to the problem <A, B> when planning by state is presented as the problem of finding on the graph the paths from A to B. Typically, the graphs are not specified but are generated as needed. Blind and directional methods of finding a path are distinguished. The blind method has two types: in-depth search and searching in-breadth. When searching in-depth, each alternative is explored to the end, ignoring the other alternatives.

The method is bad for "tall" trees, since you can easily slip past the desired branch and spend a lot of effort researching the "empty" alternatives. When searching in-breadth at a fixed level, all alternatives are investigated, and only after this is the transition to the next level carried out. The method may turn out to be worse than the in-depth search method if in the graph all the paths leading to the target vertex are located at approximately the same depth. Both blind methods are time-consuming and therefore directed search methods are needed.

The method of branches and borders. From the unfinished paths formed in the search process, the shortest one is selected and extended by one step. The resulting new unfinished paths (there are as many as there are branches at a given vertex) are considered along with the old ones, and the shortest of them is again extended by one step. The process is repeated until the first peak is reached, the decision is remembered. Then from the remaining unfinished paths are excluded longer than the completed path, or equal to it, and the remaining ones are extended according to the same algorithm as long as their length is less than the completed path. As a result, either all unfinished paths are excluded, or a complete path is formed among them, shorter than the previously received one. The last path begins to play the role of a standard, etc.

Moore Shortest Path Algorithm. The initial vertex *x0* is marked with the number *0*. Let the set of daughter vertices *Γ (xi)* of the vertex xi be obtained in the course of the algorithm at the current step. Then all the vertices obtained earlier are deleted from it, the remaining vertices are marked with a label increased by one compared to the vertex label *xi*, and pointers to *xi* are drawn from them. Next, on the set of marked vertices that are not yet listed as pointer addresses, the vertex with the lowest label is selected and daughter vertices are constructed for it. Vertex marking is repeated until a target vertex is obtained.

Dijkstra's algorithm for determining paths with minimal cost is a generalization of Moore's algorithm by adding arcs of variable length.

Doran and Michi low-cost search algorithm. It is used when the cost of searching is high compared to the cost of the optimal solution. In this case, instead of selecting the vertices that are least distant from the beginning, as in the Moore and Dijkstra algorithms, a vertex is selected for which the heuristic estimate of the distance to the target is the smallest. With a good assessment, you can quickly get a solution, but there is no guarantee that the path will be minimal.

Algorithm of Hart, Nilson and Raphael. The algorithm combines both criteria: the cost of the path to the top g (x) and the cost of the path from the top *h(x)* – in additive evaluation function;

$f(x) = g(x) + h(x)$. Provided that $h(x)<hp(x)$, where $hp(x)$- the actual distance to the target, the algorithm guarantees finding the optimal path.

2. Task planning

This method leads to good results because often problem-solving has a hierarchical structure. However, it is not necessary to require the main task and all subtasks to be solved by the same methods. Reduction is useful for representing the global aspects of the problem, and when solving more specific problems, the state planning method is preferable. The state planning method can be considered as a special case of the planning method using reductions because each application of the operator in the state space means reducing the original problem to two simpler, one of which is elementary. In the general case, the reduction of the original problem cannot be reduced to the formation of two subproblems of which at least one was elementary.

The search for planning in the task space consists in sequentially reducing the original problem to more and more simple until only elementary problems are obtained. A partially ordered set of such tasks will make up the solution to the original problem. It is convenient to represent the partition of the problem into alternative sets of subproblems in the form of an AND/OR graph. In such a graph, every vertex, except the terminal one, has either conjunctively connected daughter vertices (AND is a vertex) or disjunctively connected (OR is a vertex). In the particular case, in the absence of And – vertices, there is a graph of the state space. End vertices are either final (they correspond to elementary problems) or dead ends. The initial vertex (the root of the AND/OR graph) represents the original problem. The purpose of the search on the AND/OR graph is to show that the initial vertex is solvable.

Solvable are the final vertices (AND – vertices) for which all daughter vertices are solvable, and OR – vertices for which at least one daughter vertex is solvable. The resolving graph consists of decidable vertices and indicates the decidability of the initial vertex. The presence of dead ends leads to unsolvable vertices. Insoluble are dead-end vertices, AND are vertices for which at least one daughter vertex is unsolvable, and OR are vertices for which each daughter vertex is unsolvable.

The Cheng and Sleig algorithm is based on the conversion of an arbitrary AND/OR – graph into a special OR – graph, each OR – branch of which has AND – vertices only at the end. The transformation uses the representation of an arbitrary AND/OR graph as an arbitrary formula of propositional logic with the further transformation of this arbitrary formula into a disjunctive normal form. Such a transformation allows further use of the Hart, Nielson and Raphael algorithm.

Key Operator Method. Let task <A, B> given, and it is known that the operator f must be included in the solution of this problem. Such an operator is called a key. Let the state C be necessary for the application of f, and the result of the application is f(c). Then A –

the vertex <A, B> generates three daughter vertices: <A, C>, <C, f (c}> and <f (c), B>, of which the middle is an elementary problem. C> and <f (c), B> key operators are also selected, and the indicated reduction procedure is repeated as long as possible. As a result, the original problem <A, B> is divided into an ordered set of subtasks, each of which is solved by the planning method in the state space.

There are alternatives for choosing key operators, so in general, there will be an AND/OR graph. In most tasks, it is not possible to select the key operator, but only to specify the set containing it. In case, for the problem <A, B>, the difference between A and B is calculated and associated with an operator that eliminates this difference. The latter is the key.

Method for planning a common problem solver (CPS). CPS was the first most famous planner model. It was used to solve the problems of integral calculus, inference, grammar analysis, and others. CPS combines two basic principles of search: analysis of goals and means and recursive problem-solving. In each search cycle, CPS solves three types of standard tasks in a rigid sequence: convert object A to object B, reduce the difference D between A, apply the operator f to object A. Solving the first problem determines the difference D, the second is the suitable operator f, and the third is the required condition application of C. If C does not differ from A, then the operator f is applied, otherwise C is presented as the next target and the cycle repeats, starting with the task of "converting A to C". In general, the ARI strategy performs a reverse search – from a given goal B to the required means of achieving it C, using the reduction of the original problem <A, B> to tasks <A, C> and <C, B>. It should be noted that the CPS assumes the independence of differences from each other, which implies a guarantee that the reduction of some differences will not lead to an increase in others.

3. Planning using inference

Such planning involves: a description of states in the form of correctly constructed formulas (CCF) of some logical calculus, a description of operators in the form of either CCF, or rules for transferring some CCF to others. Representation of operators in the form of CCF allows you to create deductive planning methods, the representation of operators in the form of translation rules – planning methods with elements of deductive inference.

The deductive method of planning the QAS system, CPS did not live up to its expectations mainly due to the poor presentation of tasks. An attempt to rectify the situation led to the creation of the QAS question and answer system. The system is designed for an arbitrary subject area and is capable of answering the question by inference: is it possible to achieve state B from A? The principle of resolutions is used as a method of automatic output. To guide the logical conclusion, QAS uses various strategies, mainly of a syntactic nature, taking into account the formalism of the principle of resolutions. The operation of QAS showed that the conclusion in such a system is slow, detailed, which is unusual for human reasoning.

STRIPS system production method. The most commonly used languages for representing planning domains and specific planning problems, such as STRIPS and PDDL for Classical Planning, are based on state variables. In this method, the operator represents the products P, A => B, where P, A and B-sets of CCF are first-order predicate calculus, P expresses the conditions for using the product core A => B, where

B contains a list of added CCF and a list of excluded CCF postconditions. The method repeats the CPS method with the difference that the standard tasks of determining differences and applying suitable operators are solved based of the principle of resolutions. A suitable operator is selected in the same way as in the CPS based on the principle of "analysis of means and goals". The presence of a combined method of planning made it possible to limit the process of logical inference to a description of the state of the world, and leave the heuristic process of generating new descriptions "from the goal to the means of achieving it".

Each possible state of the world is an assignment of values to the state variables, and actions determine how the values of the state variables change when that action is taken. Since a set of state variables induce a state space that has a size that is exponential in the set, planning, similarly to many other computational problems, suffers from the curse of dimensionalityality and the combinatorial explosion.

An alternative language for describing planning problems is that of **hierarchical task networks**, in which a set of tasks is given, and each task can be either realized by a primitive action or decomposed into a set of other tasks.

This does not necessarily involve state variables, although in more realistic applications state variables simplify the description of task networks.

The method of the hierarchical production system of the solver ABSTRIPS. In this method, the search space is divided into hierarchy levels using the drilldown of products used in the STRIPS method. Each PPF letter included in the set P of product application conditions is assigned a weight *j, j = 0, k*, and at the *i-th* planning level carried out by the STRIPS method, only letters j of weight is taken into account. Thus, at the *k-th* level, the products are described in the least detail, at the zero-most detail as in the method of the STRIPS system. Such a partition allows for planning at the jth level to use the solution of the *(j + 1)* level as the skeleton of the solution of the *j-th* level, which increases the overall search efficiency.

A production method using macrooperators. Macrooperators are generalized solutions to problems obtained by the STRIPS method. The use of macrooperators makes it possible to reduce the search for a solution, however, this raises the problem of simplifying the macrooperator used, the essence is to select the required part for a given difference and to exclude unnecessary operators from the latter.

Newell and Simon's Advanced Planning Method based on the idea of further improving the CPS method: the first problem is solved in a simplified (due to ranking of differences) planning area, and then an attempt is made to clarify the solution in relation to a more detailed, initial problem area.

Conditional planning. Deterministic planning was introduced with the STRIPS planning system, which is a hierarchical planner. Action names are ordered in a sequence and this is a plan for the robot. Hierarchical planning can be compared with an automatically generated behavior tree (Neufeld, X. etc., 2017). The disadvantage is, that a normal behavior tree is not so expressive as a computer program. That means, the notation of a behavior graph contains action commands, but no loops or if-then-statements. Conditional planning overcomes the bottleneck and introduces an elaborated notation which is similar to a control flow, known from other programming languages like Pascal. It is very similar to program synthesis, that means a planner generates a sourcecode which can be executed by an interpreter (Sanelli V. etc., 2017).

An early example of a conditional planner is "Warplan-C" which was introduced in the mid-1970s (Peot M., Smith D., 1992). What is the difference between a normal sequence and a complicated plan, which contains if-then-statements? It has to do with uncertainty at the runtime of a plan. The idea is that a plan can react to sensor signals which are unknown to the planner. The planner generates two choices in advance. For example, if an object was detected, then action A is executed, if an object is missing, then action B is executed (Karlsson L., 2001). A major advantage of conditional planning is the ability to handle partial plans (Liu Daphne Hao, 2008). An agent is not forced to plan everything from start to finish but can divide the problem into chunks. This helps to reduce the state space and solves much more complex problems.

We speak of "contingent planning" when the environment is observable through sensors, which can be faulty. It is thus a situation where the planning agent acts under incomplete information. For a contingent planning problem, a plan is no longer a sequence of actions but a decision tree because each step of the plan is represented by a set of states rather than a single perfectly observable state, as in the case of classical planning (Alexandre A., Hector P., Hector G., 2009). The selected actions depend on the state of the system. For example, if it rains, the agent chooses to take the umbrella, and if it doesn't, they may choose not to take it.

Mikael L. Littman showed in 1998 that with branching actions, the planning problem becomes EXPTIME-complete (Littman M., 1997; Jussi R., 2004). A particular case of contiguous planning is represented by FOND problems – for "fully-observable and non-deterministic". If the goal is specified in LTLf (linear time logic on finite trace) then the problem is always EXPTIME-complete (De Giacomo G., Rubin S., 2018) and 2EXPTIME-complete if the goal is specified with LDLf.

Conformant planning. Conformant planning is when the agent is uncertain about the state of the system, and it cannot make any observations. The agent then has beliefs about the real world, but cannot verify them with sensing actions, for instance. These problems are solved by techniques similar to those of classical planning (Palacios H. Geffner H., 2009; Albore A.; Ramírez M., Geffner H., 2011), but where the state space is exponential in the size of the problem, because of the uncertainty about the current state. A solution for a conformant planning problem is a sequence of actions. Haslum and Jonsson have demonstrated that the problem of conformant planning is EXPSPACE-complete (Haslum P., Jonsson P., 2000) and 2EXPTIME-complete when the initial situation is uncertain, and there is non-determinism in the outcomes of the action (Littman M., 1997).

QUESTIONS:

1. What is the state space planning (SS problem)?
2. What is the task of space planning (PR problem)?
3. Give a classification of the methods, which are used to solve SS- and PR-problems.
4. Explain what are the blind and directional methods for finding a solution.
5. The essence of the branch and bound method.
6. The essence of the Moore Shortest Path Algorithm.
7. The essence of Dijkstra's Algorithm.
8. The essence of the Doran and Michi Algorithm.
9. The essence of the Algorithm of Hart, Nilson and Raphael.
10. What is task planning?
11. What is inference planning?

Chapter 4. Neurons and artificial neural networks

4.1. The history of neural networks

4.2. The essence of neural networks, and the specifics of their work

4.3. Classification of Neural Networks

4.4. Neural Network training

4.5. The use of Neural Networks

4.1. THE HISTORY OF NEURAL NETWORKS

The history of neural networks begins from the time when people began to be interested in their own thinking. This "thinking" of the brain is the hallmark of man. There are many reflections on the nature of thinking, ranging from spiritual to anatomical. The discussion of this issue, which took place in the heated debates of philosophers and theologians with physiologists and anatomists, was of little use because the subject itself is very difficult to study. Those who relied on introspection and reflection came to conclusions that did not meet the level of severity of the physical sciences. The experimenters found that the brain is difficult to observe and confuses its organization. Powerful methods of scientific research, which changed our view of physical reality turned out to be powerless in understanding the person himself.

The study and use of artificial neural networks, began a long time ago – at the beginning of the 20th century, but they got vast popularity a little later. This is due, first of all, to the fact that advanced (for that time) computing devices began to appear, the capacities of which were large enough to work with artificial neural networks. In fact, at the moment, you can easily simulate a neural network of medium complexity on any personal computer.

