mathematics

Article

A Weights Direct Determination Neural Network for International Standard Classification of Occupations

Dimitris Lagios [1], Spyridon D. Mourtas [2,3,*], Panagiotis Zervas [1] and Giannis Tzimas [1]

[1] Data and Media Laboratory, Department of Electrical and Computer Engineering, University of Peloponnese, 26334 Patras, Greece
[2] Department of Economics, Mathematics-Informatics and Statistics-Econometrics, National and Kapodistrian University of Athens, Sofokleous 1 Street, 10559 Athens, Greece
[3] Laboratory "Hybrid Methods of Modelling and Optimization in Complex Systems", Siberian Federal University, Prosp. Svobodny 79, 660041 Krasnoyarsk, Russia
* Correspondence: spirmour@econ.uoa.gr

Abstract: Multiclass classification is one of the most popular machine learning tasks. The main focus of this paper is to classify occupations according to the International Standard Classification of Occupations (ISCO) using a weights and structure determination (WASD)-based neural network. In general, WASD-trained neural networks are known to overcome the drawbacks of conventional back-propagation trained neural networks, such as slow training speed and local minimum. However, WASD-based neural networks have not yet been applied to address the challenges of multiclass classification. As a result, a novel WASD for multiclass classification (WASDMC)-based neural network is introduced in this paper. When applied to two publicly accessible ISCO datasets, the WASDMC-based neural network displayed superior performance across all measures, compared to some of the best-performing classification models that the MATLAB classification learner app has to offer.

Keywords: neural networks; weights and structure determination; multiclass classification; international standard classification of occupations; machine learning

MSC: 68T10; 91B40; 65F20

Citation: Lagios, D.; Mourtas, S.D.; Zervas, P.; Tzimas, G. A weights direct determination neural network for international standard classification of occupations. *Mathematics* **2023**, *11*, 629. https://doi.org/10.3390/math11030629

Academic Editors: Alessandro Niccolai and Huawen Liu

Received: 23 December 2022
Revised: 12 January 2023
Accepted: 22 January 2023
Published: 26 January 2023

1. Introduction

In machine learning (ML), the classification problem arises in a variety of fields, including economics and finance [1], medicine [2], medical [3], and engineering [4]. Multiclass classification is a significant issue in these fields. Each instance in the learning set is a member of a different set of labels that were previously established for multiclass classification. The goal of supervised classification techniques is to build a learning model from a training set of labeled data so that it can classify new objects with unknown labels [5]. The multiclass classification problem can be solved by extending the binary classification problem using several ML approaches, such as NNs, decision trees, k-nearest neighbor, naive Bayes, and support vector machines [6]. The main focus of this paper is to handle multiclass classification tasks using NNs.

On the one hand, NNs are frequently employed for classification and regression problems, and they have been effectively used in a variety of fields, including but not limited to, economics and finance, medicine, and engineering. Particularly, in the field of economics and finance, NNs have been used to optimize portfolios [7], analyze time series [8], stabilize stochastic exchange rate dynamics [9], and forecast a variety of macroeconomic factors [10]. In the field of medicine, NNs have been used to diagnose breast cancer [11], lung cancer [12], flat foot [13], and to classify diabetic retinopathy [14], whereas in the field of engineering, they have been used to stabilize feedback control systems [15], track mobile objects [16], analyze the performance of solar systems [17], and forecast the flow behavior of alloy [18].

