applied sciences

MDPI

Article

Detection of Decision-Making Manipulation in the Pairwise Comparison Method

Michał Strada [1], Sebastian Ernst [1], Jacek Szybowski [2] and Konrad Kułakowski [1,*]

[1] Department of Applied Computer Science, AGH University of Krakow, 30-059 Kraków, Poland; mstrada@agh.edu.pl (M.S.); ernst@agh.edu.pl (S.E.)
[2] Faculty of Applied Mathematics, AGH University of Krakow, 30-059 Kraków, Poland; szybowsk@agh.edu.pl
* Correspondence: konrad.kulakowski@agh.edu.pl; Tel.: +48-12-617-3408

Abstract: Most decision-making models, including the pairwise comparison method, assume the honesty of the decision-maker. However, it is easy to imagine a situation where the decision-maker tries to manipulate the ranking results. This problem applies to many decision-making methods, including the pairwise comparison method. This article proposes three simple algorithmic methods for manipulating data using the pairwise comparison method. The proposed solutions try to mimic the behavior of a dishonest decision-maker who, acting under time pressure, chooses a simple strategy that leads to pushing through a given alternative. We also test the susceptibility to detection of the proposed manipulation strategies. To this end, we propose a convolutional neural network architecture, which we train based on generated data consisting of the original random pairwise comparison matrices and their manipulated counterparts. Our approach treats the pairwise comparison matrices as two- or three-dimensional images specific to the decision situation. In the latter case, the matrices are initially transformed into a three-dimensional map of local inconsistencies, and only data processed in this way are subjected to analysis using neural networks. The experiments indicate a significant level of detection of the proposed manipulations. In numerical tests, the effectiveness of the presented solution ranges from 88% to 100% effectiveness, depending on the tested algorithm and test parameters. The measured average computation time for the single case analyzed oscillated below one millisecond, which is a more than satisfactory result of the performance of the built implementation. We can successfully use the neural networks trained on synthetic data to detect manipulation attempts carried out by real experts. Preliminary tests with respondents also indicated high effectiveness in detecting manipulation. At the same time, they signaled the difficulty of distinguishing actual manipulation from a situation in which an expert strongly prefers one or more selected alternatives.

Keywords: pairwise comparisons; manipulation; neural network; machine learning; AHP

check for updates

Citation: Strada, M.; Ernst, S.; Szybowski, J.; Kułakowski, K. Detection of Decision-Making Manipulation in the Pairwise Comparison Method. *Appl. Sci.* **2024**, *14*, 8946. https://doi.org/10.3390/app14198946

Academic Editor: Vangelis Karalis

Received: 16 August 2024
Revised: 26 September 2024
Accepted: 1 October 2024
Published: 4 October 2024

1. Introduction

When it comes to decision-making, especially in high-stakes situations such as medical diagnoses, financial investments, political elections, or major sports tournaments, it is essential to ensure the fairness and transparency of the process [1]. Manipulation of the decision-making process can have serious consequences that can negatively affect individuals [2], society, organizations [3], or tournament results [4]. Prejudice, external pressures, bribery, or multiple factors can influence decision-makers, leading to suboptimal outcomes or harm. In political elections, propaganda, disinformation, or bribery can manipulate voters and influence elections [5], leading to long-term societal consequences. In the context of sports, various forms of cheating, including doping, collusion, and match-fixing [6], as well as coordinated fights [7], are considered unethical and have a negative financial impact on individuals who place bets.

We can take various measures to prevent manipulation and ensure transparency [1] and objectivity in decision-making. For example, the number of decision-makers can be increased to make manipulation more difficult [8], or the decision-making processes may be subject to external oversight (or review) to ensure compliance with ethical and legal standards [9]. Another critical aspect is data-driven decision-making, where objective and reliable data support decision-making. By analyzing these data, decision-makers can identify patterns and anticipate and evaluate alternatives more objectively, reducing the influence of biases and external pressures.

Researchers have proposed many methods to reduce subjectivity and increase transparency, ensuring the decision-making process is open, visible, and not manipulable [10,11]. Many of them are based on well-defined mathematical models [12–14]. Szybowski et al. [14] defined a form of optimal manipulation for a pair of alternatives. Knowing it allows one to estimate the difficulty of the manipulation for a particular model.

In this study, we would like to consider the problem of manipulative behavior for one of the most popular decision-making methods, the quantitative pairwise comparison method, and its best-known multiple criteria implementation, the AHP (Analytic Hierarchy Process). It is a commonly used approach that breaks down complex decisions into smaller, more manageable components, allowing decision-makers to evaluate alternatives based on multiple criteria [15]. The quantitative pairwise comparison method emerged at the beginning of the XX century and is attributed to Thurstone [16]. It has inspired many researchers such as Miller [17] and David [18]. Saaty's seminal paper [19], published in 1977, in which he proposed the AHP method, contributed to its growing popularity. It continues to inspire and be the research subject for those involved in the study of decision-making methods. The research concerns synthesis and aggregation of weights [20,21], inconsistencies and inconsistency indexes [22,23], incompleteness [24–26], preference representation using different representations such as interval, fuzzy, or grey numbers [27–29], and exploring new areas and methods including BWM [30], MACBETH [31], and HRE [32]. In this context, outranking decision-making methods such as PROMETHEE and ELECTRE [33] or hybrid solutions [34,35] are worth mentioning.

Using multiple-criteria decision-making methods (MCDM) can be particularly challenging when bribery and corruption are prevalent. Bribery, corruption, and manipulation can undermine the integrity of the decision-making process [36], leading to decisions based on personal or financial interests rather than on the merits of the evaluated alternatives. This can have severe consequences for stakeholders' welfare and the sustainability of the decision. As a response to this intricate issue, researchers and practitioners have embraced innovative methodologies to scrutinize, comprehend, and alleviate the influence of manipulation.

This paper delves into the intersection of pairwise comparisons, bribery, and the utilization of neural networks, presenting a comprehensive framework to prevent and combat corrupt practices. Moreover, in acknowledgment of the constraints of traditional methods in curbing bribery, there is a burgeoning interest in harnessing advanced technologies. Neural networks, machine learning tools, and a subset of artificial intelligence have showcased remarkable pattern recognition and decision-making capabilities. This paper explores the integration of neural networks as a preventive measure against bribery to create proactive systems capable of identifying and deterring corrupt activities before they escalate.

Section 3 provides a theoretical background for pairwise comparisons and machine learning. The following fundamental terms are described: the pairwise comparison (PC) matrix (Section 3.1), inconsistency (Section 3.2), error measurement (Section 3.3), and machine learning (Section 3.5). In Section 4, three simple manipulation methods in the pairwise comparison method are described in detail, and the results of their use in the form of a colored matrix are presented for each of them. The following Section 5 explains the use of the selected types of neural networks. Section 6 presents the aggregated results of the conducted experiments, while Section 7 offers a discussion of the results. The final section (Section 8) summarizes the completed work and outlines future directions for further investigation.

2. Motivation

It is widespread among those concerned with decision-making methods to believe that decision-makers, sometimes equated with experts, are honest. i.e., they express their opinions according to their most profound conviction and knowledge within a defined decision model. Hence, even when inconsistencies appear in judgments, they are most often explained by simple human frailties. This somewhat romantic assumption comes from decision-makers usually being experts in the field. Consequently, it is inappropriate for an expert to say something incompatible with their knowledge. Interestingly, in the case of electoral systems and social choice methods, expectations of decision-maker honesty are much lower. This is perhaps because the decision-makers are equated with the politicians, and the general perception is that both honesty and integrity among politicians are low (for example, a survey conducted during the 2010 general election in the UK found that 58% of respondents thought the honesty and integrity of elected politicians were low or very low [37]). Nevertheless, these two approaches have resulted in quite a lot of work on the manipulation of electoral systems ([12], pp. 127–169, [38]) and a relatively moderate number of studies on the manipulation of decision-making methods, including the method of pairwise comparison of alternatives. By addressing the topic of manipulation in the pairwise comparison method, we want to enter this gap.

Research on manipulation within decision-making methods often concerns group decision-making (GDM). In such a context, it is easier to identify strategic behaviors that may suggest a desire to influence the outcome directly. Liang et al. [39] propose a consensus model that takes into account possible manipulations and uses several heuristics to identify them, such as an overly strong preference for the chosen alternative or a decision-maker's different preferences compared to the rest of the group. An approach to prevent priority manipulation using minimum adjustment and maximum entropy methods in GDM in social networks is presented in the paper [40]. Wu et al. [41] study a framework preventing manipulations during consensus-finding processes in social network GDM. Xiong et al. [42] propose a consensus framework using historical data and employing several heuristics to help identify manipulation.

Interestingly, manipulations can involve both preferences and trust relationships. Detecting a manipulation entails a penalty that reduces the influence of a given expert or group (cluster) of experts on the outcome. Li et al. deal with consensus finding and potential manipulative behavior using the example of the social network WeChat [43]. Zhang et al. [44] established guidelines for identifying non-cooperative behaviors among DMs and utilized social network analysis to manage these behaviors effectively. Pelta and Yager [45] explore such behaviors during the aggregation process using optimization techniques to counteract the adverse effects of manipulation behaviors. Yager [46,47] delves into strategic preference manipulation behaviors primarily within the selection process. Sasaki [13] defines a game-theoretical GDM model based on aggregating individual judgments with strategic manipulation in mind. Quesada et al. [48] suggest using uniform aggregation operators to manage non-cooperative behaviors. Xu et al. [49] introduce a consensus model that handles non-cooperative behaviors and minority opinions. Although there are statistical methods (not necessarily related to GDM) used to detect fraudulent behavior in addition to GDM [50], most studies focus on group decision-making.

