applied sciences

MDPI

Article

Information-Cognitive Concept of Predicting Method for HCI Objects' Perception Subjectivization Results Based on Impact Factors Analysis with Usage of Multilayer Perceptron ANN

Andrii Pukach [1,*], Vasyl Teslyuk [1], Nataliia Lysa [1], Liubomyr Sikora [1] and Bohdana Fedyna [2]

[1] Department of Automated Control Systems, Computer Science and Information Technologies Institute, Lviv Polytechnic National University, 79013 Lviv, Ukraine; vasyl.m.teslyuk@lpnu.ua (V.T.); nataliia.k.lysa@lpnu.ua (N.L.); liubomyr.s.sikora@lpnu.ua (L.S.)

[2] Department of Computer Technologies in Publishing and Printing Processes, Institute of Printing Art and Media Technologies, Lviv Polytechnic National University, 79013 Lviv, Ukraine; bohdana.i.fedyna@lpnu.ua

* Correspondence: andriipukach@gmail.com

Abstract

An information-cognitive concept of a predicting method for obtaining specialized human–computer interaction (HCI) objects' perception subjectivization results, based on impact factors analysis, with the use of multilayer perceptron (MP) artificial neural networks (ANNs), has been developed. The main purpose of the developed method is to increase the level of intellectualization and automation of research into relevant processes of HCI objects' perception subjectivization, especially in the context of software products' comprehensive support processes. The method is based on the developed conceptual models and the developed mathematical model, as well as a specialized developed algorithm. Results prediction is carried out on the basis of a preliminary in-depth analysis of a set of unique direct chains (UDCs) of neurons of the relevant encapsulated MP ANN, built on the basis of researching the results of isolated influences of each of the previously declared impact factors and further comparing the present direct chain (of each separate investigated modeling case) with UDCs from the aforementioned sets. As an example of practical approbation of the developed method, the appropriate practically applied problem of identifying a member of the support team, whose multifactor portrait is as close as possible to the corresponding multifactor portrait of a given client's user, has been resolved.

Keywords: human–computer interaction; comprehensive software support; impact factors; interaction objects' perception subjectivization; subjectivization results forecasting

Academic Editor: Douglas O'Shaughnessy

Received: 23 July 2025
Revised: 22 August 2025
Accepted: 28 August 2025
Published: 5 September 2025

Citation: Pukach, A.; Teslyuk, V.; Lysa, N.; Sikora, L.; Fedyna, B. Information-Cognitive Concept of Predicting Method for HCI Objects' Perception Subjectivization Results Based on Impact Factors Analysis with Usage of Multilayer Perceptron ANN. *Appl. Sci.* **2025**, *15*, 9763. https://doi.org/10.3390/app15179763

1. Introduction

The problem of HCI and human–machine interaction (HMI) has become increasingly relevant, as both the quantity and the diversity, as well as the complexity and the quality of these interactions, are constantly growing, which is due, among other things, to the increasing level of complexity of the computer component of these interactions. At the same time, HCI is an extremely complex concept that includes many examples of specific interaction(s) between the human and the computer. In particular, within the framework of this research, one of the most common and relevant areas of HCI is explored, which is represented by the interaction between a person (interaction subject) and a software product (interaction object), in the context of comprehensive support of the latter. However, it is worth noting that the obtained research results are valid for any components or manifestations of HCI

and are not limited exclusively to the direction of comprehensive support of software products; however, this exact direction is considered in detail as it is the most comprehensive, clear and relevant example of HCI. At the same time, in the context of this direction, one of the most relevant scientific and applied problems nowadays is, undoubtedly, the problem of automation and intellectualization of software products' comprehensive support. It should be also noted that, in the context of this research, the focus is not only on customer service/support of software products, but rather about comprehensive support of the latter, taking into account all the processes surrounding the supported software product (as a "computer" component of HCI) during its interaction with a wide variety of subjects (as a "human" component of HCI), which covers representative units of subjects both from the developer-company side (i.e., programmers, testers, engineers, etc.), as well as the customer (i.e., service recipients, clients, end users, etc.) side.

HCI/HMI issues are investigated in lots of relevant studies, including the following examples: In [1], the authors have carried out a study of interactive HMI design in the context of adaptive automation, which improves human–machine coevolution, situational awareness, and supports appropriate cognitive and ergonomic standards; in [2], the authors have studied the issues of a scalable HCI automation; in [3], the authors have reviewed the issues of HCI design methodologies in the context of modern technologies of mobile and cloud computing, the Internet of Things, augmented and virtual reality, and other organizational information systems; in [4], the authors have explored the issues of HCI in the context of chatbots as one of the most relevant implementations of artificial intelligence (AI). Regarding the issues of software products' comprehensive support (as one of the most relevant HCI manifestations), component automation and intellectualization, there are also certain significant developments in this direction (involving the usage of AI technologies as well), which include concepts such as software testing automation, DevOps automation, automation of registration and processing of requests (both internal and/or external), decision-making automation, etc. In particular, software testing automation is one of the basic historical directions of automation (in the context of corresponding component of complex interaction with a software product), described in many works, among which, as an example, we can highlight recent works [5–7] that explore this topic quite comprehensively. Also, software testing automation is quite closely related to the Agile methodology (presented, in particular, in various comprehensive works [8–10]), where the main task is to ensure compliance with the pace of software product development. Along with this, another relevant direction of automation is DevOps, which is one of the most popular practices of interaction between the developers and the teams of information and technological maintenance of software products nowadays, and various comprehensive works in this field have been recently published [11–16].

Each of these directions solves automation problems exclusively in the context of its prerogative, area of responsibility, tasks, commercial interests, goals, etc. While at a more global level, in the context of automation and intellectualization of both software products' comprehensive support and HCI in general, another problem remains neglected, unresolved, and at the same time extremely relevant, which consists of taking into account and consideration the factor of perception subjectivization of the object(s) of interaction by the subjects of this same interaction. This factor of perception subjectivization, caused by the influence of various impact factors, affects absolutely all the processes described above, introducing appropriate (sometimes even extremely significant) adjustments into them. In turn, the described problem also includes a number of additional scientific and applied issues of various etymology and nature. However, in the context of this research, a specific scientific and applied task of researching and predicting the results of subjective perception

of the objects of software products' comprehensive support components (in scope of their automation and intellectualization) is considered.

2. Literature Review

Continuing the analysis of research in the field of HCI components' automation and intellectualization issues, especially in the area of software products' comprehensive support, we will consider the direction of automated decision-making systems, which are also part of a global processes of automation of software products' comprehensive support as a component of HCI. Classical decision-making systems are quite often based on a multicriteria optimization [17], while the context of software products' comprehensive support makes it necessary to consider additional (including human etymology) impact factors [18]. In particular, research [19] raises the issue of AI technologies spread in a wide variety of human activities and applications areas, as well as in the context of appropriate decision-making automation. In line with the scope of a previous work [20], the issues of decision-making modeling using the corresponding Decision Making and Notation (DMN) models are comprehensively disclosed, with a description of decision modeling techniques and aspects of coordination with a decision-making management and a digital transformation, implemented in the form of a chat-bot. The authors of [21] discuss the topic of decision-making automation in the field of human resources, which also has a significant impact on the comprehensive support of software products, especially in the context of DevOps interaction. A study [22] highlights research on how big data analytics can be integrated into the support of the decision-making process. In turn, the authors of [23] discuss aspects of the transition from automation to full automaticity (autonomy) in the field of robotic products and usage of AI technologies. The concept of implementing a "homo oeconomicus" as a generalized theoretical agent for usefulness maximization (not only from a classical economic point of view, but also from the point of view of software product(s) support rationalization by using modern capabilities and technologies) is presented in Ref. [24] in the form of automated solution(s). On the other hand, Ref. [25] considers an example of decision-making processes' automation using a specialized method, which includes the designing of a combined approach that involves the following two components: the rules of fuzzy logic, as well as the algorithms for data classification, where the first component detects and works out imprecise information, and the second component determines regularities or trends in the data, while the cooperative nature of the proposed approach ensures that each method complements the other, leading to a more reliable and effective decision-support systems and platforms.

Another direction of the automation of software products' comprehensive support components is registration and processing of requests and issues (both "internal", i.e., generated inside the development company, as well as "external", i.e., originated from outside of the development company). In particular, one of generalized examples of incident monitoring automation is presented in Ref. [26], where the authors have developed a corresponding model that uses Prometheus together with the Alert-manager in order to integrate an operational monitoring system via the ServiceNow for real-time incident detection, diagnosis, and resolution. In line with the scope of Ref. [27], the authors have developed and presented a transformation-based approach to automate issues' classification in the GitHub repository(-ies) by assigning them appropriate labels, and the main feature of this approach is the presence of more than one label for each issue report, as well as the usage of the RoBERTa neural network. In addition, Ref. [28] presents a method for automatic generation of the titles for registered bugs and corresponding reports, providing the possibility of focusing attention on critical problems mentioned in the titles. Continuing this topic, Ref. [29] presents a study demonstrating the ability of GPT-like models to correctly

(and automatically) classify problem reports in situations of labeled data absence, which are required for fine-tuned BERT-like Large Language Models (LLMs). In turn, Ref. [30] demonstrates the capabilities of a specially designed tool for mining automation, as well as analysis and visualization of relevant artifacts directly related to the problems/issues, based on information obtained from open-source repositories. In addition, Ref. [31] presents an automated approach, which provides the ability to recommend potential developers for a certain type, or description, of problems, combining two classical tasks of assigning developers and distributing problem types, and ensuring their solution (as a whole) simultaneously within the framework of the proposed multi-task learning approach. At the same time, Ref. [32] presents a model of an automated problem delegator, which performs automated assessment of problems' delegation correctness, thereby preventing incorrect problem delegating (which is a quite common problem in the field of software products' support). On the other hand, the authors of Ref. [33] have investigated the relevant issue of "under-development" software projects' duration assessment intellectualization based on the analysis of available resources in Scrum teams with differentiated specialization and/or separation.

Thus, in all above-described directions, an increasing level of AI technologies (including, in particular, ANN, Machine Learning (ML), etc.) application is observed. The AI technology itself (especially in the context of its practical application possibilities) is fully represented and described in Refs. [34–38]. Currently, AI is already being widely used far beyond the classical areas of exclusively the IT industry. For example, according to the data presented in Ref. [39], 71.4% of companies rely on AI in making business decisions. In a fairly comprehensive study, representing the experience of using AI in the IT services industry [40], the authors have revealed three key areas of research, which include obtaining useful information from the operational data in order to automate service management and improve human decision-making; obtaining and improving expert knowledge throughout the life cycle from solution development to continuous service improvement; providing a self-service for service requests and problem solving by means of using intuitive natural language interfaces. Ref. [41] describes the usage of AI to automate the DevOps processes in the field of software development, while Ref. [42] discusses a possibility of transitioning from an automated software testing to fully automatic testing, mainly through the usage of AI technologies. At the same time, AI is also used in the Agile project management processes, as demonstrated in Ref. [43]. Another popular area of AI practical application is chatbots, with practical usage examples presented in Refs. [44–49]. In addition, a separate large-scale area, where AI-based automation is being actively used in the scope of the software products' development and the HMI is robotics, with recent research results presented in Refs. [50–53], which emphasize the importance of AI technologies' implementation and application, as well as the appropriate methodologies in this area, in order to increase the level of automation and intellectualization of the relevant HMI systems.

Thus, as it can be seen from the overview of studies presented above, automation in the context of software products (including those integrated into the HCI/HMI systems) is quite a complex concept, which includes an extremely large number of diverse manifestations, each of which, in fact, are already being researched and developed quite autonomously. Along with this, there is a need to research the processes of software products' support automation at a level that would generalize each of the described areas. An example is the level of HCI objects' (in particular, in the given context of software products' comprehensive support: supported software products, or processes of their comprehensive support that actually act as such objects) perception subjectivization by the relevant interaction subjects, which directly or indirectly interact with these objects.

3. Materials and Methods: A General Concept

Therefore, as already mentioned above, there is a need for the development of some generalized conceptual representation of HCI objects' subjective perception processes (particularly in the context of software products' comprehensive support), one of which may be any relevant conceptual or simulation model dedicated to the representation of HCI objects' subjective perception by various subjects interacting with these objects. Let us define a conceptual model of HCI objects' subjective perception.

The main requirement for such a conceptual model is mandatory presence of input characteristics of the researched HCI object; output results of its subjective perception; impact factors that transform the input characteristics of the HCI object into the resulting perception of this same object (by each separate corresponding interaction subject). At the same time, the form of representation of such a conceptual model can be fully arbitrary. In particular, Figure 1 below shows the structure of a conceptual model (of HCI objects' subjective perception) example, presented in a graphical representation/visualization form.

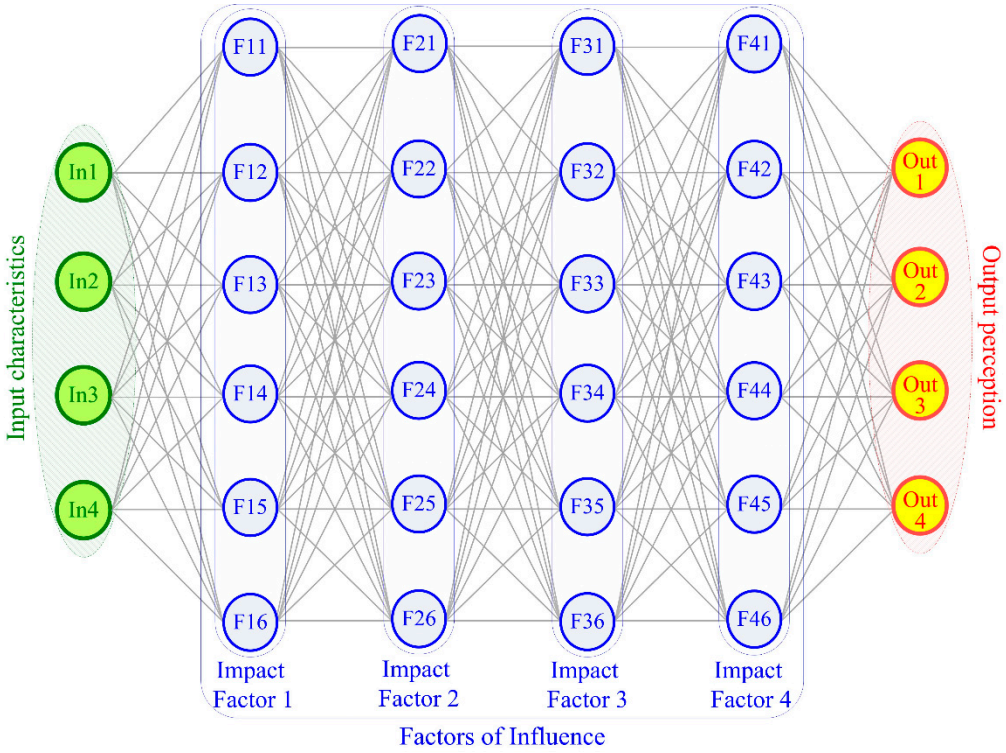

Figure 1. The structure of a conceptual model of HCI objects' subjective perception.

Thus, the proposed conceptual model (of HCI objects' subjective perception) acts as the primary material for further research and refinements, one of which is, in particular, encapsulation into this model, for example, an MP ANN (or in the general case, ANN of any other existing types or species), in order to ensure the necessary level of automation and intellectualization of the researched processes. While ANNs have become one of the basic tools of AI technology, MP itself is one of the most popular, effective, and efficient types of ANN. At the same time, inside the encapsulated MP ANN, the neurons of the input layer interpret the input characteristics of the researched object; the neurons of the output layer interpret the output results of the researched object's subjective perception, while the neurons of the hidden layers interpret the aforementioned impact factors. After the encapsulation of the MP ANN (into a conceptual model of researched HCI object's subjective perception), it is necessary to carry out the entire "standard" (for any MP ANN) set of activities, which consists, in particular, of its training on the corresponding, previously

prepared, general dataset, which represents specific cases of sets of the input characteristics and the output results of the researched HCI object's subjective perception.

Figure 2 below demonstrates an example of a trained MP ANN (trained by means of the R-system (version 3.6.3), and visualized by means of its standard built-in "plot" function) encapsulated into the relevant conceptual model of the researched HCI object's subjective perception.