On the other hand, social science research usually involves multiclass classification tasks, such as characterizing occupational mobility [19], conducting case-control studies in healthcare [20], examining the relationship between cancer and changes in occupational characteristics [21], and evaluating jobs' potential for telework [22]. Particularly, a multitude of classification systems are the foundation for the systematic surveillance of any population, from the general populace of the world to the people who make up a small- or medium-sized business or community [23]. The occupation (job title) or the industry a person works in can be used to reflect occupational exposure for a variety of work-related topics, such as administrative usage, employment statistics, social sciences, international trade and commerce, comprising surveillance and analytical methodologies [24]. There are numerous national and international classification schemes for these concepts, which are periodically updated [25]. International Standard Classification of Occupations (ISCO), in particular, was created to make it easier to compare occupational statistics across borders and to act as a guide for nations creating or updating their own national occupational classification systems. It has been developed to serve broad administrative and research purposes, whereas the international community fully endorses it as a recognized benchmark for international labor statistics [26]. It is worth mentioning that the most recent version of ISCO, known as ISCO-08, was adopted in 2008. Some case studies on occupational classification can be found in the Finnish Register of Occupational Diseases [27] and on the names of music professions [28].

The task of classifying occupations using NNs has become increasingly common during the past few years [29–33]. In particular, a mathematical model that uses transformer NNs and ISCO-08 to connect education and occupations is proposed in [29]. By using NNs and ISCO-08, ref. [30] links the necessary skills that have surfaced in the labor market from a demand viewpoint and occupations. Ref. [31] proposes a method for analyzing job titles and embedding them, and then utilizes NN models to demonstrate that they outperform humans and establish the baseline accuracy for identifying occupations based on ISCO-08. Using statistical ML and ISCO-08, a comprehensive occupational and economic framework for wage prediction is constructed in [32]. Ref. [33] describes a method to assess and choose embeddings from a large text corpus while maintaining the co-hyponyms links synthesized from ISCO-08 taxonomy, and uses the chosen embeddings as features in order to classify co-hyponym associations through NNs.

In this paper, we will use a feed-forward NN to handle multiclass classification tasks in social sciences. Back-propagation algorithms have a long history of being used to train feed-forward NNs, where the structure of the network is iteratively adjusted. However, a feature that their forerunners lacked is provided by recently introduced weights and structure determination (WASD) training algorithms. In particular, the weights direct determination (WDD) process, which is a component of all WASD algorithms, makes it easier to directly compute the ideal set of weights, preventing one from becoming stuck in local minima and ultimately helping to attain lower computational complexity [34]. We thus develop a 3-layer feed-forward WASD for multiclass classification (WASDMC)-based NN. The novel WASDMC algorithm uses the WDD process combined with a power maxout activation function to train the WASDMC-based NN. The prominent multiclass classification task of classifying occupations based on ISCO-08 is taken into consideration to examine the performance of the WASDMC-based NN. In particular, when applied to two publicly accessible ISCO datasets, the WASDMC-based NN displayed superior performance across all measures, compared to some of the best-performing classification models that the MATLAB classification learner app has to offer.

The main contribution of this paper is the use of a WASD-based NN to address multiclass classification tasks for the first time and the development of a new activation function specifically created for WASD-based NNs. The key ideas of this work can be summed up as follows:

- A novel 3-layer feed-forward WASDMC-based NN for multiclass classification tasks is presented.
- A novel power maxout activation function is employed to increase the accuracy of WASDMC-based NN.
- Two publicly accessible ISCO datasets are considered, and the WASDMC-based NN performance is compared to some of the best-performing classification models that the MATLAB classification learner app has to offer.
- A different power activation function is also used to show how the proposed power maxout activation function behaves differently from other activation functions.

The structure of the paper is described in the sections that follow. An overview of the final structure and justification for the WASDMC-based NN is given at the start of Section 2. The creation of the power maxout activation function and the design of the WDD procedure ensue. The section concludes with a detailed explanation of the WASDMC algorithm and the entire training procedure. The two ISCO datasets' data preparation for use with the WASDMC-based NN is described in Section 3. The results of the WASDMC-based NN on the two ISCO datasets are shown in Section 4, and its performance is evaluated in comparison to that of other well-known models. A brief summary and helpful information for the MATLAB package that has been made accessible on GitHub are also included in Section 4 to support the quality and computational utility of this work. Final observations are provided in Section 5.