The question arises, however, of whether expert dishonest behavior can only be investigated and detected in the context of group decision-making. The answer is of course not, and some indication may be the ways (heuristics) used in GDM to identify manipulative behavior. For example, many authors agree that one of the characteristics of manipulation is that the expert's preferences are strongly oriented towards one particular alternative [42–44]. In a set of pairwise comparisons, such an inclination implies a significant advantage of one alternative over all others in a series of individual comparisons. Such inclinations can create a kind of pattern or image in the data that can be attempted to be machine-recognized.

Following this observation, we propose a method for detecting such patterns using neural networks in this study. In doing so, we are not limited to GDM. Hence, one can use the presented solution in single- and multi-expert models. The neural network learning process requires data. For this reason, we have proposed a few simple manipulation algorithms for the artificial generation of such learning data. The primary mechanism used in these algorithms is the "strong expert orientation to a given alternative" postulated in [42–44] as one of the distinguishing features of the manipulations. The simplicity of the algorithms is intended to allow the expert's ad hoc behavior to be mimicked during the working session.

3. Preliminaries

When individuals make decisions, they usually compare possible alternatives and choose the one that suits them best. For example, when buying an orange at a store, one may opt for the larger one. When assessing products on a scale, the comparison involves their weight against a one-kilogram standard. Comparing alternatives in pairs facilitates the creation of a ranking system, helping to select the best alternative. In the pairwise comparison method, every alternative is systematically compared against others. This comparison results in a PC matrix where rows and columns correspond to alternatives and individual elements indicate the outcomes of these comparisons. These matrices serve as the basis for priority derivation methods, transforming them into weight vectors. The i-th element of the vector represents the significance of the corresponding i-th alternative. The reliability of this ranking depends on the consistency of the PC matrix. There is a consensus that the less inconsistent the set of comparisons is, the more reliable the assessment becomes. This section briefly overviews fundamental methods used to compute priorities and estimates inconsistency for pairwise comparison matrices.

3.1. PC Matrices

In the PC method, a decision-maker, often an expert in a particular field, compares the alternatives in pairs. The easiest way to show these comparisons is in the form of an $n \times n$ square matrix:

$$\boldsymbol{C} = \left[c_{ij}\right], \tag{1}$$

where n denotes the order of the pairwise comparison PC matrix (the number of alternatives) and each c_{ij} element in the matrix $\boldsymbol{C}$ indicates the result of the comparisons between the alternatives a_i and a_j. The PC matrix $\boldsymbol{C}$ has to be reciprocal, so for every $i, j \in \{1, \ldots, n\}$, the reverse comparison a_j to a_i gives

$$c_{ji} = \frac{1}{c_{ij}}. \tag{2}$$

Several methods allow us to calculate a ranking of n alternatives $A = \{a_1, \ldots, a_n\}$ using $\boldsymbol{C}$ as an input. Let us denote the ranking value of a_i as $w(a_i)$ and compose the priority vector as follows:

$$w = [w(a_1), \ldots, w(a_n)]^T. \tag{3}$$

Although there are many methods to calculate the priority vector w [51–53], we will focus on two of probably the most popular of these [15]. The EVM (eigenvalue method) is proposed by Saaty [19] and the GMM (geometric mean method) is proposed by Crawford and Williams [54].

In the EVM [15], the priority vector w is calculated as a normalized principal eigenvector of $\boldsymbol{C}$ such that ($\|\cdot\|_m = 1$ is the Manhattan norm) $\|w\| = 1$:

$$w(a_i) = \frac{\widehat{w}_i}{\sum_{j=1}^{n} \widehat{w}_j}, \tag{4}$$

where $\widehat{w}$ is the principal eigenvector of $\boldsymbol{C}$, i.e., it satisfies the following equation:

$$\boldsymbol{C}\widehat{w} = \lambda_{max}\widehat{w}. \tag{5}$$

where λ_{max} is the principal eigenvalue.

On the other hand, in the GMM, the priority of the i-th alternative is derived as a rescaled geometric mean of the ith row. Thus, the priority vector w consists of rescaled geometric means for all the rows of the matrix $\boldsymbol{C}$:

$$w = \varphi \left[\left(\prod_{j=1}^{n} c_{1j} \right)^{\frac{1}{n}}, \ldots, \left(\prod_{j=1}^{n} c_{nj} \right)^{\frac{1}{n}} \right]^{T}, \tag{6}$$

where φ is a scaling factor such that $\|w\| = 1$ and in the GMM method, the scaling factor is $\varphi = \left(\sum_{i=1}^{n} \left(\prod_{j=1}^{n} c_{ij} \right)^{\frac{1}{n}} \right)^{-1}$. It is worth mentioning that the EVM and the GMM are equivalent if $n = 3$ ([54], p. 393) but may differ for $n \geq 4$. The difference depends on the inconsistency. It can be shown that it is smaller the smaller the value taken by Koczkodaj's inconsistency index [55].

3.2. Inconsistency

It is commonly assumed that in the optimal scenario, experts comparing each pair of alternatives assign c_{ij}, a value corresponding to the actual priority ratio between the two alternatives a_i and a_j. It is, therefore, natural to expect that

$$c_{ij} = \frac{w(a_i)}{w(a_j)}. \tag{7}$$

If $c_{ij} = c_{ik}c_{kj}$ holds for all $i, j, k \in \{1, \ldots, n\}$, the PC matrix $\boldsymbol{C} = [c_{ij}]$ is said to be multiplicatively consistent. Otherwise, if this condition is not satisfied, $\boldsymbol{C}$ is considered inconsistent.

Each priority vector w induces exactly one consistent PC matrix $\boldsymbol{C}$ and vice versa: a consistent PC matrix $\boldsymbol{C}$ induces exactly one vector w. Thus, for a consistent pairwise comparison matrix, both methods mentioned, the EVM and GMM, result in the same vector w.

Like the PC matrices themselves, their inconsistency is quantitative. Thus, one may be tempted to quantify to what extent the PC matrix (decision data in the form of a set of pairwise comparisons of alternatives) is inconsistent.

One of the most recognizable and frequently used indexes is Saaty's consistency index CI [19], defined as follows:

$$CI(\boldsymbol{C}) = \frac{\lambda_{max} - n}{n - 1}. \tag{8}$$

Additionally, Saaty also introduced the consistency ratio CR [19], defined as follows:

$$CR(\boldsymbol{C}) = \frac{CI(C)}{RI(C)}, \tag{9}$$

where $RI(\boldsymbol{C})$ is the average inconsistency for the entirely random PC matrix with the exact dimensions as $\boldsymbol{C}$. As proposed by Saaty, every PC matrix $\boldsymbol{C}$ for which $CR(\boldsymbol{C}) > 0.1$ is deemed too inconsistent to form the basis of a ranking.

Another consistency index used to determine inconsistency is the Geometric Consistency Index [56], defined as follows:

$$GCI(\boldsymbol{C}) = \frac{2}{(n-2)(n-1)} \sum_{i<j} ln^2\left(c_{ij}\frac{w(a_j)}{w(a_i)}\right), \tag{10}$$

where w is derived using the GMM method (6). Both the *CI* and *GCI* depend on the method of calculating the vector of weights. There are also indexes based on the pairwise comparison matrix itself. An example of such a solution is the determinant inconsistency index CI^* proposed by *Pelaez* and *Lamata* [57], defined as follows:

$$CI^*(C) = \begin{cases} 0 & n < 3 \\ \det C & n = 3 \\ \frac{1}{\binom{n}{3}} \sum^n_{\substack{i,j,k=1 \\ i\neq j, i\neq k, j\neq k}} \det C_{ijk} & n > 3 \end{cases}, \tag{11}$$

where $\det C_{ijk}$ is the minor of a 3×3 sub-matrix C, defined as

$$\det C_{ijk} = \frac{c_{ik}}{c_{ij}c_{jk}} + \frac{c_{ij}c_{jk}}{c_{ik}} - 2. \tag{12}$$

This approach first calculates the local inconsistency values determined by the $\det C_{ijk}$. Then, the global inconsistency value for matrix C is calculated as an arithmetic mean of individual local inconsistency indicators.

The threshold value below which the matrix $\boldsymbol{C}$ is considered sufficiently consistent can be calculated empirically following the approach proposed by [19]. There are many different inconsistency indexes for PC matrices. There are at least a dozen different inconsistency indexes for pairwise comparison matrices, including Koczkodaj's index [58], Kazibudzki's index [59], Brunelli–Cavallo's index [60], Kuo's ordinal consistency indicator [61], and others [22,62]. For a 3 × 3 matrix, monotonic increasing functional relationships exist between multiple indexes, including Saaty's *CI* and the *GCI* [63]. The properties of inconsistency indexes are the subject of numerous studies [64–66].

3.3. Error Measurement

In real-life scenarios, *PC* matrices created by experts are inconsistent. Most often, the value of the inconsistency index is insufficient to detect the pairwise comparisons that correspond to a high level of inconsistency (the Koczkodaj inconsistency index may be an exception [58]). For this reason, the concept of error is often considered [67].

To define an error indicator, let us consider a *PC* matrix $\boldsymbol{C}$, where each entry c_{ij} represents an expert's subjective comparison between alternatives a_i and a_j. The ranking vector w, for $\boldsymbol{C}$, is given as $w = [w(a_1), \ldots, w(a_n)]^T$ (3); then, the error matrix [67] can be defined as follows:

$$E(\boldsymbol{C}) = \left[c_{ij}\frac{w(a_j)}{w(a_i)}\right], \quad \forall\, i,j \in \{1,\ldots,n\}. \tag{13}$$

High error values identify highly inconsistent comparisons and, thus, potentially erroneous expert judgments. For a consistent *PC* matrix $\boldsymbol{C}$, all elements of the error matrix are 1, i.e., $(e_{ij} = 1)$, where $E(C) = [e_{ij}]$.