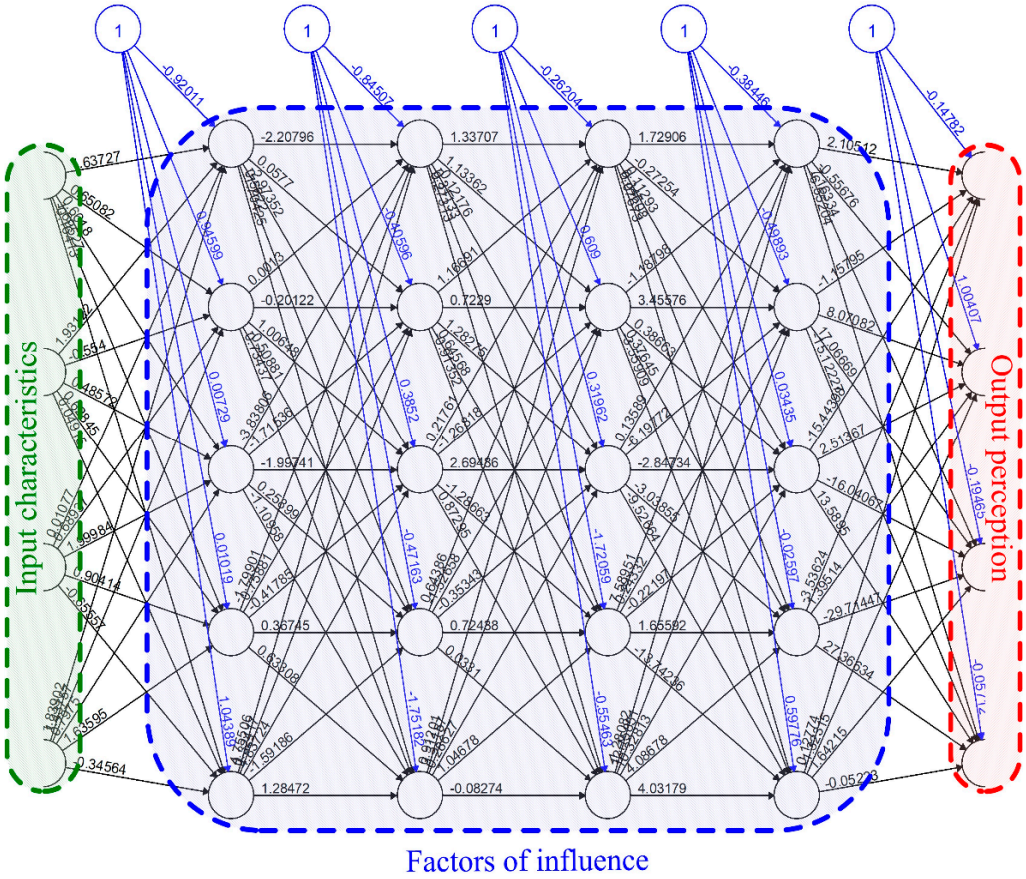

Figure 2. An example of a trained MP ANN encapsulated into the relevant conceptual model of the researched HCI object's subjective perception.

Among the possible nuances, the following should be noted. An important feature of the proposed approach is that the boundaries of the impact factors present inside the conceptual model (of the researched HCI objects' subjective perception) are blurred as a result of the encapsulation of the MP ANN. That is, before the encapsulation of MP ANNs, we could clearly separate the impact factors from each other (as can be seen in Figure 1). While after the encapsulation of MP ANNs, we, unfortunately, cannot confidently and unambiguously state that the neurons of the first hidden layer of the encapsulated MP correspond to impact factor 1 (or any other impact factors), the neurons of the second hidden layer of the encapsulated MP correspond to impact factor 2 (or any other impact factors), etc. (by analogy with the structure of the conceptual model described in Figure 1).

Thus, the previous distribution of the impact factors' boundaries (which can be clearly seen in Figure 1) can no longer be correct, since the concept of the MP itself does not provide any functional or semantic meaning for the neurons of the hidden layers because their main function is pure mathematical and procedural, so they provide exclusively the functions of training and correct functioning of the MP itself. Therefore, this nuance should be taken into account in cases where it is important and has an impact on further research. However, in the context of this specific research, the main focus of attention is not on the boundaries

of the impact factors, and therefore the described effect of blurring these boundaries will not hinder further research.

4. Method Development

The development of the declared method (for HCI objects' perception subjectivization result prediction) has been carried out on the basis of obtained conceptual models of HCI objects' subjective perception with additionally encapsulated MP ANN, and includes the following two mandatory key stages. In particular, one of these stages is the development of a specialized algorithm for HCI objects' perception subjectivization result prediction based on the impact factors, which provides the possibility of structuring the components of the researched processes, as well as providing the needed algorithmic supply for their further programming/software realization and modeling. Another stage is the development of an appropriate mathematical model (for HCI objects' perception subjectivization result prediction based on the impact factors), which provides the possibility of representation formalizing and interpretation of the researched processes in a mathematical base, including the context of implementing their further mathematical modeling. At the same time, both these key stages together provide the possibility of further software/programming implementation and computer modeling of the researched processes of HCI objects' perception subjectivization result prediction.

4.1. Development of a Specialized Algorithm for HCI Objects' Perception Subjectivization Result Prediction Based on Impact Factors

Figure 3 below presents a step-by-step flowchart of the developed specialized algorithm, which reflects all the processes needed for the comprehensive prediction of the researched HCI objects' perception subjectivization result prediction based on the previously declared impact factors.

The functioning of the developed algorithm starts with the initialization of all variables and reading the configuration of the MP ANN encapsulated into the corresponding conceptual model of the researched HCI object's subjective perception. After that, the UDCs' identification function is performed, which forms all the unique combinations (which are possible in the process of functioning of this specific MP ANN based on the available datasets) of the neurons' activation chains, starting with the neurons of the input layer, through the neurons of each of the hidden layers (all of them, starting from the first and up to the last hidden layer), and ending with the neuron(s) of the output layer.

At the same time, it is also important to note the following: it is recommended and advisable to carry out the process of the UDCs' identification at the training stage for each separate encapsulated MP ANN, because at this stage, absolutely all possible (within the framework of the available training datasets) combinations of the chains will be processed. At the stage of post-training testing of already trained MP ANNs, the test sample set theoretically should also contain all the possible combinations (according to all "good practice" rules for preparing and forming such sets for high-quality testing of trained MP ANN), but in practice, it may not contain some separate combinations (from among all the existing ones present during the MP's training procedure execution).

After that, the main cycle of reading records from the set of cases of the available dataset occurs, followed by a layer-by-layer analysis of the activation of neurons, which form a current-specific direct chain, and prediction (based on this specific direct chain) of the results, which should be hypothetically obtained at the output of this specific MP ANN, which are also, in the context of the investigated problematic, the results of the researched HCI object's (for example, the supported software product, or the processes related to its comprehensive support) perception subjectivization.

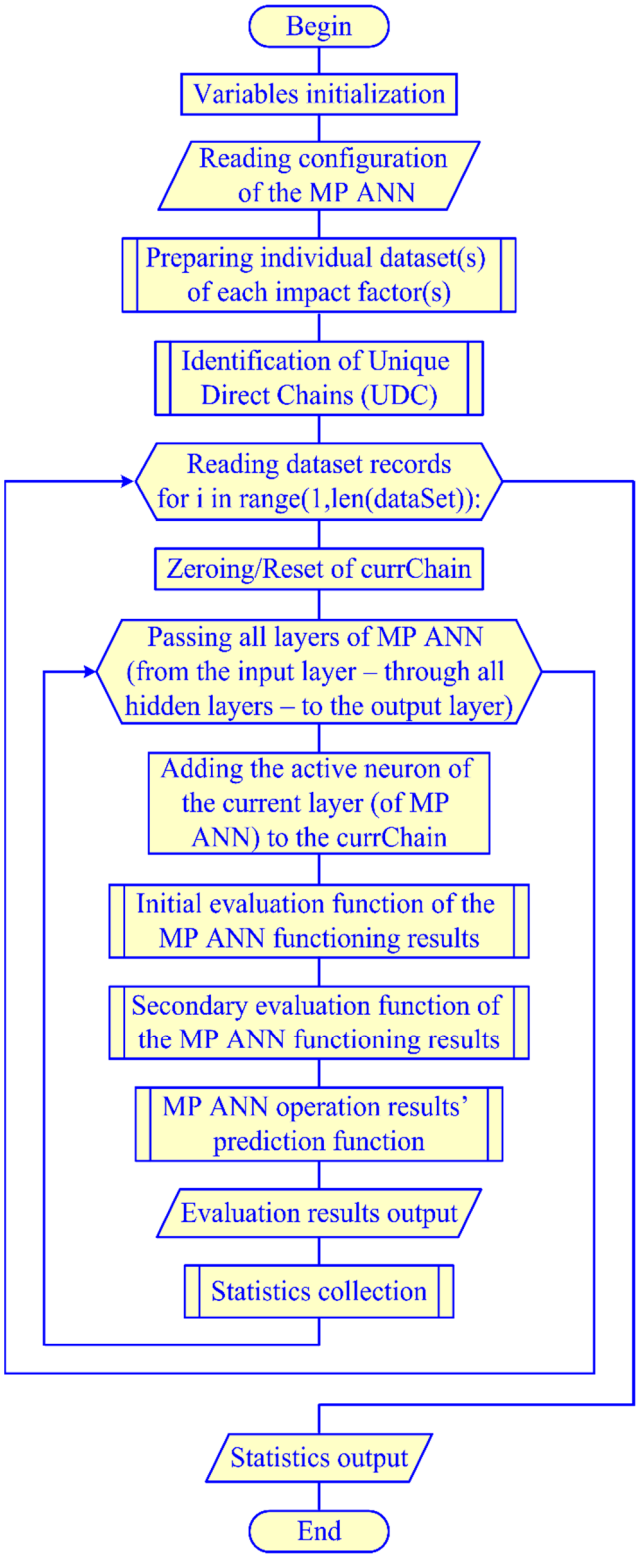

Figure 3. A flowchart of the developed algorithm for HCI objects' perception subjectivization result prediction.

At the beginning of this main cycle, the current direct chain (represented by the appropriate variable named *currChain*) is reset. Next, an internal loop of a layer-by-layer passage of the entire MP begins starting from the input layer, going through all the hidden layers (from the first one to the last one), and ending with the output layer. At the same time, at each step (i.e., at each layer of the considered MP), the corresponding neuron

(of this separate currently considered layer) is activated, which is added to current direct chain *currChain*. Immediately after, the updated current direct chain *currChain* is subjected to a comprehensive analysis by the corresponding specialized developed functions (described below in the following text), each of which corresponds to its own appropriate evaluation/prediction area.

The main purpose of the developed primary (initial) evaluation function (of evaluation the output results of current specific MP's functioning case) is an extremely simple statement of the fact of the presence of a specific impact factor(s) influencing the result of subjective perception (of the researched object) by identifying the current direct chain *currChain* among the previously identified UDCs or their constituent parts. In other words, the initial evaluation function (of the MP's output results) analyzes the presence of the current constructed (both in the middle of construction, as well as already fully constructed) direct chain *currChain*, among each of the previously identified UDCs (including occurrence of the current chain as just a constituent part of any UDC), and in case such a presence/occurrence is detected, the function states/confirms the fact of the presence of the influence of the corresponding impact factor (that exact impact factor, to which the relevant UDC corresponds) on the result of perception subjectivization of the researched object.

The main task of the secondary evaluation function (evaluation of the output results of current-specific MP's functioning case) is a simplified arithmetic calculation of the relative share of influence of each of the impact factors on the final result of perception subjectivization of the researched object. This simplified arithmetic calculation consists of counting the amount of the UDCs (which includes current-specific constructed direct chain *currChain*) for each individual impact factor, with the subsequent calculation of the share of influence of each separate impact factor as the arithmetic share between the amount of occurrences of the *currChain* in only UDCs of this particular impact factor, and the total amount of occurrences of the same *currChain* in all UDCs of absolutely all impact factors in general. Thus, unlike the previous (i.e., initial evaluation) function, this function (i.e., secondary evaluation) not only states/confirms the fact of presence of the influence of some specific impact factor(s) onto the subjectivization perception results, but also, additionally, provides a relative ratio of the share of those specific impact factor(s)' influence on the result of perception subjectivization of the researched object.

In turn, the main mission of the MP's operating final results prediction function is performing, in fact, the final stage of the results prediction, at which a comprehensive influence probability assessment of each of the impact factors on the final result of the researched object's subjective perception is carried out. Thus, unlike the previously considered (i.e., secondary evaluation) function, in this case, it involves a more complex assessment and prediction of the influence of the impact factor(s) onto the final result of subjectivization, based on the appropriate processing of the existing current direct chain *currChain* and available UDCs. Details of all calculations, performed by this final result prediction function, will be presented further in the corresponding Section 4.2 with a detailed description of the mathematical component (model) of the developed method.

After that, according to the developed algorithm, the obtained current intermediate evaluation results are displayed and the necessary collection of relevant statistics is performed in order to obtain additional indicators of the functioning results of the developed method. At this stage, both the internal loop (of a layer-by-layer passage of the entire MP ANN) as well as current iteration of the main cycle (of reading the records from the set of cases of available studied dataset) are completed, and the transition to processing the next test case (from the same current studied dataset) occurs. In turn, upon completion of processing of the entire studied dataset, all necessary statistical information, obtained during the method execution, is available for further analysis in order to identify and

provide, among other things, potential ways and options for possible improvement(s) and enhancement(s) of the developed method.

Thus, in summary, the main idea of the developed algorithm (as well as the proposed predicting method and the relevant information-cognitive concept in general) consists of providing the possibility of predicting the final operating results of a previously designed and trained MP ANN without the need to fully process all its layers of neurons (starting from the input layer, through all the hidden layers, and up to the output layer). At the same time, the MP ANN itself represents the subjectivization of the original characteristics of the investigated HCI object under the influence of previously agreed and declared impact factors, and the relation between these impact factors and the MP ANN neurons is represented by the corresponding UDCs obtained as a result of processing the relevant training dataset(s), which include the results of modeling an isolated dominant influence of each of the impact factors on the subjectivization of the original characteristics of the investigated HCI object.

4.2. Development of the Mathematical Model for HCI Objects' Perception Subjectivization Result Complex Prediction Based on Impact Factors

The basis and the key element of the developed mathematical model (for HCI objects' perception subjectivization result complex prediction) is the current direct chain *currChain* of activated neurons of the researched MP ANN, which can be represented by the following expression:

$$currChain = \cup(activeNeuron[lx]),\ \ lx \in \left[InL, HidL\left[\overline{1,n}\right], OutL\right], \tag{1}$$

where *currChain* is the currently constructed (as well as "in the process of construction") chain of activated neurons of the researched pre-trained MP ANN; $\cup(activeNeuron[lx])$ is a union (chain) consisting of active neurons of the lx layers of the researched MP; lx is an indicator of the current investigated layer (of neurons) of the researched MP; InL is an indicator of the input layer (of neurons) of the researched MP; $HidL\left[\overline{1,n}\right]$ is an indicator of the hidden layers (from the first layer until the n-th layer, where n is the total amount of hidden layers) of neurons of the researched MP; $OutL$ is an indicator of the output layer (of neurons) of the researched MP.

Thus, as it has been already mentioned earlier, *currChain* actually represents a step-by-step chain of neurons activated at each layer of the researched MP, starting from the input layer, continuing through all the hidden layers (in the direct order of their passage, that is: from the first hidden layer, through all the subsequently incremented ones, and up to the last hidden layer of the researched MP), and ending with the output layer of neurons of the researched MP.

The following expression represents the initial evaluation function of the researched MP's functioning case result prediction:

$$prelmFunc[F[j]](currChain) = \begin{cases} 1, currChain \in UDC[q], (UDC[q] \in F[j]) \\ 0, currChain \notin UDC[q], (UDC[q] \in F[j]) \end{cases}, \left(j = \overline{1,m}; q = \overline{1,k}\right), \tag{2}$$

where $prelmFunc[F[j]](currChain)$ is a logical result of execution of the initial evaluation function (for the specific impact factor $F[j]$ and the researched *currChain*), which can obtain the values of a logical "0" (indicating the absence of influence of the specific impact factor $F[j]$ on the researched *currChain*) or a logical "1" (indicating the presence of influence of the specific impact factor $F[j]$ on the researched *currChain*); *currChain* is a currently constructed (as well as "in the process of construction") chain of activated neurons of the researched pre-trained MP ANN; $F[j]$ is the j-th impact factor, which can obtain values in the range

[1...*m*]; *j* is a simple counter, dedicated to listing all the determined predefined impact factors; *m* is the a total amount of the determined predefined impact factors (in the scope of current-specific researched MP); *q* is a simple counter, dedicated to listing all the defined UDCs (in the scope of current-specific researched MP ANN); *UDC*[*q*] is the *q*-th unique direct chain (i.e., UDC), which can obtain values in the range [1...*k*]; *k* is the total amount of all the UDCs (in the scope of current-specific researched MP).