The main stages in the history of the study and application of artificial neural networks:

- 1943 McCulloch Warren; Walter Pitts (1943) formalize the concept of a neural network in a fundamental article on the logical calculus of ideas and nervous activity (McCulloch, Warren, 1961).
- 1948 Norbert Wiener together with associates publishes work on cybernetics. The main idea is the presentation of complex biological processes by mathematical models. Norbert Wiener's merit lies in the fact that he combined many of the available works on the general theory of control, designated and gave the name of a new scientific field – cybernetics (Levinson, 1966).
- 1949 Donald O. Hebb (1949) offers the first learning algorithm.
- B. Rosenblatt Frank (1958) invents a single-layer perceptron and demonstrates its ability to solve classification problems. The perceptron has gained popularity – it is used for pattern recognition, weather forecasting, etc.
- 1960 Bernard Widrow professor of electrical engineering at Stanford University. He is the co-inventor of the Widrow-Hoff least mean squares filter (LMS) adaptive algorithm with his doctoral student Ted Hoff [3]. The LMS algorithm led to the ADALINE and MADALINE artificial neural networks and to the backpropagation technique. He made other fundamental contributions to the development of signal processing in the fields of geophysics, adaptive antennas, and adaptive filtering (Abend Kenneth, 2002).
- 1969 Marvin Lee Minsky publishes a formal proof of the limitedness of the perceptron and shows that it is unable to solve some problems (the problem of "parity" and "one in a block") associated with the invariance of representations. Interest in neural networks drops sharply (Horgan J., 1993).
- 1972 Kohonen T. (1988; 2001), and Anderson John R. independently offer a new type of neural network that can function as a memory (Anderson John R. etc., 1998).

- 1974 Paul John. He is best known for his 1974 dissertation, which first described the process of training artificial neural networks through backpropagation of errors (Werbos Paul J., 1994). He also was a pioneer of recurrent neural networks (Werbos P., 1990). Werbos was one of the original three two-year Presidents of the International Neural Network Society (INNS).
- 1975 Kunihiko Fukushima представляет когнитрон – a self-organizing network designed for invariant pattern recognition, but this is achieved only by storing almost all image states.
- 1980 Fukushima published the neocognitron Fukushima K. (1980) the original deep convolutional neural network (CNN) architecture (Schmidhuber J., 2015). Fukushima proposed several supervised and unsupervised learning algorithms to train the parameters of a deep neocognitron such that it could learn internal representations of incoming data (Fukushima, K., 2007).
- 1982 after a period of oblivion, interest in neural networks increases again. John Joseph Hopfield (1982, 1984) showed that a feedback neural network can be a system that minimizes energy (the so-called Hopfield network). Neural networks and physical systems with emergent collective computational abilities (1982) (known as the Hopfield Network).
- Kohonen T. (1988; 2001) presents models of a network that learns without a teacher (Kohonen neural network), solves the problems of clustering, data visualization (Kohonen self-organizing map) and other tasks of preliminary data analysis.
- 1985 – John Joseph Hopfield and, with D.W. Tank (1985), "Neural computation of decisions in optimization problems".
- 1986 Rumelhart D.E., Hinton G.E., Williams R.J., (1986) independently and simultaneously with S.I. Bartsev and V.A. Okhonin (1986) (Krasnoyarsk group), the method of backpropagation of error is substantially developed. This is an iterative gradient algorithm that is used to minimize the error of the multilayer perceptron and obtain the desired output. The main idea of this method is to propagate error signals from the network outputs to inputs, in the direction opposite to the direct propagation of signals in normal operation (Bartsev and Okhonin, 1986). Bartsev S.I., Gilev S.E., Okhonin V.A., (1989), proposed a general method (the "duality principle") applicable to a wider class of systems, including delayed systems, distributed systems, etc.
- 2003 MacKay David (2003) to prove that apparently dissimilar methods can be considered special cases in the framework of the general mathematical model. Many types of artificial neural networks can be considered as classifiers that perform certain statistical calculations (maximum likelihood estimation).
- 2007 Geoffrey Hinton University of Toronto created deep learning algorithms for multilayer neural networks. The success is due to the fact that Hinton used the RBM – Restricted Boltzmann Machine, which he invented with Terry Sejnowski, to train the multilayer neural network while training the lower layers. The Boltzmann machine can be considered as a stochastic generative version of the John Joseph Hopfield network (1982).
- 2009 IBM (IBM achieves accurate brain simulation) demonstrated a successfully modeled cat brain. True, that his work was 643 times slower than real-time.

- 2011 IBM (DARPA SyNAPSE Program) created a neural processor that contained 256 neurons and 262,144 synapses.
- 2014 – IBM as part of the program SyNAPSE (Systems of Neuromorphic Adaptive Plastic Scalable Electronics; introduced a neurosynaptic processor for architecture TrueNorth. (Robert F., 2014).

In the first half of 2016, the world heard about many developments in the field of neural networks – their algorithms showed Google (network player AlphaGo), Microsoft (several services for image identification), startups MSQRD, Prisma etc.

Abilities of own neural networks were demonstrated in Google (in March 2016, the corporation auctioned 29 paintings by neural networks), and Microsoft (project CaptionBot, recognizes images in pictures and automatically generates captions to them; project WhatDog, determines the breed of the dog from the photograph; HowOld service, which determines the age of a person in the picture). Such entertainment services are created not to solve global problems that neural networks are aimed at, but to demonstrate the capabilities of a neural network and conduct its training.

As in many cases, tasks of high complexity require the use of not one, but several methods of solution or their synthesis. No exception artificial neural networks. From the very beginning of this century, the work of various researchers actively describes neuro-fuzzy networks, cell-neural network models. Neural networks are also used, for example, to configure the parameters of fuzzy control systems, and successfully used for the synthesis of control systems for dynamic objects.

Algorithms of neural networks are widely used in economics. Using neural networks, the task of developing algorithms for finding an analytical description of the laws governing the functioning of economic objects (enterprise, industry, region) is solved. These algorithms are applied to forecasting some "output" indicators of objects. The use of neural network methods allows us to solve some problems of economic and statistical modeling, increase the adequacy of mathematical models, to get it closer to economic reality. As economic, financial, and social systems are very complex and the result of human actions and counteractions, the creation of a complete mathematical model, taking into account all possible actions and counteractions, is a very difficult (if solvable) task. In systems of such complexity, it is natural and most effective to use models that directly mimic the behavior of society and the economy. This is what the methodology of neural networks can offer.

Artificial neural networks are used in various fields of life. Especially often in unmanned vehicles, applications for PC and mobile devices, in search engines, and so on. Many text and voice recognition programs work based on neural networks. Neural networks start to be used by neurospecialists for a deeper and more thorough study of not only human intelligence, but also the mind itself. In recent years, artificial neural networks can be found in medicine and especially in the field of information technology.

In general, there is no doubt about the further integration of artificial intelligence methods among themselves and with other methods of solving problems.

QUESTIONS:

1. What are the main stages in the history of the study and application of Artificial Neural Networks?
2. Give an example of developments in the field of Neural Networks.
3. What are the areas of application of Neural Networks?

4.2. THE ESSENCE OF NEURAL NETWORKS, AND THE SPECIFICS OF THEIR WORK

Neural networks are one of the directions in the development of Artificial Intelligence Systems. The idea is to simulate as closely as possible the work of the human nervous system, ability to learn and correct mistakes. The main feature of any neural network – it is able to independently learn and act based on previous experience, making fewer mistakes.

A neural network imitates not only activity but also the structure of the human nervous system. The network is a collection of neurons connected to each other in a certain way. Consider one neuron is shown in Figure 4.1.

In most cases, each "neuron" refers to a particular network level. Input data is processed sequentially on all levels of the network. The parameters of each "neuron" can vary depending on the results obtained on previous sets of input data, thus changing the order of operation of the system.

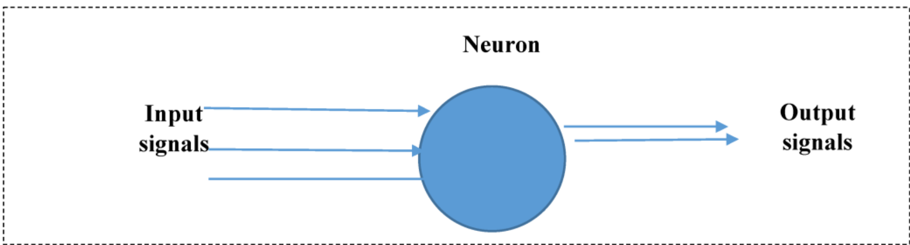

Figure 4.1. Schematic representation of a neuron

A neuron is an element that calculates the output signal (according to a certain rule) from a set of input signals. The basic sequence of actions of one neuron is as follows:

- receiving signals from previous network elements;
- combination of input signals;
- calculation of the output signal;
- transmission of the output signal to the following elements of the neural network.

Neurons can be connected in completely different ways, this is determined by the structure of a particular network. But the essence of the neural network is always the same. Based on the totality of the signals entering the network input, an output signal (or several output signals) is generated at the output. A neuron calculates the weighted sum of the input signals, and then converts the resulting sum using a given non-linear function. A set consisting of a threshold level and all weights is called neuron parameters. First, the neuron calculates the weighted sum and then, using the activation function, it calculates the output signal.

The neuron activation function is a function that calculates the output signal of a neuron. The sum of all the products of the signals and the weights of these signals are fed to the input of this function.

Neuron Models:

- **Perceptron.** This artificial neuron (IN) model was proposed in 1943, also called the McCulloch-Pitts model. In this model, a neuron is considered a binary element.
- **Sigmoid neuron.** A neuron of this type eliminates the main disadvantage of the perceptron – the discontinuity of the activation function.
- **Instar Grossberg.** Instar features that distinguish it from neurons of the previously considered types are the following:
  - ➢ the activation function f (ui) f is often linear, i.e., yi = ui;
  - ➢ the input vector X is normalized so that its Euclidean norm is 1;
  - ➢ instance training is possible both with a teacher and without him.
- **Neurons like WTA** (Winner Takes All - the winner gets everything) are always used by groups in which they compete with each other.
- **Neuron Hebb, D.O. Hebb, D.** (1949). Studying the behavior of natural nerve cells, he recorded an increase in the bond of two interacting cells with their simultaneous excitation. Neuron training can be done with or without a teacher. A feature of the Hebb rule is the ability to achieve a weight of large value due to the multiple summations of the increment in the training cycles.
- **Radial neuron.** Neurons of this type are significantly different from previously considered. They are used only in groups, making up the first level in multilayer radial networks.

Consider the most commonly used activation functions:

a) **Threshold function.** This is a simple piecewise linear function. If the input value is less than the threshold, then the value of the activation function is equal to the minimum acceptable, otherwise – the maximum allowed.

b) **Linear threshold.** This is a simple piecewise linear function. It has two linear sections, where the activation function is identically equal to the minimum and maximum allowable value and there is a section in which the function strictly monotonously increases.

c) **Sigmoid function or sigmoid.** This is a monotonically increasing differentiable S-shaped nonlinear function. Sigmoid allows you to amplify weak signals and not be saturated from strong signals.

d) **Hyperbolic tangent.** This function takes an arbitrary real number at the input, and at the output gives a real number in the range from –1 to 1. Like a sigmoid, a hyperbolic tangent can be saturated. However, unlike the sigmoid, the output of this function is centered around zero.

A neural network can be simplified as the form of a black box that has inputs and outputs. And inside this box, a huge number of neurons are the main stages of the network. Since each neuron can receive several input signals, when modeling a neural network, it is necessary to set a certain rule for combining all these signals. And quite often the rule of summing the weighted values of relationships is used.

Each connection in a network of neurons can characterize using three factors:

- the first is the element from which the connection originates;
- the second is the element to which the connection is directed;
- the third is the weight of the connection.

Particular attention is given to the third factor. The weight of the link determines whether the signal transmitted over that link will be amplified or attenuated. Consider this example, the output signal of neuron 1 is 5. The weight of the connection between neurons is 2. Thus, to determine the input signal of neuron 2 coming from neuron 1, it is necessary to multiply the value of this signal by the weight of the connection (5 * 2).

And if there are a lot of signals, they all add up. As a result, at the input of a neuron, we get the following:

$$net_j = \sum_{i=1}^{N} x_i * w_{ij}$$

net_j – this is the result of combining all input signals for neuron j (combined input of a neuron);

N – the number of elements transmitting their output signals to the input of signal j;

w_{ij} – the weight of the connection connecting neuron i to neuron j.

Summing up all the weighted input signals, we get the combined input of the network element. Most often, the structure of connections between neurons is represented in the form of a matrix W, which is called a weight matrix.

The matrix element *w*ij, as in the formula, determines the weight of the connection going from element i to element j. A simple neural network is shown in Figure 4.2.

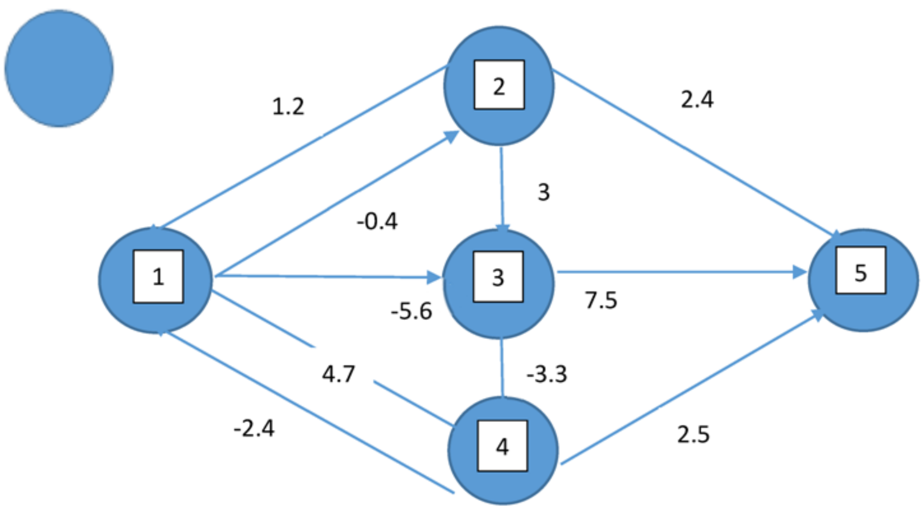

Figure 4.2. An example of a neural network

The weight matrix of a neural network will have the following form:

$$W = \begin{bmatrix} 0 & -0.4 & -5.6 & 4.7 \\ 1.2 & 0 & 3 & 0 \\ 0 & 0 & 0 & -3.3 \\ 2.4 & 0 & 0 & 0 \end{bmatrix}$$

For example, from the second element to the third there is a connection, the weight of which is 3. We look at the matrix, the second row, the third column – the number 3, that's right.

Consider the output signals. For each network element, there is a certain rule according to which its output value is calculated from the value of the combined input of the element. This rule is called the activation function. And the output value itself is called neuron activity. Absolutely any mathematical functions can play the role of activation functions.