3.4. Distance Measurement

When both the manipulated matrix and the original matrix are known, the distance between these matrices can be taken as a measure of manipulation. Hence, we have adopted three distance measures to describe the experiments performed. The first of them is the incompatibility indicator $D_{cmp}(A,B)$ introduced by *Saaty* [68]:

$$D_{cmp}(A,B) = \frac{1}{n^2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}b_{ji}\right), \tag{14}$$

The second is the distance indicator $D_{AC}(A,B)$ suggested by *Agoston* and *Csato* [69]

$$D_{AC}(A,B) = \frac{1}{n^2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n}(a_{ij}b_{ji} - 1)\right), \tag{15}$$

And lastly is algorithmic distance, proposed by Tekiele et al. [70]:

$$D_{TBF}(A,B) = \sum_{i=1}^{n}\sum_{j=1}^{n}(\log a_{ij} - \log b_{ij}). \tag{16}$$

Obviously, the values of all indicators described above are inversely proportional to the similarity of the matrices.

3.5. Machine Learning

Machine learning (ML) is a field of artificial intelligence (AI) concerned with algorithms that can learn from the offered data without being explicitly programmed. The term itself was coined by Samuel [71]. The primary branches of ML include the following [72]:

- *supervised learning*, where models are trained to predict the values of output variables (labels) based on the input variables (features) by presenting the model with labeled instances. The function that maps features to labels can be implemented using a broad spectrum of algorithms;
- *unsupervised learning*, where the instances are unlabeled, and the models instead try to learn patterns that can be discovered in the data;
- *reinforcement learning*, where the data are not explicitly labeled and the algorithm only receives a performance score to guide its actions.

As *PC* matrices can be treated as data instances, this paper mainly focuses on supervised learning. One of the most common challenges associated with this technique is to obtain accurate predictions (the pairwise comparison method can also be used for score prediction [73]) (label values), while maintaining the ability to generalize [74], i.e., to predict the values of labels for yet unseen instances or, in other words, to avoid overfitting to the training data.

The standard approach to supervised learning involves splitting the dataset into the training set and the testing set at a particular proportion, e.g., 80:20. The algorithm is trained using the former. Then, the prediction quality is verified for the latter, which the algorithm has not yet seen. Various metrics can be used for this purpose, depending on the task (classification or regression) and the character of the data.

Both the training set and the testing set should be representative of the entire dataset. Hence, it is expected that shuffling be performed before the division. However, this can still result in some bias; therefore, a common approach is to use k-fold cross validation, where the algorithm is trained and tested for different splits of the initial dataset [75]. Convolutional neural networks (CNNs) represent a foundational element in deep learning, exhibiting considerable capacity for processing and comprehending visual data. Inspired by the hierarchical organization of neurons in the human visual system, CNNs emulate this structure, enabling them to extract intricate features from images and other spatial data. Critical components of CNNs are as follows [76]:

- *convolutional layers* serve as the foundation of CNNs, functioning as feature extractors that capture patterns and structures from raw input data. These layers detect edges, textures, and other salient features by convolving learnable filters across the input, enabling the network to discern complex visual patterns.
- *pooling layers* play a pivotal role in spatial downsampling, reducing the computational burden while retaining essential information. These layers distill the most salient features through operations such as max pooling and average pooling, facilitating robustness and generalization.

- *activation functions* infuse nonlinearities into the network, empowering it to learn intricate mappings between input and output. Activation functions like Rectified Linear Unit (ReLU) enable CNNs to capture and model intricate relationships within data by introducing complexity and expressiveness.
- *fully connected layers* integrate the high-level features extracted by preceding layers, culminating in decision-making or classification. These layers, analogous to the brain's association areas, synthesize abstract representations into actionable insights, guiding the network's predictions.

Training convolutional neural networks (CNNs) is an iterative parameter optimization process. During this process, the network learns to minimize the disparity between predicted outputs and ground truth labels. Through back-propagation and stochastic gradient descent, CNNs adjust their internal parameters (weights and biases), gradually converging toward an optimal configuration that maximizes predictive accuracy.

CNNs can be applied in various solutions, from recognizing handwritten digits to identifying plant species; convolutional neural networks (CNNs) excel in assigning labels or categories to images based on their content, enabling automated analysis and categorization. Empowered to localize and identify objects within images or videos, CNNs revolutionize tasks ranging from autonomous driving to surveillance and security, facilitating real-time analysis and decision-making. By segmenting images into semantically meaningful regions, CNNs facilitate new avenues in medical imaging, environmental monitoring, and urban planning, enabling precise analysis and interpretation. CNNs allow accurate and reliable facial recognition systems, empowering applications such as biometric authentication, surveillance, and personalized marketing.

4. Problem Statement

Let us consider a scenario where a decision-maker, an expert in a particular field, must compare multiple alternatives while facing external pressures limiting the available time for making judgments. In a typical situation, the expert creates a PC matrix $\boldsymbol{C}$ ((1)) and calculates the ranking vector w ((5) and (6)), such that $w(a_r) > w(a_p)$, where $r, p \in \{1, \ldots, n\}$. Assuming that the expert may have a personal interest in promoting an alternative a_p over a reference alternative a_r, one possible solution is to create a matrix $\boldsymbol{C}$, keeping in mind that alternative a_p is intended to be better than a_r. In this paper, we focus on an alternative approach—modifying the initial PC matrix $\boldsymbol{C}$ to the matrix $\boldsymbol{C}'$ where $w'(a_r) < w'(a_p)$. To achieve this, we have developed several manipulation algorithms based on the observation that increasing c_{pr} generally increases $w(a_p)$ at the expense of all other priority values. Discussing this strategy, it is worth noting that Csató and Petróczy's study revealed that the EVM (5) is not monotonic, meaning that increasing c_{pr} does not always increase $w(a_p)$. At the same time, this effect does not occur with the GMM approach. However, Monte Carlo experiments showed that this phenomenon is negligible [77]. Consequently, in practical scenarios, we can disregard this aspect for moderately inconsistent PC matrices.

4.1. Naive Algorithm

The first manipulation algorithm is the simplest one. Assuming that the goal is to increase the priority of alternative a_p, we will try to increase all elements in the pth row of the PC matrix $\boldsymbol{C}$.

Let α be a positive real number such that $\alpha > \min\{\max\{c_{ij}\}, n\}$ where $i, j \in \{1, \ldots, n\}$. In the first step of the algorithm, we update all elements in the pth row (except c_{pp}, which must be 1) to α and all elements in the pth column to $1/\alpha$. Without loss of generality, we may assume that $p = 2$. Thus, the initial form of $\boldsymbol{C}$ and the final form, denoted by $\boldsymbol{C}'$, are as follows:

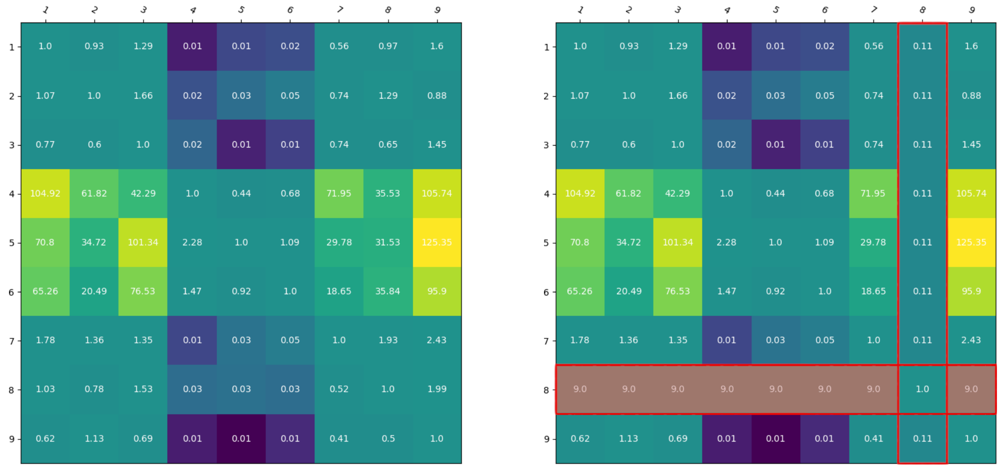
$$\boldsymbol{C} = \begin{bmatrix} 1 & c_{12} & \cdots & c_{1n} \\ c_{21} & 1 & & c_{2n} \\ \vdots & & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & 1 \end{bmatrix} \xrightarrow[\text{attack}]{} \begin{bmatrix} 1 & 1/\alpha & \cdots & c_{1n} \\ \alpha & 1 & & \alpha \\ \vdots & & \ddots & \vdots \\ c_{n1} & 1/\alpha & \cdots & 1 \end{bmatrix}' = \boldsymbol{C}'$$

The "naive" algorithm can be written compactly as follows:

Algorithm 1 Naive algorithm

```
function ATTACK(it takes matrix $C$ and index $p$ on input)
    set the initial value for $\alpha$ and generate a list of indexes to modify
    For each index and in the list to be modified, set as element $c_{pi}$ the value of
    $\alpha$ and, due to the reciprocity preservation, its inverse i.e. $c_{ip}$ set to $\alpha^{-1}$.
end function
```

Figure 1 shows the effect of the *naive* algorithm in the form of a colored PC matrix (i.e., heat map) (the values in the heat map correspond to the c_{ij} values. The color changes continuously from purple (lowest value) to yellow (highest value)). It is visible that the 8th row and column have been modified, while the rest of the values remain unchanged.