Thus, if the *currChain* is a part of some specific *UDC*[*q*] (which, in turn, belongs to the appropriate impact factor *F*[*j*]), then $prelmFunc[F[j]](currChain) = 1$, while in the opposite case (i.e., if the *currChain* is not a part of this specific *UDC*[*q*]), then, accordingly, $prelmFunc[F[j]](currChain) = 0$.

The next component of the developed mathematical supply (of the developed predicting method and its information-cognitive concept in general) is the mathematical representation model of the secondary evaluation function (of the researched MP's functioning result prediction), represented by the following corresponding expression, given and described below:

$$scondaryFunc[F[j]](currChain) = \frac{count(currChain \in UDC[q])}{count\left(currChain \in UDC\left[\overline{1,k}\right]\right)}, \begin{pmatrix} UDC[q] \in F[j]; \\ j = \overline{1,m}; \\ q = \overline{1,k} \end{pmatrix}, \quad (3)$$

where $scondaryFunc[F[j]](currChain)$ is the obtained result of execution of the secondary evaluation function (for the specific impact factor *F*[*j*] and the researched *currChain*), which is calculated as an arithmetic fraction between the amount of occurrences of *currChain* among the UDCs related to this specific impact factor *F*[*j*], and the total amount of occurrences of this same *currChain* among absolutely all available UDCs (related to absolutely all available impact factors); *currChain* is the currently constructed (as well as "in the process of construction") chain of activated neurons of the researched pre-trained MP ANN; $count(currChain \in UDC[q])$ is the amount of occurrences/presence of the researched *currChain* in those UDCs (among all available UDCs) which are related to this specific impact factor *F*[*j*] (condition $UDC[q] \in F[j]$); $count\left(currChain \in UDC\left[\overline{1,k}\right]\right)$ is the amount of occurrences/presence of the researched *currChain* in all available UDCs (regardless of their affiliation/relation to any particular impact factor); *F*[*j*] is the *j*-th impact factor that can obtain values in the range [1...*m*]; *j* is a simple counter, dedicated to listing all the determined predefined impact factors; *m* is the total amount of the determined predefined impact factors (in the scope of current-specific researched MP); *UDC*[*q*] is the *q*-th unique direct chain (i.e., UDC) that can obtain values in the range [1...*k*]; *q* is a simple counter, dedicated to listing all the defined UDCs (in the scope of current-specific researched MP ANN); *k* is the total amount of all the UDCs (in the scope of current-specific researched MP).

Thus, expression (3) provides the possibility to calculate the relative share of the influence of each of the impact factors on the final result of perception subjectivization of the researched HCI object.

The final component of the developed mathematical model is represented by expression (4), which implements, in fact, the most complex assessment of the probability of influence of each of the impact factors on the final result of perception subjectivization of the researched HCI object:

$$predictFunc[F[j]](currChain) = \frac{\sum_{o=1}^{length(currChain)} (F[j](currChain[o]).globalInflue\%)}{\sum_{o=1}^{length(currChain)} \left(F\left[\overline{1,m}\right](currChain[o]).globalInflue\%\right)}, \quad (4)$$

where $predictFunc[F[j]](currChain)$ is the probability of the influence of the separate specific impact factor $F[j]$ on the final result of the researched HCI object's subjective perception (based on the researched *currChain*), which is calculated as the fraction of the sum of relative values of the global influence of this specific impact factor $F[j]$ performed on each neuron of the researched *currChain* divided by the sum of relative values of the global total influence of all the declared impact factors $F\overline{[1,m]}$ (taken together) performed on each of these same neurons of the same researched *currChain*; *currChain* is the currently constructed (as well as "in the process of construction") chain of activated neurons of the researched pre-trained MP ANN; $F[j](currChain[o]).globalInflue\%$ is the relative value of the global influence of the specific impact factor $F[j]$ performed on the particular o-th neuron of the researched *currChain*; j is a simple counter, dedicated to listing all the determined predefined impact factors; $F\overline{[1,m]}(currChain[o]).globalInflue\%$ is the sum of relative values of the global total influence of all the declared impact factors $F\overline{[1,m]}$ (taken together) performed on the particular o-th neuron of the researched *currChain*; $currChain[o]$ is the o-th neuron of the current direct chain *currChain*; o is a simple counter, dedicated to listing all separate neuron(s) (i.e., $currChain[o]$) of the present researched current direct chain *currChain*; $length(currChain)$ is the total length of current direct chain *currChain*.

Therefore, the main purpose of the aforementioned component (of the developed mathematical model), represented by expression (4), is, actually, a complex assessment of the probability of influence of each of the impact factors on the final result of perception subjectivization of the researched HCI object, which is defined as the arithmetical fraction between the summarized value of indicators of the influence of a specific impact factor $F[j]$ on all neurons of current investigated chain *currChain*, and the summarized value of indicators of the influence of all declared impact factors $F[1...m]$ on all same neurons of the same current investigated chain *currChain*.

At the same time, aforementioned relative values of the global influence of each impact factor(s) performed on all the corresponding neuron(s) of the researched MP (namely those neurons, which, in fact, appear in all possible constructed researched direct chains *currChain* that are also represented by the corresponding declared set of UDCs) are determined (calculated) in advance using expression (5), which is given below:

$$F[j](neur[o]).globalInflue\% = \frac{F[j](neur[o]).localInflue \times F[j].Index}{\sum_{s=1}^{m}(F[s](neur[o]).localInflue \times F[s].Index)}, \quad (5)$$

where $F[j](neur[o]).globalInflue\%$ is the relative value of the global influence of the specific impact factor $F[j]$ performed onto a particular specific investigated o-th neuron, i.e., $neur[o]$ (in the scope of current-specific researched MP ANN); $F[j](neur[o]).localInflue$ is the share of influence of the specific impact factor $F[j]$ onto the specific o-th neuron ($neur[o]$) within the local dataset for this same specific impact factor $F[j]$; o is a simple indicator, dedicated to the definition of the separate specific investigated neuron $neur[o]$; j is a simple counter, dedicated to listing all the determined predefined impact factors; $F[j].Index$ is the index of the impact factor $F[j]$ itself, i.e., the share of presence of this separate specific impact factor among the set of all declared impact factors; $F[j](neur[o]).localInflue \times F[j].Index$ is the arithmetic product between the share of influence of the specific impact factor $F[j]$ (performed onto the specific o-th neuron) and the index of this same specific impact factor $F[j]$, which represents, in fact, the share of the influence of the specific impact factor $F[j]$ onto the specific neuron $neur[o]$ within the global dataset (i.e., for all the declared impact factors); $\sum_{s=1}^{m}(F[s](neur[o]).localInflue \times F[s].Index)$ is an arithmetic sum of the shares of all the declared impact factors within the global dataset; s is a simple counter, dedicated

to enumerating all the declared impact factors from the first to the m-th one (where m is, respectively, the total amount of the declared impact factors).

Therefore, the main purpose of the aforementioned component (of the developed mathematical model), represented by expression (5), is, actually, a calculation of the relative values of the global influence performed by each impact factor(s) on each corresponding neuron(s) of the researched MP ANN, which is defined as the arithmetical fraction between the arithmetic product between the share of influence of the specific impact factor $F[j]$ on the specific o-th neuron ($neur[o]$) within the local dataset for this same specific impact factor $F[j]$ and the index of this same impact factor $F[j]$ itself (i.e., the share of presence of this separate specific impact factor $F[j]$ in the global dataset among all the declared impact factors F $[1...m]$), and the summary of the shares of presence of all the declared impact factors F $[1...m]$ (within the global dataset) on the same specific o-th neuron ($neur[o]$).

In fact, the value of the share of influence of a certain specific impact factor $F[j]$, performed on a certain particular neuron $neur[o]$, is nothing more than the share of belonging of this particular neuron $neur[o]$ to this particular specific impact factor $F[j]$. So, it only remains to describe the expressions for calculating quantities such as the share of a local influence of the impact factor(s); and an index of the impact factor(s).

Thus, the corresponding equations for calculating values of these abovementioned quantities are provided below in expressions (6) and (7), respectively.

$$F[j](neur[o]).localInflue = \frac{count(neur[o] \in (UDC[q] \in F[j]))}{count(neur[u] \in UDC[q])}, \begin{pmatrix} j = \overline{1,m}; \\ u = \overline{1,p}; \\ q = \overline{1,k} \end{pmatrix}, \left(o = \overline{1,p}\right), \tag{6}$$

where $F[j](neur[o]).localInflue$ is the share of influence performed by the specific impact factor $F[j]$ onto the specific o-th neuron ($neur[o]$) within the local dataset for this same specific impact factor $F[j]$; $count(neur[o] \in (UDC[q] \in F[j]))$ is the amount number of occurrences of the specific o-th neuron $neur[o]$ in all those particular UDCs related to the specific impact factor $F[j]$; $count(neur[u] \in UDC[q])$ is the total amount number of occurrences of all neurons in all available UDCs (regardless of the affiliation of these UDCs to any of the defined impact factors); $neur[o]$ is the o-th researched neuron; $neur[u]$ is the u-th neuron while enumerating all the neurons of the UDCs using the corresponding counter $u = \overline{1,p}$; $UDC[q]$ is the q-th UDC, which can obtain values in the range $[1...k]$; $o = \overline{1,p}$ is a simple counter, dedicated to enumerating all the neurons of all the UDCs, for each of which the share value of the local influence of all impact factors (onto each of these neurons) is being calculated; $j = \overline{1,m}$ is a simple counter, dedicated to enumerating all the defined impact factors; $u = \overline{1,p}$ is a simple counter, dedicated to enumerating all the neurons, which belong to the specified $UDC[q]$ (condition $\in UDC[q]$); $q = \overline{1,k}$ is a simple counter, dedicated to enumerating all the defined UDCs; k is the total amount number of all the defined UDCs; m is the total amount number of all the defined pre-declared impact factors; p is the number of neurons that belong to the specified $UDC[q]$ (condition $\in UDC[q]$).

Thus, as mentioned before, expression (6), in fact, represents the affiliation of each neuron (from among all the neurons, which are a part of all the defined UDCs) to each of the declared impact factors, but locally, which means within the scope of the corresponding local datasets of these impact factors.

Therefore, in order to be able to determine the belonging of these same neurons to these same impact factors, but within the basic/general (i.e., "global") dataset, it is necessary to additionally determine the index of each of the impact factors as the share of presence of each of these impact factors within the entire generalized global dataset (without reference to any specific impact factor, but instead in general, i.e., for all of them considered together).

In addition, expression (7) in fact provides such a possibility to determine the indexes of the defined impact factors.

$$F[j].Index = \frac{count(UDC[q] \in F[j])}{count\left(UDC\left[\overline{1,k}\right]\right)}, \left(q = \overline{1,k}; j = \overline{1,m}\right), \quad (7)$$

where $F[j].Index$ is the index of the impact factor $F[j]$ itself (i.e., the share of presence of this separate specific impact factor $F[j]$ in the global dataset among all the declared impact factors F $[1...m]$); $count(UDC[q] \in F[j])$ is the amount of the UDCs related to the specified impact factor $F[j]$; $count\left(UDC\left[\overline{1,k}\right]\right)$ is the total amount of absolutely all the defined UDCs; $q = \overline{1,k}$ is a simple counter, dedicated to enumerating all available defined UDCs; k is the total amount number of all available defined UDCs; $j = \overline{1,m}$ is a simple counter, dedicated to enumerating all the defined impact factors; m is the total amount number of all the defined pre-declared impact factors.

Thus, expressions (1)–(7) represent, in fact, the developed mathematical model of complex prediction of the HCI objects' perception subjectivization results based on impact factors, which makes it possible to predict the final result of the corresponding pre-trained MP's functioning (for each particular investigated modeling case) even before the final completion of this functioning (within this same particular investigated modeling case).

5. Method Modeling and Results Analysis

A modeling of the developed method (for HCI objects' perception subjectivization result prediction based on impact factor analysis with usage of MP ANN) has been carried out by the appropriate specialized software, developed in Python 3.12 [54] using IDE Thonny 4.1.4 [55,56], as well as the R-system (version/release 3.6.3) [57,58] used for pre-training the designed MP ANN, encapsulated into the corresponding conceptual model(s) of the researched HCI objects' subjective perception (in particular, the supported software product(s), or the processes of their comprehensive support as a considered example(s) of such HCI objects).

It should be noted that a model that is not overly complicated or cumbersome was specially chosen as a presented experimental example, precisely so that each successive step of applied use (i.e., modeling) of the developed method could have been explained in maximum detail and quite clearly, including all the intermediate results obtained at each step of its functioning, presented in the relevant tables, as well as a detailed explanation of its software realization implemented by means of the aforementioned software development environment consisting of Python 3.12 and Thonny 4.1.4 IDE.

Thus, Figure 2 presents a corresponding example of a designed and trained experimental MP ANN, which will be considered as an example of all further steps and calculations (in the scope of the developed method's modeling and approbation), as well as the presentation and analysis of the obtained results. This MP ANN has been designed and trained, as already noted earlier, in the R-system 3.6.3 environment.

As input data for training the designed encapsulated MP ANN, a corresponding CSV-file has been used (the structure of which is shown in Figure 4), which consists of a set of values for the input layer's neurons (contained in columns 1–4); a reference mono-value of the result at the model's output, represented by a single decimal (as well as additionally normalized) number (contained in column 5); and a set of reference resulting multi-values for the output layer's neurons (contained in columns 6–9), which, in fact, represent the mono-value(s) (in column 5), transformed from the decimal representation form into a set of multi-values for each separate neuron of the output layer (found in columns 6–9, respectively).

DataForTrainingEncapsulatedANNMPexperimentExampleCase.csv

```
   1  I1,I2,I3,I4,Odecim,O1,O2,O3,O4
   2  0.3,0.3,0.3,0.2,0.1,1,0,0,0
   3  0.4,0.9,0.4,0.6,0.5,0,0,1,0
   4  0.8,0.3,0.3,0.4,0.3,0,1,0,0
   5  0.6,0.6,0.5,0.8,0.5,0,0,1,0
   6  0.6,0.4,0.8,0.5,0.5,0,0,1,0
   7  0.2,0.6,0.5,0.8,0.5,0,0,1,0
   8  0.6,0.4,0.3,0.7,0.3,0,1,0,0
   9  0.6,0.4,0.6,0.7,0.5,0,0,1,0
  10  0.6,0.9,0.9,0.8,0.7,0,0,0,1
  11  0.2,0.1,0.1,0.3,0.1,1,0,0,0
  12  0.2,0.4,0.4,0.9,0.3,0,1,0,0
  13  0.5,0.3,0.6,0.8,0.5,0,0,1,0
  14  0.8,0.7,0.3,0.9,0.5,0,0,1,0
  15  0.5,0.2,0.8,0.8,0.5,0,0,1,0
       . . .
       . . .
       . . .
4191  0.7,0.5,0.5,0.2,0.3,0,1,0,0
4192  0.8,0.7,0.1,0.5,0.5,0,0,1,0
4193  0.6,0.5,0.1,0.1,0.3,0,1,0,0
4194  0.1,0.3,0.2,0.3,0.1,1,0,0,0
4195  0.9,0.6,0.1,0.4,0.3,0,1,0,0
4196  0.4,0.7,0.4,0.3,0.3,0,1,0,0
4197  0.6,0.7,0.4,0.4,0.5,0,0,1,0
4198  0.4,0.7,0.3,0.9,0.5,0,0,1,0
4199  0.2,0.3,0.3,0.3,0.1,1,0,0,0
4200  0.1,0.7,0.7,0.5,0.3,0,1,0,0
4201  0.5,0.4,0.6,0.2,0.3,0,1,0,0
```

Figure 4. The structure of a CSV file with input data for training of an encapsulated MP ANN.

The structure of the dataset(s), regardless of whether it is used for MP training, for structuring data that represent an isolated absolute dominant influence of only one separate specific impact factor (with absolutely minimized influence of the other factors), or for further MP modeling simulation of the researched processes of HCI object's perception subjectivization, is always the same, and consists of two main categories of the data: the first category contains the values of all the neurons of the input layer of the considered MP ANN (as can be seen from the dataset provided in Figure 4, which includes the values from columns "I1", "I2", "I3" and "I4" that correspond to the relevant neurons of the input layer of an example MP ANN considered in this research); the second category contains the expected resulting values of all the neurons of the output layer of the same considered MP ANN, which we expect to obtain as a result of its functioning (as can be seen from the dataset provided in Figure 4, which includes values from columns "Odecim", "O1", "O2", "O3" and "O4", where "Odecim" is just a "one-value" result (represented by one single decimal value, which replaces the batch of values present in columns "O1", "O2", "O3" and "O4") that is used for testing purposes only in order to investigate the possibility of dataset optimization. It has already designed and introduced, but will only be used in further research and not in the current research. In the other columns, i.e., "O1", "O2", "O3" and "O4" (which are used as the main result fields and are considered in the scope of current research presented in this paper) correspond to the relevant neurons of the output layer of the example MP ANN considered in this research.

