applied system innovation

MDPI

Article

Case-Based Reasoning with an Artificial Neural Network for Decision Support in Situations at Complex Technological Objects of Urban Infrastructure

Igor Glukhikh and Dmitry Glukhikh *

Information Systems Department, University of Tyumen, 6 Volodarskogo Street, 625003 Tyumen, Russia; igluhih@utmn.ru

* Correspondence: gluhihdmitry@gmail.com; Tel.: +79-3232-000-36

Abstract: The article considers the tasks of intellectual support for decision support in relation to a complex technological object. The relevance is determined by a high level of responsibility, together with a variety of possible situations at a complex technological facility. The authors consider case-based reasoning (CBR) as a method for decision support. For a complex technological object, the problem defined is the uniqueness of the situations, which is determined by a variety of elements and the possible environmental influence. This problem complicates the implementation of CBR, especially the stages of comparing situations and a further selection of the most similar situation from the database. As a solution to this problem, the authors consider the use of neural networks. The work examines two neural network architectures. The first part of the research presents a neural network model that builds upon the multilayer perceptron. The second part considers the "Comparator-Adder" architecture. Experiments have shown that the proposed neural network architecture "Comparator-Adder" showed higher accuracy than the multilayer perceptron for the considered tasks of comparing situations. The results have a high level of generalization and can be used for decision support in various subject areas and systems where complex technological objects arise.

Keywords: case-based reasoning; decision making; neural network; neural network architecture; perceptron; comparator; adder; control of complex systems

Citation: Glukhikh, I.; Glukhikh, D. Case-Based Reasoning with an Artificial Neural Network for Decision Support in Situations at Complex Technological Objects of Urban Infrastructure. *Appl. Syst. Innov.* **2021**, *4*, 73. https://doi.org/10.3390/asi4040073

Academic Editor: Luís Bragança

Received: 4 August 2021
Accepted: 17 September 2021
Published: 27 September 2021

1. Introduction

1.1. The Relevance of Research

Modern systems of urban infrastructure (power supply systems, gas, water, and heat supply systems) are complex technological objects (CTO). Their safety and stability processes are important not only for to enable city systems, but also for the protection of the ecology, people's lives, and health. The emergency situation has dangerous and fatal consequences. The prevention of such situations and their removal represent a relevant task in managing complex technological objects.

The dangerous situation prevention relates to monitoring and recognition and to retrieving solutions for neutralizing an incident. The implementation of both tasks in one software–hardware complex leads to the modern concept of intelligence monitoring and decision-making systems (IMDS) [1]. The decision-making process to neutralize (prevent) an arising dangerous situation is aimed at finding an action program (for personnel of operating, service organizations, operational dispatch service, and support services), which should convert the current emergency situation into a target, standard situation.

Two approaches are possible. One assumes the development "from scratch" of an effective action program (AP) based on the existing criteria, limitations, and standards. The second assumes a choice from the ready-made versions of the AP that will most fully correspond to the current situation. The second approach is good in that it does not require

a lengthy development process (this is important in conditions of time pressure) and is applicable in cases where there is no formalization of target functions and selection criteria.

The approach in which the solution is retrieved from the already available database of ready-made solutions examples is called case-based reasoning (CBR) in artificial intelligence systems [2,3]. The CBR method proves to be effective in many applied tasks, including in the tasks of making decisions in emergency or undesirable situations [2–5].

The main idea of CBR is to find the same or a similar situation from past experience in the IMDS knowledge base in a hazardous situation case at CTO, and to apply the solution that was used previously.

To implement CBR inference in each specific subject area, it is necessary to solve a number of problems, where the key is the task to compare and retrieve similar situations. Decision-making in situations at urban infrastructure facilities must take into account many different conditions, not only the state of this or that technological equipment, but also many other factors, e.g., the environment, the state and availability of support systems and personal, the availability of resources, climatic conditions, etc. All these aspects give rise to a wide variety of possible situations, as well as difficulties in their formalized description and comparison for the solution retrieval.

Under such conditions, classical comparison methods based on the distance metrics in the parameter space do not achieve the required accuracy of the results. Therefore, one of the modern directions of research in CBR systems development and application has become the solution search for the problem of comparing and retrieving situations in complex areas [6–9].

1.2. Problem Description and Proposed Method

An important role in CBR is played by the procedure for evaluating the similarity of a situation with another, that is, determining Sim (.).

A sign of similarity between situations can be their belonging to the same class, i.e., Sim (.) $\in$ {0, 1}, with the value 1 when two situations belong to the same class, and the value 0 otherwise.