Let us cite as an example several of the most frequently used ones:

- threshold function – if the value of the combined input is lower than a certain value (threshold), then the activity is zero, if it is higher, it is one.
- logistic function.

Consider another example, which is frequently used in the literature to explain the essence of the work of neural networks.

The task of the example is to calculate the ratio using a neural network XOR is shown in figure 4.3. That is, we will submit different types of signals to the input and the output should be the result of the XOR operation for the values supplied to the input:

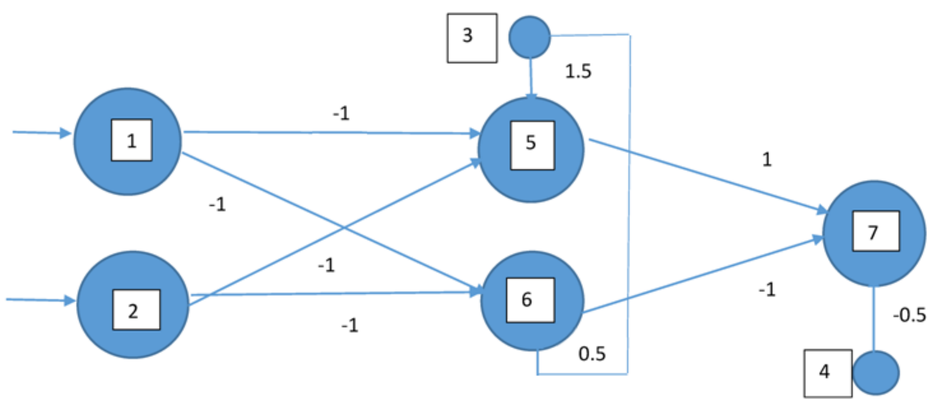

Figure 4.3. Relationship Calculation Example XOR

Elements 1 and 2 are input and element 7 is output. Neurons 5 and 6 are called hidden, since they are not connected with the external environment. Thus, three layers are obtained – input, hidden and output. Elements 3 and 4 are called offset elements. Their output signal (activity) is always equal to 1.

To calculate the combined input in this network, the rule of summing the weighted links is used, and the threshold function will act as a function of activity. If the combined input of an element is less than 0, then the activity is 0, if the input is more than 0, then the activity is 1.

We apply one to the input of neuron 1, and zero to the input of neuron 2. In this case, the output is 1 (0 XOR 1 = 1). We calculate the output value manually to demonstrate the operation of the network:

The combined input of element 5: net5 = 1 * (-1) + 0 * (-1) + 1 * 1.5 = 0.5.

The activity of the element is 5: 1 (0.5> 0).

The combined input of element 6: net6 = 1 * (-1) + 0 * (-1) + 1 * 0.5 = -0.5.

The activity of the element is 6: 0.

The combined input of element 7: net7 = 1 * (1) + 0 * (-1) + 1 * (-0.5) = 0.5.

The activity of element 7, and at the same time, the output value of the network is 1.

All possible values can be used as input signals (0 and 0, 1 and 0, 0 and 1, 1 and 1), the output will always show a value that corresponds to the table of the operation XOR.

In this case, all the values of the weight coefficients were known in advance, but the main feature of neural networks is that they can themselves adjust the weight values of all the connections in the process of learning the network.

QUESTIONS:

1. Define Neural Networks.
2. What is a neuron?
3. What is the basic sequence of actions of a neuron?
4. What are the models of neurons.
5. What are the most commonly used activation functions.
6. What mathematical functions can act as activation functions?

4.3. CLASSIFICATION OF NEURAL NETWORKS.

The classification of neural networks by the nature of training is divided into:

- neural networks using training with a teacher;
- neural networks using teacherless learning.

Neural networks using teacher training. Learning with a teacher assumes that for each input vector there is a target vector representing the desired output. Together they are called a training pair. Typically, the network is trained on several training pairs. An output vector is presented, the network output is calculated and compared with the correspon-ding target vector. Further, the weights are changed under the algorithm that seeks to minimize error. The vectors of the training set are presented sequentially, the errors are calculated, and the weights are adjusted for each vector until the error over the entire training array reaches an acceptable level.

Neural networks using teacherless learning. Learning without a teacher is a much more plausible model of learning in terms of the biological roots of Artificial Neural Networks.

It was developed by Kohonen T. (2001) and many others, it does not need a target vector for outputs and does not require comparison with predefined ideal answers. The training set consists only of input vectors. The training algorithm adjusts the weights, so that consistent output vectors are obtained, i.e., the presentation of sufficiently close input vectors gives the same outputs. The learning process identifies the statistical properties of the training set and groups similar vectors into classes.

Classification of Neural Networks **by the type of adjustment of weights** divides them into:

- networks with fixed connections – the weights of the neural network are selected immediately based on the conditions of the problem;
- networks with dynamic connections – during the training process, the synaptic scales are adjusted.

Classification of neural networks **by type of input information** divides them into:

- analog – input information is presented in the form of real numbers;
- binary – all input information is presented in the form of zeros and ones.

In fully connected Neural Networks is shown in figure 4.4, where each neuron transmits an output signal to other neurons, including to itself. All input signals are supplied to all neurons. The output signals of the network can be all or some output signals of neurons after several cycles of the network.

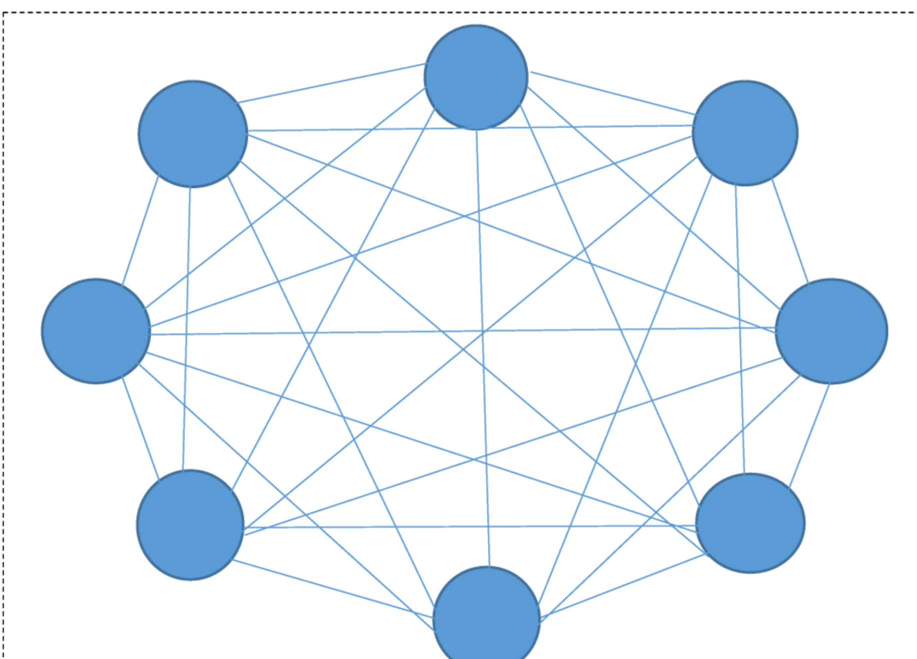

Figure 4.4. Full Neural Network Diagram

In multilayer (layered) Neural Networks, neurons are combined into layers is shown in figure 4.5. The layer contains a collection of neurons with single input signals.

The number of neurons in a layer can be any and does not depend on the number of neurons in other layers. In general, a network consists of layers numbered from left to right. External input signals are fed to the inputs of the neurons of the input layer (it is often numbered as zero), and the outputs of the network are the output signals of the last layer. In addition to the input and output layers, a multilayer neural network has one or more hidden layers. Connections from the outputs of neurons of a certain layer q to the inputs of neurons of the next layer $(q + 1)$ are called sequential.

Types of Multilayer Neural Networks

Monotonous. This is a special case of layered networks with additional conditions for communications and neurons. Each layer, except for the last (output) is divided into two blocks: exciting and inhibitory. The connections between the blocks are also divided into inhibitory and exciting. If excitatory connections lead from block neurons to block neurons, it means that any output signal of the block is a monotonic non-decreasing function of any output signal of the block. If these connections are only braking, then any output signal of the block is a non-increasing function of any output signal of the block. For neurons of monotonic networks, a monotonic dependence of the output signal of a neuron on the parameters of the input signals is necessary.

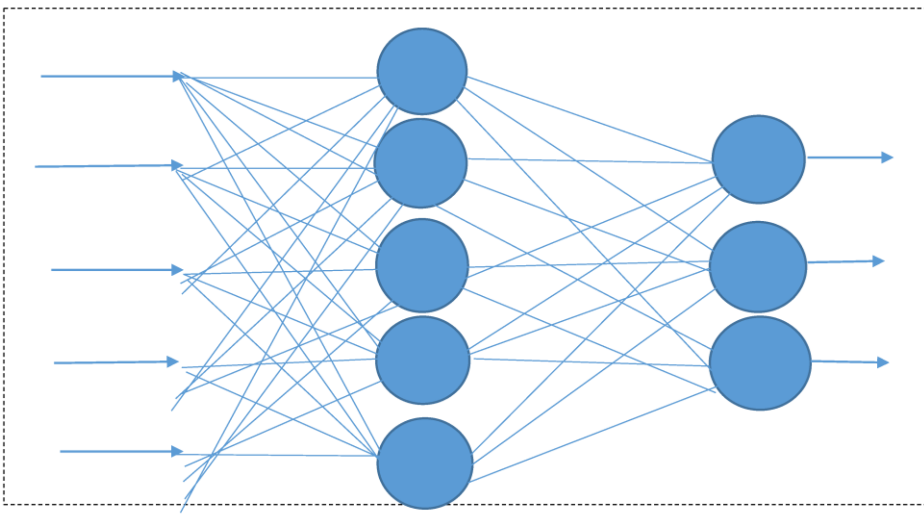

Figure 4.5. Multilayer network diagram with serial communications.

Networks without feedback

The neurons of the input layer receive input signals, convert and transmit them to the neurons of the first hidden layer, and so on to the output, which gives signals to the interpreter and user.

If another option is not specified, then each output signal of the *q-th* layer is fed to the input of all neurons of the *(q + 1)* layer; however, it is possible to connect the *q-th* layer with an arbitrary layer.

Among multilayer networks without feedbacks, distinguished are fully connected (the output of each neuron of the *q-th* layer is associated with the input of each neuron of the *(q + 1))* and partially fully connected. A classic version of layered networks is fully connected direct distribution networks is shown in Figure 4.6.

Feedback Networks.

In networks with feedback, information from subsequent layers is transmitted to the previous ones. The following types of feedback Neural Networks are distinguished:

- layered cyclic, characterized that the layers are closed in a ring: the last layer transmits its output signals to the first; all layers are equal and can both receive input signals and give output;

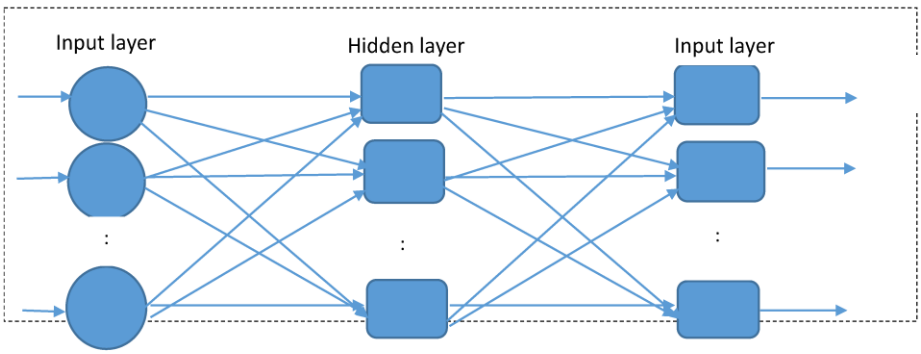

Figure 4.6. Scheme of a multilayer (two-layer) fully connected direct distribution network

- layered-fully-connected consist of layers, each of which is a fully-connected network, and the signals are transmitted from layer to layer and inside the layer; in each layer, the operation cycle is divided into three parts: receiving signals from the previous layer, exchanging signals within the layer, generating an output signal and transmitting to the next layer;
- fully connected-layered, similar in structure to layered-fully-connected, but functioning differently: they do not separate the phases of exchange within the layer and transmission to the next, at each beat, the neurons of all layers receive signals from neurons of both their own layer and subsequent ones.

In figures 4.7 and 4.8, there is shown a diagram of partially recurrent networks of Elman and Jordan.

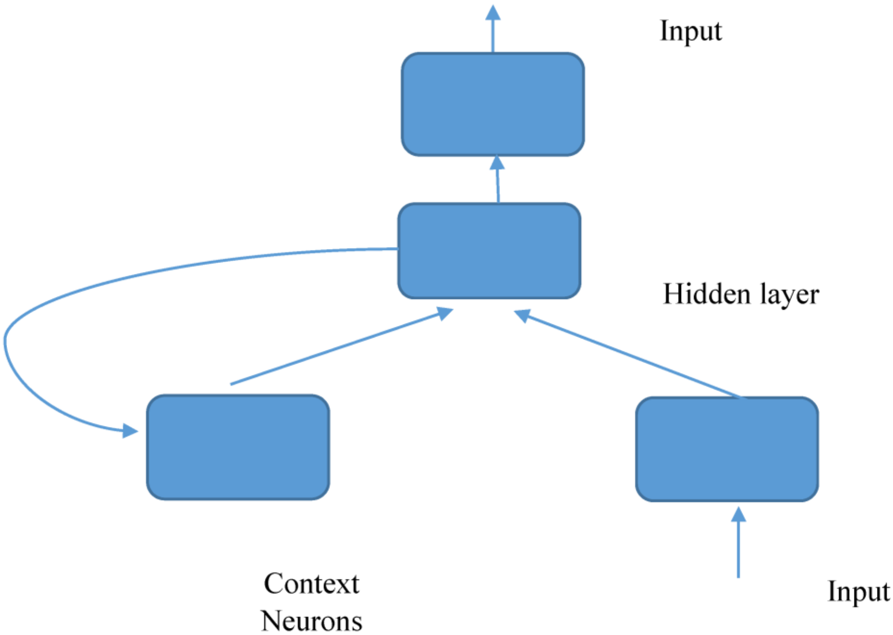

Figure 4.7. Elman Network Diagram

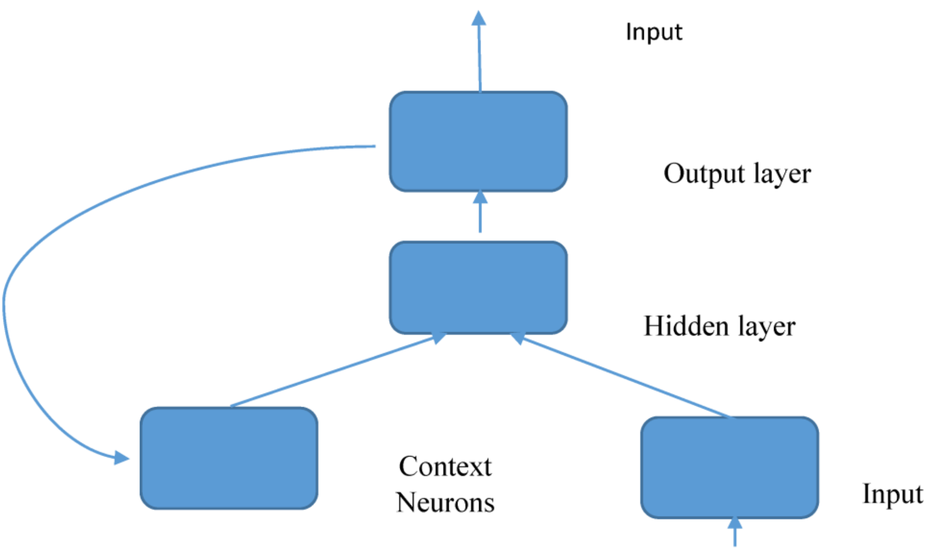

Figure 4.8. Scheme of Jordan networks

A weakly connected Neural Networks is shown in figure 3.9., where neurons are located in nodes of a rectangular or hexagonal lattice. Every neuron is connected with four others (von Neumann), six others (Golei) or eight others (Moore), which are its nearest neighbors.