The main characteristics of the considered dataset (used as a relative example within the framework of the research presented in scope of this paper) are as follows. In particular,

the sample size of an exemplary basic MP training dataset is 4200 records allocated for the training of the considered (and not too complicated) exemplary MP ANN, while the other dataset (i.e., modeling) consists of another 500 records allocated for modeling of the same considered, but this time already trained, exemplary MP ANN. In general, the dataset train/test split ratio, used in the scope of this research, fully corresponds to the limits of suggested (quite popular and widely used by most researchers while training and testing a wide variety of different MP ANNs) distribution ratio, according to which the size of a training subset should be about ~90% of the total size of both training plus testing subsets together, while the size of a testing (e.g., modeling) subset should be about ~10% of the total size of both training plus testing subsets together.

We would also like to pay additional attention to the fact that the quality and the informativeness of the dataset are not so directly affected by the total amount of records contained inside it, but rather by the coverage area of all possible options and variations of the UDCs, which, subsequently, will be later used to predict the modeling results of the corresponding considered MP ANN. However, it is also worth noting that an increase in the total amount of records of a dataset still contributes (to a certain extent) to increasing the probability of detecting additional possible options and variations of the UDCs, but not always proportionally and reliably.

The input-feature dimensions, in the case of the exemplary basic dataset, fully correspond to the dimensionality of the input and the output layers of neurons of the corresponding related MP ANN, according to which each dataset record contains four values for the input layer neurons (marked by the corresponding columns “I1”, “I2”, “I3” and “I4” in the dataset) and four values for the output layer neurons (marked by the corresponding columns “O1”, “O2”, “O3” and “O4” in the dataset). There is also one additional intermediate value located between the aforementioned input and output groups of values; it is marked with the column “Odecim” and, as already being mentioned earlier in the previous paragraph, it is actually not used within the framework of this separate research, but it should have been already designed and implemented at this stage of investigation for the possibility of further research into the relevant optimization of the dataset for better scalability.

Regarding the dimensionality of the input-feature values (of the considered exemplary dataset) themselves, as will be additionally noted below in the following paragraph, it is absolutely normalized, as a result of which, in particular, the values in the columns “I1”, “I2”, “I3” and “I4” can vary in the range [0.1–0.9] (with step = 0.1, as an example), while the values in the columns “O1”, “O2”, “O3” and “O4” can only acquire the binary values “0” or “1”.

Also, Table 1 below provides distribution statistics of the used dataset for both its training and modeling parts (subsets).

In addition, Figure 5 below demonstrates the same distribution statistics in a more visually appealing histogram form of representation.

Additionally, we would like to emphasize that the data in a CSV file(s) (i.e., the dataset) should be presented in a completely depersonalized normalized representation form. Depersonalization is an extremely important, and even mandatory, condition for working with any sensitive and/or confidential information. On the other hand, normalization is a no less mandatory or important stage of preparing input data for further correct processing of these data by any relevant MP ANN, designed in the scope of the proposed approach.

Table 1. Distribution statistics of used dataset for both training and modeling parts.

Distribution statistics of training part of the dataset				
Value	I1	I2	I3	I4
0.1	536	548	567	456
0.2	494	508	476	481
0.3	487	506	459	420
0.4	432	401	446	394
0.5	425	439	417	448
0.6	430	431	397	454
0.7	402	432	447	434
0.8	491	429	467	505
0.9	503	506	524	608
Value	**O1**	**O2**	**O3**	**O4**
0	3402	2749	2979	3470
1	798	1451	1221	730
Distribution statistics of modeling part of the dataset				
Value	**I1**	**I2**	**I3**	**I4**
0.1	70	70	54	5
0.2	60	66	62	17
0.3	60	70	55	30
0.4	54	58	67	45
0.5	54	62	61	60
0.6	69	50	62	82
0.7	47	54	45	79
0.8	48	35	39	88
0.9	38	35	55	94
Value	**O1**	**O2**	**O3**	**O4**
0	474	260	298	468
1	26	240	202	32

In summary, depersonalization and normalization play the role of those main "compensators" that ensure the validity of the developed approach in the context of processing the vast majority of various existing datasets, because any of these datasets, after applying the abovementioned depersonalization and normalization approaches to them, will obtain a similar data representation, with the difference only in the discreteness step of their change in values (but always with a constant range of values). It is for this reason that the preliminary depersonalization and normalization of the input data of any real existing dataset(s) (before their further processing by any appropriate MP ANN, designed in the scope of the proposed approach) is an extremely important, mandatory and critical condition for the correct functioning, as well as the robustness, of the proposed approach as a whole. It is for this reason that, in the context of data processing and results representation, performed within the framework of this research and presented in this paper, such an exemplary dataset has been used and demonstrated, which fully meets the specified requirements for the data depersonalization and normalization. In addition, as it has been already mentioned earlier, the generalized effectiveness and efficiency of the developed approach mostly depend not only on the total amount of data present in the prepared and processed datasets, but also on the presence of a greater number of UDC variations among all these data present inside the prepared and processed dataset(s). In turn, a greater number of the UDCs directly depend on the relevant unique combinations of the input data that which may be located, in particular, at the edges of the relevant data sample distribution, which may be mistakenly perceived by the standard methods of statistical

data processing as a statistical error. Therefore, it is also worth noting that the use of any additional (including, for example, statistical) methods of processing the datasets should be carried out only when taking into consideration the mandatory need to implement an additional verification of the differences between the statistical errors in the data and the presence of the unique input combinations in the scope of these same data, in order to prevent the latter from an accident loss, ensuring the future possibility of identifying a greater number of relevant available UDCs in the scope of the considered dataset(s).

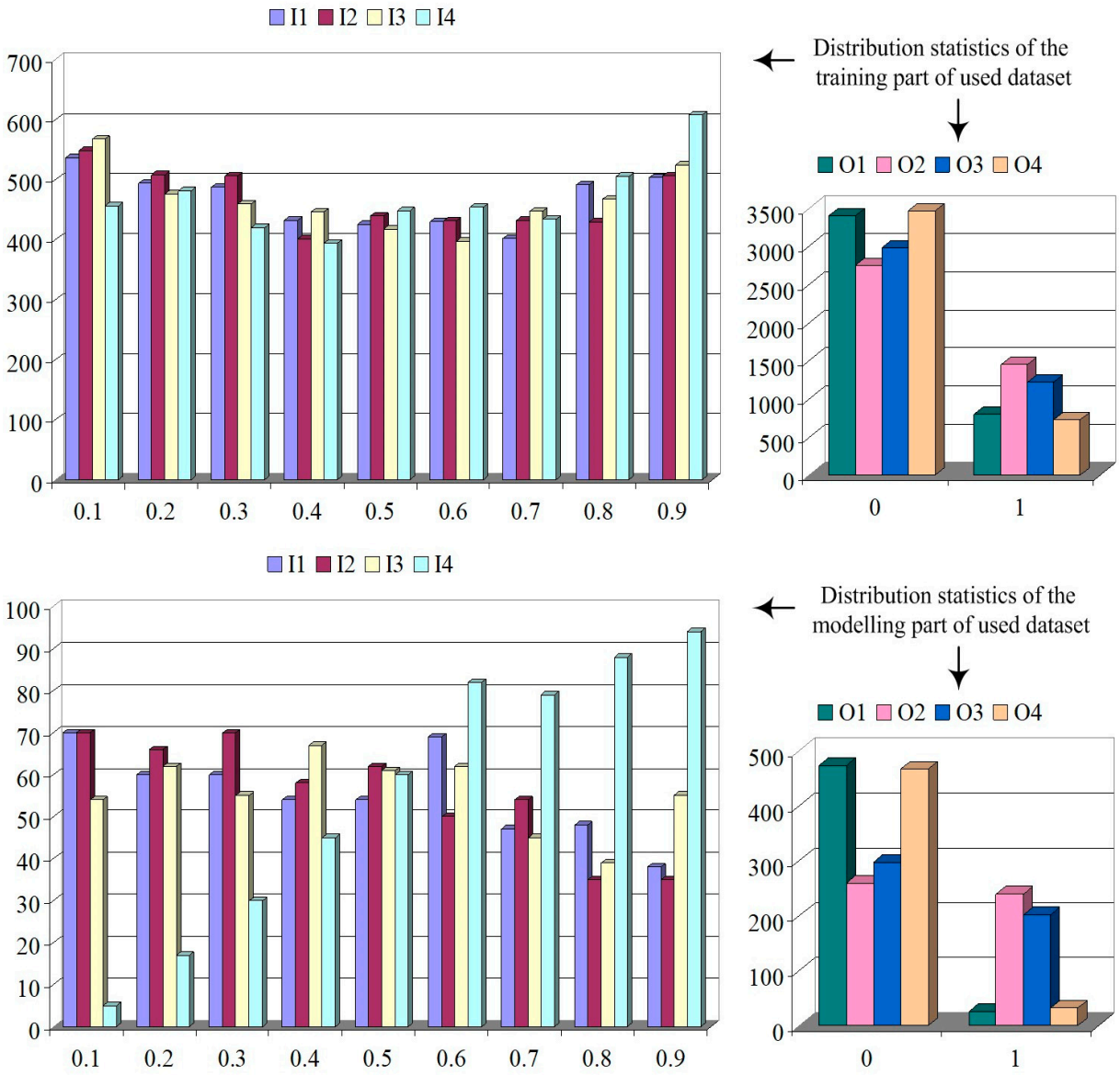

Figure 5. A visualization of the distribution statistics of used dataset.

In addition, it is also necessary to prepare a separate training dataset(s) for each of the previously agreed and declared impact factors. The structure and the format of each of these datasets are absolutely identical to the one presented above in Figure 4. The only difference between these datasets and the general training dataset is that each individual dataset (of the corresponding separate individual specific impact factor) should display the data with corresponding input and output values, reflecting the maximum (absolutely dominant and isolated) influence of only this single separate specific impact factor on the result(s) of perception subjectivization of the researched object, while the influence of the rest of the impact factors should be minimized as much as possible (ideally, it should be

decremented to an "absolute zero" level). That is, for the specific example of the designed MP ANN (encapsulated into the relevant conceptual model of perception subjectivization of the researched object) at the output stage, we should obtain four separate datasets for four declared impact factors.

Next, according to the algorithm of the proposed approach, the already trained MP ANNs were tested on each of these individual datasets for each of the separated impact factors, and based on the results of this testing, the following components were obtained:

- The UDCs;
- The index value(s) for each of the declared impact factors;
- The values of the local influence of each of the declared impact factors onto each neuron(s) of each of the available UDCs (within the local datasets for each separate considered impact factor);
- The values of the global influence of each of the impact factors onto each neuron(s) of each of the available UDCs (beyond just an individual dataset of each separate impact factor).

Accordingly, for the considered experimental example of the method's modeling, the following UDCs have been obtained, which are presented below in Table 2.

Table 3 below presents the index values for each of the declared impact factors, obtained based on the dataset's analysis.

Table 4 below presents the obtained values of the local influence of each of the declared impact factors on each neuron(s) of each of the available UDCs (within the local datasets for each separate impact factor).

Thus, with the values of the local influence for each of the declared impact factors, as well as the index values for each of these same impact factors, presented in the corresponding abovementioned Tables 2 and 3, it becomes possible to obtain the values of the global influence of each of the declared impact factors on each neuron(s) of each of the available UDCs (beyond just an individual dataset of each separate impact factor), which are presented below in Table 5.

Table 2. The UDCs, obtained for the considered experimental example of the method's modeling.

Impact Factor	UDC
Factor 1	I [0]–>HLN [0][1]–>HLN [1][2]–>HLN [2][3]–>HLN [3][4]–>O [0] I [0]–>HLN [0][3]–>HLN [1][2]–>HLN [2][3]–>HLN [3][4]–>O [0] I [1]–>HLN [0][1]–>HLN [1][2]–>HLN [2][3]–>HLN [3][4]–>O [0] I [1]–>HLN [0][3]–>HLN [1][2]–>HLN [2][3]–>HLN [3][4]–>O [0] I [2]–>HLN [0][1]–>HLN [1][2]–>HLN [2][3]–>HLN [3][3]–>O [0] I [2]–>HLN [0][1]–>HLN [1][2]–>HLN [2][3]–>HLN [3][4]–>O [0] I [2]–>HLN [0][3]–>HLN [1][2]–>HLN [2][3]–>HLN [3][4]–>O [0] I [3]–>HLN [0][1]–>HLN [1][2]–>HLN [2][3]–>HLN [3][4]–>O [0] I [3]–>HLN [0][3]–>HLN [1][2]–>HLN [2][3]–>HLN [3][4]–>O [0] I [3]–>HLN [0][4]–>HLN [1][2]–>HLN [2][3]–>HLN [3][4]–>O [0]
Factor 2	I [0]–>HLN [0][1]–>HLN [1][2]–>HLN [2][3]–>HLN [3][1]–>O [1] I [0]–>HLN [0][3]–>HLN [1][2]–>HLN [2][3]–>HLN [3][1]–>O [1] I [0]–>HLN [0][4]–>HLN [1][2]–>HLN [2][3]–>HLN [3][1]–>O [1] I [1]–>HLN [0][1]–>HLN [1][2]–>HLN [2][3]–>HLN [3][1]–>O [1] I [1]–>HLN [0][3]–>HLN [1][2]–>HLN [2][3]–>HLN [3][1]–>O [1] I [1]–>HLN [0][4]–>HLN [1][2]–>HLN [2][3]–>HLN [3][1]–>O [1] I [2]–>HLN [0][1]–>HLN [1][2]–>HLN [2][3]–>HLN [3][1]–>O [1] I [2]–>HLN [0][4]–>HLN [1][2]–>HLN [2][3]–>HLN [3][1]–>O [1] I [3]–>HLN [0][1]–>HLN [1][2]–>HLN [2][3]–>HLN [3][1]–>O [1] I [3]–>HLN [0][4]–>HLN [1][2]–>HLN [2][3]–>HLN [3][1]–>O [1]

Table 2. *Cont.*

Impact Factor	UDC
Factor 3	I [0]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][1]–>O [2] I [0]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][3]–>O [2] I [0]–>HLN [0][4]–>HLN [1][2]–>HLN [2][2]–>HLN [3][3]–>O [2] I [0]–>HLN [0][4]–>HLN [1][2]–>HLN [2][3]–>HLN [3][3]–>O [2] I [1]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][1]–>O [2] I [1]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][3]–>O [2] I [1]–>HLN [0][4]–>HLN [1][2]–>HLN [2][2]–>HLN [3][3]–>O [2] I [1]–>HLN [0][4]–>HLN [1][2]–>HLN [2][3]–>HLN [3][3]–>O [2] I [2]–>HLN [0][1]–>HLN [1][2]–>HLN [2][3]–>HLN [3][3]–>O [2] I [2]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][3]–>O [2] I [2]–>HLN [0][4]–>HLN [1][2]–>HLN [2][2]–>HLN [3][3]–>O [2] I [2]–>HLN [0][4]–>HLN [1][2]–>HLN [2][3]–>HLN [3][3]–>O [2] I [3]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][3]–>O [2] I [3]–>HLN [0][4]–>HLN [1][2]–>HLN [2][3]–>HLN [3][3]–>O [2]
Factor 4	I [0]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][1]–>O [3] I [0]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][2]–>O [3] I [1]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][1]–>O [3] I [1]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][2]–>O [3] I [2]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][1]–>O [3] I [2]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][2]–>O [3] I [3]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][1]–>O [3] I [3]–>HLN [0][4]–>HLN [1][2]–>HLN [2][1]–>HLN [3][2]–>O [3]

Table 3. Obtained index values for each of the declared impact factors.

Factor 1	Factor 2	Factor 3	Factor 4
0.0647401	0.4718856	0.406147	0.0572273

Table 4. The values of the local influence of each of the impact factors onto each neuron(s) of each of available UDCs (within the local datasets for each separate impact factor).

Neuron	F1local	F2local	F3local	F4local
I [0]	0.3291814947	0.2538631347	0.3087674714	0.4309623431
I [1]	0.1903914591	0.2615894040	0.2706480305	0.2510460251
I [2]	0.2241992883	0.2538631347	0.2325285896	0.1757322176
I [3]	0.2562277580	0.2306843267	0.1880559085	0.1422594142
HLN [0][1]	0.3612099644	0.1920529801	0.0101651842	0.0000000000
HLN [0][3]	0.6209964413	0.0165562914	0.0000000000	0.0000000000
HLN [0][4]	0.0177935943	0.7913907285	0.9898348158	1.0000000000
HLN [1][2]	1.0000000000	1.0000000000	1.0000000000	1.0000000000
HLN [2][1]	0.0000000000	0.0000000000	0.2566709022	1.0000000000
HLN [2][2]	0.0000000000	0.0000000000	0.1499364676	0.0000000000
HLN [2][3]	1.0000000000	1.0000000000	0.5933926302	0.0000000000
HLN [3][1]	0.0000000000	1.0000000000	0.0038119441	0.5794979079
HLN [3][2]	0.0000000000	0.0000000000	0.0000000000	0.4205020921
HLN [3][3]	0.0124555160	0.0000000000	0.9961880559	0.0000000000
HLN [3][4]	0.9875444840	0.0000000000	0.0000000000	0.0000000000

Table 5. The values of the global influence of each of the declared impact factors on each neuron(s) of each of available UDCs.