However, for a detailed classification, especially at the beginning of the system operation, there may not be a sufficient amount of information. In addition, during operation, new situations may appear that go beyond the previously created classification, which will require retraining of the system. Therefore, in CBR, the classification is used only at a sufficiently high level to separate typical situations.

More detailing within types is performed using a set of practical cases that may appear during the system operation. This is what makes it possible, if necessary, to find in the knowledge base those cases that will most closely correspond to the current problem situation.

As a consequence, some special similarity function [10] becomes more important than their belonging to one or another class for evaluating the similarity of the situations among themselves in CBR. The function value will show the similarity of situations among themselves.

Calculation of the similarity function can be performed based on metrics (weighted Euclidean, Manhattan metrics, etc.), by calculating the distance ρ between situations in the attribute space that describe these situations [6]. Then, Sim (.) = $-\ \rho$ or Sim (.) = $(1 - \rho)$ in the case of normalization of distance values.

This method has shown its effectiveness in many machine learning problems and in case-based inference problems when relatively simple and homogeneous objects are considered.

On a complex object case, it is necessary to take into account the states of its various elements and the connections between them, which are described by many quantitative and categorical parameters.

An attempt to compare situations in the multidimensional attributive space of a complex object faces some problems:

- The need to create local similarity metrics and their aggregation into global metrics.

- The need for expert judgment when ranking the importance of attributes.
- The need to identify collisions, i.e., cases where the difference in situations in one attribute or local metrics can be compensated for by their similarity in other attributes.

The problems described are associated with time-consuming tasks that require expert intervention. At the same time, the more complex the object, the higher the labor intensity, and the higher the probability of error and collisions.

This set of problems requires new ways to assess the proximity of situations where expert intervention is minimized, and thereby the accuracy and speed of decision support processes is increased.

As a solution to the described problems, we consider the use of neural networks. Artificial intelligence will speed up the decision-making process, which is especially important in critical situations, eliminate the human factor, and reduce the labor intensity of processes.

The purpose of this work is to develop and research neural network architecture to evaluate the similarity of situations on a complex technological object of urban infrastructure.

The work is based on previous studies, where we define the situation on a complex object through a set of elements states and connections between them [11,12].

The article is organized as follows.

Section 2 provides a brief overview of studies on the possibility of using neural networks to evaluate the proximity of the situation. Further, a formalized concept of a complex technological object is introduced, with a definition and formal presentation of situations arising at a complex object. Then, the process of forming a training data set is described. To do this, many situations in the building's heat supply point are used, on the basis of which two data-set are formed, namely training and validation. Section 3 considers neural network architectures for the situation comparison problem. Two developed architectures are proposed, namely multilayer perceptron and a complex architecture, Comparator-Adder. Section 4 discusses the results obtained and presents some conclusions. In particular, the developed neural network architecture "Comparator-Adder" showed higher accuracy than the multilayer perceptron for the considered problems of situation comparison.

2. Materials and Methods

2.1. Background and Related Work

Methods for learning similarity measures have been a topic of research in the CBR community for many years [13,14]. The possibilities of using neural networks for evaluating the proximity of situations are actively studied. Recent studies show positive results [7,8,15].

Thus, work [7] describes an experiment evaluating the quality of a car by determining its "similarity" in the space of the considered parameters with other known cars. The model under consideration is described by vectors. Those vectors concatenation is fed to the input of the multilayer perceptron, and the output is a signal, the value of which is projected onto the scale of quality classes. With a sufficiently large volume of the training sample, the model showed higher accuracy compared to the classical K-nearest neighbors method based on weighted local metrics.

The work [8] uses Siamese neural networks to compare cases. First, the input vectors are converted into the embeddings, namely vectors reflecting important features of the situations being compared. Next, on the output neuro-classifier, the value of the similarity function between embeddings is calculated. This study considered cases from the field of decision-making in the aquaculture industry, and the experiments showed a sufficiently high accuracy of the results for comparing and retrieving these cases when deriving decisions in the CBR system.

The study [10] proposes a typology of models for evaluating similarity functions, taking into account the ways of forming two components of the evaluation process, namely identifying important features and forming the embeddings, and comparing the embeddings to evaluate the similarity of situations.

Our proposed work is the continuation of applicability studies concerning neural network architectures for solving the problem of comparison and retrieval of situations in CBR systems as applied to complex technological objects of urban infrastructure.

This paper considers two versions of the architectures: the multilayer perceptron and the Comparator-Adder architecture. Software implementation and experiments were carried out in the Google Colaboratory environment in Python using the Keras and Tensorflow libraries. The error function MeanSquaredError was selected as one of the standard functions of the Keras library, with an estimate of the calculation accuracy using the Mape metric (mean absolute percentage error).