	1	2	3	4	5	6	7	8	9
1	1.0	0.93	1.29	0.01	0.01	0.02	0.56	0.97	1.6
2	1.07	1.0	1.66	0.02	0.03	0.05	0.74	1.29	0.88
3	0.77	0.6	1.0	0.02	0.01	0.01	0.74	0.65	1.45
4	104.92	61.82	42.29	1.0	0.44	0.68	71.95	35.53	105.74
5	70.8	34.72	101.34	2.28	1.0	1.09	29.78	31.53	125.35
6	65.26	20.49	76.53	1.47	0.92	1.0	18.65	35.84	95.9
7	1.78	1.36	1.35	0.01	0.03	0.05	1.0	1.93	2.43
8	1.03	0.78	1.53	0.03	0.03	0.03	0.52	1.0	1.99
9	0.62	1.13	0.69	0.01	0.01	0.01	0.41	0.5	1.0

(**a**) colored matrix ***C*** (heat map)

	1	2	3	4	5	6	7	8	9
1	1.0	0.93	1.29	0.01	0.01	0.02	0.56	0.11	1.6
2	1.07	1.0	1.66	0.02	0.03	0.05	0.74	0.11	0.88
3	0.77	0.6	1.0	0.02	0.01	0.01	0.74	0.11	1.45
4	104.92	61.82	42.29	1.0	0.44	0.68	71.95	0.11	105.74
5	70.8	34.72	101.34	2.28	1.0	1.09	29.78	0.11	125.35
6	65.26	20.49	76.53	1.47	0.92	1.0	18.65	0.11	95.9
7	1.78	1.36	1.35	0.01	0.03	0.05	1.0	0.11	2.43
8	9.0	9.0	9.0	9.0	9.0	9.0	9.0	1.0	9.0
9	0.62	1.13	0.69	0.01	0.01	0.01	0.41	0.11	1.0

(**b**) colored matrix ***C***′ (heat map)

Figure 1. Pairwise comparison matrices for the *naive* algorithm ($\alpha = 9$) are presented as a heat map. Lighter colors indicate larger values, whereas darker colors indicate smaller ones.

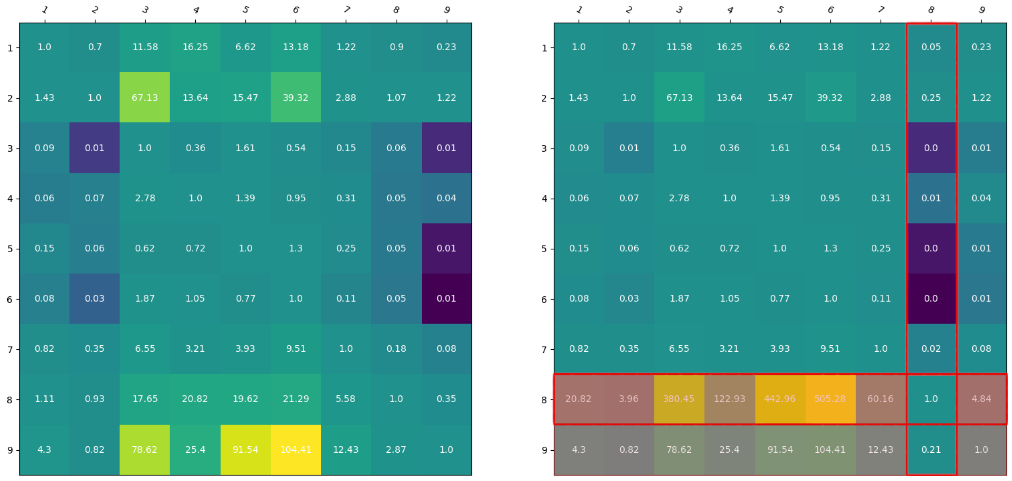
4.2. Basic Algorithm

The *basic* manipulation algorithm is an improved version of the *naive* one in Section 4.1 and the *row* algorithm [78], but the main idea behind the manipulations remains the same. The goal is to improve the ranking of alternative a_p by increasing the values of all elements in the pth row of $\boldsymbol{C}$. Unlike the naive approach, the attack function accepts an additional input argument, denoted as r, representing the reference alternative's index.

Let α be a positive real number randomly selected from the right-side open interval $[1.1, 5)$. Similar to the *naive* algorithm, the first step of the algorithm updates the value of all elements in the pth row, except for the c_{pp} element. The value of the c_{pi} is updated by the α factor multiplied by the corresponding element in the rth row (c_{ri} element). The reciprocity constraint (2) is then applied to change all elements in the pth column. Compared with the *naive* algorithm in the *basic* version, we update the value of c'_{pi} to the value $\alpha c'_{ri}$. Without the loss of generality, assuming $p = 2$ and $r = n$, the initial form of the *PC* matrix (denoted by $\boldsymbol{C}$) and the final form (denoted by $\boldsymbol{C}'$) are as follows:

$$\boldsymbol{C} = \begin{bmatrix} 1 & c_{12} & \cdots & c_{1n} \\ c_{21} & 1 & & c_{2n} \\ \vdots & & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & 1 \end{bmatrix} \xrightarrow[\text{attack}]{} \begin{bmatrix} 1 & 1/c_{n1}\alpha & \cdots & c_{1n} \\ \alpha c_{n1} & 1 & & \alpha \\ \vdots & & \ddots & \vdots \\ c_{n1} & 1/\alpha & \cdots & 1 \end{bmatrix}' = \boldsymbol{C}'$$

The following Figure 2 shows the effect of the *basic algorithm* in the form of a colored PC matrix. The values in the 8th row have been updated with the values from the 9th row multiplied by the α factor (and similarly all reciprocal elements), while the rest remain unchanged.



(**a**) colored matrix $\boldsymbol{C}$ (heat map) (**b**) colored matrix $\boldsymbol{C}'$ (heat map)

Figure 2. Pairwise comparison matrices for the "basic" algorithm ($\alpha = 4.84$) are presented as a heat map. Lighter colors indicate larger values, whereas darker colors indicate smaller ones.

4.3. Advanced Algorithm

This section describes the *advanced algorithm*, which extends the *basic algorithm* (Section 4.2) by adding a stopping constraint. The goal is to improve the ranking of alternative a_p by increasing the values of as few elements as possible in the pth row and column of matrix $\boldsymbol{C}$.

The α factor in this algorithm has the same properties as in the *basic algorithm*. The c_{pi} element in the pth row and its reciprocal counterpart c_{ip} in the pth column are updated sequentially, except for the c_{pp} element, as long as the ranking value (3) of the alternative a_p is lower than the ranking value of the reference alternative a_r or all feasible elements are updated. The updated value of the c_{pi} element is equal to the corresponding c_{ri} element multiplied by the α factor. Unlike the basic approach, we stop updating the elements of the matrix C when $w(a'_p)$ becomes more significant than $w(a'_r)$.

Without loss of generality, we may assume that $p = 2$ and $r = n$. Thus, the initial form of $\boldsymbol{C}$, the intermediate form denoted by $\boldsymbol{C}^{\text{step}}$, and the final form denoted by $\boldsymbol{C}'$ are as follows:

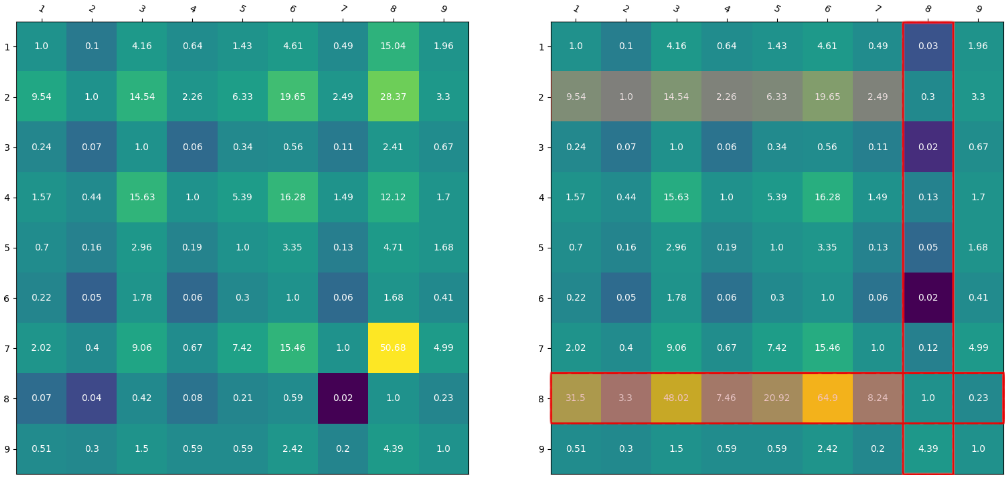
$$\boldsymbol{C} = \begin{bmatrix} 1 & c_{12} & \cdots & c_{1n} \\ c_{21} & 1 & & c_{2n} \\ \vdots & & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & 1 \end{bmatrix} \xrightarrow[\text{first step}]{} \begin{bmatrix} 1 & 1/c_{n1}\alpha & \cdots & c_{1n} \\ \alpha c_{n1} & 1 & & c_{2n} \\ \vdots & & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & 1 \end{bmatrix}' = \boldsymbol{C}^1$$

$$\downarrow \text{next } n-2 \text{ consecutive steps}$$

$$\boldsymbol{C}^{n-2} = \begin{bmatrix} 1 & c_{12} & \cdots & c_{1n} \\ c_{21} & 1 & & c_{2n} \\ \vdots & & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & 1 \end{bmatrix} \xrightarrow[\text{last step}]{} \begin{bmatrix} 1 & 1/c_{n1}\alpha & \cdots & c_{1n} \\ \alpha c_{n1} & 1 & & \alpha \\ \vdots & & \ddots & \vdots \\ c_{n1} & 1/\alpha & \cdots & 1 \end{bmatrix}' = \boldsymbol{C}'$$

It is important to note that when the stopping criteria are met, the number of algorithm steps may be lower than those ($n-1$ steps) shown in the example above.