Neuron	F1global	F2global	F3global	F4global
I [0]	0.0213112429	0.1197943576	0.1254049822	0.0246628113
I [1]	0.0123259621	0.1234402728	0.1099228856	0.0143666862
I [2]	0.0145146843	0.1197943576	0.0944407891	0.0100566803
I [3]	0.0165882107	0.1088566119	0.0763783431	0.0081411222
HLN [0][1]	0.0233847692	0.0906270358	0.0041285591	0.0000000000
HLN [0][3]	0.0402033717	0.0078126755	0.0000000000	0.0000000000
HLN [0][4]	0.0011519591	0.3734458887	0.4020184409	0.0572273000
HLN [1][2]	0.0647401000	0.4718856000	0.4061470000	0.0572273000
HLN [2][1]	0.0000000000	0.0000000000	0.1042461169	0.0572273000
HLN [2][2]	0.0000000000	0.0000000000	0.0608962465	0.0000000000
HLN [2][3]	0.0647401000	0.4718856000	0.2410046366	0.0000000000
HLN [3][1]	0.0000000000	0.4718856000	0.0015482097	0.0331631006
HLN [3][2]	0.0000000000	0.0000000000	0.0000000000	0.0240641994
HLN [3][3]	0.0008063714	0.0000000000	0.4045987903	0.0000000000
HLN [3][4]	0.0639337286	0.0000000000	0.0000000000	0.0000000000

The final step at this stage of functioning of the developed method is the conversion of the absolute values of the global influence of each of the declared impact factors (presented in Table 5) into the corresponding relative values (presented in Table 6). A calculation of these relative values is quite simple, and to achieve this, two steps must be performed: the first step is to sum each row of Table 5; and the second step is to calculate the proportion of the value from each cell of the row (from the Table 5) to the sum for this same row found in the first step, which can then be written in the same cell of Table 6.

Table 6. Obtained relative values of the global influence of each of the declared impact factors on each neuron(s) of each of the available UDCs.

Neuron	%F1global	%F2global	%F3global	%F4global
I [0]	0.0731909004	0.4114193126	0.430688328	0.084701459
I [1]	0.0473973731	0.4746683967	0.422689603	0.055244628
I [2]	0.0607801029	0.5016377356	0.395469908	0.042112253
I [3]	0.0790049148	0.5184529857	0.363768257	0.038773842
HLN [0][1]	0.1979405549	0.7671132257	0.034946219	0
HLN [0][3]	0.8372903237	0.1627096763	0	0
HLN [0][4]	0.0013815050	0.4478608384	0.48212692	0.068630737
HLN [1][2]	0.0647401000	0.4718856000	0.406147	0.0572273
HLN [2][1]	0.0000000000	0.0000000000	0.645593057	0.354406943
HLN [2][2]	0.0000000000	0.0000000000	1	0
HLN [2][3]	0.0832530535	0.6068250913	0.309921855	0
HLN [3][1]	0.0000000000	0.9314814015	0.003056098	0.065462501
HLN [3][2]	0.0000000000	0.0000000000	0	1
HLN [3][3]	0.0019890505	0.0000000000	0.998010949	0
HLN [3][4]	1.0000000000	0.0000000000	0	0

Therefore, according to expression (4) presented above, the values from Table 6 can be used in calculating a complex assessment of the probability of the influence of each of the declared impact factors on the final result of perception subjectivization of the researched object.

For further modeling and functioning, as well as the practical approbation, of the developed method, additional corresponding specialized software (which has been developed by the authors, in the scope of this research, using Python 3.12) was used, with the

help of which the following results were obtained for the same experimental example case considered in this section of research.

In particular, Figure 6 below demonstrates the results obtained by the functioning of this specialized developed software.

Also, Table 7 below provides an additional detailed explanation of the resulting data presented in Figure 6.

Table 7. An explanation of the results obtained by functioning of the specialized developed software (presented in Figure 6).

The Element of the Resulting Data	Explanation
# 001	A sequence number of the dataset record(s) for modeling the researched HCI object (i.e., the supported software product, or the processes of its comprehensive support)
Input values 0.3 0.3 0.3 0.2	An input value entering the corresponding input layer of neurons of the encapsulated trained MP ANN
Res 0.1	The reference mono-value of the MP's modeling output result, represented by a single normalized decimal number
Ideal 1000	A set of reference multi-values for the corresponding neurons of the output layer of the researched MP ANN
I[x] -> I[0]->	MP input layer's neuron, activated during the simulation/modeling of the current dataset record
-> OutAN#(%)HLN[0][x] -> ->1(0.1)1(0.4)1(0.4)1(0.1)HLN[0][3]-> ->1(0.9)1(0.6)0(0.4)0(0.1)HLN[1][2]->	OutAN# (%) is the prediction result, obtained after passing the previous layer of the MP (OutAN#—an ordinal number of the output layer's active neuron of the MP; (%)—the probability value indicating that this particular neuron will be activated upon completion of the MP's functioning iteration upon processing the current dataset record)
1	"1" blue color in the position OutAN# indicates that the output neuron with the sequence number of this specific position is responsible for activation at the output of the MP
1	"1" with gray fill in the position OutAN# indicates that the output layer's neuron with the sequence number of this specific position will be guaranteed to be activated (according to the data from the processed CSV file of current dataset)
1	"1" with yellow fill in the position OutAN# indicates that the output layer's neuron with the sequence number of this specific position has been determined by the method as the one that will be activated at the output of the MP's current functioning iteration (however, according to the data from the processed CSV file of current dataset, a completely different neuron will be guaranteed to be activated)
1	"1" with green fill in the position OutAN# indicates that the output layer's neuron with the sequence number of this specific position has been determined by the method as the one that will be activated at the output of the MP's current functioning iteration (and according to the data from the processed CSV file of current dataset, exactly the same neuron will be guaranteed to be activated)
-> O[x] O[2]	MP output layer's neuron activated during processing of current record of the dataset

Table 7. *Cont.*

The Element of the Resulting Data	Explanation
Act 0010	A set of multi-values for the corresponding neurons of the output layer of the MP, obtained as a result of the MP's current functioning iteration
OK? Y N	The verification of whether the actual result is identical to the reference one. –"Y" on a green background indicates that the actual result is identical to the reference one; –"N" on a red background indicates that the actual result is not identical to the reference one.
Fmax 3 1 2 2 1	An example for understanding the predicted result (an ordinal number of an active neuron of the processed layer of the researched MP) of the specific functioning iteration, obtained after passing each layer of the researched MP, from the left to the right: 3—the predicted result, obtained after passing the input layer of neurons of the researched MP. 1—the predicted result, obtained after passing the 1st hidden layer of neurons of the researched MP (a green background means that this result is identical to the reference one). 2—the predicted result, obtained after passing the 2nd hidden layer of neurons of the researched MP (the absence of a green background indicates a noncompliance with the reference result). 2—the predicted result, obtained after passing the 3rd hidden layer of neurons of the researched MP (the absence of a green background indicates a noncompliance with the reference result). 1—the predicted result after passing the 4th (in this particular considered experimental case, it is the last one) hidden layer of neurons of the researched MP (a green background means that this result is identical to the reference one).
Clc% 80%	A completion percentage of processing all the layers of researched MP (for its current functioning/modeling iteration) at the time of result prediction completion with guaranteed determination of an unambiguous (and correct) prediction result. The lower the value of "Clc%", the more effective the prediction is. For example, Clc% = 00% indicates that a guaranteed unambiguous correct prediction result has been already achieved after processing only the input layer of neurons of the researched MP (which means that calculation of absolutely all the subsequent hidden layers of this specific MP (up to its output layer) can be fully omitted), and therefore the efficiency of such particular prediction is the maximal possible value. On the other hand, Clc% = 100% indicates that a guaranteed unambiguous correct prediction result was not achieved even after processing all the layers of neurons of the researched MP, so the effectiveness of performed prediction, in this particular case, is the minimal possible value.

In fact, the main functional purpose of this specialized developed software (as a corresponding programming implementation of the developed method) is, in fact, a step-by-step construction of the direct chains of the researched MP's activated neurons with their simultaneous analysis in full accordance with the developed method and the algorithm, which, as a result, makes it possible to predict the result(s) obtained at the output of the researched MP ANN.

Figure 6. An example of functioning of the specialized developed software, dedicated to modeling the operation of the developed method.

In most cases, based on the testing(s) performed in the scope of this research, the prediction result can be obtained without the need to calculate all the hidden layers of neurons of the researched MP ANN. However, in some cases, there are also possible scenarios, for example, when there is a need to calculate absolutely all the hidden layers of the researched MP ANN (with the Clc% = 100%; see explanation in Table 7); as well as scenarios when the developed approach (method) allows for the avoidance of calculations of some (or, even, all) hidden layers, and even without their calculation, the developed method guarantees an unambiguous and correct prediction of the real value obtained at the output of the researched MP ANN as a result of its functioning. All this is possible because of the developed method and its constituent components: the appropriate mathematical supply (i.e., mathematical model) and the specialized algorithmic supply, which have been presented and described, in detail, above within the scope of this research.

Thus, the modeling, performed on the basis of this specialized developed software (which, in fact, is a programming implementation of the method, developed and presented in scope of this research), makes it possible to predict the results of functioning of the researched MP ANN, which, in the context of the declared scientific and applied problem, interprets the perception subjectivization of the researched HCI objects, particularly in the context of this specific research (the objects, or the processes, of a comprehensive support of software products) and provides all the necessary capabilities for further corresponding increases in the level of automation and intellectualization of these processes.

In particular, within the scope of the presented experimental example of the developed method's modeling and approbation, the following results were obtained, presented below in Figure 7. Thus, from the obtained results of the method's modeling, based on a specific experimental case of the designed encapsulated researched MP ANN, the following conclusions were made (on average, the results are averaged/generalized based on the conducted research):

- In 37% of cases, the final (and correct) result of prediction was obtained at the stage of processing the input layer of neurons of the researched encapsulated MP ANN;
- In 0.2% of cases, the final (and correct) result of prediction was obtained immediately after processing the 1st hidden layer of neurons of the researched encapsulated MP ANN;
- In 10.8% of cases, the final (and correct) result of prediction was obtained immediately after processing the 2nd hidden layer of neurons of the researched encapsulated MP ANN;
- In 13.4% of cases, the final (and correct) result of prediction was obtained immediately after processing the 3rd hidden layer of neurons of the researched encapsulated MP ANN;
- In 29.6% of cases, the final (and correct) result of prediction was obtained immediately after processing the 4th (which in this considered particular experimental example is the last one) hidden layer of neurons of the researched encapsulated MP ANN;
- In 9% of cases, the final (and correct) result of prediction was not (unfortunately) obtained even after complete processing of all the layers of neurons of the researched encapsulated MP ANN.

Thus, this is a quite positive result for the prediction efficiency because an additional interpretation of the obtained results indicates the following (for this specific experimental example of conducted research):

- OutL—the need to calculate the output layer of neurons of the researched MP is absent in 91% of modeling cases;

- 4thHL + OutL—the need to calculate the 4th hidden layer of neurons of the researched MP (and, accordingly, the next (i.e., output) layer of neurons of this same MP) is absent in 61.4% of modeling cases;
- 3rdHL + allNext—the need to calculate the 3rd hidden layer of neurons of the researched MP (as well as all the subsequent layers of neurons of this same MP) is absent in 48% of modeling cases;
- 2ndHL + allNext—the need to calculate the 2nd hidden layer of neurons of the researched MP (as well as all the subsequent layers of neurons of this same MP) is absent in 37.2% of modeling cases;
- 1stHL + allNext—the need to calculate the 1st hidden layer of neurons (as well as absolutely all the subsequent layers of neurons) of the researched MP is absent in 37% of modeling cases.

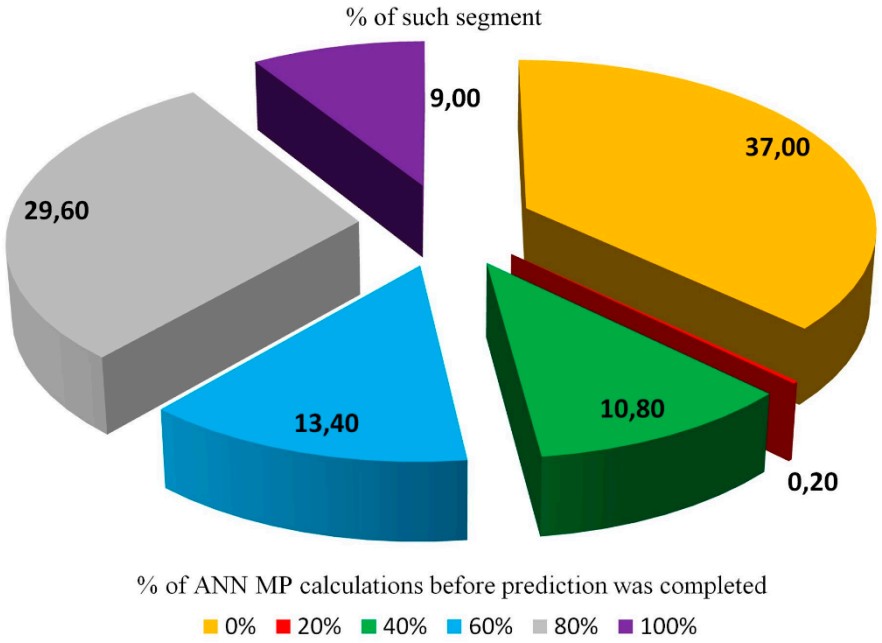

Figure 7. A visualization of obtained indicators of predicting effectiveness of the results of functioning of the researched MP ANN, which represents the perception subjectivization of a relevant investigated HCI object(s).

The corresponding diagram of predicting effectiveness of the results of the considered experimental case is presented below in Figure 8.

The obtained results demonstrate a positive growth trend in the indicator of absence of the need to calculate additional layers of neurons of the researched MP(s), starting from the input layer and moving towards the output layer, i.e., the closer to the output layer of the MP, the less important it is to calculate each subsequent layer(s). This trend is fairly expected because with the "growth" of each researched formatted (i.e., in the process of formation) direct chain of neurons, the number of impact factors, to which it can simultaneously belong, decreases.

Another equally important and significant achievement of the developed method is the assignment to each of the neurons of the hidden layers of the researched encapsulated MP ANN(s), i.e., the corresponding indicators (i.e., markers) of the affiliation of these neurons to the certain impact factor(s), which, in turn, opens additional opportunities for further research in this field. In particular, this achievement provides possibilities for the formation (restoration, or reproduction) of the distribution boundaries of the impact factors, as the neurons of the encapsulated MP now can be united based on their belonging to the most common specific impact factor(s). In addition, this achievement creates a basis for the

acquisition by the neurons of the hidden layers (of the researched MP ANN)—a certain essential content and purpose, which they have always been automatically deprived of before—since the very concept of the MP itself does not imply the presence of any functional and/or semantic load/meaning(s) for the neurons of the hidden layers, so they served exclusively a quite pure arithmetic (i.e., "calculational") function providing and ensuring the possibility of training and correct functioning/operating of the MP itself.