2.2. Formation of a Training Dataset

2.2.1. Representation of the Situation at the CTO

For research, a complex technological object was considered, namely a house heat point. The technological scheme is an independent two-circuit heating system, where an external coolant through a heat exchanger transfers thermal energy to the coolant of the home heating system.

Elements of a complex object are formed into groups:

- technological (input pipe, pump, heat exchanger, interior pipe);
- providing (IT, electricity, other equipment);
- personnel (electrician, plumber, emergency service);
- environment (premises, neighbor house, nature event, nature object).

The elements of the "personnel" and "environment" groups are not directly related to the object but are considered part of it since they can influence it. For example, snowfall can make it difficult for personnel to access the facility and affect the composition of the solution to a problem situation.

In order to take into account the peculiarities of a complex technological object, we [11] introduce a formal representation of a complex object O through its elements and relations between them:

$$O = \{O_i \mid i = 1, 2, \ldots, N\}, \quad (1)$$

where O_i with $i \in I_1$ denotes complex object elements, and O_i with $i \in I_2$ denotes relations between complex object elements; $I_1 \cap I_2 = \varnothing$ denotes sets of indices of elements and relationships between them; N denotes the number of considered elements and connections in a complex object.

Let each of $O_i \in O$ corresponds to its set of possible states $S_i = \{S_{ij} \mid j = 1, 2, \ldots, M_j\}$ and at any time moment, the elements O_i can be in one of these states.

Definition 1. The situation at a complex technological object is a set of those states in which the elements O_i are at a given time.

The situation can be formally represented through the matrix of states, where one in the column corresponds to the state of the element (Table 1).

Table 1. State matrix that reflects the situation at the CTO.

Sit_{act}	Workable	Broken	Working	Stopped	Not Available	Available	Influence	Not Influence
Input pipe	1	-	-	-	-	-	-	-
Interior pipe	1	-	-	-	-	-	-	-
Heat exchanger	1	-	-	-	-	-	-	-
Pump	-	-	1	-	-	-	-	-
Other equipment	-	-	1	-	-	-	-	-
IT	-	-	1	-	-	-	-	-
Electricity	-	-	-	-	-	1	-	-
Emergency service	-	-	-	-	-	1	-	-
Plumber	-	-	-	-	-	1	-	-
Electrician	-	-	-	-	-	1	-	-
Neighbor house	-	-	-	-	-	-	-	1
Nature object	-	-	-	-	-	-	-	1
Nature event	-	-	-	-	-	-	-	1
Premises	-	-	-	-	-	-	-	1

2.2.2. Forming a Dataset for Experiments

For the experiment, the dataset has been prepared with 150 pairs of situations in the state matrix view at the object "house heat point" with known values of Sim. At the same time, 50% has been formed by similar situations and another 50% by dissimilar situations.

The situation's similarity Sim was determined by an expert review using the method previously described in our study [12]:

$$\text{Sim}\ (\text{Sit}_z,\ \text{Sit}_{act}) = \sum \beta_i \times (1 - d_i), \tag{2}$$

where β_i is the normalized element importance factor; d_i is the distance between the states in which the i-th element of a complex object is in the compared situations.

The distance between the states of an element is determined by the following formula:

$$d_i = \|S_{i,act} - S_{i,z}\|, \tag{3}$$

where $S_{i,act}$, $S_{i,z}$ is the state of the i-th element in the current situation and the z-th situation, respectively.

For example, the possible states for the pump are ordered in the interval [0,1] in such a way that S_1 is at point 0, S_3 is at point 1, and the rest of the states take values between them. It is visualized in Figure 1.

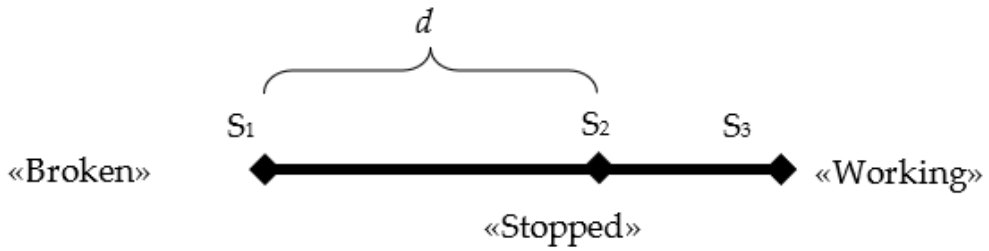

Figure 1. Streamlined states of the element "pump".

Moreover, this method implies an expert evaluation of the importance of an element in each situation.

A set of matrix pairs has been converted to the embeddings in order to bring the required form for the functioning of neural networks. Namely, the matrices have been transformed in each pair into vectors x and y, respectively.