2. The WASDMC-Based NN Model

This section describes the 3-layer feed-forward WASDMC-based NN for multiclass classification tasks. Its structure, which has m input and n hidden layer neurons, can be as shown in Figure 1. In particular, Layer 1 is the input layer, receiving the input values $X_1, X_2, \dots, X_m$, and allocating them to the relevant neuron in Layer 2, which has a maximum number of activated neurons of n, with equal weight 1, whereas Layer 3 is the output layer and has one activated neuron. The WDD process is used to acquire the weights $W_i, i = 1, 2, \dots, n-1$ in the neurons that connect Layer 2 and Layer 3 neurons. The NN model can achieve low hidden layer utilization by employing the WASDMC algorithm.

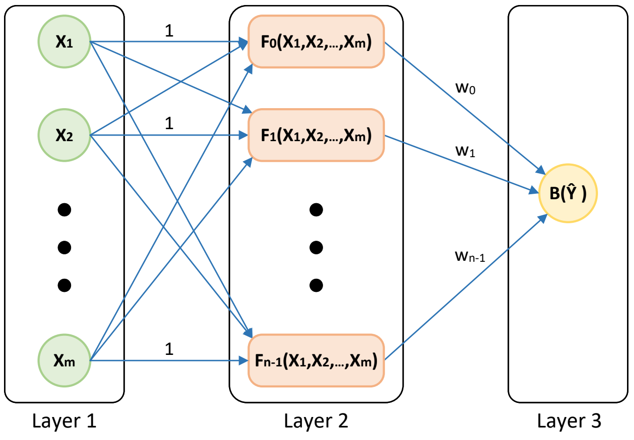


Figure 1. Structure of the WASDMC-based neural network.

2.1. *The WDD Process with the Power Maxout Activation Function*

The WDD process is a key component of any WASD algorithm since it eliminates the need for time-consuming, frequently unreliable iterative computations to find the ideal weights matching to the present hidden layer structure. Apparently, compared to conventional weight determination methods, the WDD process helps to achieve both speed and lower computational complexity, while avoiding some of the associated difficulties [34]. Here, detailed explanations of crucial theoretical foundations and analyses are offered for the WASDMC-based NN design. The Taylor polynomial (TAPO) approximation theorem [35] is first established as follows.

Theorem 1. *When a target function $f(\cdot)$ has the $(K+1)$-order continuous derivative on the interval $[a_1, a_2]$ and K is a non-negative integer, the following applies:*

$$f(x) = A_K(x) + E_K(x), \quad x \in [a_1, a_2], \tag{1}$$

where $A_K(x)$ and $E_K(x)$, respectively, denote the K-order TAPO approximation of $f(x)$ and the error term.

Considering the value of the r-order derivative of $f(x)$, which is denoted by $f^{(r)}(u)$ at the point u, $f(x)$ can be approximately represented as seen below:

$$f(x) \approx A_K(x) = \sum_{r=0}^{K} \frac{f^{(r)}(u)}{r!}(x-u)^r, \quad u \in [a_1, a_2], \tag{2}$$

where $r!$ denotes the factorial of r.

Proposition 1. *To approximate multivariable functions, one may utilize the TAPO approximation Theorem 1. Consider the target function $f(x_1, x_2, \dots, x_h)$ with h variables and $(K+1)$-order continuous partial derivatives in an origin's neighborhood $(0, \dots, 0)$. Then, the K-order TAPO $A_K(x_1, x_2, \dots, x_h)$ about the origin is the following:*

$$A_K(x_1, x_2, \dots, x_h) = \sum_{r=0}^{K} \sum_{r_1+\cdots+r_h=r} \frac{x_1 \cdots x_h}{r_1 \cdots r_h} \left(\frac{\partial^{r_1+\cdots+r_h} f(0, \cdots, 0)}{\partial x_1^{r_1} \cdots \partial x_h^{r_h}} \right), \tag{3}$$

where $r_1, r_2, \dots, r_h$ are non-negative integers.