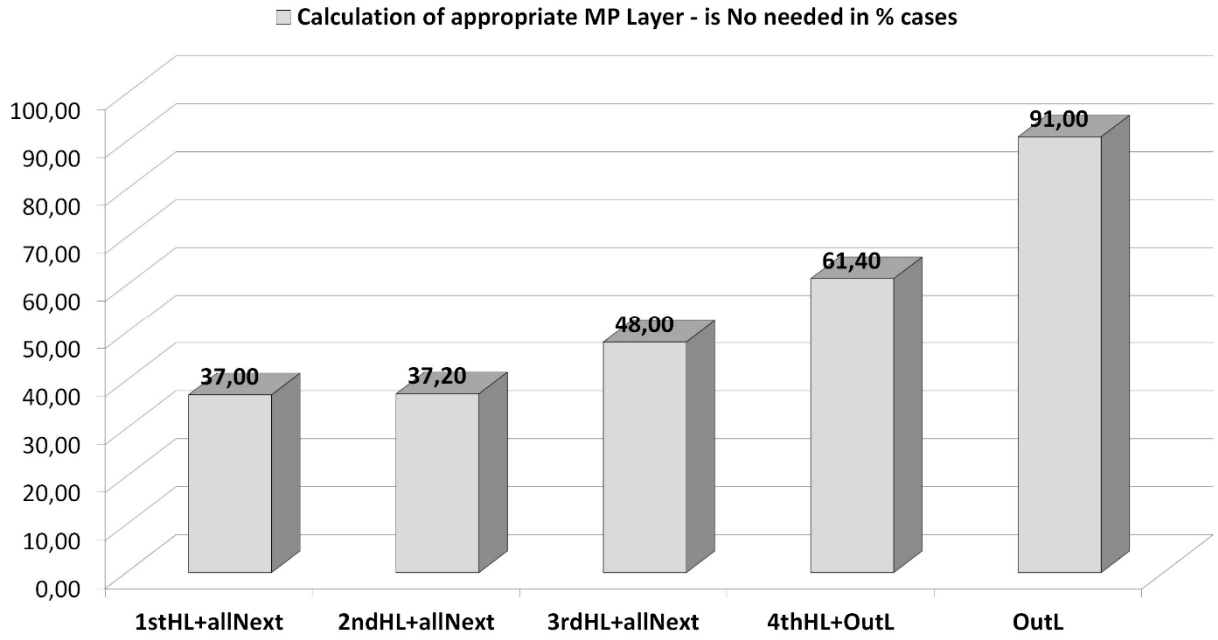

Figure 8. A diagram demonstrating the effectiveness of the results prediction based on the considered experimental case of performed research of the proposed method's modeling.

Additionally, this aforementioned feature to unite the MP's hidden layer neurons according to their belonging to a certain (common for all of them) impact factor(s) has been successfully used during a practical approbation of the developed method while solving the practically applied problem of identifying a member of the support team (of a defined software product) whose multifactor portrait is as close as possible to the corresponding multifactor portrait of a given client's user of the same supported software product.

First of all, let us define the concept of a multifactor portrait as an individual set of averaged indicators of the influence of each of the declared impact factors on the subjective perception of the researched HCI object (i.e., the supported software product, or the processes of its comprehensive support) by the corresponding subject of interaction with this object.

Thus, in the case of a declared practically applied problem, one subject of interaction with the object (i.e., the defined supported software product) is the client user, while the other subjects of this interaction are the members of the support team (of the same defined supported software product). First, by modeling the functioning of obtained trained MP ANN (encapsulated into the relevant previously designed conceptual model of perception subjectivization of the researched HCI object, i.e., defied supported software product, or processes of its comprehensive support) performed on the basis of records of the corresponding dataset (which represents the individualistic perception of this specific HCI object by each separate specific researched subject of interaction), as a result of processing each separate record of this dataset, a corresponding direct chain of neurons can be obtained. After that, by summing the values (for each separate specific impact factor) of the local influence of each of the impact factors on each of the neurons of the resulting direct chain,

we can obtain a certain multifactor set of values (i.e., one specific numerical value for each separate specific impact factor), which represents the shares of the influence of each of the declared impact factors on the result of perception subjectivization of the researched object by the separate researched subject of this interaction in the scope of one specific considered (situational) case, represented by this specific considered record of the processed dataset. Thus, after processing all the records of the dataset, we can obtain a set of multifactor sets of values, on the basis of which the arithmetic mean value is calculated for each of the impact factors, and then obtained arithmetic mean values are converted from an absolute form to a relative form, which, in fact, will represent the corresponding multifactor portrait of this specific researched subject (of interaction) in the context of its individualistic perception of the specific researched HCI object (i.e., the defined supported software product, or the processes of its comprehensive support).

Table 8 below presents the given values of the multifactor portraits of the employees of a defined software product's support team, as well as a given client user of the same defined software product.

Table 8. Multifactor portraits of a defined software product's support team members, as well as the given client user.

Subject of Interaction	Factor #1	Factor #2	Factor #3	Factor #4
Employee 1	0.2703	0.3812	0.1963	0.1522
Employee 2	0.3504	0.2289	0.1806	0.2401
Employee 3	0.2295	0.3781	0.2152	0.1772
Employee 4	0.2618	0.2593	0.2491	0.2298
Employee 5	0.1907	0.3412	0.1758	0.2923
Employee 6	0.3317	0.2911	0.1665	0.2107
Employee 7	0.2902	0.1984	0.3056	0.2058
Employee 8	0.1893	0.3271	0.3297	0.1539
Employee 9	0.2095	0.2105	0.2862	0.2938
Employee 10	0.3104	0.2358	0.2394	0.2144
Employee 11	0.3026	0.2817	0.2123	0.2034
Employee 12	0.2281	0.3096	0.2274	0.2349
Employee 13	0.2761	0.3125	0.2509	0.1605
Employee 14	0.2862	0.3359	0.1908	0.1871
Employee 15	0.3219	0.2724	0.2302	0.1755
Employee 16	0.3426	0.2996	0.1831	0.1747
Employee 17	0.2678	0.3243	0.2917	0.1162
Employee 18	0.2336	0.3527	0.2035	0.2102
Employee 19	0.3154	0.2638	0.2316	0.1892
User 1	0.3085	0.2182	0.2954	0.1779

Figure 9 below shows a graphical interpretation of the multifactor portraits of the researched support team members and a given client user, with an additional indication of the identified support team employee whose multifactor portrait is as close as possible to the multifactor portrait of the given client user.

The identification of the required support team employee has been carried out based on the data presented in Table 9, which consists of the comparative characteristics of the multifactor portraits of the support team members with the multifactor portrait of the given client user.

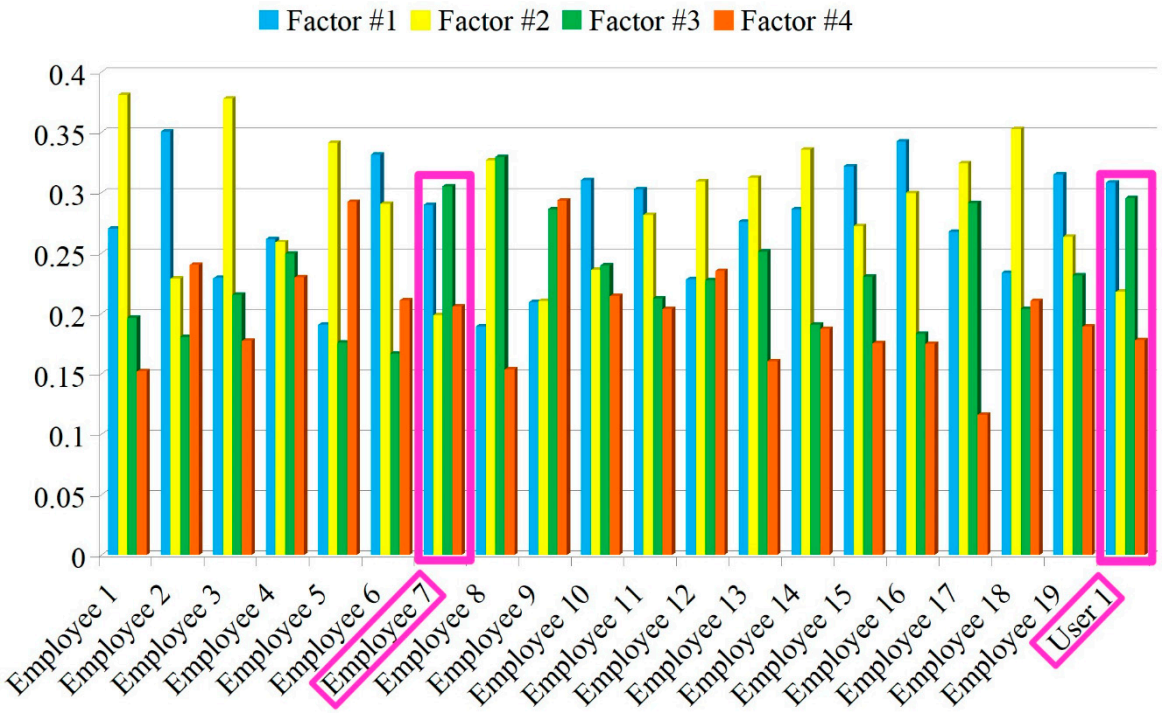

Figure 9. A graphical interpretation of the multifactor portraits of the support team members and the given client user of the researched HCI object, i.e., the defined supported software product.

Table 9. Comparative characteristics of the multifactor portraits of the support team members and the given client user.

Subject of Interaction	dF1	dF2	dF3	dF4	Sum(dF [1–4])
Employee 1	0.0382	0.163	0.0991	0.0257	0.326
Employee 2	0.0419	0.0107	0.1148	0.0622	0.2296
Employee 3	0.079	0.1599	0.0802	0.0007	0.3198
Employee 4	0.0467	0.0411	0.0463	0.0519	0.186
Employee 5	0.1178	0.123	0.1196	0.1144	0.4748
Employee 6	0.0232	0.0729	0.1289	0.0328	0.2578
Employee 7	0.0183	0.0198	0.0102	0.0279	0.0762
Employee 8	0.1192	0.1089	0.0343	0.024	0.2864
Employee 9	0.099	0.0077	0.0092	0.1159	0.2318
Employee 10	0.0019	0.0176	0.056	0.0365	0.112
Employee 11	0.0059	0.0635	0.0831	0.0255	0.178
Employee 12	0.0804	0.0914	0.068	0.057	0.2968
Employee 13	0.0324	0.0943	0.0445	0.0174	0.1886
Employee 14	0.0223	0.1177	0.1046	0.0092	0.2538
Employee 15	0.0134	0.0542	0.0652	0.0024	0.1352
Employee 16	0.0341	0.0814	0.1123	0.0032	0.231
Employee 17	0.0407	0.1061	0.0037	0.0617	0.2122
Employee 18	0.0749	0.1345	0.0919	0.0323	0.3336
Employee 19	0.0069	0.0456	0.0638	0.0113	0.1276

The data, presented in Table 9 above, reflect the absolute difference in the values of the indicators between the multifactor portraits of each of the considered software product's support team employees and the multifactor portrait of the given client user, as outlined below:

- Separate values for each specific impact factor:
  - dF1—a difference in Factor #1;

- dF2—a difference in Factor #2;
- dF3—a difference in Factor #3;
- dF4—a difference in Factor #4.

- Total values, i.e., all the defined impact factors together:
  - Sum(dF [1–4]).

In addition, Figure 10 below represents a graphic visualization of the obtained comparative characteristics of the multifactor portraits of the considered support team members and the given client user.

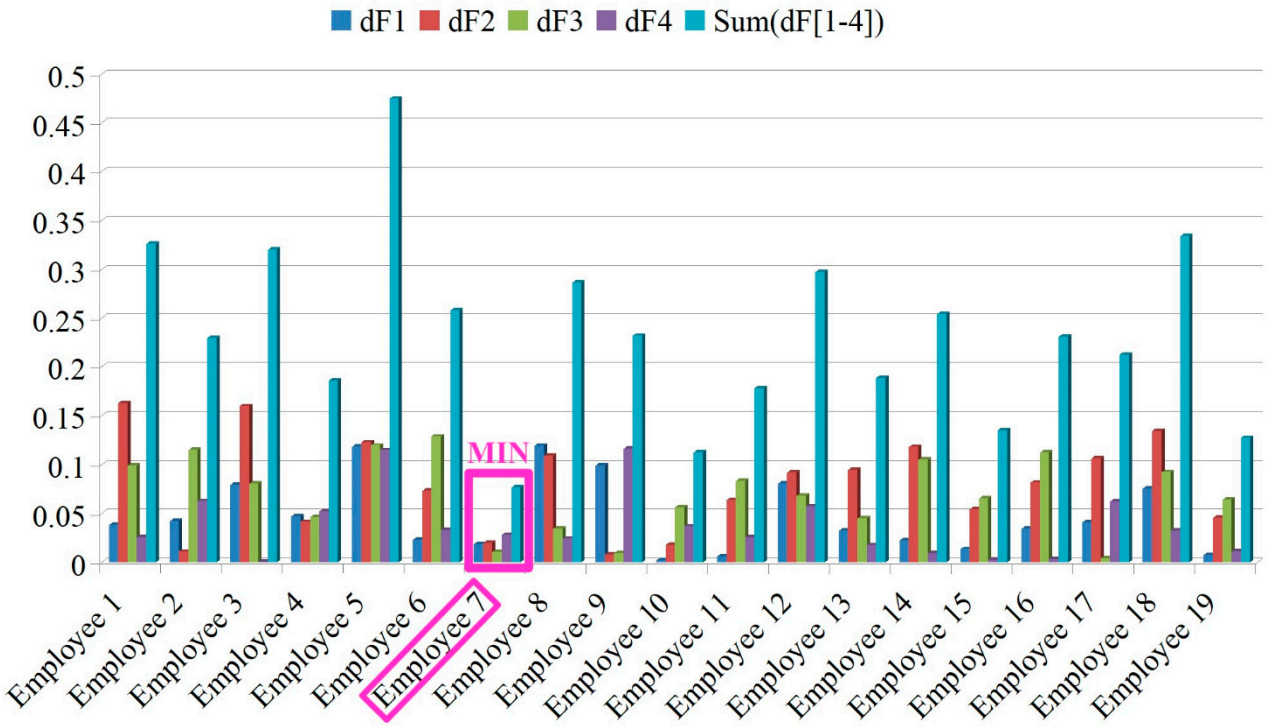

Figure 10. A graphic visualization of the comparative characteristics of the multifactor portraits of the considered support team members and the given client user.

The process of identification of the required employee consists of finding a support team member with the minimum value of comparative characteristics (representing the differences between its multifactor portrait and the multifactor portrait of the given client user). In this particular case, an example of an identified support team member is "Employee 7" (see Figure 10, as well as Figure 9).

Therefore, the given practically applied problem of identifying a member/employee of the support team (of defined software product) whose multifactor portrait is as close as possible to the corresponding multifactor portrait of the given client's user (of the same supported software product) has been solved by means of the developed method for HCI objects' perception subjectivization result prediction based on impact factor analysis with usage of MP ANN. In turn, the solving of this problem provides possibilities for increasing the efficiency level of a customer support service/department (of the considered declared software product), since the subjective vision/perception of the supported software product between the given client user and the corresponding identified support team employee is as close as possible, which ensures an improvement in the level of their mutual understanding in the context of the researched HCI object (i.e., the supported software product) of their joint interaction.

6. Discussion

In the context of a comparison of the developed method presented in this research (i.e., HCI objects' perception subjectivization result prediction method based on impact factor analysis with usage of MP ANN) with the outcomes of existing similar approaches, the following conclusions have been obtained. In particular, the authors of Ref. [59] investigated methods dedicated to the prediction of software reliability, and presented their own developed method of selecting the best model for predicting software reliability. Their proposed method is a composite of Hesitant Fuzzy (HF), Analytic Hierarchy Process (AHP), and the Technique for Order of Preference by Similarity to Ideal Solution (known as TOPSIS). In the mentioned study, it was found that the most preferred model for predicting software reliability, which can be found in the presented (within the framework of this separate study, conducted by the authors) structure and the hierarchy, is the neuro-fuzzy computing model. Despite this, the proposed method does not provide the possibility of taking into account the factors influencing the subjectivization of software's perception by the relevant interaction subjects (e.g., members of the relevant teams of the software developer company) in the context of software reliability research and prediction.

At the same time, in line with the scope of Ref. [60], the authors investigated various existing quality models with defined parameters for predicting the quality of a Component-Based Software (CBS). In particular, according to the presented study, several models (for example, Boehm's, McCall's, FURPS, ISO 9126, Dromey's) have been developed to assess the quality using the hierarchically related characteristics of quality indicators. This study focuses on the analysis and evaluation of the CBS quality and prediction model using software quality assurance models as well as the quality characteristics. The main concept of this study is to measure the software quality characteristics using the quality prediction parameters, providing the ability to determine the quality factors for the CBS, including using various computational intelligence methods to predict optimal CBS quality. However, as in the previous case of the existing approach, the proposed models do not take into account the factors of subjectivization of a software perception in the context of analysis, evaluation, and prediction of the quality of the latter.

In addition, in the scope of Ref. [61], the authors developed a specialized tool that provides the ability to measure the perception of social and human factors (SHFs) (which, in particular, include problem-solving skills, a cognitive aspect and social interaction, as well as a number of other additional researched SHFs, with 13 in total) by software development team(s) members, as those SHFs can affect the performance of these members of the researched teams. The developed tool confirms the relevance and the significance of the impact of the aforementioned SHFs on the members of the researched software development teams, and also provides the possibility to both identify and rank the SHF(s), as well as their impact. In addition, the developed tool could be useful for measuring the perception of those declared SHFs (which may affect the performance of the researched teams), and implement relevant improvements based on the obtained results. At the same time, the developed approach does not provide the possibility of predicting the results of the impact of declared SHF on the productivity of the researched subjects, i.e., the members of appropriate software development teams.