Vector x = (x_{ij} | i = 1, 2, . . . , N, j = 1, 2, . . . , M_i), whose elements take the value 0 or 1, and x_{ij} = 1, if i-th element of O_i is in a state S_{ij}, and 0—otherwise. A single set of 8 states was formed for each of the 14 elements of a complex object, when preparing the data set. The situation presented in the matrix of states (Table 1) will have the vector form formalized, in which the length is 112 positions:

|10000000100000001000000001000000100000 . . . 00000001000000100000001|

Thus, a training dataset (TDS) is obtained for further experiments through pass to formalization in the form of the embeddings. TDS contains 150 pairs of situations, i.e., pairs of vectors (x, y) with known Sim values.

3. Results

3.1. Multilayer Perceptron

The first part of the study considered the neural network model based on a multilayer perceptron, which was used in [7]. There have been changing parameters, i.e., the number of input neurons. The optimal architecture and the number of hidden layers have been selected during experiments. A source code snippet illustrating the multilayer perceptron model is shown below (Listing 1).

In the course of the experiments, the concatenation of two vectors (x, y) with a total length of 224 positions was fed to the input of the neural network. The value Sim (x, y) was calculated at the output.

Next, the neural network trained on the TDS operation was checked using a validation dataset (VDS). This contained a vector representation of 40 pairs of situations that were absent in the TDS.

Listing 1. Defining and compiling a multilayer perceptron to compare situations defined by vectors in state space

```
model = Sequential()
model.add(Dense(448, input_dim = 224, activation = 'relu'))
model.add(Dense(224, activation = 'swish'))
model.add(Dropout(0.3, seed = 2))
model.add(Dense(112, activation = 'relu'))
model.add(Dense(56, activation = 'swish'))
model.add(Dense(28, activation = 'relu'))
model.add(Dropout(0.5))
model.add(Dense(14, activation = 'swish'))
model.add(Dense(1, activation = 'relu'))
# Compile model
model.compile(loss = 'MeanSquaredError', optimizer = 'RMSprop', metrics = ['mape']).
```

Figure 2 shows the graphs of the change in the error and the value of the loss function during training of the neural network on the training dataset.

As a result, the following results were obtained:

- On TDS: Mape absolute error indicator 5.57% with the minimum value of the loss function 0.0032.
- On VDS: The absolute error of Mape 17.96% with the minimum value of the loss function 0.026.

It can be seen that fairly high calculation accuracy is achieved on the training data set. However, the results on the validation set were significantly worse than on the TDS.

Attempts to improve the quality of Sim prediction by increasing the number of network layers and introducing regularization layers did not lead to noticeable improvements in the quality of the results.

The most likely reason for this difference in computational accuracy is the small amount of training data. However, it is important to note that there may not be a large number of situation examples in a real system, especially at the beginning of the operation of a CBR system. We propose a more complex architecture, namely Comparator-Adder for application in a small amount of training data case.

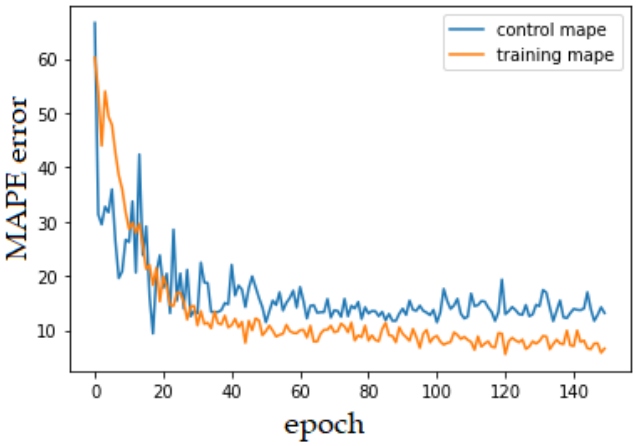

Figure 2. Change in MAPE error on training and control samples when training a multilayer perceptron.

3.2. Comparator-Adder Architecture

The proposed architecture Comparator-Adder is shown in Figure 3. We compare separately the vectors (x_i, y_i), the concatenation of which with a length of 16 positions entered the input of its *i*-th comparator. The comparator is implemented as a multilayer fully connected neural network that determines the similarity between these vectors. N = 14 comparators were trained on their part of the TDS, with each comparator giving a prediction regarding the similarity of the elements in two compared situations.