The WDD process limits the input data to only be real numbers. As a result, we assume the input $X = [X_1, X_2, \dots, X_m] \in \mathbb{R}^{1\times m}$ and the target vector $Y \in \mathbb{R}$. Based on the power activated multi-input NNs in [34], the relationship between the input variables $X_1, X_2, \dots, X_m$ and the output target Y of the NN can be represented by the nonlinear function shown below:

$$f(X_1, X_2, \dots, X_m) = Y. \tag{4}$$

Therefore, according to Proposition 1, the K-order TAPO $A_K(X_1, X_2, \dots, X_m)$ can map (4) as follows:

$$A_K(X_1, X_2, \dots, X_m) = \sum_{r=0}^{n-1} z_r w_r, \tag{5}$$

where $z_r = R_r(X_1, X_2, \dots, X_m) \in \mathbb{R}^{1\times mn}$ is a power activation function, $w_r \in \mathbb{R}^{mn}$ is the weight that refers to z_r, and r indicates both the power value and the number of the hidden layer neurons.

On WASD-based NNs, a variety of activation functions are used, including Chebyshev and Euler polynomials, signum, power, sine, and square wave [36–40]. However, because each activation function has its own combination of empirical and mathematical characteristics, it might be more appropriate for a certain problem. When dealing with multiclass classification tasks, the maxout activation function [41] is regarded as one of the most efficient activation functions. After modifying the maxout activation function to be employed in (5), the following elementwise power maxout activation function is proposed:

$$R_r(X_i) = \begin{cases} X_i^r & , X_i^r = \max(X^{\odot r}) \\ 0 & , \text{otherwise} \end{cases}, \quad \text{for } i = 1, 2, \dots, m, \tag{6}$$

where the superscript $()^{\odot}$ denotes the Hadamard (or elementwise) exponential.

For a given number of samples $s \in \mathbb{N}$, we assume the standardized input matrix $X = [X_1, X_2, \ldots, X_m] \in \mathbb{R}^{s \times m}$ and the target vector $Y \in \mathbb{R}^s$. Further, setting $z_{s,r} = R_r(X_1, X_2, \ldots, X_m) \in \mathbb{R}^{s \times mn}$, the input-activation matrix becomes

$$Z = \begin{bmatrix} z_{1,0} & z_{1,1} & \cdots & z_{1,n-1} \\ z_{2,0} & z_{2,1} & \cdots & z_{2,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ z_{s,0} & z_{s,1} & \cdots & z_{s,n-1} \end{bmatrix} \in \mathbb{R}^{s \times mn}, \tag{7}$$

and the weight vector becomes $W = [w_0, w_1, \ldots, w_{n-1}]^{\mathrm{T}} \in \mathbb{R}^{mn}$. Note that the superscript $()^{\mathrm{T}}$ denotes transposition. Then, instead of iterative weight training in conventional NNs, the weights of the WASDMC-based NN in Figure 1 are computed using the WDD process, which is stated in the following Lemma 1.

Lemma 1. *The K-order TAPO NN's steady-state weights can be directly obtained as shown below [35]:*

$$W = Z^{\dagger} Y, \tag{8}$$

where the superscript $()^{\dagger}$ denotes pseudoinversion.