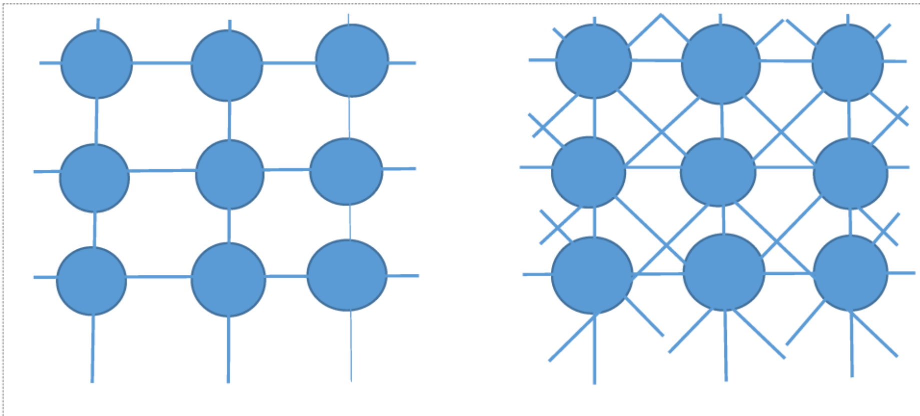

Figure 4.9. Scheme of weakly connected neural networks (with local connections)

Neural Networks can be divided according to the types of structures of neurons into homogeneous and heterogeneous. Homogeneous networks consist of one type of neurons with a single activation function, and a heterogeneous network includes neurons with different activation functions.

There are binary and analog networks. The first of them operate only with binary signals and the output of each neuron can take the value of either a logical zero (inhibited state) or a logical unit (excited state).

Another classification divides Neural Networks into synchronous and asynchronous. In the first case, at each moment of time, only one neuron changes state, in the second, the state changes immediately in a whole group of neurons, as a rule, in the entire layer. Algorithmically, the course of time in neural networks is determined by the iterative execution at the same type of action on neurons.

The choice of the structure of the Neural Network is carried out in accordance with the features and complexity of the task. Optimal configurations already exist for solving certain types of tasks. If the problem cannot be reduced to any of the known types, it is necessary to solve the complex problem of synthesizing a new configuration. In this case, it is necessary to be guided by the following basic rules:

- network capabilities increase with numerous network neurons, the density of connections between them and the number of layers;
- the introduction of feedback along with an increase in network capabilities raises the issue of dynamic network stability;
- the complexity of the network functioning algorithms, the introduction of several types of synapses enhances the power of the neural network.

The problem of synthesis of a neural network depends on the problem that must be solved. In most cases, the best option is obtained based on intuitive selection, although the literature provides evidence that for any algorithm, there is a neural network that can implement it.

The use of Neural Networks to approximate data:

Probabilistic neural network (PNN)

Probabilistic Neural Networks relate to neural networks of radial basis type. PNN was proposed by Specht D.F. (1990, 1992) as improving statistical methods of pattern recognition. PNNs are for classification purposes.

Probabilistic Neural Network has at least three layers: input, radial and output. Radial elements are taken one for each training observation. Each of them represents a Gaussian function centered on this observation. Each class has one output element. Each such element is connected to all radial elements belonging to its class, and it has a zero connection with all other radial elements. Thus, the output element simply adds up the responses of all elements belonging to its class. The values of the output signals are obtained proportional to the nuclear estimates of the probability of belonging to the corresponding classes and normalizing them by one, we obtain the final estimates of the probability of belonging to the classes.

The basic PNN model has two modifications

In the first case, the proportions of classes in the training set correspond to their proportions in the entire studied population (or the so-called a priori probabilities). If the a priori probabilities will differ from the proportions in the training set, the network will produce the wrong result. This can be taken into account (if a priori probabilities have become known) by correcting factors for various classes.

The second version of the modification is based on the fact that any estimate issued by the network is based on noisy data and will inevitably lead to separate classification errors. It is advisable to consider the value of some types of errors. The probabilities given by the network should be multiplied by loss factors reflecting the relative cost of classification errors.

A probabilistic Neural Network has a single learning control parameter whose value the user chooses, as well as the degree of smoothing (or the deviation of the Gauss function). PNNs are not very sensitive to the choice of the smoothing parameter.

The most important benefits of PNN:

- the output value has a probabilistic meaning;
- can generate accurate predictive estimates of the target probability;
- the network is learning fast;
- much faster than multi-layer perceptron networks;
- PNNS may be more accurate than multi-layer perceptron networks;
- relatively insensitive to emissions.

While training such a network, time is spent almost exclusively on submitting training observations, and the network works as fast as possible. PNNs are especially useful in trial experiments, due to the short training time, but a large number of trial tests can be quickly completed.

A significant drawback of such networks is volume. PNN actually contains all the training data, so it requires a lot of memory and can work slowly. PNN is slower than multilayer network perceptron when classifying new cases.

Generalized Regression Neural Network (GRNN)

GRNN arranged similarly PNN, but it is designed to solve regression problems, not classification (Speckt, 1990; Bishop, 1995). As in the case of a PNN network, a Gaussian nuclear function is placed at the location of each training observation. GRNN copies all the training observations inside itself and uses it to evaluate the response at an arbitrary point. The final output estimate of the network is obtained as a weighted average of the outputs for all training observations, where the weights reflect the distance from these observations to the point at which the assessment is made.

The first intermediate layer of the network GRNN consists of radial elements. The second intermediate layer contains elements that help evaluate a weighted average. A special procedure is used for this. Each output in this layer has own element, which forms a weighted sum for it. To get a weighted average from a weighted amount, this amount must be divided by the sum of the weighting factors. The last amount is calculated by the special element of the second layer. After that, the actual division is performed in the output layer. Thus, the number of elements in the second intermediate layer is one more than in the output layer. As a rule, when the regression problems appear, it is required to evaluate one output value, and, accordingly, the second intermediate layer contains two elements.

It is possible to modify GRNN so that the radial elements correspond not to individual training cases, but to their clusters. This reduces the size of the network and increases the speed of learning.

Pluses: GRNN learns almost instantly

GRNN disadvantages:

- does not have the ability to extrapolate data;
- it can turn out big and slow.

Line network

In terms of approximation, the simplest model is a linear one, in which the fitting function is determined by the hyperplane. In the classification problem, the hyperplane is positioned so that separates two classes (linear discriminant function); in the regression problem, the hyperplane must pass through the given points. A linear model is usually written using an NxN matrix and a displacement vector N.

In the language of Neural Networks, a linear model is represented by a network without intermediate layers, which in the output layer contains only elements with a linear function. Weights correspond to matrix elements, and thresholds correspond to components of the displacement vector. During operation, the network actually multiplies the input vector by the weight matrix, and then the displacement vector is added to the resulting vector. A linear network is a good starting point for assessing the quality of constructed Neural Networks.

Kohonen Network

Kohonen Networks are fundamentally different from all other types of networks (Kohonen, 1982; Haykin, 1994; Fausett, 1994).

The Kohonen Network is learning to understand the data structure. One possible use of such networks is exploratory data analysis. The Kohonen Network can recognize clusters in the data, as well as establish the proximity of classes. In this way, the user can improve his understanding of the data structure in order to further refine the Neural Network model. If classes are recognized in the data, they can be designated, after that, the network will be able to solve classification problems. Kohonen networks can also be used in classification problems where classes are already defined - then the advantage will be that the network can reveal similarities between different classes.

Another possible application is the discovery of new phenomena. The Kohonen Network recognizes clusters in training data and assigns all data to particular clusters. If the network encounters that the data set is unlike any of the known samples, it will not be able to classify a set and thereby reveal novelty.

The Kohonen Network has only two layers: input and output, composed of radial elements (the output layer is also called the topological map layer). Elements of a topological map are located in a certain space – usually two-dimensional.

Deep neural network (DNN) – it is an Artificial Neural Network (ANNs) with several layers between the input and output layers (Bengio Y., 2009; Schmidhuber J., 2015).

Deep neural networks are generally interpreted in terms of the universal approximation theorem (Cybenko G., 1989; Hornik K., 1991; Haykin S., 1999; Hassoun M., 1995; Lu Z., etc. 2017) or probabilistic inference (Bengio Y., LeCun Y., Hinton G., 2015; Bengio Y.; Courville A., Vincent P., 2013; Schmidhuber J., 2015; Deng L.; Yu D., 2014).

Deep Neural Networks appeared not so long ago, in the last decade. Before that, mathematicians have proved that three layers of a Neural Network are quite enough to solve any problems. But nature tells us something else: if you divide the reaction time of a person by the time of passage of a nerve impulse through one neuron, you get about 10-20. This is a rough estimate of the number of layers of our natural Neural Network, its depth. Apparently, nature is so disposed for a reason, and this is necessary for something. Indeed, it turned out that difficult tasks are more convenient to solve. The initial layers in the Neural Network play the role of converters, capable of translating the original complex structure of the object (for example, an image, an audio signal or text) into a set of attributes, in applied statistics and machine learning, we are already used to dealing with such vector feature descriptions of objects. Researchers and engineers have always come up with features, based on an understanding of the essence of the problem. This was an important stage of modeling, and in each application, it is a unique creative work. Deep Neural Networks claim to fully automate this stage. If previously developed signs (feature engineering), then with the advent of Deep Neural Networks began to develop network architectures (architecture engineering). This is a higher-level activity in the sense that now there is no need to ponder too small details of the problem being solved. With lessknowledge, more data to collect, but solve the problem more precisely.

DNNis the correct method of mathematical transformations to turn outgoing data into output, regardless of linear or non-linear correlation. The network moves through the layers, calculating the probability of each exit. For example, a DNN that is trained to recognize dog breeds will walk through a given image and calculate the likelihood that the dog in the image belongs to a particular breed. The user can view the results and select the probabilities that the network should display (above a certain threshold, for example), and return the suggested label to the network. Each mathematical transformation is considered a layer, and complex DNNs have many layers, hence the name "deep" networks.

DNN can simulate complex nonlinear relationships. DNN architectures generate compositional models in which an object is expressed as a multi-level composition of primitives (Szegedy C., Toshev A., Erhan D., 2013). Additional levels allow composing elements from lower levels, potentially modeling complex data with fewer units than a small network with similar indicators (Bengio Y., 2009; Bengio Y.; LeCun Y., Hinton G., 2015).

The deep architecture includes many options for several basic approaches. Each architecture has found success in certain areas. It is not always possible to compare the performance of several architectures if they were not evaluated on the same datasets. DNN-networks with a direct connection in which data is transmitted from the input level to the output level without feedback. First, the DNN creates a map of virtual neurons and assigns random numerical values or "weights" to the connections between them. Weights and input data are multiplied and return an output signal from 0 to 1. If the network does not accurately recognize a specific pattern, the algorithm will adjust the weighting factors. Thus, the algorithm can make certain parameters more significant until it determines the correct mathematical manipulations for complete data processing.

Recurrent Neural Networks (RNNs), in which data can flow in any direction, are used for applications such as language modeling (Gers F., Schmidhuber J., 2001). Long short-term memory is particularly effective for this use (Hochreiter S., Schmidhuber J., 1997).

Convolutional deep neural networks (CNNs) are used in computer vision (LeCun Y. et al., 1998). CNNs also have been applied to acoustic modeling for automatic speech recognition (ASR) (Sainath, Tara N. Mohamed, Abdel-Rahman; Kingsbury, Brian; Ramabhadran, Bhuvana, 2013).

In March 2019, Yoshua Bengio, Geoffrey Hinton and Yann LeCun were awarded the Turing Award for conceptual and engineering breakthroughs that have made deep neural networks a critical component of computing.

QUESTIONS:

1. How are Neural Networks divided by the nature of training?
2. How are Neural Networks divided by the type of tuning weights?
3. List the types of multilayer Neural Networks.
4. What are the feedback networks?
5. What is the essence of using Neural Networks for data approximation?
6. Describe the Probabilistic neural network.
7. What are the modifications of the basic PNN model?
8. What are the most important benefits of PNN?

9. Describe the generalized regression Neural Network (GRNN).
10. Advantages and disadvantages of GRNN?
11. What is the essence of a linear network?
12. What is the essence of the Kohonen network?
13. What is the essence of the Deep Neural Network?
14. What is the essence of Recurrent Neural Networks?
15. What is the essence of Convolutional deep Neural Networks?

4.4. NEURAL NETWORK TRAINING

The most important property of Neural Networks is the ability to learn from environmental data and as a result of training, increase productivity. Productivity increases over time following certain rules. Neural Network training occurs through an interactive process for adjusting synaptic weights and thresholds. In the ideal case, a Neural Network gains knowledge about the environment at each iteration of the learning process.

Many activities are associated with the concept of training, and because of this, it is difficult to give this process a unique definition. Moreover, the learning process depends on the point of view. That is what makes it virtually impossible for any exact definition of this concept to appear. For example, the learning process from the point of view of a psychologist is fundamentally different from the point of view of a school teacher. From the perspective of the neural network, can probably use the following definition:

Learning is a process in which the free parameters of a neural network are tuned by modeling the environment in which this network is embedded. The type of training is determined by the way these parameters are adjusted.