Additionally, in the context of a comparison of the developed approach presented in this research to a standard feed-forward inference [62] method, it has been established that the latter is more intended for the training of neural networks, while the prediction of modeling results is not its prerogative. Another more interesting approach is the early-exit method, which has been considered in the scope of recent studies [63–65]. It is worth noting that the "early-exit" is an extremely broad conceptual platform that can include a wide variety of solutions. In particular, the basic implementation of the early-exit concept

uses side branches. The early-exit neural networks insert multiple side branches into the conventional architecture of a Deep Neural Network (DNN). The inference process in such DNNs can be stopped at an early stage on a particular side branch when it meets a predefined confidence criterion that is directly related to the input data classifying difficulty, due to which simpler inputs can be classified on earlier side branches using the features extracted by the initial neural layers, while more complex inputs require processing of all the following layers up to the output layer. In particular, some early-exit DNN models offer a solution by treating the entire DNN structure as a single optimization problem and jointly optimizing the weighted losses of all the side branches. In addition, another effective advantage of the early-exit concept is that it provides a solution to the quite common problem of the overtraining of neural networks, as each early-exit side branch acts as a "regularizer" for other branches, helping to prevent this same overtraining. In addition, early-exit DNNs provide an additional gradient update signal through the exit points during backpropagation, helping to prevent another fairly common problem with gradient shrinking. Also, early-exit DNNs add a discriminative effect to the lower layers, increasing the network's ability to learn efficiently, and thus early-exit DNNs mitigate the vanishing gradient problem, providing more stable and efficient training. Early-exit DNN architecture typically includes a backbone DNN model, which is a regular DNN, with additional side branches attached to it. These side branches can be created using conv-layers, fc-layers, or a combination of both of them. During the inference, the input data are processed layer-by-layer until a side branch is reached, where a prediction is made and the confidence is estimated. If the confidence is sufficient, the inference process stops and returns a prediction; otherwise, it continues until it reaches the next side branch. If no side branch can provide a sufficiently confident prediction, the input is processed to the final output layer, which always returns a prediction. But along with the undeniable advantages of the early-exit concept, it also has some disadvantages. In particular, there are some design limitations of early-exit DNNs, as discussed below. Designing an effective early-exit DNN involves several key decisions, namely choosing the DNN backbone architecture, structuring the side branches, determining their placement within the backbone, and formulating the optimal early-exit policy. As a recent and actively developing research area, early-exit DNNs, unfortunately, still do not have standardized strategies for all these aforementioned aspects. The choice of the early-exit policy significantly affects the number of early predictions on the side branches, which, in turn, affects the overall efficiency of this whole approach in general. Different studies have used different methods to determine the confidence in the prediction, and this section digs deep into the previous investigations by examining how the side branches were constructed and the strategies adopted to place them on the trunk model, as well as the policies formulated to define the trust criteria at each separate exit point. Therefore, to achieve an optimal performance, the side branches must be carefully designed with appropriate neural layers and strategically placed along the backbone of the original parent DNN. At the same time, the design of an effective early-exit policy is crucial, as it significantly affects the general performance. However, as this area of research is still actively developing, there is currently a significant lack of optimized configurations for constructing and placing side branches along the trunk model, as well as developing relevant effective early-exit strategies and policies. Therefore, the existing studies actually highlight the significant design issues and research challenges with the aim of stimulating further research in this area. In contrast to the aforementioned early-exit approach, the developed (in the scope of the current research) information-cognitive concept of a predicting method for HCI objects' perception subjectivization results based on impact factor analysis with usage of multilayer perceptron ANN is, in fact, a full-fledged (developed completely separately and independently from scratch)

alternative to the early-exit approach, and unlike the latter, offers the completely full and finalized functional cycle of development and implementation, without requiring any extra redesign or retraining of any considered existing trunk neural networks for encapsulating any additional side branches (as required by the early-exit approach), working universally with any existing DNN architecture, analyzing only their UDCs (regardless of the DNN's architectural and/or constructive features). Furthermore, an early-exit is a concept for DNNs only, while the proposed information-cognitive concept of a predicting method for HCI objects' perception subjectivization results is not only the same universal solution for DNNs, but also, at the same time, is a highly specialized solution for researching relevant problems of HCI objects' perception subjectivization. In addition, the developed approach for predicting DNN modeling (e.g., functioning, operating) results (MP is a classic representative of this family of neural networks) based on the UDC comparative analysis can also be used in the early-exit concept as an additional mechanism for generating a confidence criterion, which additionally confirms the relevance of the developed approach proposed by the information-cognitive concept of a predicting method for HCI objects' perception subjectivization results, presented within the framework of the current research.

Another research that complements and actually completes the list of similar existing approaches is Ref. [66], a study dedicated to the prediction of ANN output results based on a high dimensional gradient-based rule extraction. In particular, this study uses a three-layer ANN and physiological signals as inputs to predict the results of a subjective belief experiment, which consists of a comparison with a human perception, exploring whether the speaker's speech content can be better assessed as manipulative, by using the ANN's physiological signal data processing. First, the concepts of the decision rules, the characteristics' input signal(s), and the gradient sensitivity analysis are introduced, and both the setup and the training of the considered researched ANNs are briefly described. Then, the process of the rule extraction and the prediction of new input patterns are explained. In addition, a dimensionality reduction for simpler rules and some further improvement methods, including gradient regression, the removal of oblique rules and an in-depth analysis of their beneficial effects, are also introduced in the scope of the considered existing approach. Finally, some limitations of the gradient rule extraction, as well as the potential solutions, are discussed. Thus, the considered study mainly focuses on explaining ANN inference using different rule extraction methods, which include using a weighted decision rule based on sorting according to the gradient magnitude and adding an oblique-type rule, using LOWESS regression in the gradient approximation, removing the oblique-type rule, and maintaining the sorted order. The main conclusion is that by using slightly complex rules, such as the oblique-type rule, the prediction accuracy can be improved while introducing unwanted complexity. So, the trade-off between the simplicity and accuracy is to choose fewer dimensions but with higher magnitude in order to avoid noisy dimensions. Using this mechanism to create simple and relatively accurate rules can help researchers to find the correlations between the factors and the response. In a broader sense, rule extraction also combines symbolic and connectionist approaches with artificial intelligence. The extracted rules belong to the symbolic approaches, while the ANN itself belongs to the connectionist approach. So, the existing research actually presents good examples of constructing symbolic rules and using various methods to better approximate the behavior of the researched ANN. However, the described existing high dimensional gradient-based rule extraction approach also has a number of disadvantages. In particular, the gradient-based rule extraction has no theoretical basis and is not reliable enough, so only a shadow ANN and a careful selection of activation function are suitable for this method. In addition, the main problem is that the magnitude of the gradient cannot proportionally reveal the probability of the ANN's output change, and the gradient itself

suffers from different network structures, when the gradient magnitude unexpectedly increases or decreases beyond a threshold value for two consecutive epochs. Therefore, gradient compensation could be a potential solution. However, in this case, the collected gradients must obtain a convex combination of the previous epoch and the current epoch, in order to control the gradients, so that they do not behave in an undesirable manner, which significantly complicates the proposed solution. At the same time, in contrast to this particular approach, the developed information-cognitive concept of a predicting method for HCI objects' perception subjectivization results does not require encapsulation at the very early stages of the design and the training of used trunk ANNs. It is quite universal and adaptive to any architectures or configurations of the researched ANNs, and provides a significantly more simple and faster method of further automation of the proposed approach, in addition to software implementation.

Thus, the main advantage of the developed information-cognitive concept of a predicting method for HCI objects' perception subjectivization results based on impact factor analysis with the use of MP ANNs, presented within the scope of this research, is that, unlike the existing similar approaches, it provides the ability to predict the resulting impact of each of the declared impact factors on the perception subjectivization of the researched HCI object(s) by each individual subject of the interaction with this object. In turn, such an advantage and feature of the developed method and the concept provides and opens up an extremely wide range of opportunities for studying the derivative relevant processes, where the factor(s) of perception subjectivization (of the researched objects) plays a key role. First, the focus is on ensuring the possibility of improving the interaction between the HCI subjects and the HCI object(s) by taking into account the subjective vision and perception (by these subjects) of the common object of their joint interaction. In addition, the proposed approach does not require its encapsulation at the very initial pre-training stage(s) of ANN design, which allows its use in already existing fully designed ANNs without the need for their complete re-evaluation and/or re-design. Additionally, it ensures a significant reduction in the total resources needed for its software (e.g., programming) implementation due to its simplicity, being mainly based on string operations without highly complicated high-math instruments.

An additional feature of the developed information-cognitive concept of a predicting method for HCI objects' perception subjectivization results, as well as the relevant developed algorithm, consists of its ability to predict MP's operating results not by using arithmetic calculations (which are performed by MP itself in order to obtain these actual operating results at its output layer of neurons) but instead by using string operations, each time comparing the constructed current chain (i.e., "*currChain*") with all the previously identified UDCs, trying to find full or partial entry of the former into the latter. So, within the framework of the standard architecture of the most existing processors, such string operations (used for MP's operating result prediction) require significantly more time and resources in comparison with the arithmetic calculations (used by MP's direct result calculation). In turn, this leads to the algorithm's time and space complexity increasing, and the larger the designed and investigated models are, the more complex the string operations, and the greater the time required and resource costs for their processing in comparison to the arithmetic approach. Therefore, this actually represents another direction in the prospect of further studies and investigations that could be performed as a potential improvement(s) of the proposed approach developed and presented in the scope of the current research.

In summary, the proposed information-cognitive concept of the relevant developed predicting method ensures the possibility of resolving a complex problem of the prediction of HCI objects' perception subjectivization results, which consists of two main parts: the

problem of evaluation of the HCI objects' perception subjectivization as an irrational component of the HCI; and the problem of results prediction of the HCI objects' perception subjectivization (represented by the relevant pre-trained MP ANN, encapsulated into the corresponding conceptual models of perception subjectivization of the researched HCI objects). Resolving this complex problem, in turn, increases the level of automation and intellectualization of relevant researched processes, as well as HCI in general. In comparison with the outcomes of existing similar approaches, the proposed information-cognitive concept fills their remaining gaps. As an example, in the scope of evaluating various HCI factors, the existing approaches do not provide possibilities for predicting the level(s) of impact of various factors (including precisely those causing the perception subjectivization of HCI objects, which themselves are already extremely understudied, since the majority of existing methods for studying the factors that influence HCI mainly investigate rational factors of influence, while perception subjectivization is an irrational factor). On the other hand, in line with the scope of the prediction of DNN results, the existing approaches do not provide possibilities for evaluating or taking into consideration various relevant impact factors, including those that cause HCI objects' perception subjectivization. In turn, the main practical significance of the proposed approach and obtained research results is its role as a relevant instrument for predicting and assessing such an irrational component of HCI as a perception subjectivization of the object(s) of this interaction, which is extremely important in practice, especially in the context of comprehensive support of software products, where the perception subjectivization of the supported software product itself (or the processes of its comprehensive support) often acts as a kind of "stumbling block" precisely because of the differences in the vision and/or the perception of a single common object of interaction by different subjects (representatives of customer support teams, business analysts, developers, clients and customers, ordinary end users, etc.) of this same interaction. Generally speaking, the problem of differences and/or incompatibilities in the perception subjectivization of a common object of interaction between different subjects of this same interaction is extremely important and relevant, especially in the context of an understudied irrational component such as HCI and/or any other type(s) or manifestation(s) of an intersubjective interaction(s) in general.

7. Conclusions

An information-cognitive concept of a predicting method for HCI objects' perception subjectivization results based on impact factor analysis with the use of MP ANNs has been developed in the scope of this research. The developed method provides the possibility of predicting the results of functioning of a pre-trained MP ANN(s), encapsulated into the corresponding conceptual models of perception subjectivization of the researched HCI objects (including, in particular, the objects of comprehensive support of software products). The process of predicting the results of functioning of the researched encapsulated (pre-trained) MP ANN is carried out based on the analysis of the direct chain of neurons, which is built/constructed from the MP's neurons, activated layer-by-layer at each step of the MP's operation (functioning) based on the calculation of the current layer of the researched MP: starting from the input layer of neurons, continuing with all the hidden layers (from the first one to the last one), and ending with the output layer of neurons of the researched MP ANN (which represents the researched perception subjectivization of the investigated HCI object by the subjects of this same interaction).

The proposed method is based on the developed conceptual models of HCI objects' perception subjectivization and the developed mathematical model, as well as a specialized developed algorithm, dedicated to the prediction of HCI objects' perception subjectivization results based on the impact factor analysis with the use of an MP ANN. In particular, the

conceptual models of HCI objects' perception subjectivization provide the possibility of representation formalizing of these relevant researched processes (of HCI objects' perception subjectivization). At the same time, the developed mathematical model acts as a mathematical supply of the developed method, and provides the possibility of mathematical modeling of the researched processes of HCI objects' perception subjectivization. In addition, the developed specialized algorithm, which describes the main steps and procedures of the researched processes of the prediction of HCI objects' perception subjectivization results, provides the possibility of further software (i.e., programming) implementation/supply of the developed method, in order to ensure the possibility of further computer modeling and investigation of the researched processes.

The modeling of the developed method, as well as its practical approbation, has been conducted on numerous relevant experimental cases, one of which has been presented in scope of this research, in maximum detail, as the most illustrative example. In order to ensure the possibility of carrying out all the necessary modeling of the proposed method, an appropriate specialized software has been developed by means of the Python 3.12 programming language and Thonny 4.1.4 IDE, which makes it possible to investigate the processes of construction of the corresponding current direct chains of activated neurons of the researched MP ANN (which, in turn, represents the perception subjectivization of the researched HCI object based on previously declared impact factors) with simultaneous prediction of the results of full operation of this same MP ANN. A detailed analysis of the obtained modeling results has been carried out, as well as their further representation and visualization in the most informative and understandable form. In addition, thanks to the developed method, the concept of a multifactor portrait has been defined, which (as a practical approbation of the developed method) allowed us to solve the given example of the practically applied problem of identifying a member of the support team (of defined software product) whose multifactor portrait is as close as possible to the corresponding multifactor portrait of a given client's user of the same supported software product.

In summary, we state the positive effect of the developed information-cognitive concept of a predicting method for HCI objects' perception subjectivization results, which is confirmed by the obtained indicators of the effectiveness of the results prediction of the considered experimental research cases (on a basis of which the developed method has been tested), in the context of the scientific and applied problem of increasing the level of intellectualization and automation while studying the relevant processes of HCI objects' perception subjectivization (in particular, in the context of comprehensive support of software products).

However, perhaps a more important feature is that the developed method actually allows the prediction of the results of functioning of any MP ANN (by additionally introducing the concept of impact factors), which opens up completely new prospects for further research in this direction.

As a prospect for further research, authors see the potential of applying the developed method (as well as its components) in the context of solving the scientific and applied problems of perception subjectivization of various objects, including not only those considered in the scope of the software products' comprehensive support, HCI and/or HMI, but also objects of any other intersubjective interaction(s) in general, which significantly expands the horizon of possibilities for further research.

Author Contributions: Conceptualization, A.P. and V.T.; methodology, A.P.; software, A.P.; validation, A.P. and V.T.; formal analysis, A.P., V.T., N.L., L.S. and B.F.; investigation, A.P.; resources, A.P., N.L. and B.F.; data curation, A.P.; writing—original draft preparation, A.P.; writing—review and editing, A.P., V.T., N.L., L.S. and B.F.; visualization, A.P.; supervision, V.T., N.L. and L.S.; project administration, A.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article.