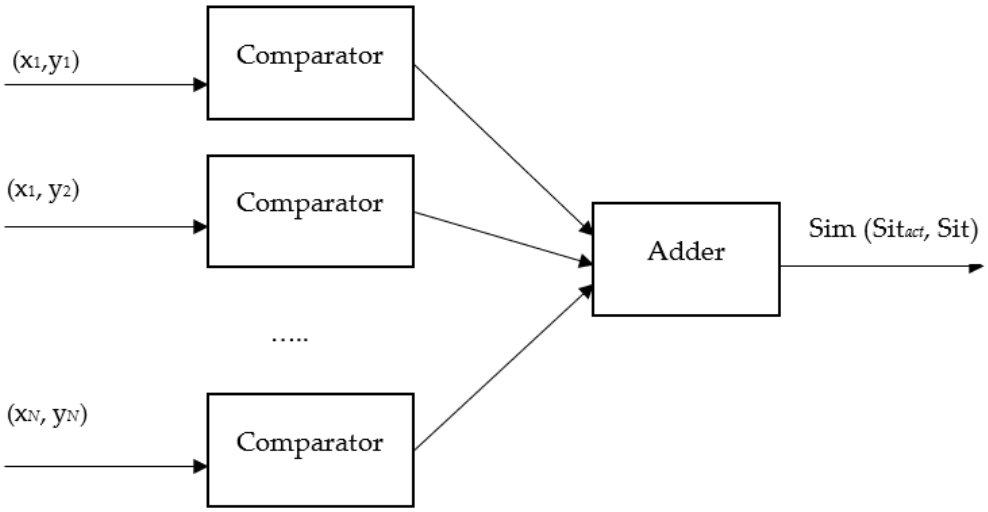

Figure 3. The neural network architecture "Comparator-Adder" for calculation Sim (Sit_{act}, Sit).

Comparator outputs as rounded values of Sigmoid activation functions 0 or 1 come to the input of the adder, the purpose of which is to calculate the final evaluation Sim (x, y) ⇔ Sim (Sit_{act}, Sit).

The optimal parameters of each component of architecture have been selected during experiments. A source code snippet illustrating the comparator model is shown below (Listing 2).

Listing 2. The defining and compiling a comparator

```
model1 = Sequential()
model1.add(Dense(32, input_dim = 16, activation = 'relu'))
model1.add(Dense(8, activation = 'relu'))
model1.add(Dropout(0.5))
model1.add(Dense(1, activation = 'sigmoid'))
# Compile model
model1.compile(loss = 'binary_crossentropy', optimizer = 'adam', metrics = ['accuracy']).
```

The following is the source code snippet for the adder model (Listing 3).

Listing 3. The defining and compiling an adder

```
model_summ = Sequential()
model_summ.add (Dense (28, input_dim = 14, activation = 'relu'))
model_summ.add (Dense (28, activation = 'swish'))
#model_summ.add (Dropout (0.3, seed = 2))
model_summ.add (Dense (14, activation = 'relu'))
#model_summ.add (Dropout (0.5))
model_summ.add (Dense (7, activation = 'swish'))
model_summ.add (Dense (1, activation = 'linear'))
# Compile model
model_summ.compile (loss = 'MeanSquaredError', optimizer = 'RMSprop', metrics = ['mape']).
```

Figure 4 shows the graphs of the change in the error and the value of the loss function during training of the neural network on the training dataset. During the experiments, the following final results were obtained at the output of the adder:

- On TDS: Mape absolute error indicator—5.29% with the minimum value of the loss function 0.0022;
- On VDS: The absolute error of Mape is 9.55% with the minimum value of the loss function 0.0084.

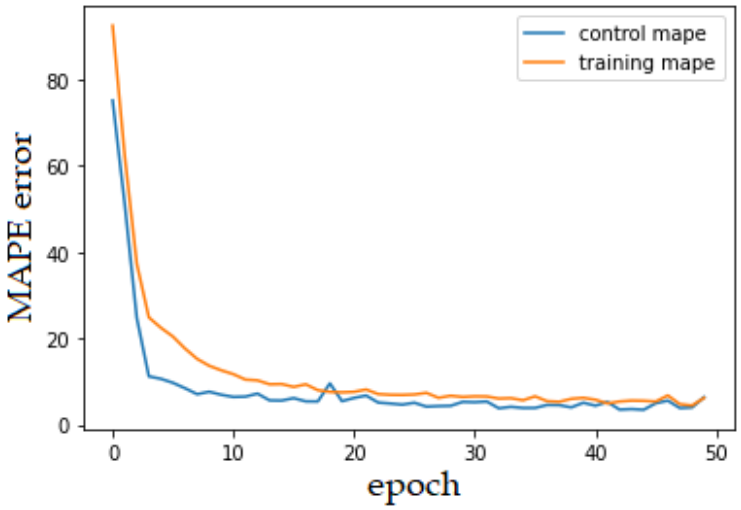

Figure 4. Change in the MAPE error on the training and control sample (on the TDS) when training the adder, to the input of which signals from the comparators are received.