Figure 3 shows the heat maps for the *advanced* algorithm. Some values in the 8th row have been updated with the values from the 2nd row multiplied by the α factor (and similarly all reciprocal elements), while the rest remain unchanged. The algorithm stopped after the 7th iteration of making changes to the initial *PC* matrix $\boldsymbol{C}$, promoting alternative a'_8 over alternative a'_2.



(**a**) colored matrix $\boldsymbol{C}$ (heat map)

(**b**) colored matrix $\boldsymbol{C}'$ (heat map)

Figure 3. Pairwise comparison matrices for the "advanced" algorithm ($\alpha = 3.3$) are presented as a heat map. Lighter colors indicate larger values, whereas darker colors indicate smaller ones.

5. Detecting Manipulation Using Machine Learning

As mentioned in Section 3.5, the primary focus of the presented work is on supervised learning methods. One of the most important tasks when building such models is to choose the most appropriate algorithm for the data's characteristics.

We can consider PC matrices as mappings in the form $f : A \times A \to \mathbb{R}$, where the domain of the function f is the Cartesian product of alternatives (or indexes of alternatives), and the values are in the set of real numbers. In this sense, they are similar to raster images understood as mapping a set of points $(x, y) \in R^2$ into color space, e.g., the images of handwritten digits in the MNIST dataset [79]. They can be seeded as 2-D images, representing the comparison values, e.g., as shades of gray.

However, most ML algorithms transform the input object into a vector. This may result in the algorithm's inability to "see" certain phenomena in the data. For instance, if the naive manipulation maximizes the values in a single matrix column, visually forming a vertical line, this fact may not be apparent after the data have been flattened, as presented in Figure 4.

Therefore, exploring ML algorithms that can detect 2-D relationships between values seems reasonable. One of the most prominent classes in this category is convolutional neural networks (CNNs) [76]. Compared to fully connected neural networks, these significantly reduce the number of weights (parameters) by utilizing convolution kernels (also called filters).

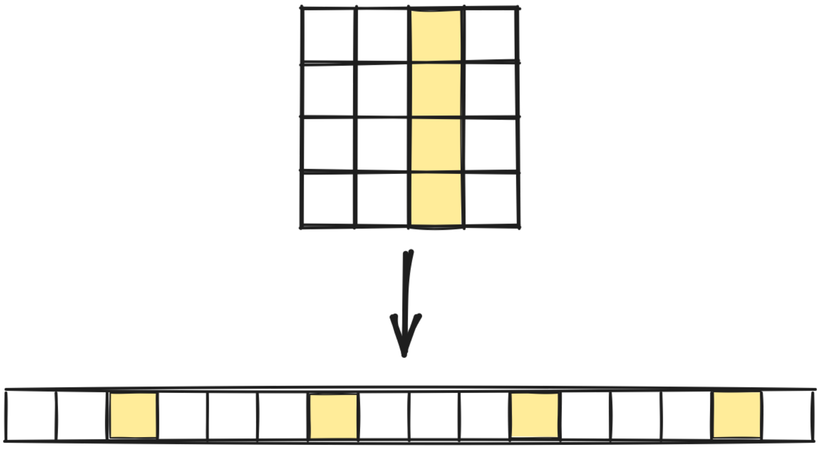


Figure 4. A way to transform data for neural network processing. In this case, a two-dimensional (2-D) PC matrix is flattened to a one-dimensional (1-D) vector.

6. Experiments and Results

This section explores the potential of using CNNs to detect manipulations in PC matrices, describes the tuning process of the network structure, and presents the results of the experiments. To test the neural network's accuracy in detecting the use of the described algorithms, we first conducted a training process and then a series of experiments. A separate network was prepared to detect the use of each of the three attack algorithms. We repeated the neural network training process on PC matrix C of different sizes n.

To ensure that the results of the described algorithms are comparable, we have chosen the following standard input parameters:

- α—for the *naive* algorithm, it is equal to the size of the PC matrix $\boldsymbol{C}$; otherwise, it is a uniformly distributed random number between 1.1 and 5;
- r—index of the reference alternative, is always equal to the index of the highest ranked alternative evaluated from the initial matrix $\boldsymbol{C}$;
- p—index of the alternative to be promoted, randomly selected and different from r.

All matrices used in the experiments were generated as follows:

- first, a ranking vector w (3) of length n was generated;
- next, the ranking vector w was transformed into a consistent PC matrix $\boldsymbol{C}$ using (7);
- finally, the matrix $\boldsymbol{C}$ was disturbed to ensure that its inconsistency (9) was greater than 0.

To introduce disturbances, each value of the c_{ij} element, where $j > i$, was multiplied by a uniformly distributed random number chosen from the experimentally chosen interval $[0.5, 2]$. Subsequently, each c_{ji} was modified using (2).

We generated and attacked $10,000$ PC matrices for each $n \in \{3, 4, \ldots, 9\}$ and for each tested manipulation algorithm. A total of $420,000$ samples of different sizes were generated and manipulated. Next, matrices were divided into two sets: one set, containing 20% of the samples, was used to evaluate the model, and the other set, containing 80% of the samples, was used to train the model.

The attack detection rate $[\%]$ (the percentage of the samples in which the learned network was able to identify whether the matrix had been manipulated or not correctly) is shown in Table 1.

Table 1. Attack detection rate.

n	Naive Algorithm	Basic Algorithm	Advanced Algorithm
3	64	99	88
4	78	99	92
5	89	99	93
6	96	99	93
7	98	∼100	92
8	99	∼100	92
9	∼100	99	90

In the context of binary classification (manipulation detection), a fundamental understanding of the concepts of *false positives* (FPs) and *false negatives* (FNs) is essential for the evaluation and enhancement of predictive model performance. These concepts underpin the metrics employed in model evaluation and serve as the basis for sophisticated analytical techniques such as ROC (Receiver Operating Characteristic) analysis [80]. The numbers of *false positives* and *false negatives* were determined for a series of different matrix sizes, with a total of 4000 test matrices utilized per size for this purpose (Table 2).

Table 2. False positive and false negative numbers.

n	Naive Algorithm	Basic Algorithm	Advanced Algorithm
5	FP: 86, FN: 354	FP: 0, FN: 40	FP: 235, FN: 48
6	FP: 166, FN: 21	FP: 1, FN: 16	FP: 179, FN: 51
7	FP: 72, FN: 5	FP: 0, FN: 9	FP: 148, FN: 76
8	FP: 29, FN: 5	FP: 0, FN: 2	FP: 146, FN: 57
9	FP: 11, FN: 2	FP: 0, FN: 8	FP: 138, FN: 53

Moreover, we investigated the impact of introducing an upper threshold for the pairwise comparison values in the *basic* algorithm. The attack detection rate (for 7 × 7 matrices) in such a scenario is shown in Table 3:

Table 3. Upper threshold impact on the attack detection rate.

Upper Threshold Value	Attack Detection Rate [%]
9	70
20	79
50	86
100	93
200	97

Using the distance indicators ((14)–(16)) between the original and manipulated matrix allow us to tracking the size of the resulting manipulation. In the case of the "advanced" algorithm, this allowed us to detect a weak positive Pearson's correlation between the values of distance indicators (14) and (15), and the inconsistency value determined by Saaty's *CI* (Figure 5). This means that some cases of manipulation attack using this method can be detected by observing the inconsistency level of the PC matrix.

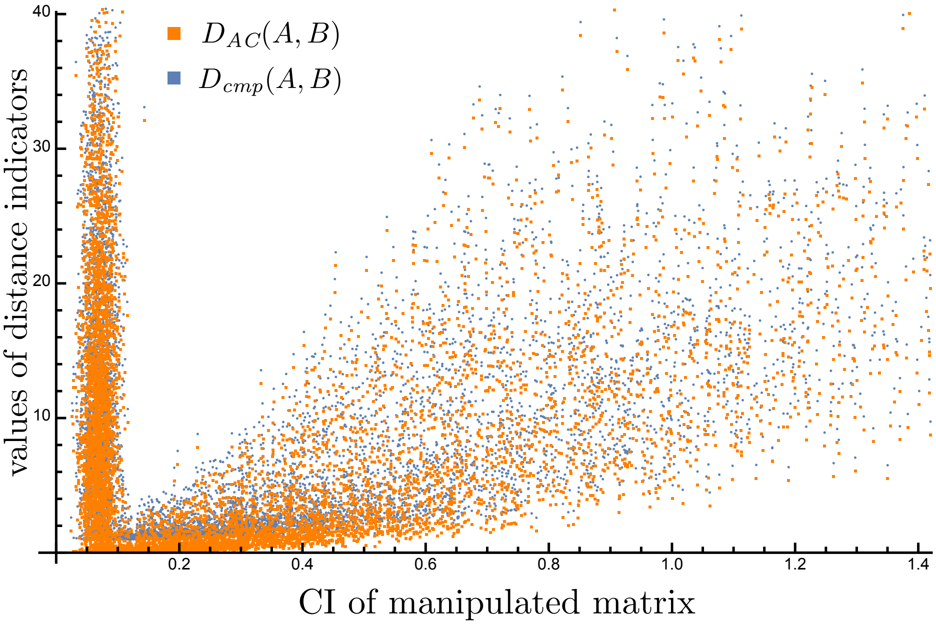


Figure 5. Weak positive correlation between inconsistency of manipulated matrix and distance indicators comparing original and manipulated matrices measured for 9-by-9 matrices. The "advanced" algorithm was used for manipulation.

To evaluate the model, we used preprocessed sample matrices. Our initial attempt to train a neural network on bare matrices $\boldsymbol{C}$ and $\boldsymbol{C}'$ yielded unacceptable results. Next, we attempted to use the error matrix (13) in the preprocessing step to achieve better results. However, the detection rate for the *basic* algorithm oscillated around 65%, which is also unacceptable.