This definition of the learning process of a Neural Network involves the following sequence of events:

- stimuli from the external environment enter the neural network;
- as a result of the first paragraph, the free parameters of the Neural Network are changed;
- after a change in the internal structure, the Neural Network responds to excitations differently.

The above list of clear rules for solving the problem of learning a neural network is called a learning algorithm. It is easy to guess that there is no universal learning algorithm suitable for all Neural Network architectures. There is only a set of tools, represented by a variety of training algorithms, which have their advantages. Learning algorithms differ from each other by the way they adjust the synaptic weights of neurons. Another distinguishing characteristics in the way the neural network is trained, is connected with the outside world. In this context, it is a learning paradigm which operates within associated with a model of the environment in a given Neural network.

We can distinguish learning algorithms with a teacher and without a teacher.

The process of teaching with a teacher is a network of sample training examples. Each sample is fed to the inputs of the network, then it is processed inside the structure of the Neural Network, the output signal of the network is calculated, which is compared with the corresponding value of the target vector, which represents the required output of the network. Then, according to the rule, an error is calculated, and the weighting coefficients of the connections within the network change, depending on the selected algorithm. The vectors of the training set are presented sequentially, errors are calculated, and the weights are adjusted for each vector until the error over the entire training array reaches an acceptably low level.

For managed network learning, training data is taken from historical information. For example, it can be previous values of stock prices and the FTSE index, examples of the positions of the robot and its correct reaction, etc.

Then the Neural Network is trained by using one or another controlled learning algorithm (the most famous of them is the backpropagation method proposed in Rumelhart et al., 1986), in which the available data are used to adjust the weights and threshold values of the network in such a way as to minimize the forecast error on the training set. If the network is well trained, it acquires the ability to model a (unknown) function that relates the values of the input and output variables, and subsequently, such a network can be used to predict in a situation where the output values are unknown.

For managed network training, the user must prepare a set of training data. These data are examples of input data and their corresponding outputs. The network learns to establish a connection between the first and second one. The training data set is a set of observations for which the values of the input and output variables are indicated.

Data collection for a Neural Network

Initial selection of variables is intuitive. To begin with, it makes sense to include all the variables that may affect the result – in the subsequent stages this set is reduced.

Neural networks can work with numerical data lying in a certain limited range. This creates problems in cases where the data has a non-standard scale or there are missing values when the data is non-numeric. Numeric data is scaled to a range suitable for the network, and missing values can be replaced by the average value of this variable for all available training examples (Bishop, 1995).

A more difficult task is to work with non-numeric data. Most often, non-numeric data are presented in the form of nominal variables. Neural Networks do not give good results when working with nominal variables, which can take many different values.

The question about observations you need to have to train the network is often difficult. Numerous heuristic rules are known, which link the number of necessary observations with the size of the network (the number of observations should be ten times the number of connections in the network). In fact, this number also depends on the (previously unknown) complexity of the mapping that the Neural Network seeks to reproduce. As the number of variables increases, the number of required observations grows nonlinearly, so even with a small number of variables, a huge number of observations may be required. This difficulty is known as the "curse of dimensionality".

In many tasks, it is necessary to deal with not quite reliable data. The values of some variables may be distorted by noise or partially absent. Although Neural Networks are generally noise resistant. However, stability has a limit. For example, emissions, values lying very far from the normal range of a variable can distort the learning outcome. In such cases, it is better to try to detect and remove these outliers (either by deleting the relevant observations, or by converting outliers into missing values). If there is enough data, the observations with missing values from consideration should be removed.

What will be the output variables is always known (in the case of controlled learning). As for the input variables, their correct choice sometimes presents great difficulties (Bishop, 1995).

The curse of dimensionality

Each additional input element of the network is a new dimension in the data space. From this point of view, it becomes clear: in order to densely "populate" an N-dimensional space and "see" a data structure, need to have quite a few points. The required number of points rapidly increases with the increasing dimensionality of space. Most types of neural networks (in particular, the MLP multilayer perceptron) are less affected by the dimensional curse than other methods because the network can follow the projections of sections of multidimensional space into small-dimensional spaces (for example, if all weights coming from some input element are equal to zero, then MLP completely ignores this input variable). However, the curse of dimensionality remains a serious problem, and network performance can be significantly improved by eliminating unnecessary input variables. In fact, in order to reduce the effect of the dimensional curse, it is advisable to exclude even those input variables that carry some small information.

Internal dependencies between variables

Typically, two or more interrelated variables can together carry essential information, which is not contained in any subset of them. None of the variables individually carries any useful information (the classes will look completely mixed), but looking at both variables together, the classes are easy to separate. Thus, variables cannot be selected independently.

Redundancy of variables

It happens that the same information is more or less repeated in different variables. It may that as inputs it is enough to take only a part of several correlated variables, and this choice can be arbitrary. In such situations, instead of the entire set of variables, it is better to take part of them – this avoids the curse of dimensionality. So, the choice of input variables is an extremely important stage in the construction of a Neural Network. Before you start directly, it makes sense to pre-select the variables, using knowledge in the subject area and standard statistical criteria.

Learning a Neural Network without a teacher is a much more plausible model of learning in terms of the biological roots of Artificial Neural Networks. When learning without a teacher, the learning set consists only of input vectors. The training algorithm adjusts the network weights so that consistent output vectors are obtained, i.e., so that the presentation of sufficiently close input vectors gives the same outputs. The learning process, therefore, identifies the statistical properties of the training set and groups similar vectors into classes. Presenting a vector from a given class at the input will give a certain output vector, but before training, it is impossible to predict which output will be produced by this class of input vectors. Therefore, the outputs of such a network should be transformed into some understandable form, due to the learning process. This is not a serious problem. It is usually not difficult to identify the connection between the input and output established by the network.

Multilayer Perceptron training (MLP)

This network architecture is now used most often. She was offered by Rumelhart, McClelland (1986). Each network element builds a weighted sum of its inputs, adjusted in the form of a term, then passes this activation value through the transfer function, and the output value of this element is obtained. Elements are organized in a layered topology with direct signal transmission. Such a network can easily be interpreted as an input-output model, in which weights and threshold values (offsets) are free parameters of the model. Such a network can simulate a function of almost any degree of complexity, and the number of layers and the number of elements in each layer determine the complexity of the function. The determination of the number of intermediate layers and the number of elements is an important issue in the design MLP (Haykin, 1992, 1994).

The number of input and output elements is determined by the conditions of the problem. Assume that the input variables are intuitively selected and that they are all significant. As an initial approximation, one intermediate layer is taken, and the number of elements in it is equal to half the sum of the number of input and output elements. After determining the number of layers and elements in each of them, you need to find values for the weights and thresholds of the network that would minimize the forecast error generated by the network. Training algorithms serve these purposes. Using historical data, weights and thresholds are automatically adjusted to minimize this error. In essence, this process is a fitting of the model, which is implemented by the network, to the existing training data. The error for a specific network configuration is determined by running through the network all available observations and comparing the actual output values with the desired (target) values. All differences are summed up in the so-called error function, the value of which is the network error. As the error function, the sum of the squared errors is most often taken, i.e., when all errors of the output elements for all observations are squared and then summed. When working with ST Neural Networks, the so-called root-mean-square error (RMS) is generated – the value described above is normalized to the number of observations and variables, after which the square root is extracted from it – this is a very good measure of error, averaged over the entire training set and overall output elements.

The notion of a surface error is very useful in these considerations. Each of the weights and thresholds of the network (i.e., the free parameters of the model; their total number is denoted by N) corresponds to one dimension in multidimensional space. $N + 1$ dimension corresponds to a network error. For all possible combinations of weights, the corresponding network error can be represented by a point in the $N + 1$-dimensional space, and all such points form some surface there – the error surface. The main goal of training Neural Network is to find the lowest point on this multidimensional surface. The error surface has a complex structure and can have local minima (points that are the lowest in a certain neighborhood but lying above the global minimum).

By analytical means it is impossible to determine the position of the global minimum on the surface of errors; therefore, training a Neural Network essentially consists in studying the surface of errors. Based on the random initial configuration of weights and thresholds (i.e., a randomly taken point on the error surface), the learning algorithm gradually searches for a global minimum. As a rule, the gradient (slope) of the error surface at a given point is calculated, and then this information is used to move down the slope. In the end, the algorithm stops at a lower point, which may turn out to be just a local minimum (in case of lucky, a global minimum).

For training Neural Networks without a teacher, a training method by Hebb. D., Oja E. is used.

The rule for training an individual indicator neuron is necessarily local. It is based on information directly accessible to the neuron itself – the values of its inputs and outputs. This rule, bearing the name of a Canadian scientist (Hebb. D., 1949), plays a fundamental role in neurocomputing because, as in a bud, the basic properties of self-organization of Neural Networks. The Hebb Neural Network learning rule is also called the activity multiplication rule. The rule can be applied to various types of Neural Networks with different activation functions; when training a neuron with a linear activation function, stabilization is not achieved even when forgetting is used.

Hebb education, in practice, in its pure form is not applicable because it leads to an unlimited increase in the amplitude of the balance (Caporale N., Dan Y., 2008). This drawback can be eliminated by adding, which prevents the increase of weights. In 1991 E. Oja suggested a modification of the Hebb rule, when taught by Oja E., the neuron weight vector is located on the hyper-sphere, in the direction that maximizes the projection of the input vectors. Such training seeks to maximize the sensitivity of a single output indicator to multidimensional input information, providing an example of optimal information compression (Oja E., Karhunen J., 1985, 1995).

Mathematically, the learning process can be described as the process of functioning, the Neural Network generates an output signal Y, realizing some function Y = G (X). If the network architecture is specified, then the form of the function G is determined by the values of the synaptic weights and the offset network.

Suppose that a solution to a problem is a function $Y = F(X)$, which specified by the input-output data $(X1, Y1), (X2, Y2), \ldots, (XN, YN)$, for which $Yk = F(Xk)$ $(k = 1, 2, \ldots, N)$. Learning consists in finding (synthesizing) a function G that is close to F in the sense of some error function E. If a lot of training examples are chosen - pairs (XN, YN) (where $k = 1,2, \ldots, N$) and the method of calculating the error function E, the training of a Neural Network turns into a multidimensional optimization problem having a large dimension, and since the function E can have an arbitrary form of training in the general case, it is a multi-extreme non-convex optimization problem.

To solve this problem, the following (iterative) algorithms can be used:

1. Local optimization algorithms with the calculation of partial derivatives of the first order:

- gradient algorithm (steepest descent method);
- methods with one-dimensional and two-dimensional optimization of the objective function in the direction of the anti-gradient;
- conjugate gradient method;
- methods that take into account the direction of the anti-gradient at several steps of the algorithm.

2. Local optimization algorithms with the calculation of partial derivatives of the first and second order:

- Newton's method;
- optimization methods with sparse Hessian matrices;
- quasi-Newtonian methods;
- Gauss-Newton method;
- Levenberg-Markar method (Levenberg 1944; Marquardt 1963; Bishop 1995) – the fastest learning algorithm of all that is implemented in the ST Neural Networks package, but, unfortunately, there are several important limitations to its use. It is applicable only for networks with one output element, it only works with the error function, the sum of squares, and requires a lot of memory.

3. Stochastic optimization algorithms:

- search in a random direction;
- simulated annealing;
- Monte Carlo method (numerical method of statistical tests).

4. Global optimization algorithms (global optimization problems are solved by enumerating the values of variables on which the objective function depends).

The Kohonen network is mainly for an unmanaged learning (Kohonen 1982; Haykin 1994; Fausett 1994). In guided learning, the observations that make up the training data, along with the input variables, also contain the corresponding output values, and the network must restore the mapping that transfers the first to the second. In the case of uncontrolled learning, the training data contains only the values of the input variables.

The Kohonen network is trained by the method of successive approximations. Starting with a randomly selected source location of the centers, the algorithm gradually improves it so as to capture the clustering of training data. As a result of the iterative training procedure, the network is organized in such a way that the elements corresponding to the centers located close to each other in the input space will be located close to each other and on the topological map. The topological layer of the network can be thought of as a two-dimensional lattice, which must be so mapped into the N-dimensional space of inputs in order to preserve the original data structure as much as possible. Of course, in an attempt to imagine N-dimensional space on a plane, many details will be lost; however, this technique is sometimes useful, as it allows the user to visualize data that cannot be understood in any other way.

Kohonen's main iterative algorithm sequentially goes through one after another of numerous eras, at each epoch, it processes each of the training examples, and then applies the following algorithm:

- select the winning neuron (one closest to the input example);
- adjust the winning neuron so that it becomes more similar to this input example (taking the weighted sum of the previous center of the neuron and the training example).

In the algorithm for calculating the weighted sum, a gradually decreasing coefficient of the learning speed is used, so that at each new era the correction becomes more and more subtle. As a result, the position of the center will be established in a certain position, which satisfactorily represents those observations for which this neuron was the winner.

The property of topological ordering is achieved in the algorithm using the additional use of the concept of neighborhood. A neighborhood is a few neurons surrounding a winning neuron. Like the speed of learning, the size of the neighborhood decreases with time, so at first, a rather large number of neurons belong to it; at the very last stages, the neighborhood becomes zero (i.e., consisting only of the most winning neuron). In fact, Kohonen algorithm, the adjustment is applied not only to the winning neuron, but to all neurons from its current neighborhood.

The result of the change in neighborhoods is initially quite large sections of the network are "pulled" towards training examples. The network forms a rough structure of a topological order, in which similar observations activate groups of closely lying neurons on a topological map. With each new era, the learning speed and the size of the neighborhood decrease, thereby revealing subtler differences within the map sections, which ultimately leads to fine-tuning of each neuron. Often training is deliberately divided into two phases: shorter, with a high learning speed and large surroundings, and longer with a slow learning speed and zero or almost zero neighborhoods.

After the network is trained to recognize the data structure, it can be used as a visualization tool for data analysis. Using the data displayed in the window – Win Frequencies, (where for each neuron it is calculated how many times it won in the processing of training examples), it is possible to determine whether the card is divided into separate clusters. Furthermore, possible process of individual observations and see how the topological map changes in this case – this allows to understand whether the clusters have any meaningful meaning. After the clusters are identified, neurons of the topological map are marked with meaningful labels. After the topological map in the form described here is constructed, new observations can be submitted to the network input. If the winning neuron was previously marked with a class name, the network carries out the classification. Otherwise, it is considered that the network has not made any decision.