Acknowledgments: The authors would like to express appreciation for the editors and the anonymous reviewers for their insightful suggestions that helped to improve the quality of this paper.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AHP	Analytic hierarchy process
AI	Artificial intelligence
ANN	Artificial neural networks
CBS	Component-based software
DMN	Decision making and notation
DNN	Deep Neural Network
GPT	Generative pre-training transformer
HCI	Human–computer interaction
HF	Hesitant fuzzy
HLN	Hidden layer neuron
HMI	Human–machine interaction
IDE	Integrated development environment
LLM	Large language model
ML	Machine learning
MP	Multilayer perceptron
SHFs	Social and human factors
UDC	Unique direct chain

References

1. Pollini, A.; Giacobone, G.A.; Zannoni, M.; Pucci, D.; Vignali, V.; Falegnami, A.; Tomassi, A.; Romano, E. Human–Machine Interaction Design in Adaptive Automation. *Procedia Comput. Sci.* **2025**, *253*, 1034–1044. [CrossRef]
2. Mentler, T.; Flegel, N.; Pöhler, J.; Laerhoven, K.V. Does Automation Scale? Notes From an HCI Perspective. In Proceedings of the AutomationXP23: Intervening, Teaming, Delegating—Creating Engaging Automation Experiences, Hamburg, Germany, 23 April 2023; pp. 1–5. Available online: https://ceur-ws.org/Vol-3394/short3.pdf (accessed on 1 May 2025).
3. Dhakecha, H. A Methodological Study of Human–Computer Interaction: A Review. *Int. J. Res. Appl. Sci. Eng. Technol. (IJRASET)* **2022**, *10*, 195–200. [CrossRef]
4. Shlyakhetko, O.; Fedushko, S.; Kryvinska, N. Intelligent Human–Computer Interaction in Innovative AI Solutions in Travel and Its Impact on the E–Society. In Proceedings of the 3rd Workshop on Artificial Intelligence for Cultural Heritage, Co–Located with the 23nd International Conference of the Italian Association for Artificial Intelligence, Bolzano, Italy, 25–28 November 2024; Volume 3865, pp. 1–6. Available online: https://ceur-ws.org/Vol-3865/07_paper.pdf (accessed on 5 May 2025).
5. Parsa, S. Software Testing Automation. In *Springer eBooks*; Springer: Cham, Switzerland, 2023; 580p. [CrossRef]
6. Nama, P. Integrating AI in testing automation: Enhancing test coverage and predictive analysis for improved software quality. *World J. Adv. Eng. Technol. Sci.* **2024**, *13*, 769–782. [CrossRef]
7. Gracic, S.; Vukovic, V. Traditional and Non–traditional Computing Mechanisms in Software Testing Automation Through Users', Developers' and Managers' Lens: A Systematic Literature Review. *J. Mechatron. Autom. Identif. Technol.* **2021**, *6*, 17–26.
8. Karamustafa, E.Y. Agile human resources practices in decentralized autonomous organizations: A conceptual study. *Int. J. Humanit. Soc. Dev. Res.* **2023**, *7*, 1–7. [CrossRef]
9. Hašek, F.; Mohelská, H. Selection of a suitable agile methodology—Case study. *Hradec Econ. Days* **2021**, *11*, 225–231. [CrossRef]
10. Orantes–Jiménez, S. –D.; Pérez–Castillo, Y.–J.; Aguilar–Jaúregui, M.–E. Study to Delimit the Factors That Contribute to the Adoption of an Agile Methodology. *J. Syst. Cybern. Inform.* **2022**, *20*, 21–29. [CrossRef]

11. Khan, H.H.; Afzal, M.; Zubair, S.; Atif, M.; Hamid, K. DevOps methodology impact on software projects to lead successes and failure through Kubernetes. *J. Jilin Univ. (Eng. Technol. Ed.)* **2023**, *41*, 610–620. [CrossRef]
12. Masud, S.M.R.A.; Masnun, M.; Sultana, A.; Ahmed, F.; Begum, N. DevOps Enabled Agile: Combining Agile and DevOps Methodologies for Software Development. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 278–283. [CrossRef]
13. Salman Asif, S.; Anala, M.R. DevOps with Modern Software Development. *Int. J. Soft Comput. Eng.* **2020**, *9*, 18–21. [CrossRef]
14. Ogala, J.O. A Complete Guide to DevOps Best Practices. *Int. J. Comput. Sci. Inf. Secur. (IJCSIS)* **2022**, *20*, 1–6. [CrossRef]
15. Jayakody, J.A.V.M.K.; Wijayanayake, W.M.J.I. DevOps Adoption in Information Systems Projects; A Systematic Literature Review. *Int. J. Softw. Eng. Appl.* **2022**, *13*, 39–53. [CrossRef]
16. Gawade, S.; Nirp, N.K. DevOps: Beginning and more to know. *Int. Res. J. Mod. Eng. Technol. Sci.* **2022**, *4*, 1555–1562. [CrossRef]
17. Sydorenko, O.; Lysa, N.; Sikora, L.; Martsyshyn, R.; Miyushkovych, Y. Optimizing Aircraft Routes in Dynamic Conditions Utilizing Multi–Criteria Parameters. *Appl. Sci.* **2025**, *15*, 6044. [CrossRef]
18. Pukach, A.; Teslyuk, V.; Lysa, N.; Sikora, L. Development of Impact Factors Reverse Analysis Method for Software Complexes' Support Automation. *Designs* **2025**, *9*, 58. [CrossRef]
19. Fayad, A. Development and Future Scope of AI in the Workplace. *Am. Int. J. Bus. Manag. (AIJBM)* **2024**, *7*, 11–20.
20. Estrada–Torres, B.; del–Río–Ortega, A.; Resinas, M. DemaBot: A Tool to Automatically Generate Decision–Support Chatbots. In Proceedings of the Demonstration & Resources Track, Best BPM Dissertation Award, and Doctoral Consortium at BPM 2021 Co–Located with the 19th International Conference on Business Process Management, Rome, Italy, 6–10 September 2021; Volume 2973, pp. 1–5. Available online: https://ceur-ws.org/Vol-2973/paper_278.pdf (accessed on 18 May 2025).
21. Vijai, C.; Mariyappan, M.S.R. Robotic Process Automation (RPA) in Human Resource Functions. *Adv. Manag.* **2023**, *16*, 30–37. [CrossRef]
22. Elgendy, N.; Elragal, A. Big Data Analytics in Support of the Decision Making Process. *Procedia Comput. Sci.* **2016**, *100*, 1071–1084. [CrossRef]
23. Rayhan, A. Artificial Intelligence in Robotics: From Automation to Autonomous Systems. *IEEE Trans. Robot.* **2023**, *39*, 2241–2253. [CrossRef]
24. Hoßfeld, S. Optimization on Decision Making Driven by Digitalization. *Econ. World* **2017**, *5*, 120–128. [CrossRef]
25. Alagu Karthikeyan, A.; Jagadeesha, R.D.; Raha, S.; Patil, H.; Williams, P. Fuzzy logic systems with data classification—A cooperative approach for intelligent decision support. *ICTACT J. Soft Comput.* **2023**, *14*, 3212–3217. [CrossRef]
26. Divate, M.; Singh, K. Technology acceptance model for incidents monitoring through cloud based platforms. *Online J. Distance Educ. e–Learn.* **2023**, *11*, 1678–1684.
27. Nadeem, A.; Sarwar, M.U.; Malik, M.Z. Automatic Issue Classifier: A Transfer Learning Framework for Classifying Issue Reports. In Proceedings of the 2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), Wuhan, China, 25–28 October 2021; pp. 421–426. [CrossRef]
28. Tian, X.; Wu, J.; Yang, G. BUG–T5: A Transformer–based Automatic Title Generation Method for Bug Reports. In Proceedings of the 2022 3rd International Conference on Big Data & Artificial Intelligence & Software Engineering, Guangzhou, China, 21–23 October 2022; Volume 3304, pp. 45–50. Available online: https://ceur-ws.org/Vol-3304/paper06.pdf (accessed on 22 May 2025).
29. Colavito, G.; Lanubile, F.; Novielli, N.; Quaranta, L. Leveraging GPT–like LLMs to Automate Issue Labeling. In Proceedings of the 21st International Conference on Mining Software Repositories (MSR '24), Lisbon, Portugal, 15–16 April 2024; Association for Computing Machinery: New York, NY, USA, 2024; pp. 469–480. [CrossRef]
30. Reza, S.M.; Mahmud, S.U.; Badreddin, O. G–Issue: Analyzing Lifetime and Evolution of Issue–related Artifacts from Open Source Repositories. In Proceedings of the IEEE/ACM 37th International Conference on Automated Software Engineering (ASE '22), Rochester, MI, USA, 10–14 October 2022; p. 6. Available online: https://smreza.com/publications/ase-2022.pdf (accessed on 25 May 2025).
31. Aung, T.W.W.; Wan, Y.; Huo, H.; Sui, Y. Multi–triage: A multi–task learning framework for bug triage. *J. Syst. Softw.* **2022**, *184*, 111133. [CrossRef]
32. Annunen, R. Issue Delegator: Correct Delegation of Issue Tickets. University of Oulu; Degree Programme in Computer Science and Engineering, 2024; 40p. Available online: https://oulurepo.oulu.fi/bitstream/handle/10024/47782/nbnfioulu-202402131734.pdf (accessed on 27 May 2025).
33. Teslyuk, V.; Batyuk, A.; Voityshyn, V. Method of Software Development Project Duration Estimation for Scrum Teams with Differentiated Specializations. *Systems* **2022**, *10*, 123. [CrossRef]
34. Li, R. *Artificial Intelligence Revolution: How AI Will Change Our Society, Economy, and Culture*; Skyhorse, Simon and Schuster N.Y.: New York, NY, USA, 2020; 288p, ISBN 978-1-5107-5299-3. Available online: https://scholar.google.com/scholar_lookup?title=Artificial+Intelligence+Revolution:+How+AI+Will+Change+Our+Society,+Economy,+and+Culture&author=Li,+R.&publication_year=2020 (accessed on 29 May 2025).
35. Aceves–Fernandez, M.A. *Artificial Intelligence—Emerging Trends and Applications*; InTech eBooks; IntechOpen Limited: London, UK, 2018; 464p, ISBN 978-1-83881-615-5. [CrossRef]

36. Stone, P.; Brooks, R.; Brynjolfsson, E.; Calo, R.; Etzioni, O.; Hager, G.; Hirschberg, J.; Kalyanakrishnan, S.; Kamar, E.; Kraus, S.; et al. Artificial Intelligence and Life in 2030: The One Hundred Year Study on Artificial Intelligence. *arXiv* **2016**, arXiv:2211.06318. [CrossRef]
37. Hopgood, A.A. *Intelligent Systems for Engineers and Scientists: A Practical Guide to Artificial Intelligence*, 4th ed.; CRC Press: Boca Raton, FL, USA, 2021; 514p. [CrossRef]
38. Sheikh, H.; Prins, C.; Schrijvers, E. Artificial Intelligence: Definition and Background. In *Mission AI*; Research for Policy; Springer: Cham, Switzerland, 2023; pp. 15–41. [CrossRef]
39. Gu, S. Exploring the Role of AI in Business Decision–Making and Process Automation. *Int. J. High Sch. Res. (IJHSR)* **2024**, *6*, 94–101. [CrossRef]
40. Kloeckner, K.; Davis, J.S.; Fuller, N.; Lanfranchi, G.; Pappe, S.; Paradkar, A.; Shwartz, L.; Surendra, M.; Wiesmann, D. Transforming the IT Services Lifecycle with AI Technologies. In *SpringerBriefs in Computer Science*; Springer: Cham, Switzerland, 2018; 100p. [CrossRef]
41. Srivastava, S. Utilizing AI systems to automate DevOps processes within the field of Software Engineering. *Int. J. Sci. Res. Eng. Dev.* **2023**, *6*, 906–910.
42. Trudova, A.; Dolezel, M.; Buchalcevova, A. Artificial Intelligence in Software Test Automation: A Systematic Literature Review. In Proceedings of the 15th International Conference on Evaluation of Novel Approaches to Software Engineering, Prague, Czech Republic, 5–6 May 2020; pp. 181–192. [CrossRef]
43. Tadimarri, A.; Jahan, M.; Prakash, S.; Tomar, M. Efficiency Unleashed: Harnessing AI for Agile Project Management. *Int. J. Multidiscip. Res.* **2024**, *6*, 1–13. [CrossRef]
44. Joshi, H. Artificial Intelligence in Project Management: A Study of The Role of Ai–Powered Chatbots in Project Stakeholder Engagement. *Indian J. Softw. Eng. Proj. Manag.* **2024**, *4*, 20–25. [CrossRef]
45. Alazzam, B.A.; Alkhatib, M.; Shaalan, K. Artificial Intelligence Chatbots: A Survey of Classical versus Deep Machine Learning Techniques. *Inf. Sci. Lett.* **2023**, *12*, 1217–1233. [CrossRef]
46. Mariciuc, D.F. A bibliometric review of chatbots in the context of customer support. *J. Public Adm. Financ. Law* **2022**, *26*, 206–217. [CrossRef]
47. Murár, P.; Piatrov, I. Chatbots and Customer Service: AI as a Key Tool for Customer Interaction. In *Media & Marketing Identity*; University of Ss. Cyril and Methodius: Trnava, Slovakia, 2024; pp. 498–505. [CrossRef]
48. Ebsen, T.; Segall, R.S.; Aboudja, H.; Berleant, D. A Customer Service Chatbot Using Python, Machine Learning, and Artificial Intelligence. *J. Syst. Cybern. Inform. J. Syst. Cybern. Inform.* **2024**, *22*, 38–46. [CrossRef]
49. Crowder, J. AI Chatbots. In *Synthesis Lectures on Engineering, Science, and Technology*; Morgan & Claypool Publishers: San Rafael, CA, USA, 2024; 165p. [CrossRef]
50. Paygude, V.; Thorat, J.; Kakade, M.; Waghmode, S.; Shimpi, R. Robotics and Automation Using Artificial Intelligence. *Int. J. Ingenious Res. Invent. Dev. (IJIRID)* **2024**, *3*, 1–6. [CrossRef]
51. Raj, N.; Verma, H.; Sharma, K.; Kumar, M.; Desai, S.G. An Intelligent Human–Machine Interface for Robotics Control. *Int. J. Sci. Res. Sci. Eng. Technol.* **2019**, *6*, 386–391. [CrossRef]
52. Nawalagatti, A.; Kolhe, P.R. A Comparative Study on Artificial Cognition and Advances in Artificial Intelligence for Social–Human Robot Interaction. *Int. J. Robot. Res. Dev.* **2018**, *8*, 1–10. [CrossRef]
53. Amalia, K. Advancements in Robotic Systems and Human Robot Interaction for Industry 4.0. *J. Robot. Spectr.* **2023**, *1*, 100–110. [CrossRef]
54. Python Release Python 3.12.0. Python.org. Available online: https://www.python.org/downloads/release/python-3120/ (accessed on 20 June 2025).
55. Thonny, Python IDE for Beginners. Thonny.org. Available online: https://thonny.org/ (accessed on 20 June 2025).
56. Thonny. Release Version 4.1.4 Thonny. GitHub. 2023. Available online: https://github.com/thonny/thonny/releases/tag/v4.1.4 (accessed on 20 June 2025).
57. R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing. 2024. Available online: https://www.r-project.org/ (accessed on 20 June 2025).
58. Previous Releases of R for Windows, 2020. R–Project.org. Available online: https://cran-archive.r-project.org/bin/windows/base/old/3.6.3/ (accessed on 20 June 2025).
59. Sahu, K.; Alzahrani, F.A.; Srivastava, R.K.; Kumar, R. Evaluating the Impact of Prediction Techniques: Software Reliability Perspective. *Comput. Mater. Contin.* **2021**, *67*, 1471–1488. [CrossRef]
60. Yadav, S.; Kishan, B. Analysis and Assessment of Existing Software Quality Models to Predict the Reliability of Component–Based Software. *Int. J. Emerg. Trends Eng. Res.* **2020**, *8*, 2824–2840. [CrossRef]
61. Machuca–Villegas, L.; Gasca–Hurtado, G.P.; Morillo Puente, S.; Restrepo Tamayo, L.M. An Instrument for Measuring Perception about Social and Human Factors that Influence Software Development Productivity. *JUCS—J. Univers. Comput. Sci.* **2021**, *27*, 111–134. [CrossRef]

62. Manoharan, S.; Sathish, R. Population Based Meta Heuristics Algorithm for Performance Improvement of Feed Forward Neural Network. *J. Soft Comput. Paradig.* **2020**, *2*, 36–46. [CrossRef]
63. Laskaridis, S.; Kouris, A.; Lane, N.D. Adaptive Inference through Early–Exit Networks: Design, Challenges and Directions. In Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning (EMDL'21), Virtual, 25 June 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 1–6. [CrossRef]
64. Rahmath, P.H.; Srivastava, V.; Chaurasia, K.; Pacheco, R.G.; Couto, R.S. Early–Exit Deep Neural Network—A Comprehensive Survey. *ACM Comput. Surv.* **2024**, *57*, 1–37. [CrossRef]
65. Di Francesco, A.G.; Bucarelli, M.S.; Nardini, F.M.; Perego, R.; Tonellotto, N.; Silvestri, F. Early–Exit Graph Neural Networks. *arXiv* **2025**, arXiv:2505.18088. [CrossRef]
66. Luoyu, C. High Dimensional Gradient–Based Rule Extraction for ANN Output Prediction. In Proceedings of the 4th ANU Bio–inspired Computing conference, Canberra, Australia, 22–23 July 2021; pp. 1–10. Available online: https://users.cecs.anu.edu.au/~Tom.Gedeon/conf/ABCs2021/paper1/ABCs2021_paper_121.pdf (accessed on 29 June 2025).