Finally, in the preprocessing step, we map a square matrix $M \subset \mathbb{R}^2$ to a cube $D \subset \mathbb{R}^3$ using the transformation

$$\boldsymbol{D}(\boldsymbol{C}) = \left[d_{i,j,k}\right], \quad d_{i,j,k} = \det \begin{bmatrix} c_{i,i} & c_{i,j} & c_{i,k} \\ c_{j,i} & c_{j,j} & c_{j,k} \\ c_{k,i} & c_{k,j} & c_{k,k} \end{bmatrix}, \quad \forall\, i,j,k \in \{1,\dots,n\} \tag{17}$$

This transformation converts the original matrix C into its three-dimensional map of local inconsistencies calculated in the same way as proposed by Pelaez and Lamata (12) [81]. Thus, in this approach, the object of neural network identification is not the PC matrix itself but its representation, which is a three-dimensional distribution of local inconsistencies. This approach comes across the heuristic according to which dishonest experts make changes "on the fly" that are often not locally consistent [11].

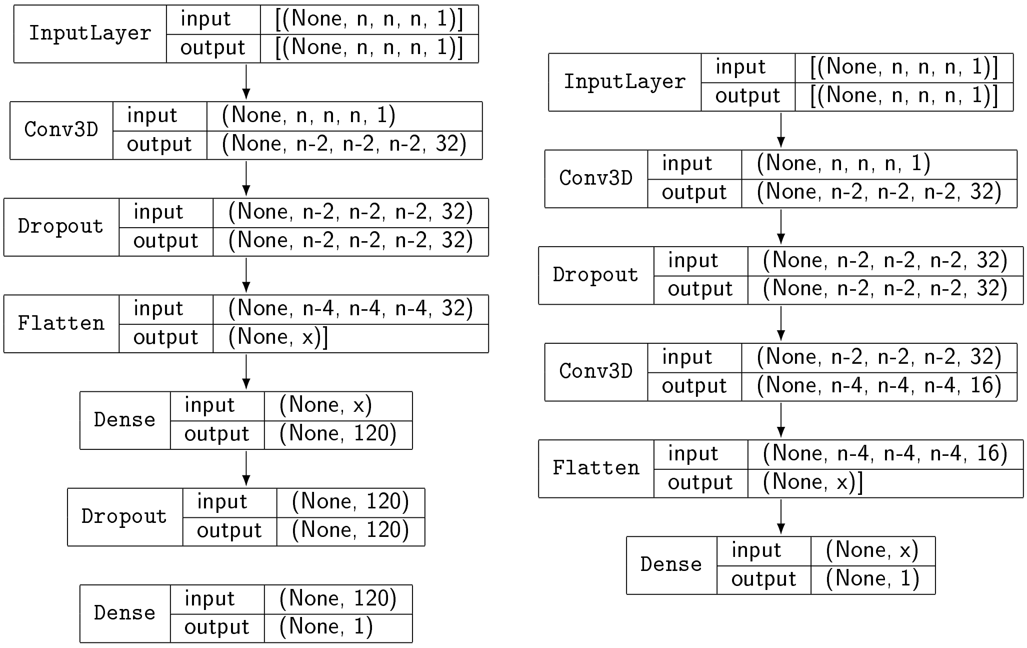
Since our matrix representation is of dimension $n \times n \times n$, we used 3-dimensional convolutional layers (Section 3.5). The concept of the architecture is a classical approach used for CNNs, with a block of convolutional layers followed by a densely connected section. In subsequent layers of the convolutional block, the size of the feature maps decreases. At the same time, the number of the filters grows, allowing further layers to become more specialized in detecting more complex phenomena in the matrices. The 3-dimensional structures are then flattened and fed to a sequence of dense layers which, in the end, yield the manipulation flag. To tune the network, we implemented an automated process using the KerasTuner module to tune the hyperparameters (https://keras.io/keras_tuner/, (accessed on 1 December 2023)).

The following parameters were subjected to the tuning process:

- the number of hidden (convolutional) layers;
- the number of feature maps in each layer, parametrized by the index of the layer;
- the size of filters;

- the learning rate (with logarithmic sampling);
- the optimization algorithm: stochastic gradient descent [82], the Nesterov accelerated gradient (NAG) [83] and adaptive moment estimation (ADAM) [84].

As a result of the tuning process, two architectures were selected. Both of them were parametrized based on the size of the input matrix. Figure 6 shows the design of the two models for different values of n:


InputLayer
input [(None, n, n, n, 1)]
output [(None, n, n, n, 1)]
Conv3D
input (None, n, n, n, 1)
output (None, n-2, n-2, n-2, 32)
Dropout
input (None, n-2, n-2, n-2, 32)
output (None, n-2, n-2, n-2, 32)
Flatten
input (None, n-4, n-4, n-4, 32)
output (None, x)]
Dense
input (None, x)
output (None, 120)
Dropout
input (None, 120)
output (None, 120)
Dense
input (None, 120)
output (None, 1)
InputLayer
input [(None, n, n, n, 1)]
output [(None, n, n, n, 1)]
Conv3D
input (None, n, n, n, 1)
output (None, n-2, n-2, n-2, 32)
Dropout
input (None, n-2, n-2, n-2, 32)
output (None, n-2, n-2, n-2, 32)
Conv3D
input (None, n-2, n-2, n-2, 32)
output (None, n-4, n-4, n-4, 16)
Flatten
input (None, n-4, n-4, n-4, 16)
output (None, x)]
Dense
input (None, x)
output (None, 1)

(**a**) $n = 3, 4$ (**b**) $n = 5 — 9$

Figure 6. Design of neural networks used in the experiments.

It is worth noting that the network's design is simple. However, the detection rate is high, reaching 100% in some cases. For $n = 5$, the Flatten layer is unnecessary and is only included for consistency within the model. For $n = 3$ and $n = 4$, the second convolutional layer has been replaced with a Dense and a Dropout layer, and the Flatten layer has been rearranged. The x value is automatically generated and depends on the value of n.

We also conducted a preliminary test of a neural network trained to detect manipulations using a naive algorithm on actual data we collected. In the first survey, we asked respondents to make pairwise comparisons of vacation destinations. In the second survey, we asked them to "promote" the chosen alternative while suggesting that its victory in direct comparisons would translate into its final score. CNN showed remarkable efficiency in detecting manipulation. At the same time, however, it recorded a noticeable number of false positives (FPs). This was because several people questioned had clear preferences for the selected country, and the overall picture of these preferences was similar to the naive manipulation pattern built. Such a result suggests that in practice, the investigator conducting the decision process should confirm in each such case through direct communication with the expert that the strong preference for the chosen alternative is their honest and sincere choice.

7. Discussion

The experiments show that convolutional neural networks can accurately identify the manipulation methods described in Section 6 with high accuracy. These methods aim to provide a straightforward approach that aligns with the manipulative actions of an expert operating under time constraints and following the decision-making process [85]. We may observe this dishonest behavior in contexts such as sports or other clearly defined decision-making scenarios [86], where variables can be manipulated directly. The solution we present can be used in principle in any decision-making process using the pairwise comparison method. The prerequisite is access to suitable computing software. Detecting potential manipulation in practice can mean manually checking the data and having an expert decide on them or automatically excluding the indicated data as suspicious. Adopting one strategy or another depends on the person overseeing the decision-making process.

While conducting the experiments, we discovered that simple indicators such as the error matrix (13) or inconsistency indexes (9) and (10) are insufficient for adequately training the neural network. We think some features of the manipulation algorithms are only detectable if more than two ranking values are compared simultaneously. Based on this observation, we introduced the determinant factor (17). We have also investigated other methods for transforming $\mathbb{R}^2$ PC matrices to $\mathbb{R}^3$ factors. However, we are still researching their impact on detecting attacks by neural networks.

In the experimental setup, we used 2-D and 3-D convolutional layers with the same number of hidden neurons and layers for both cases. We achieved the optimal results for the 3-D models, although models with only one hidden layer can also produce acceptable results. The detection rates for the 2-D network, which has a similar design as shown in Figure 6, with the error matrix as a preprocessing step (Table 4).

Table 4. Attack detection rate (using 2-D approach).

n	Naive Algorithm	Basic Algorithm	Advanced Algorithm
5	89	86	79
6	95	77	80
7	98	68	82
8	99	67	83
9	∼100	58	82

We used the (17) factor in the preprocessing step because the detection rate of the basic algorithm falls below our threshold. It is exciting that in contrast to the other algorithms, the *naive* algorithm (Table 1) exhibits an increasing attack detection rate as n increases. Our interpretation is that in smaller matrices, pairwise comparison values are more often similar to each other. Nevertheless, our primary focus was on the attack detection rate. In addition, we calculated the number of *false positives* and *false negatives* (Table 2). Based on these values, we believe there is an opportunity to improve the architecture of our networks.

The manipulation detection time for each algorithm increases depending on the matrix's size under consideration. In the small range of problem sizes studied, the increase is more or less linear despite the amount of data growing quadratically or cubically. This may be due to the relatively small size of the input data, which does not saturate the computational capabilities of the hardware. Computation time measurements are included in Table 5. The inconsistency of the matrices considered did not affect the results.

Table 5. Manipulation detection time in μs for three algorithms: "naive", "basic", and "advanced".

n	Naive Algorithm	Basic Algorithm	Advanced Algorithm
5	491	504	488
6	704	546	592
7	686	660	651
8	759	793	767
9	945	964	945

8. Summary

Practice suggests that all known decision-making methods are susceptible to sophisticated manipulation, and decision-making methods based on quantitative pairwise comparisons are no exception. In this work, we proposed a manipulation model and three attack algorithms. We also analyzed the effectiveness of neural networks in detecting attacks for different sizes of pairwise comparison matrices. At the preprocessing stage, we considered various methods to ensure the optimal data format processed in subsequent phases. Detecting manipulation is a significant challenge for researchers. In future research, we plan to investigate this issue further, particularly in the context of different and more complex manipulations for complete and incomplete pairwise comparison matrices. Further investigation is required into the preprocessing step for more advanced manipulation techniques, as well as the neural network models used to detect them. Therefore, this matter will be a challenge for us in future research.