When solving classification problems in Kohonen networks, the so-called access threshold is used. Due to the fact that in such a network the level of neuron activation is the distance from it to the input example, the access threshold plays the role of the maximum distance at which recognition occurs. If the activation level of the winning neuron exceeds this threshold value, then the network is considered not to have made any decision. Therefore, when all neurons are labeled and the thresholds are set at the right level, the Kohonen network can serve as a detector of new phenomena.

Backpropagation algorithm

The most famous version of the neural network learning algorithm is the so-called back propagation algorithm; (Haykin, 1994; Fausett, 1994). Compared to other algorithms, it requires less memory and usually quickly reaches an acceptable level of error, although it can converge rather slowly to the exact minimum of error.

There are modern second-order algorithms, such as the conjugate gradient method and the Levenberg-Markard method. The backpropagation algorithm is the easiest to understand, and in some cases, it has certain advantages. Heuristic modifications of this algorithm have also been developed that work well for certain classes of problems – rapid distribution (Fahlman, 1988) and Delta-Delta with a dash (Jacobs, 1988).

In the backpropagation algorithm, the error surface gradient vector is computed. This vector indicates the direction of the shortest descent on the surface from a given point, if we move along it, the error decreases. The sequence of such steps will ultimately lead to a minimum of one type or another. With long strides, convergence will be faster, but there is a danger of jumping over the solution or going in the wrong direction.

In practice, the step value is taken proportional to the steepness of the slope (so that the algorithm slows down near the minimum) with some constant called the learning speed. The right choice of learning speed depends on the specific task and is usually carried out empirically; this constant may also depend on time, decreasing as the algorithm advances.

Usually, this algorithm is modified in such a way as to include the term of momentum (or inertia). This helps to advance in a fixed direction, so if several steps were taken in the same direction, the algorithm "increases the speed", which (sometimes) avoids a local minimum, as well as faster to pass flat sections.

Thus, the algorithm acts iteratively, and steps are called epochs. At each epoch, all training observations are fed to the network input, the output values of the network are compared with the target values, and an error is calculated. The error value, as well as the error surface gradient, is used to adjust the weights, after which all actions are repeated. The initial network configuration is randomly selected, and the learning process stops either when a certain number of epochs have passed, or when the error reaches a certain level of smallness, or when the error stops decreasing (the user can select the necessary stopping condition himself).

Deep learning (also known as deep structured learning) is part of a broader family of machine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised (Bengio Y., LeCun Y., Hinton G., 2015; Bengio Y., Courville A., Vincent P., 2013; Schmidhuber J., 2015).

Deep learning – a set of machine learning methods (with a teacher, with the partial involvement of a teacher, without a teacher, with reinforcement) based on learning feature/ representation learning, and not specialized algorithms for specific tasks. Many methods were known as far back as 1980, but the results were unimpressive as long as advances in the theory of Artificial Neural Networks (pre-training of Neural Networks using a special case of an undirected graphic model, the so-called limited Boltzmann machine) and the computing power of the mid-2000 (primarily Nvidia, and now Google tensor processors) made it possible to create a complex technological architecture of Neural Networks.

Deep learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases surpassing human expert performance (Krizhevsky A., Sutskever I., Hinton G., 2012; Ciresan D., Meier U., Schmidhuber J., 2012).

Some researchers assess that the October 2012 ImageNet victory anchored the start of a "deep learning revolution" that has transformed the AI industry.

QUESTIONS:

1. How do you understand Neural Network training? Sequence of events.
2. How to understand learning algorithms with and without a teacher?
3. Describe data collection for the Neural Network.
4. Describe the selection of variables and the decrease in dimension.
5. What is the "curse of dimensionality"?
6. Describe the internal dependencies between the variables.
7. What are the features of multilayer perceptron training?
8. How is the Kohonen network trained?
9. Describe the backpropagation algorithm.
10. What is the essence of Deep learning?

4.5. THE USE OF NEURAL NETWORKS

The average human brain has around 86 billion neurons. Now we can build neural networks from millions of neurons. Not so long ago, scientists managed to create a dynamic model of the nematode nervous system (about 300 neurons).

Already today, artificial neural networks are used in many areas, but before they can be used in humans' lives or in significant material resources, it is necessary to solve important issues regarding the reliability of their work. Therefore, the level of permissible errors should be determined based on the nature of the problem. Some problems with the analysis of reliability issues arise due to the assumption that computers are completely error-free, while artificial Neural Networks can be inaccurate even when they are functioning properly. In fact, computers, like people, can also be wrong. The first – due to various technical problems or errors in the programs, the second – due to carelessness, fatigue or unprofessionalism. Therefore, for particularly complex tasks the systems must duplicate and insure each other. This means that in solving such problems Neural Networks should not act as the only means, but as additional ones to prevent special situations or take control when the problem is not solved in a standard way and any delays can lead to disaster.

Another difficulty in using neural networks is that traditional neural networks are unable to explain how they solve the problem. The internal presentation of learning outcomes is often so complex that it cannot be analyzed, except in some simple cases, which are usually not of interest.

Recently, active attempts have been made to combine artificial Neural Networks and expert systems. In such a system, an Artificial Neural Network can respond to most relatively simple cases, and all the rest are referred for examination to the expert system. As a result, complex cases are accepted at a higher level, while possible with the collection of additional data or even with the involvement of experts.

Neural Network application packages developed by numerous companies allow users to work with different types of Neural Networks and with different ways of training them. They can be either specialized (for example, to predict the stock price), or quite universal. Neural Networks can be used in almost all fields and areas of human activity: economics, medicine, communications and security of security systems, information processing.

The class of tasks that can be solved using a neural network is determined by how the network works and how it is trained. During operation, the neural network takes the values of the input variables and provides the values of the output variables. Thus, a network can be used in situations where there is certain known information, and it is necessary to obtain some information that is not yet known (Fausett, 1994).

At the beginning of the 2000, one of the basic concepts of the development of a Multilayer Neural Network has been the desire to increase the number of layers, and this involves ensuring the choice of the structure of the Neural Network that is adequate to the task, the development of new methods for generating coefficients tuning algorithms. The advantages of neural networks are: a property of the so-called gradual degradation – when individual elements fail, the quality of the system decreases gradually; increased resistance to changes in the parameters of circuits that implement them (for example, very significant changes in weights do not lead to errors in the implementation of a simple logical function of two variables), etc.

The widespread use of Neural Network algorithms in the field of complex formalizable, weakly formalizable and unformalizable problems has led to the creation of a new direction in computational mathematics – neuromathematics. Neuromathematics includes Neural Network algorithms for solving the following problems: pattern recognition; optimization and extrapolation of functions; graph theory; cryptographic tasks; solving real and Boolean systems of linear and non-linear equations, ordinary one-dimensional and multidimensional differential equations, partial differential equations, etc. Based on the theory of neural networks, a new section of the modern control theory of complex non-linear and multidimensional, multiply connected dynamic systems is created – neuro-control, including neural network methods identification of complex dynamic objects; the construction of neuroregulators in the control loops of complex dynamic objects, etc.

Typical tasks that can be solved with the help of Neural networks and neurocomputers are: the task of classification, automation of forecasting, automation of decision-making, control, encoding and decoding of information, pattern recognition, etc.

Here are examples of some applications:

An example of the successful application of neural computing in the field of economics, in particular the financial sector, are the credit risk management systems that are successfully used in some well-known US banks.

Other important areas of application of neural computing in the economy are forecasting the situation in the stock market, real estate valuation, forecasting the dynamics of exchange rates, optimization of commodity and cash flows, automatic reading of checks and forms and more.

The provision of credit. It is required to determine whether there is a high risk of providing a loan to an individual who has applied for a request. As a result of a conversation with him, his income, previous credit history, etc. are known. Such calculations are based on the assessment of credit history, the dynamics of company development, the stability of the main financial indicators, and the baggage of senior officials. Neural Network technologies make it possible to effectively perform calculations and accurately identify potential defaulters.

In medicine, Neural Networks are used in the diagnosis of diseases. In particular, an example of diagnostic systems is a software package for cardiodiagnostics, developed by R Informati. Similar systems have been used successfully in some hospitals in England to prevent myocardial infarction and other cardiovascular diseases, which makes it possible to reduce their level.

Neural Network technologies are also used in the diagnosis of cancer. Scientists from Duke University (USA) have developed a neural system for the detection of malignant tissue, which is successfully used to diagnose breast cancer.

Neural Networks have practical applications in the design and optimization of communication networks. And also successfully solve an important task in the field of telecommunications – finding the optimal path of traffic between nodes. In addition to flow routing management, Neural Networks are used to obtain efficient solutions of a new design of telecommunications networks, as well as for fast encoding and decoding of data, compression of video information and more.

In the field of security systems, Neural Networks are needed for face identification, voice recognition, crowd recognition, license plate recognition, analysis of aerospace images, monitoring of information flows, detection of counterfeits. In the field of information processing, Neural Networks can be used for processing checks, signature recognition, fingerprints and voice.

Thus, in the modern world, Neural Networks are not a distant future. Scientists from all over the world are involved in neuroinformatics and research of neural networks in various fields. With the help of artificial Neural Networks can process, analyze and summarize information, which is similar to the work of the human brain. Neutron Networks are used in economics, medicine, communications, security and security systems, information input and processing. Of course, this list is not complete, but it gives an idea of the nature of the use of Neural Network technologies.

You can trace individual milestones in the technology of training Neural Networks.

In addition to the new victories of machine intelligence over humans in increasingly complex games (from chess to Go), one of the most important points is the competition – ImageNet recognition of objects in photographs, which started in 2011. To train algorithms to classify images or find objects on images – these tasks have been around for decades. Prior to the start of ImageNet, progress in computer vision was progressing extremely slowly, at a fraction of a percent annually, despite the considerable efforts of a huge scientific community. At the beginning of the competition, the accuracy of the methods existing at that time did not exceed 75%. Everyone understood that 25% of errors are a bad result for real business applications. The first breakthrough occurred in 2012, when the team of a British scientist Geoffrey Everest Hinton, used his solution based on deep Neural Networks, reduced the share of erroneous classifications from 25 to 12% (Krizhevsky A., Sutskever I., Hinton G., 2012). This is a lot. Since then, other solutions, except for Neural Networks, have not won at Imag. From that moment a continuous improvement in the quality of recognition. By 2015, the share of errors was brought up to 3.5%. A person is mistaken in about 5% of cases. That is, contemporary machines are able to recognize images better than humans. When such a milestone is taken, it is a signal for business: the problem can be solved automatically. Breakthroughs of the same scale have occurred in other areas, such as speech recognition.

Three prerequisites played a role simultaneously. **Firstly**, the development of electronic computing. The ability to store, process, transfer terabytes of data used to be affordable only to large corporations. Now it can be done on a personal computer. **Secondly**, crowdsourcing appeared – a cheap way to form huge training samples by the efforts of thousands of users via the Internet. Project ImageNet would not be so successful if people did not manually classify a million pictures. **Thirdly,** mathematics also did not stand still. Huge experience has been accumulated in the field of mathematical statistics, numerical optimization methods, machine learning, neural networks.

ReLU – Rectified Linear Unit. From the point of view of mathematics, Neural Networks are a composition of linear functions and a nonlinear activation function. For a long time, they used activation functions like sigmoid or arctangent. And at some point, they decided to try a very simple piecewise linear function, composed of two half-lines. It suddenly turned out that this improves accuracy, simplifies the Neural network, and reduces training time by several times. This solution was non-standard at that time, later it found theoretical justification.

In the theory of Neural Networks, whole series of discoveries have taken place that have led to qualitatively new methods in the last decades. Dozens of years ago, no one could have imagined that the explosive growth of Neural Network capabilities would occur largely due to the development of video cards and graphic processors (GPUs) and would actually be invested by tens of millions of gamers around the world who created a massive demand for high-performance computing. Today, thanks to a large part of these technologies, computer vision and unmanned vehicle control systems, interactive and recommendation systems, computer and personalized medicine are being created.

The main progress continues to take place in the academic environment, but the development and implementation of advanced technologies require the effective integration of the efforts of the academic community and business with the support of the state.

For example, Apple Siri's voice assistant is based on speech recognition technology, which has been developing for forty years as part of a series of DARPA (Department of Defense Advanced Research Projects). Scientists from a dozen leading US universities took part in these works. Another modern trend is that corporations are increasingly making their developments open, providing them not only to the academic community but also to everyone, including direct competitors. For example, Google allows freely use its TensorFlow technology to build neural networks. Facebook develops open technology PyTorch, many companies and universities have already joined. The staff of the research divisions of many large companies, such as Google, Facebook, Apple, in fact, the same scientists, yesterday's graduate students or postdoctoral students, and often university professors. They publish a lot and actively participate in scientific conferences. So it is sometimes difficult to understand where the line is between the scientific community and the business community.

Artificial Neural Networks are currently an important extension of the concept of computation. They have already allowed to cope with several difficult problems and promise the creation of new programs and devices that can solve problems that only a human can do. Modern neurocomputers are mainly used in software products and therefore rarely use their potential of "parallelism". The era of true parallel Neural computations will begin with the appearance on the market of a large number of hardware implementations – specialized neurochips and extension cards designed for speech, video, static images and other types of figurative information.

Over time, household appliances should also adapt to their owner, a harbinger of which is the neural network adaptive control unit in the new Samsung vacuum cleaner. Security systems will recognize their owners by voice, appearance and several other unique characteristics. The life support systems of "smart" electronic houses will also be developed, which will become even more adaptive and trainable. In production and in various industrial systems, intelligent Neural Network controllers will be able to recognize potentially dangerous situations, notify people about them and take adequate and, most important, timely measures. Data streams in computer networks and cellular networks will also be optimized using neurotechnologies.

A lot of hopes about Neural Networks today are associated with hardware implementations, but the time for their mass market entry has not yet come. They are either available as part of specialized devices, or are quite expensive, and often both. Considerable time is spent on their development, during which software implementations on the latest computers are only an order of magnitude less productive, which makes the use of neuroprocessors unprofitable. But all this is only a matter of time – Neural Networks have to go the same way that computers have only recently developed, increasing their capabilities and productivity, capturing new areas of application as new tasks arise and the technical basis for their development.

Another area of application of Neural networks is using it in specialized software agents – in robots designed to process information, not for physical work. Intelligent assistants should make it easier for users to work with information and communicate with the computer. Their distinctive feature will be the desire to understand as best as possible what is required of them, by observing and analyzing the behavior of their master, trying to detect some patterns in this behavior and promptly offer their services to perform certain operations. For example, to filter news messages, with tips on resolving a problem that has arisen or for backing up documents the user is working on That is why Neural Networks that can generalize data and find patterns in them are a natural component of such software agents.