It can be seen data the Comparator-Adder architecture on the validation dataset showed results closer to the data on the TDS than in the study of the perceptron.

The next stage of the study is the analysis of the results of calculations according to the methodology presented in work [8] compliance with the requirements for similarity metrics:

- Sim (Sit_k, Sit_k) = 1
- Sim (Sit_k, Sit_z) = Sim (Sit_z, Sit_k),
- Sim (Sit_k, Sit_k) $\geq$ Sim (Sit_k, Sit_z),

where k, z denote some indices of situations with different values.

In conditions of uncertainty and a lack of training information, the first two requirements are formulated less rigorously: Sim (Sit_k, Sit_k) $\rightarrow$ 1, Sim (Sit_k, Sit_z) $\approx$ Sim (Sit_z, Sit_k).

For this, a subset of various situations in the validation dataset has been selected. Next, the similarity function between them has been calculated using the trained neural network Comparator-Adder. Table 2 shows the calculation results.

Table 2. Sim (.) Calculation data using the Comparator-Adder architecture on examples of situations in the validation file.

Sim(.)	**Sit_1**	**Sit_2**	**Sit_3**	**Sit_4**	**Sit_5**
Sit_1	**0.947**	0.691	0.536	0.624	0.492
Sit_2	0.687	**0.96**	0.38	0.471	0.359
Sit_3	0.59	0.496	**0.912**	0.442	0.802
Sit_4	0.624	0.472	0.338	**0.894**	0.402
Sit_5	0.58	0.454	0.807	0.539	**0.907**

As can be seen from the table, the results obtained correspond to the requirements in their not strict formulation, which approves the most similarities gets with comparison

the situation with itself. It does say about sufficient precision to work under conditions of uncertainty and a lack of training data.

Thus, the experiments have shown that the proposed neural network architecture "Comparator-Adder" for the considered problems of comparison of situations showed: (a) higher accuracy on the same validation file than a multilayer perceptron; (b) the computed evaluation of the similarity of situations meets the requirements for the similarity metric.

4. Discussion

In this work, we considered the key problem of case-based reasoning. At issue is situation similarity evaluation with regard to preventing dangerous situations that arise at complex technological objects of urban infrastructure.

The variety of elements and their states at the complex object lead to high labor intensity or make it impossible to apply classical metrics of situation comparison and retrieval when elements are represented in attribute space and it is necessary to estimate distance in this space.

To improve the selection processes, we suggested using neural networks. Their use will speed up the decision-making process, which is especially important in critical situations, eliminate the human factor, and reduce the labor intensity of the processes.

To solve similar problems in similar difficult conditions, a number of studies [7,8,10,16] have also shown the promise of using the method of neural networks. Neural networks make it possible to determine the similarity of situations through machine learning using examples of pairs of similar or dissimilar situations.

Our proposed architecture Comparator-Adder neural network is based on the idea of Siamese neural networks [8,17–19], which are used to compare images or other signals. The architecture may be an example of the further development of such networks in relation to working with tabular data. In Siamese networks, two channels of neural network computations are organized to encode input images (signals) with their subsequent comparison at the output, decision element, which defines the class "similar" or "dissimilar".

The developed architecture compares separate parts of the input vectors of two situations. Each part corresponds to its own element of a complex technological object. Thus, N-comparison channels are organized in the form of N-trained neural network comparators. The outputs of comparators are fed to the "Adder" neural network. The adder at its output calculates the value of the similarity function, by which one can estimate the degree of similarity of two situations in their general representation.

The experiments showed that a trained neural network demonstrates sufficiently high accuracy under the conditions of a minimal training sample (we used a training date of 150 operations) when evaluating similar actions on the validation file (MAPE less than 10%).

The accuracy estimate obtained when processing the validation data set is a test of the neural network's reliability, while the reliability is digitized and estimated by the MAPE error.

The work [8] compares the results of studies on the application of various models of Siamese networks to assess the similarity function. Experiments carried out in the study showed that after 1000 epochs of training, the retrieval performance (measured as described in the [8]) is 90% ($\pm$0.7%) for esnn, 85.57% ($\pm$3.4%) for chopra, and 82.32% ($\pm$8.7%) for gabel.

Thus, we can say that the efficiency of the neural network presented in our work is sufficient in comparison with similar studies.