Author Contributions: Conceptualization, M.S., S.E. and K.K.; methodology, M.S, S.E., J.S. and K.K.; software, M.S. and S.E.; validation, M.S., S.E. and K.K.; investigation, M.S.; writing—original draft preparation, M.S.; writing—review and editing, K.K., J.S. and S.E.; supervision, K.K. and S.E.; funding acquisition, K.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been supported by the National Science Centre, Poland, as a part of the project SODA no. 2021/41/B/HS4/03475. Jacek Szybowski has also been partially supported by the Polish Ministry of Science and Higher Education within the internal task of AGH University of Krakow no. 11.11.420.004. This research was funded in whole or in part by the National Science Centre, Poland 2021/41/B/HS4/03475. For the purpose of open access, the author has applied a CC-BY public copyright license to any Author-Accepted Manuscript (AAM) version arising from this submission.

Data Availability Statement: All data can be obtained using the algorithms described in this paper.

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

References

1. Bac, M. Corruption, Connections and Transparency: Does a Better Screen Imply a Better Scene? *Public Choice* **2001**, *107*, 87–96. [CrossRef]
2. Kontek, K.; Sosnowska, H. Specific Tastes or Cliques of Jurors? How to Reduce the Level of Manipulation in Group Decisions? *Group Decis. Negot.* **2020**, *29*, 1057–1084. [CrossRef]
3. Gorsira, M.; Steg, L.; Denkers, A.; Huisman, W. Corruption in Organizations: Ethical Climate and Individual Motives. *Adm. Sci.* **2018**, *8*, 4. [CrossRef]
4. Kendall, G.; Lenten, L.J. When sports rules go awry. *Eur. J. Oper. Res.* **2017**, *257*, 377–394. [CrossRef]
5. Faliszewski, P.; Hemaspaandra, E.; Hemaspaandra, L.A.; Rothe, J. Llull and Copeland Voting Computationally Resist Bribery and Constructive Control. *J. Artif. Intell. Res. (JAIR)* **2009**, *35*, 275–341. [CrossRef]
6. Csató, L. Quantifying incentive (in)compatibility: A case study from sports. *Eur. J. Oper. Res.* **2022**, *302*, 717–726. [CrossRef]
7. Duggan, M.; Levitt, S.D. Winning Isn't Everything: Corruption in Sumo Wrestling. *Am. Econ. Rev.* **2002**, *92*, 1594–1605. [CrossRef]
8. Pelta, D.A.; Yager, R.R. Analyzing the Robustness of Decision Strategies in Multiagent Decision Making. *Group Decis. Negot.* **2014**, *23*, 1403–1416. [CrossRef]
9. Frank, B.; Li, S.; Bühren, C.; Qin, H. Group Decision Making in a Corruption Experiment: China and Germany Compared. *Jahrb. Natl. Stat.* **2015**, *235*, 207–227. [CrossRef]
10. Dong, Y.; Zha, Q.; Zhang, H.; Herrera, F. Consensus Reaching and Strategic Manipulation in Group Decision Making with Trust Relationships. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *51*, 6304–6318. [CrossRef]
11. Kułakowski, K.; Szybowski, J.; Mazurek, J.; Ernst, S. Resilient heuristic aggregation of judgments in the pairwise comparisons method. *Inf. Sci.* **2024**, *657*, 119979. [CrossRef]
12. Brandt, F.; Conitzer, V.; Endriss, U.; Lang, J.; Procaccia, A.D. (Eds.) *Handbook of Computational Social Choice*; Cambridge University Press: Cambridge, UK, 2016; pp. 1–554.
13. Sasaki, Y. Strategic manipulation in group decisions with pairwise comparisons: A game theoretical perspective. *Eur. J. Oper. Res.* **2023**, *304*, 1133–1139. [CrossRef]
14. Szybowski, J.; Kułakowski, K.; Ernst, S. Almost optimal manipulation of pairwise comparisons of alternatives. *J. Glob. Optim.* **2024**, *90*, 243–259. [CrossRef]
15. Mazurek, J. *Advances in Pairwise Comparisons: Detection, Evaluation and Reduction of Inconsistency*; Multiple Criteria Decision Making; Springer Nature: Cham, Switzerland, 2023.

16. Thurstone, L.L. A Law of Comparative Judgment, reprint of an original work published in 1927. *Psychol. Rev.* **1994**, *101*, 266–270. [CrossRef]
17. Miller, J.R. The Assessment of Worth: A Systematic Procedure and Its Experimental Validation. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1966.
18. David, H.A. The method of paired comparisons. In Proceedings of the Fifth Conference on the Design of Experiments in Army Research Developments and Testing, the U.S. Army Biological Warfare Laboratories Fort Detrick, Frederick, MD, USA, 4–6 November, 1959, pp. 1–16.
19. Saaty, T.L. A scaling method for priorities in hierarchical structures. *J. Math. Psychol.* **1977**, *15*, 234–281. [CrossRef]
20. Srdjevic, B.; Srdjevic, Z. Prioritisation in the analytic hierarchy process for real and generated comparison matrices. *Expert Syst. Appl.* **2023**, *225*, 120015. [CrossRef]
21. Ossadnik, W.; Schinke, S.; Kaspar, R.H. Group Aggregation Techniques for Analytic Hierarchy Process and Analytic Network Process: A Comparative Analysis. *Group Decis. Negot.* **2015**, *25*, 421–457. [CrossRef]
22. Brunelli, M. A survey of inconsistency indices for pairwise comparisons. *Int. J. Gen. Syst.* **2018**, *47*, 751–771. [CrossRef]
23. Kułakowski, K.; Talaga, D. Inconsistency indices for incomplete pairwise comparisons matrices. *Int. J. Gen. Syst.* **2020**, *49*, 174–200. [CrossRef]
24. Ágoston, K.C.; Csató, L. Inconsistency thresholds for incomplete pairwise comparison matrices. *Omega* **2022**, *108*, 102576. [CrossRef]
25. Alrasheedi, M.A. Incomplete pairwise comparative judgments: Recent developments and a proposed method. *Decis. Sci. Lett.* **2019**, *8*, 261–274. [CrossRef]
26. Kułakowski, K.; Szybowski, J.; Prusak, A. Towards quantification of incompleteness in the pairwise comparisons methods. *Int. J. Approx. Reason.* **2019**, *115*, 221–234. [CrossRef]
27. Park, J.H.; Park, I.Y.; Kwun, Y.C.; Tan, X. Extension of the TOPSIS method for decision making problems under interval-valued intuitionistic fuzzy environment. *Appl. Math. Model.* **2011**, *35*, 2544–2556. [CrossRef]
28. Duleba, S.; Çelikbilek, Y.; Moslem, S.; Esztergár-Kiss, D. Application of grey analytic hierarchy process to estimate mode choice alternatives: A case study from Budapest. *Transp. Res. Interdiscip. Perspect.* **2022**, *13*, 100560. [CrossRef]
29. Bartl, D.; Ramík, J. A new algorithm for computing priority vector of pairwise comparisons matrix with fuzzy elements. *Inf. Sci.* **2022**, *615*, 103–117. [CrossRef]
30. Brunelli, M.; Rezaei, J. A multiplicative best–worst method for multi-criteria decision making. *Oper. Res. Lett.* **2019**, *47*, 12–15. [CrossRef]
31. Bana e Costa, C.; De Corte, J.M..; Vansnick, J. On the Mathematical Foundation of MACBETH. In *Multiple Criteria Decision Analysis: State of the Art Surveys*; Figueira, J.; Greco, S.; Ehrgott, M., Eds.; Springer: Boston, MA, USA; Dordrecht, The Netherlands; London, UK, 2016; pp. 421–463.
32. Kędzior, A.; Kułakowski, K. Multiple-Criteria Heuristic Rating Estimation. *Mathematics* **2023**, *11*, 2806. [CrossRef]
33. Greco, S.; Ehrgott, M.; Figueira, J.R., Eds. *Multiple Criteria Decision Analysis: State of the Art Surveys*; International Series in Operations Research & Management Science; Springer: New York, NY, USA, 2016.
34. Patil, A.N.; Bhale, N.; Raikar, N.; Prabhakaran, M. Car Selection Using Hybrid Fuzzy AHP and Grey Relation Analysis Approach. *Int. J. Perform. Eng.* **2017**, *13*, 569–576. [CrossRef]
35. Tomczyk, M.K.; Kadziński, M. On the elicitation of indirect preferences in interactive evolutionary multiple objective optimization. In Proceedings of the '20 Genetic and Evolutionary Computation Conference, Cancún, Mexico, 8–12 July, 2020; pp. 569–577. [CrossRef]
36. Orgad, K.; Avinatan, H.; Noam, H. New Approximations for Coalitional Manipulation in General Scoring Rules. *JAIR* **2019**, *64*, 109–145. [CrossRef]
37. McGurran, D. What Do People Really Think of Politicians? *BBC News*, 28 January 2012. Available online: https://www.bbc.com/news/uk-england-16760660 (accessed on 12 March 2024).
38. Taylor, A.D. *Social Choice and the Mathematics of Manipulation*; Outlooks; Cambridge University Press: Cambridge, UK, 2005. [CrossRef]
39. Liang, X.; Guo, J.; Liu, P. A consensus model considers managing manipulative and overconfident behaviours in large-scale group decision-making. *Inf. Sci.* **2024**, *654*, 119848. [CrossRef]
40. Sun, Q.; Wu, J.; Chiclana, F.; Wang, S.; Herrera-Viedma, E.; Yager, R.R. An approach to prevent weight manipulation by minimum adjustment and maximum entropy method in social network group decision making. *Artif. Intell. Rev.* **2022**, *56*, 7315–7346. [CrossRef] [PubMed]
41. Wu, J.; Cao, M.; Chiclana, F.; Dong, Y.; Herrera-Viedma, E. An Optimal Feedback Model to Prevent Manipulation Behavior in Consensus Under Social Network Group Decision Making. *IEEE Trans. Fuzzy Syst.* **2021**, *29*, 1750–1763. [CrossRef]
42. Xiong, K.; Dong, Y.; Zha, Q. Managing Strategic Manipulation Behaviors Based on Historical Data of Preferences and Trust Relationships in Large-Scale Group Decision-Making. *IEEE Trans. Fuzzy Syst.* **2024**, *32*, 1479–1493. [CrossRef]
43. Li, S.; Rodríguez, R.M.; Wei, C. Managing manipulative and non-cooperative behaviors in large scale group decision making based on a WeChat-like interaction network. *Inf. Fusion* **2021**, *75*, 1–15. [CrossRef]
44. Zhang, H.; Palomares, I.; Dong, Y.; Wang, W. Managing non-cooperative behaviors in consensus-based multiple attribute group decision making: An approach based on social network analysis. *Knowl.-Based Syst.* **2018**, *162*, 29–45. [CrossRef]