QUESTIONS:

1. Describe the difficulty of using Neural Networks?
2. What are Neural Network application packages?
3. What are the typical tasks that can be used for the Neural Neurons?
4. Describe the relationship of artificial neural networks and the expansion of the concept of computing.
5. List the prospects for using Neural Networks.

References

Aaron Masih (2019), *Augmented Intelligence, not Artificial Intelligence, is the Future*, https://medium.com/datadriveninvestor/augmented-intelligence-not-artificial-intelligence-is-the-future-f07ada7d4815.

Abend Kenneth (2002), *The 2001 Benjamin Franklin Medal in Engineering presented to Bernard Widrow*, Journal of the Franklin Institute, 3 (339), pp. 283-294, doi:10.1016/S0016-0032(01)00044-8.

Alexandre Albore, Hector Palacios, Hector Geffner (2009), *A Translation-Based Approach to Contingent Planning*, International Joint Conference of Artificial Intelligence (IJCAI), Pasadena, CA: AAAI.

Albore Alexandre, Ramírez Miquel, Geffner Hector (2011), *Effective heuristics and belief tracking for planning with incomplete information*, Twenty-First International Conference on Automated Planning and Scheduling (ICAPS).

Allen Paul, Mark Greaves (2011), *The Singularity Isn't Near*, MIT Technology Review, https://www.technologyreview.com/2011/10/12/190773/paul-allen-the-singularity-isnt-near/.

American National Standards Institute (1975), *ANSI/X3/SPARC Study Group on Data Base Management Systems; Interim Report*, FDT(Bulletin of ACM SIGMOD) 7:2.

Anderson John R., Reder Lynne M., Simon Herbert A., K. Anders Ericsson, Robert Glaser (1998), Diane Ravitch (ed.), *Radical constructivism and cognitive psychology*, Brookings Papers on Education Policy, 6 (1), pp. 227-278.

Armstrong S., Sotala K. (2012), *How We're Predicting AI or Failing to Beyond*, Artificial Dreams, pp. 52-75.

Baum Seth (2017), *A Survey of Artificial General Intelligence Projects for Ethics, Risk, and Policy (November 12, 2017)*, Global Catastrophic Risk Institute Working Paper 17-1.

Bartsev S.I., Okhonin V.A. (1986), *Adaptive information processing networks*, Krasnoyarsk: Institute of Physics SB RAS, Preprint №59, 20.

Bartsev S.I., Gilev S.E., Okhonin V.A. (1989), *The duality principle in the organization of adaptive information processing networks*, [in the book:] *Dynamics of chemical and biological systems*, Novosibirsk: Nauka, pp. 6-55.

BBC News 29 August 2019, *Elon Musk and Jack Ma disagree about AI's threat*, https://www.bbc.com/news/technology-49508091.

BBC News 20 January 2020, *Google boss Sundar Pichai calls for AI regulation*, https://www.bbc.com/news/technology-51178198.

By Ben Dickson (May 8, 2018), *What Ginni Rometty teaches us about the true meaning of AI*, https://bdtechtalks.com/2018/05/08/ibm-ceo-augmented-intelligence-ai/.

Bengio Yoshua (2009), *Learning Deep Architectures for AI*, Foundations and Trends in Machine Learning, 2 (1), pp. 1-127, doi:10.1561/2200000006.

Bengio Yoshua, LeCun Yann, Hinton Geoffrey (2015), *Deep Learning*, Nature, 521 (7553), pp. 436-444, doi:10.1038/nature14539.

Bengio Y., Courville A., Vincent P. (2013), *Representation Learning: A Review and New Perspectives*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 35 (8), pp. 1798-1828, doi:10.1109/tpami.2013.50. PMID 23787338.

Berners-Lee Tim, Hendler James, Lassila Ora (2001), *The Semantic Web A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities*, Scientific American, 284 (5), pp. 34-43, doi:10.1038/scientificamerican 0501-34.

Bezem M (1988), *Consistency of rule-based expert systems*, [in:] *International Conference on Automated Deduction*, Lecture Notes in Computer Science, 310, pp. 151-161, doi:10.1007/BFb0012830, ISBN 3-540-19343-X.

Bishop C.M. (1995), *Neural Networks for Pattern Recognition November Oxford University Press*, Inc.198 Madison Ave, New York, NYUnited States, p. 482, ISBN 978-0-19-853864-6.

Bostrom Nick (2003), *Ethical Issues in Advanced Artificial Intelligence. Cognitive, Emotive and Ethical Aspects of Decision Making in Humans and in Artificial Intelligence*, (eds.) Iva Smit and George E. Lasker. Windsor, International Institute for Advanced Studies in Systems Research, Cybernetics, 2003, 2, pp. 12-17.

Bostrom Nick (2012), *The Superintelligent Will: Motivation and Instrumental Rationality in Advanced Artifi cial Agents. Minds and Machines*, (ed.) Vincent C. Müller, Journal for Artifi cial Intelligence, Philosophy and Cognitive Science, 22 (2), pp. 71-85.

Brooks Rodney (2013), *Robots at work: towards a smarter factory*, The Futurist, 47 (3), pp. 24-27.

Caporale N., Dan Y. (2008), *Spike timing-dependent plasticity: a Hebbian learning rule*, Annual Review of Neuroscience, 31, pp. 25-46, doi:10.1146/annurev.neuro.31.060407.125639.

Carl S. Gibson et al., *VAX 9000 SERIES*, Digital Technical Journal of Digital Equipment Corporation, Vol. 2, Number 4, Fall 1990, pp. 118-129.

Chaitin Gregory J. (2006), *The Limits of Reason*, Scientific American 294, no. 3, pp. 74-81.

Charles W. Bachman (1973), *The Programmer as Navigator. ACM Turing Award lecture*, Communications of the ACM, Volume 16, Issue 11, pp. 653-658, ISSN 0001-0782, doi:10.1145/355611.362534.

Ciresan Dan, Meier U., Schmidhuber J. (2012), *Multi-column deep neural networks for image classification*, 2012 IEEE Conference on Computer Vision and Pattern Recognition: journal, pp. 3642-3649, doi:10.1109/cvpr.2012.6248110.

Clocksin W. (2003), *Artificial intelligence and the future*, Philosophical Transactions of the Royal Society A, 361 (1809), pp. 1721-1748, doi:10.1098/rsta.2003.1232.

Coats P.K. (1988), *Why expert systems fail*, Financial Management, 17 (3), pp. 77-86. doi:10.2307/3666074.

DARPA SyNAPSE Program. Artificial Brains: The quest of build entient machines, URL:http://www.artificialbrains.com/darpa-synapse-program.

DeepMind Ethics & Society homepage (2017), https://deepmind.com/applied/deepmind-ethics-society/.

Deng L., Yu D. (2014), *Deep Learning: Methods and Applications* (PDF), Foundations and Trends in Signal Processing, 7 (3-4), pp. 1-199, doi:10.1561/2000000039.

De Giacomo Giuseppe, Rubin Sasha (2018), *Automata-Theoretic Foundations of FOND Planning for LTLf and LDLf Goals*, IJCAI.

Dyson: *Help with your Dysonhж*, https://web.archive.org/web/20110827002857/http://www.english.dysoncanada.ca/support/help.asp?article=32.

Edureka (2019), *What is Cognitive AI? Is It the Future?,* https://www.edureka.co/blog/cognitive-ai/

Quora.com (2018), *What is cognitive artificial intelligence?*, https://www.quora.com/What-is-cognitive-artificial-intelligence

Fahlman S.E. (1988), *Faster learning variations on back-propagation: An empirical study*, [in:] D. Touretzky, G. Hinton, T. Sejnowski (eds.), Proceedings of the Connectionist Models Summer *School,* San Mateo: Morgan Kaufmann, pp. 38-51.

Fausett L., *Fundamentals of Neural Networks - Architectures, Algorithms, and Applications*, Prentice Hall Inc., Englewood Cliffs, NJ 1994.

Feigenbaum Edward A. (2003)*, Some challenges and grand challenges for computational intelligence*, Journal of the ACM, 50 (1), pp. 32-40, doi:10.1145/602382.602400.

Fukushima Neocognitron (1980), *A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position*, Biological Cybernetics, 36 (4), pp. 193-202, doi:10.1007/bf00344251.

Fukushima K. (2007), *Neocognitron*, Scholarpedia.,2 (1), p. 1717, doi:10.4249/scholarpedia.1717.

Gelernter D. (2010), *Dream Logic, the Internet and Artificial Thought*, Edge Foundation, June 22, 2010, 3 (339), pp. 283-294, doi:10.1016/S0016-0032(01)00044-8.

Gers Felix A., Schmidhuber Jürgen (2001), *LSTM Recurrent Networks Learn Simple Context Free and Context Sensitive Languages*, IEEE Transactions on Neural Networks, 12 (6), pp. 1333-1340, doi:10.1109/72.963769.

George F. Luger, William A. Stubblefield (1989), *Rule Based Expert System Shell: example of code using the Prolog rule based expert system shell*, California: Benjamin/Cummings Publishers Co.; Inc.

Ghallab Malik, Nau Dana S., Traverso Paolo (2004), *Automated Planning: Theory and Practice*, Morgan Kaufmann, ISBN 1-55860-856-7.

Grace Katja, Salvatier John, Dafoe Allan, Zhang Baobao, Evans Owain (2018), *Viewpoint: When Will AI Exceed Human Performance? Evidence from AI Experts,* Journal of Artificial Intelligence Research, 62, pp. 729-754, doi:10.1613/jair.1.11222.

Haraway D. (1988), *Situated Knowledges: The Science Question in Feminism and the Privilege of Partial Perspective,* Feminist Studies Vol. 14, No. 3. pp. 575-599.

Herlyn W (2012), *PPS im Automobilbau*, Hanser Verlag, München, pp. 81-122.

Haslum Patrik, Jonsson, Peter (2000), *Some Results on the Complexity of Planning with Incomplete Information,* Recent Advances in AI Planning. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1809, pp. 308-318, doi:10.1007/10720246_24.

Hayes-Roth Frederick, Waterman Donald, Lenat Douglas (1983), *Building Expert Systems*, Addison-Wesley, ISBN 978-0-201-10686-2.

Hayes P., Ford K. (1995), *Turing test considered harmful*, Proceedings of the international joint conference on artificial intelligence, Montreal, vol. 1, pp. 972-977.

Haykin S., *Blind equalization formulated as a self-organized learning process,* [in:] *26th Asilomar Conference Signals,* Systems Comp., pp. 346-350 vol.1. IEEE, 1992. 545.

Haykin S., *Neural Networks – A Comprehensive Foundations*, Macmillan College Publishing Co., Englewood Cliffs, NJ 1994.

Hebb D.O. (1949), *The Organization of Behavior: A Neuropsychological Theory*, New York: Wiley and Sons, ISBN 9780471367277.

Hochreiter Sepp, Schmidhuber Jürgen (1997-11-01), *Long Short-Term Memory*, Neural Computation, 9 (8), pp. 1735-1780, doi:10.1162/neco.1997.9.8.1735.

Horgan J. (1993), *Profile: Marvin L. Minsky: The Mastermind of Artificial Intelligence (англ.)*, Scientific American journal, New York: Scientific American, Inc, 5 (269), pp. 14-15.

Holland J.H. (1992), *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, Press 55, Hayward St. Cambridge MA United States, 228, ISBN:978-0-262-08213-6.

Hopfield J.J., Tank D.W. (1985), *Neural computation of decisions in optimization problems*, Biological Cybernetics 55, pp. 141-146.

Hopfield J.J. (1982), *Neural networks and physical systems with emergent collective computational properties*, Proc. Nat. Acad. Sci. (USA) 79, pp. 2554-2558.

Hopfield J.J. (1984), *Neurons with graded response have collective computational properties like those of two-sate neurons*, Proc. Nat. Acad. Sci. (USA) 81, pp. 3088-3092.

IBM July 16, 2018, *AI systems must be transparent and explainable Building trust in the age of AI – How businesses can build fairness into their machine learning models* https://www.ibm.com/blogs/watson/2018/07/trust-in-the-age-of-ai-build-fairness-into-machine-learning-models/.

IBM Watson Studio, *IBM Distributed Deep Learning (DDL)*, Last updated: April 28, 2020, https:dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ml_dlaas_ibm_ddl.html.

Jackson Peter (1998), *Introduction to Expert Systems* (3 ed.), Addison Wesley, 2, ISBN 978-0-201-87686-4.

Jacobs R.A. (1988), *Increased rates of convergence through learning rate adaptation*, Neural Networks, 1, pp. 295-307.

James Essinger (2014), *Ada's Algorithm: How Lord Byron's Daughter Ada Lovelace Launched the Digital Age*, Melville House, ISBN 9781612194080.

John McCarthy (1996), *Defending AI research: a collection of essays and reviews*, CSLI lecture notes: no. 49. Center for the Study of Language and Information, Cambridge University Press, 360.

John McCarthy (2007), *What is artificial intelligence? Computer Science Department Stanford University*, http://jmc.stanford.edu/articles/whatisai/whatisai.pdf.

Jussi Rintanen (2004), *Complexity of Planning with Partial Observability*, Int. Conf. Automated Planning and Scheduling, AAAI.

Kaplan A., Haenlein M. (2019), *Siri, Siri, in my hand: Who's the fairest in the land? On the interpretations, illustrations, and implications of artificial intelligence*, Business Horizons, 62, s. 15-25, doi:10.1016/j.bushor.2018.08.004.

Karlsson L., (2001), *Conditional progressive planning under uncertainty*, IJCAI. pp. 431-438.

Kendal S.L., Creen M. (2007), *An introduction to knowledge engineering*, London, Springer, p. 285, ISBN 978-1-84628-475-5.

Kohonen T. (1988), *Learning Vector Quantization*, Neural Networks, 1 (suppl 1), 303.

Kohonen T. (2001), *Self-Organizing Maps (Third Extended Edition)*, New York, p. 501, ISBN 3-540-67921-9.

Krizhevsky Alex, Sutskever Ilya, Hinton Geoffrey E. (2012), *ImageNet classification with deep convolutional neural networks,* Nips'12, Curran Associates Inc., pp. 1097-1105.

Kuzior Aleksandra, Kwilinski Aleksy, Tkachenko Volodymyr (2019), *Sustainable development of organizations based on the combinatorial model of artificial intelligence,* Entrep. Sustain. Iss., 7 (2), pp. 1353-1376, DOI: 10.9770/jesi.2019.7.2(39).