The basic idea of our approach to comparing and selecting situations at a complex technological object is to divide tasks into two large stages. At first, the states of the elements of a complex object and the connections between them independently of each other are recognized. As a result, the embeddings that represent situations in the state space are formed. In a second step, the embeddings for the quantitative evaluation of such situations are applied using a neural network. The practical significance of this approach

is due to the different methods and technologies that can be used to recognize the states of dissimilar and diverse elements. So, machine classification methods, and in particular neural network classifiers, can be used in the case of a good description of elements with quantitative data with a sufficient amount of training examples for recognizing states. In cases where there are not enough training examples, it is advisable to use expert knowledge bases and inference systems to recognize the states of elements, including in conditions of uncertainty.

Thus, our approach offers a new opportunity for creating hybrid case-based reasoning models [20–24] and contributes to solving the actual problem of integrating two concepts of artificial intelligence, namely knowledge-based systems and machine learning [25,26].

However, the proposed neural network has some limitations. Namely, it is associated with uncertainty regarding the states of the elements. In such situations, an element with some probability can assume one of several states, which must be taken into account when selecting a similar situation from the base. The implementation of one-hot encoding is impossible in such a situation. Changes in the architecture of the neural network to add the ability to handle situations with uncertainty are the goal of further research.

5. Conclusions and Further Work

In this work, we continued to study the problem of intelligence monitoring and decision-making in emergencies at complex technological objects. Such facilities are present in various urban infrastructure systems (power supply systems, gas, water, and heat supply systems) and in large production, mining, or processing enterprises.

The system reaction speed is important for prompt decision-making and the choice of action programs to eliminate hazardous situations from the system, operational dispatch services, and maintenance personnel. The case-based reasoning method has a high potential to address this need, since it uses ready-made options for action and does not require the development of new solutions when a problem situation arises. At the same time, it is necessary to solve the problem of comparing situations and selecting the one in the knowledge base that is most similar to the current situation.

We have proposed a solution to this problem using trained neural networks. In the course of our research, we relied on the ontological model of a complex technological object proposed earlier [11] and the representation of situations on such an object through the states of its elements and connections between them. In this work, the following main results are obtained:

- The formalization of the representation of the situation with the help of embeddings is proposed, which represent situations in the state space.
- A neural network architecture, Comparator-Adder, is developed to assess the similarity of situations.
- Using the example of the house heat point system, a training and validation dataset was prepared to test the efficiency of the proposed solutions.
- Experiments were carried out to assess the accuracy in predicting the similarity of situations using the proposed neural network architecture. Experiments have shown the applicability of this model for problems of comparison and selection of situations by means of their representation in the state space.

This work does not study the issues of classifying the states of elements and their connections. We consider that various methods and tools will be used to solve these problems, including machine learning methods and knowledge-based systems methods. A further research plan includes the detailing of these methods for different conditions and cases, as well as the development of a general architecture for a hybrid case-based reasoning.

The results have a high level of generalization and can be used for decision support in various subject areas and systems where complex technological objects arise.

Author Contributions: Conceptualization, I.G. and D.G.; methodology, I.G.; formal analysis, I.G.; writing—original draft preparation, I.G. and D.G.; writing—review and editing, I.G. and D.G.;

visualization, D.G.; funding acquisition, I.G. All authors have read and agreed to the published version of the manuscript.

Funding: The research was funded by RFBR and Tyumen Region, project number 20-47-720004.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Glossary

CTO	complex technological objects
IMDS	intelligence monitoring and decision-making systems
AP	action program
CBR	case-based reasoning
TDS	training data-set
VDS	validation data-set