45. Pelta, D.A.; Yager, R.R. Decision Strategies in Mediated Multiagent Negotiations: An Optimization Approach. *IEEE Trans. Syst. Man, Cybern.-Part A Syst. Hum.* **2010**, *40*, 635–640. [CrossRef]
46. Yager, R.R. Penalizing strategic preference manipulation in multi-agent decision making. *IEEE Trans. Fuzzy Syst.* **2001**, *9*, 393–403. [CrossRef]
47. Yager, R.R. Defending against strategic manipulation in uninorm-based multi-agent decision making. *Eur. J. Oper. Res.* **2002**, *141*, 217–232. [CrossRef]
48. Quesada, F.J.; Palomares, I.; Martínez, L. Managing experts behavior in large-scale consensus reaching processes with uninorm aggregation operators. *Appl. Soft Comput.* **2015**, *35*, 873–887. [CrossRef]
49. Xu, X.; Du, Z.; Chen, X. Consensus model for multi-criteria large-group emergency decision making considering non-cooperative behaviors and minority opinions. *Decis. Support Syst.* **2015**, *79*, 150–160. [CrossRef]
50. Forrest, D.; McHale, I.G. Using statistics to detect match fixing in sport. *IMA J. Manag. Math.* **2019**, *30*, 431–449. [CrossRef]
51. Choo, E.U.; Wedley, W.C. A common framework for deriving preference values from pairwise comparison matrices. *Comput. Oper. Res.* **2004**, *31*, 893–908. [CrossRef]
52. Koczkodaj, W.W.; Orlowski, M. An orthogonal basis for computing a consistent approximation to a pairwise comparisons matrix. *Comput. Math. Appl.* **1997**, *34*, 41–47. [CrossRef]
53. Kou, G.; Lin, C. A cosine maximization method for the priority vector derivation in AHP. *Eur. J. Oper. Res.* **2014**, *235*, 225–232. [CrossRef]
54. Crawford, R.; Williams, C. A note on the analysis of subjective judgement matrices. *J. Math. Psychol.* **1985**, *29*, 387–405. [CrossRef]
55. Kułakowski, K.; Mazurek, J.; Strada, M. On the similarity between ranking vectors in the pairwise comparison method. *J. Oper. Res. Soc.* **2022**, *73*, 2080–2089. [CrossRef]
56. Aguarón, J.; Moreno-Jiménez, J.M. The geometric consistency index: Approximated thresholds. *Eur. J. Oper. Res.* **2003**, *147*, 137–145. [CrossRef]
57. Peláez, J.I.; Lamata, M.T. A new measure of consistency for positive reciprocal matrices. *Comput. Math. Appl.* **2003**, *46*, 1839–1845. [CrossRef]
58. Koczkodaj, W.W.; Urban, R. Axiomatization of inconsistency indicators for pairwise comparisons. *Int. J. Approx. Reason.* **2018**, *94*, 18–29. [CrossRef]
59. Kazibudzki, P.T. On Estimation of Priority Vectors Derived from Inconsistent Pairwise Comparison Matrices. *J. Appl. Math. Comput. Mech.* **2022**, *21*, 52–59. [CrossRef]
60. Brunelli, M.; Cavallo, B. Distance-based measures of incoherence for pairwise comparisons. *Knowl.-Based Syst.* **2020**, *187*, 104808. [CrossRef]
61. Kuo, T. An Ordinal Consistency Indicator for Pairwise Comparison Matrix. *Symmetry* **2021**, *13*, 2183. [CrossRef]
62. Pant, S.; Kumar, A.; Ram, M.; Klochkov, Y.; Sharma, H.K. Consistency Indices in Analytic Hierarchy Process: A Review. *Mathematics* **2022**, *10*, 1206. [CrossRef]
63. Cavallo, B. Functional relations and Spearman correlation between consistency indices. *J. Oper. Res. Soc.* **2020**, *71*, 301–311. [CrossRef]
64. Brunelli, M.; Fedrizzi, M. Axiomatic properties of inconsistency indices for pairwise comparisons. *J. Oper. Res. Soc.* **2015**, *66*, 1–15. [CrossRef]
65. Bozóki, S.; Rapcsák, T. On Saaty's and Koczkodaj's inconsistencies of pairwise comparison matrices. *J. Glob. Optim.* **2008**, *42*, 157–175. [CrossRef]
66. Csató, L. Axiomatizations of inconsistency indices for triads. *Ann. Oper. Res.* **2019**, *280*, 99–110. [CrossRef]
67. Saaty, T.L. Decision making—The Analytic Hierarchy and Network Processes (AHP/ANP). *J. Syst. Sci. Syst. Eng.* **2004**, *13*, 1–35. [CrossRef]
68. Saaty, T.L. Relative Measurement and Its Generalization in Decision Making. Why Pairwise Comparisons are Central in Mathematics for the Measurement of Intangible Factors. The Analytic Hierarchy/Network Process. *Estad. Investig. Oper./Stat. Oper. Res. (RACSAM)* **2008**, *102*, 251–318. [CrossRef]
69. Ágoston, K.C.; Csató, L. A lexicographically optimal completion for pairwise comparison matrices with missing entries. *Eur. J. Oper. Res.* **2024**, *314*, 1078–1086. [CrossRef]
70. Tekile, H.A.; Brunelli, M.; Fedrizzi, M. A numerical comparative study of completion methods for pairwise comparison matrices. *Oper. Res. Perspect.* **2023**, *10*, 100272. [CrossRef]
71. Samuel, A.L. Some Studies in Machine Learning Using the Game of Checkers. *IBM J. Res. Dev.* **1959**, *3*, 210–229. [CrossRef]
72. Russel, S.; Norvig, P. *Artificial Intelligence: A Modern Approach*, 4th ed.; Pearson Education Inc.: London, UK, 2020.
73. Koczkodaj, W.W.; Kakiashvili, T.; Szymańska, A.; Montero-Marin, J.; Araya, R.; Garcia-Campayo, J.; Rutkowski, K.; Strzałka, D. How to reduce the number of rating scale items without predictability loss? *Scientometrics* **2017**, *111*, 581–593. [CrossRef] [PubMed]
74. Bousquet, O.; Luxburg, U.; Rätsch, G. *Advanced Lectures on Machine Learning*; Springer: Berlin/Heidelberg, Germany, 2004. [CrossRef]
75. Fushiki, T. Estimation of prediction error by using K-fold cross-validation. *Stat. Comput.* **2009**, *21*, 137–146. [CrossRef]
76. Geron, A. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd ed.; O'Reilly Media: Sebastopol, CA, USA, 2019.
77. Csató, L.; Petróczy, D.G. On the monotonicity of the eigenvector method. *Eur. J. Oper. Res.* **2020**, *292*, 230–237. [CrossRef]

78. Strada, M.; Kułakowski, K. Manipulation of individual judgments in the quantitative pairwise comparisons method. *arXiv* **2022**, arXiv:2211.01809.
79. Deng, L. The MNIST Database of Handwritten Digit Images for Machine Learning Research. *IEEE Signal Process. Mag.* **2012**, *29*, 141–142. [CrossRef]
80. Fawcett, T. An introduction to ROC analysis. *Pattern Recognit. Lett.* **2006**, *27*, 861–874. [CrossRef]
81. Lamata, M.T.; Peláez, J.I. A method for improving the consistency of judgements. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* **2002**, *10*, 677–686. [CrossRef]
82. Bottou, L. *On-line Learning and Stochastic Approximations*; Publications of the Newton Institute, Cambridge University Press: Cambridge, UK, 1999; pp. 9–42.
83. Nesterov, Y.E. A method of solving a convex programming problem with convergence rate $O\left(\frac{1}{k^2}\right)$. *Doklady Akademii Nauk SSSR* **1983**, *269*, 543–547.
84. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015; Conference Track Proceedings.
85. Fülöp, J. Introduction to Decision Making Methods. BDEI-3 workshop, Washington, November 2005
86. Baker, D.; Bridges, D.; Hunter, R.; Johnson, G.; Krupa, J.; Murphy, J.; Sorenson, K. *Guidebook to Decision-Making Methods*; Westinghouse Savannah River Company: Aiken, SC, USA, 2001; Volume 45.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.