Kuzior Aleksandra, Postrzednik-Lotko Katarzyna (2020), *Natural language and gettering of information. In: Sustainable economic development and advancing education excellence in the era of global pandemic. Proceedings of the 36th International Business Information Management Association Conference (IBIMA)*, Granada, Spain, (ed.) Khalid S. Soliman. [B.m.], International Business Information Management Association, pp. 13479-13486.

Kuzior Aleksandra, Czajkowski Waldemar (2021), *Introduction to Epistemology and Philosophy of Mind*, Lublin, Wydawnictwo Naukowe Tygiel.

Lawler R. (2019), *John Carmack takes a step back at Oculus to work on human-like AI*, https://www.engadget.com/2019-11-13-john-carmack-agi.html

LeCun Y. et al. (1998), *Gradient-based learning applied to document recognition. Proceedings of the IEEE*, 86 (11), pp. 2278-2324, doi:10.1109/5.726791.

Leondes Cornelius T. (2002), *Expert systems: the technology of knowledge management and decision making for the 21st century*, pp. 1-22, ISBN 978-0-12-443880-4.

Leith P. (2010), *The rise and fall of the legal expert system, in European Journal of Law and Technology*, Vol 1, Issue 1, http://ejlt.org//article/view/14/1.

Levinson N. (1966), *Norbert Wiener 1894-1964*, Bull. Amer. Math. Soc. 72 (1 Part 2), pp. 1-33, doi:10.1090/S0002-9904-1966-11450-7.

Littman Michael L. (1997), *Probabilistic Propositional Planning: Representations and Complexity,* Fourteenth National Conference on Artificial Intelligence, MIT Press., pp. 748-754, Retrieved 2019-02-10.

Liu Daphne Hao (2008), *A survey of planning in intelligent agents: from externally motivated to internally motivated systems (Technical report)*, Technical Report TR-2008-936, Department of Computer Science, University of Rochester.

Liu Feng, Shi Yong, Liu Ying (2017), *Intelligence Quotient and Intelligence Grade of Artificial Intelligence,* Annals of Data Science, 4 (2), pp. 179-191, doi:10.1007/s40745-017-0109-0.

Luca-d3.com (2019), *What is Cognitive Intelligence? How is it different to Artificial Intelligence?* https://luca-d3.com/data-speaks/technology-dictionary/cognitive-intelligence.

Luger D.F. (2003), *Artificial Intelligence: Strategies and Methods for Solving Complex Problems,* M.: Publishing House. Williams House, 865.

Luger F. (2008), *George Artificial Intelligence: Structures and Strategies for Complex Problem Solving Addison-Wesley Publishing CompanyUnited States*, ISBN:978-0-321-54589-3.

MacGregor Robert (1991), *Using a description classifier to enhance knowledge representation,* IEEE Expert. 6 (3), pp. 41-46, doi:10.1109/64.87683.

Mak B., Schmitt B.H., Lyytinen K. (1997), *User participation in knowledge update of expert systems*, Information & Management, 32 (2), pp. 55-63, doi:10.1016/S0378-7206(96)00010-9.

MacKay David J.C. (2003), *Information Theory, Inference, and Learning Algorithms,* NewYork: Cambridge University Press, pp. 22-47.

Matthew West Julian Fowler (1999), *Developing High Quality Data Models Archived 2008-12-21 at the Wayback Machine*, The European Process Industries STEP Technical Liaison Executive (EPISTLE).

Marcus G. (2014), *What comes after the Turing test?*, The New Yorker, http://www.newyorker.com/tech/elements/what-comes-after-the-turing-test.

McDonald R., Hall K., Mann G. (2010), *Distributed Training Strategies for the Structured Perceptron. Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL*, Association for Computational Linguistics, pp. 456-464.

McCulloch Warren, Walter Pitts (1943), *A Logical Calculus of Ideas Immanent in Nervous Activity*, Bulletin of Mathematical Biophysics. 5 (4), pp. 115-133, doi:10.1007/BF02478259.

McCulloch Warren (1961), *What is a number, that a man may know it, and a man, that he may know a Number*, General Semantics Bulletin (26 & 27), pp. 7-18.

McKinsey Global Institute (2017), *A future that works: Automation, employment, and productivity*, McKinsey Global Institute. file:///C:/Users/admin/Downloads/MGI-A-future-that-works-Full-report%20(1). pdf.

Mettrey W. (1987), *An Assessment of Tools for Building Large Knowledge-Based Systems*, AI Magazine, 8(4), p. 81.

Michie M. (1996), *Turing's test and conscious thought. Machines and thought the legacy of Alan Turing*, Millican PJR, Oxford: Clarendon Press, vol. 1, pp. 27-51.

Moor J.H. (2003), *The status and future of the Turing test. The Turing test the elusive standard of artificial intelligence*, Dordrecht: Kluwer, pp. 197-214.

Neufeld Xenija, Mostaghim Sanaz, Sancho-Pradel Dario, Brand, Sandy (2017), *Building a Planner: A Survey of Planning Systems Used in Commercial Video Games*, IEEE Transactions on Games, IEEE.

Norwig P., Russell S. (2005), *Artificial Intelligence: A Modern Approach (AIMA)*, M.: Publishing. Williams House, 1424.

Oja E., Karhunen J. (1985), *On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix*, Journal of Math. Analysis and Applications, 106, pp. 69-84.

Oja E., Karhunen J. (1995), *Signal separation by nonlinear Hebbian learning*, [in:] M. Palaniswami et al. (eds.), *Computational Intelligence – A Dynamic System Perspective*, pp. 83-97. IEEE Press,

Paul Boghossian (2007), *Fear of Knowledge: Against relativism and constructivism*, Oxford: Clarendon Press, ISBN 978-0199230419, Chapter 7, pp. 95-101.

Palacios Hector, Geffner Hector (2009), *Compiling uncertainty away in conformant planning problems with bounded width*, Journal of Artificial Intelligence Research, 35, pp. 623-675. doi:10.1613/jair.2708.

Pearl J. (2009a), *Causality: Models, Reasoning, and Inferenc*, Cambridge University Press, New York.

Peot Mark A, Smith David E (1992), *Conditional nonlinear planning. Artificial Intelligence Planning Systems*, Elsevier, pp. 189-197.

Pham H.N., Triantaphyllou E. (2008), *The Impact of Overfitting and Overgeneralization on the Classification Accuracy in Data Mining*, Soft Computing for Knowledge Discovery and Data Mining, pp. 391-431.

Pham H.N., Triantaphyllou E. (2008), *Prediction of diabetes by employing a new data mining approach which balances fitting and generalization*, Computer and Inf. Science, pp. 11-26.

Philipson A. John (2014), *Humphrys Grills the Robot who passed the Turing test and is not impressed*, http://www.telegraph.co.uk/culture/tvandradio/bbc/10891699/John-Humphrys-grills-the-robot-who-passed-the-Turing-test-and-is-not-impressed.html.

Ramamoorthy A., Yampolskiy R. (2018), *Beyond Mad? The Race For Artificial General Intelligence,* Journal: ICT Discoveries, Special Issue No. 1, 2 Feb. 2018, pp. 1-8.

Ray Kurzweil (2012), *How to Create a Mind: The Secret of Human Thought Revealed*, Viking; ISBN: 9780670025299, p. 336.

Rosenblatt Frank (1958), *The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain,* Cornell Aeronautical Laboratory, Psychological Review, vol. 65, No. 6, pp. 386-408, doi:10.1037/h0042519.

Rodney Brooks (January 1991), *Intelligence without representation*, Artificial Intelligence, 47 (1-3), pp. 139-159, CiteSeerX 10.1.1.308.6537, doi:10.1016/0004-3702(91)90053-M.

Robert F. (2014), *Service. A million spiking-neuron integrated circuit with a scalable communication network and interface*, Science. 8 August 2014, vol. 345, No. 6197, pp. 668-673.

Rumelhart D.E., Hinton G.E., Williams R.J. (1986), Learning *Internal Representations by Error Propagation*, [in:] *Parallel Distributed Processing*, Cambridge, MA, MIT Press. vol. 1, pp. 318-362.

Sainath Tara N., Mohamed Abdel-Rahman, Kingsbury Brian, Ramabhadran,Bhuvana (2013), *Deep convolutional neural networks for LVCSR. 2013 IEEE International Conference on Acoustics*, Speech and Signal Processing, pp. 8614-8618, doi:10.1109/icassp.2013.6639347.

Sanelli V., Cashmore M., Magazzeni D. and Iocchi L. (2017), *Short-term human robot interaction through conditional planning and execution*, Proc. of International Conference on Automated Planning and Scheduling (ICAPS).

Satya Nadellaq (2017), *Hit Refresh: The Quest to Rediscover Microsoft's Soul and Imagine a Better Future for Everyone*, HarperBusinessq Language, 304, ISBN-10: 0062959727q ISBN-13: 9780062959720q.

Schmidhuber Jürgen (2015), *Deep Learning*, Scholarpedia, 10 (11), pp. 1527-54, doi:10.1162/neco.2006.18.7.1527.

Schmidhuber J. (2015), *Deep Learning in Neural Networks: An Overview*, Neural Networks, 61, pp. 85-117, arXiv:1404.7828. doi:10.1016/j.neunet.2014.09.003. PMID 25462637.

Schwab K., Davis N. (2018), *Shaping the Future of the Fourth Industrial Revolution*, Penguin UK, 288, ISBN 9781984822611.

Szegedy Christian, Toshev Alexander, Erhan Dumitru (2013), *Deep neural networks for object detection*, Advances in Neural Information Processing Systems, pp. 2553-2561.

Shah H. (2013), *Conversation, deception and intelligence: Turing's question-answer game. Alan Turing: his life and impact, Building a brain: intelligent machines, practice and theory, part III*, Oxford: Elsevier, pp. 614-620.

Shah H, Warwick K. (2010), *Testing Turing's five-minutes, parallel-paired imitation game*, Kybernetes, 39(3), pp. 449-465, doi: 10.1108/03684921011036178.

Shah H., Warwick K. (2010), *Hidden interlocutor misidentification in practical Turing tests,* Mind Mach., 20, pp. 441-454, doi: 10.1007/s11023-010-9219-6.

Shah H., Warwick K., Bland I., Chapman C.D., Allen M.J. (2012), *Turing's imitation game: role of error-making in intelligent thought,* [in:] *Turing in Context II*, Brussels, pp. 31-32.

Simon H.A. (1981), *The Sciences of the Artificial (second edition; first edition 1969),* MIT Press, Cambridge, Mass.

Sloman A. (2014), *Personal communication at the Royal Society. Stanford University Online Encyclopedia of Philosophy,* The Turing test, http://plato.stanford.edu/entries/turing-test/.

Sterrett S.G. (2003)., *Turing's two tests for Intelligence,* [in:] Moor J.H. (ed.), *The Turing test – the elusive standard of artificial intelligence Dordrecht,* Kluwer, pp. 79-97.

Shan N., Ziarko W. (1995), *Data-based acquisition and incremental modification of classification rules,* Computational Intelligence, 11 (2), pp. 357-370, doi:10.1111/j.1467-8640.1995.tb00038.x.

Stanley Cavell (2002), *Knowing and Acknowledging, Must We Mean What We Say*?, Cambridge University Press, pp. 238-266.

Shulman C., Bostrom N., (2012), *How hard is artificial intelligence? Evolutionary arguments and selection effects,* Journal of Consciousness Studies, 19(7-8), pp. 7-8.

Schwab Klaus (2016), *The Fourth Industrial Revolution,* Crown Business, ISBN 978-1-5247-5886-8.

Specht D.F. (1990), *Probabilistic Neural Networks and the polynomial adaline as complementary techniques for classification,* IEEE Trans. Neural Networks, Jan, pp. 111-121.

Specht D.F. (1990), *Probabilistic neural networks,* Neural Networks, vol. 3, pp. 109-118.

Specht D.F. (1992), *Enhancements to the probabilistic neural networks,* Proc IEEE Int Joint Conf. Neural Networks, pp. 761-768.

Stuart J. Russell, Peter Norvig (2003), *Artificial Intelligence: A Modern Approach (2nd ed.),* Upper Saddle River, New Jersey: Prentice Hall, 24, ISBN 0-13-790395-2.

Traiger S. (2000), *Making the right identification in the Turing test,* Mind Mach, 2000,10, pp. 561-572, doi: 10.1023/A:1011254505902.

Tkachenko Volodymyr, Kuzior Aleksandra, Kwilinski Aleksy (2019), *Introduction of artificial intelligence tools into the training methods of entrepreneurship activities,* J. Entrep. Educ., 22(6), pp. 1-10.

Turing A. (1950), *Computing machinery and intelligence mind,* 59, pp. 433-460.

Warren S. McCulloch, Walter Pitts (1943), *A logical calculus of the ideas immanent in nervous activity,* Bulletin of Mathematical Biology, New York: Springer New York. vol. 5, № 4, pp. 115-133.

Werbos Paul J. (1994), *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting,* New York: John Wiley & Sons, ISBN 0-471-59897-6.

Werbos P. (1990), *Backpropagation Through Time: What It Does and How to Do It,* Proceedings of the IEEE. 78 (10), pp. 1550-1560, doi:10.1109/5.58337.

Wong Bo K., Monaco John A. (1995), *Expert system applications in business: a review and analysis of the literature,* Information and Management, 29 (3), pp. 141-152, doi:10.1016/0378-7206(95)00023-p.

Yanase J., Triantaphyllou E. (2019), *A Systematic Survey of Computer-Aided Diagnosis in Medicine: Past and Present Developments,* Expert Systems with Applications, 138, p. 112821, doi:10.1016/j.eswa.2019.112821.

European Funds
Knowledge Education Development

Republic of Poland

NAWA
POLISH NATIONAL AGENCY FOR ACADEMIC EXCHANGE

European Union
European Social Fund

Research, preparation of materials and preparation of the textbook were carried out under the project – grant no. PPI/KAT/2019/1/00015/U/00001 "Cognitive technologies – second-cycle studies in English" and were carried under the KATAMARAN program Polish National Agency for Academic Exchange (NAWA). The program is co-financed by the European Social Fund under the Knowledge Education Development Operational Program, a non-competition project entitled "Supporting the institutional capacity of Polish universities through the creation and implementation of international study programs" implemented under Measure 3.3. Internationalization of Polish higher education, specified in the application for project funding no. POWR.03.03.00-00-PN 16/18.

Wydawnictwo TYGIEL

www.wydawnictwo-tygiel.pl

ISBN 978-83-66489-58-5

9 788366 489585