References

1. Juraev, Z.S.; Muhamediyeva, D.T.; Sotvoldiev, D.M. Construction of hybrid intellectual monitoring and decision-making systems. *J. Phys. Conf. Ser.* **2020**, *1546*, 012083. [CrossRef]
2. Aamodt, A.; Plaza, E. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Commun.* **1994**, *7*, 39–59. [CrossRef]
3. Eremeev, A.; Varshavskiy, P.; Alekhin, R. Case-Based Reasoning Module for Intelligent Decision Support Systems. In Proceedings of the First International Scientific Conference "Intelligent Information Technologies for Industry" (IITI'16), Rostov-on-Don—Sochi, Russia, 16–21 May 2016; Springer Science and Business Media LLC: Berlin/Heidelberg, Germany, 2016; Volume 450, pp. 207–216.
4. Huang, K.; Nie, W.; Luo, N. Scenario-based marine oil spill emergency response using hybrid deep reinforcement learning and case-based reasoning. *Appl. Sci.* **2020**, *10*, 5269. [CrossRef]
5. Jiang, X.; Wang, S.; Wang, J.; Lyu, S.; Skitmore, M. A decision method for construction safety risk management based on ontology and improved CBR: Example of a subway project. *Int. J. Environ. Res. Public Health* **2020**, *17*, 3928. [CrossRef] [PubMed]
6. Perner, P. Case-Based Reasoning—Methods, Techniques, and Applications. In *Transactions on Petri Nets and Other Models of Concurrency XV*; Springer Science and Business Media LLC: Berlin/Heidelberg, Germany, 2019; pp. 16–30.
7. Case-based reasoning research and development. In Proceedings of the Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 149–164.
8. Mathisen, B.M.; Bach, K.; Aamodt, A. Using extended siamese networks to provide decision support in aquaculture operations. *Appl. Intell.* **2021**, 1–12. [CrossRef]
9. Nikpour, H.; Aamodt, A. Inference and reasoning in a Bayesian knowledge-intensive CBR system. *Prog. Artif. Intell.* **2021**, *10*, 49–63. [CrossRef]
10. Mathisen, B.M.; Aamodt, A.; Bach, K.; Langseth, H. Learning similarity measures from data. *Prog. Artif. Intell.* **2020**, *9*, 129–143. [CrossRef]
11. Glukhikh, I.; Glukhikh, D. Case based reasoning for managing urban infrastructure complex technological objects. In Proceedings of the CEUR Workshop 2021, Online. 28–29 May 2021; Volume 2843, p. 038.
12. Glukhikh, I.; Glukhikh, D. Situations representation and retrieve in the case-based reasoning system for managing a complex technological object. In Proceedings of the CEUR Workshop 2021, Online. 28–29 May 2021; Volume 2922, p. 017.
13. Aha, D.W. Case-based learning algorithms. In Proceedings of the 1991 DARPA Case-Based Reasoning Workshop, Washington, DC, USA, 31 May 1991; Volume 1, pp. 147–158.
14. Chen, D.; Burrell, P. Case-based reasoning system networks: A review. *Neural. Comput. Applic.* **2001**, *10*, 264–276. [CrossRef]
15. Kenny, E.M.; Keane, M.T. Twin-systems to explain artificial neural networks using case-based reasoning: Comparative tests of feature-weighting methods in ANN-CBR twins for XAI. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence; International Joint Conferences on Artificial Intelligence Organization, Macau, China, 10–16 August 2019; pp. 2708–2715.
16. Hoffmann, M.; Bergmann, R. Informed machine learning for improved similarity assessment in process-oriented case-based reasoning. *arXiv* **2021**, arXiv:2106.15931.
17. Acconcjaioco, M.; Ntalampiras, S. One-shot learning for acoustic identification of bird species in non-stationary environments// Cornel University. *arXiv* **2021**, arXiv:2105.00202.

18. Deshpande, A.M.; Minai, A.A.; Kumar, M. One-shot recognition of manufacturing defects in steel surfaces. *Procedia Manuf.* **2020**, *48*, 1064–1071. [CrossRef]
19. Hsiao, S.-C.; Kao, D.-Y.; Liu, Z.-Y.; Tso, R. Malware image classification using one-shot learning with siamese networks. *Procedia Comput. Sci.* **2019**, *159*, 1863–1871. [CrossRef]
20. Abdelwahed, M.F.; Mohamed, A.E.; Saleh, M.A. Solving the motion planning problem using learning experience through case-based reasoning and machine learning algorithms. *Ain Shams Eng. J.* **2020**, *11*, 133–142. [CrossRef]
21. Guo, Y.; Zhang, B.; Sun, Y.; Jiang, K.; Wu, K. Machine learning based feature selection and knowledge reasoning for CBR system under big data. *Pattern Recognit.* **2021**, *112*, 107805. [CrossRef]
22. Zhai, Z.; Martínez, J.F.; Martínez, N.L.; Díaz, V.H. Applying case-based reasoning and a learning-based adaptation strategy to irrigation scheduling in grape farming. *Comput. Electron. Agric.* **2020**, *178*, 105741. [CrossRef]
23. Guo, Y.; Chen, W.; Zhu, Y.-X.; Guo, Y.-Q. Research on the integrated system of case-based reasoning and Bayesian network. *ISA Trans.* **2019**, *90*, 213–225. [CrossRef] [PubMed]
24. Leake, D.; Ye, X.; Crandall, D. Supporting case-based reasoning with neural networks: An illustration for case adaptation. In Proceedings of the CEUR Workshop 2021, Online. 28–29 May 2021; Volume 2846, p. 1.
25. Bhatia, A.; Pinto, A. Supporting automated construction of knowledge-bases for safety critical applications: Challenges and opportunities. In Proceedings of the CEUR Workshop 2021, Online. 28–29 May 2021; Volume 2846, p. 37.
26. Saker, M.K.; Zhou, L.; Eberhart, A.; Hitzler, P. Neuro-symbolic artificial intelligence: Current trends. *arXiv* **2021**, arXiv:2105.05330 [cs.AI].