2.2. The WASDMC Algorithm and the NN Model

The WASDMC algorithm is liable for training the NN model. To avoid overfitting, the data X must first be normalized in the range $[0, 1]$ [34]. After that, the WDD process is used on all the training samples of the data X. Particularly, setting $r = 0$, the Z matrix is constructed according to (6) and (7), and the weights of the NN are directly obtained through (8). To calculate the NN predictions $\hat{Y}$, the following formula is used:

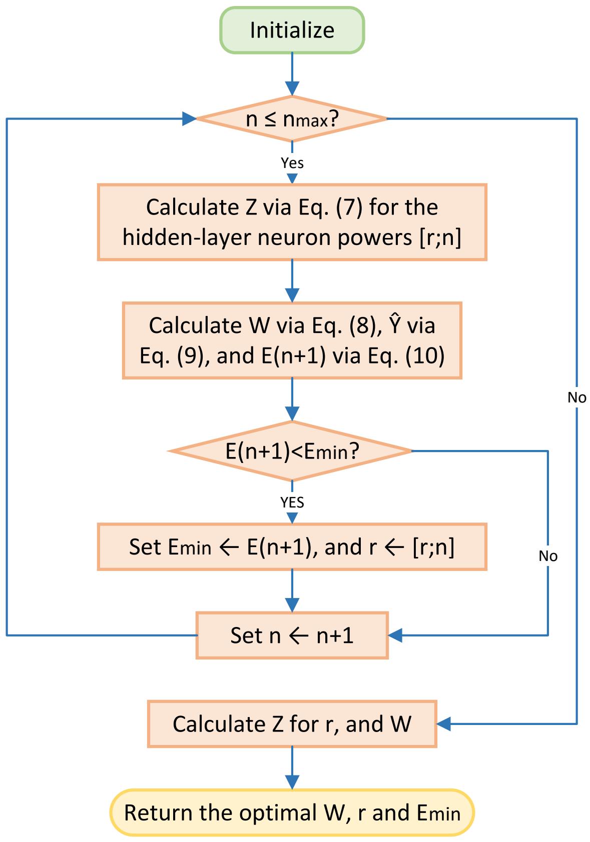
$$\hat{Y} = \mathrm{round}(ZW), \tag{9}$$

where $\mathrm{round}(\cdot)$ denotes a round function. It is significant to note that since the WDD process limits the input data to only be real numbers, the output of ZW is also a real number. Given this, the round function in (9) is used to ensure that this output rounds to the closest positive integer that corresponds to a single class. Then, the mean absolute error (MAE) between the predictions $\hat{Y}$ and the target value Y is measured. Keep in mind that the MAE is a measurement of errors representing the same phenomenon between paired observations and is typically used as a loss function in ML for classification tasks. Furthermore, MAE values that are closer to zero are preferable, and they are calculated as follows:

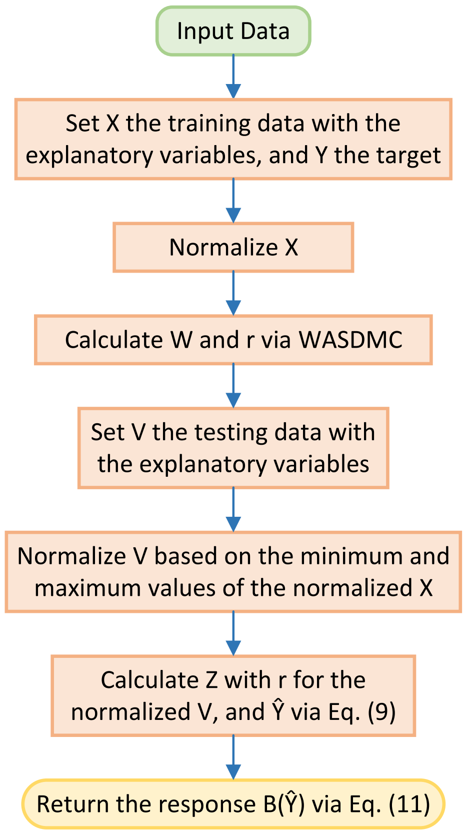
$$\mathrm{MAE} = \frac{1}{s} \sum_{q=1}^{s} |Y_q - \hat{Y}_q|, \tag{10}$$

where s implies the number of samples. In this way, the WASDMC algorithm validates the data in the entire training set X.

The WASDMC algorithm then iteratively selects the optimal activation function for each r depending on the MAE. Given a maximum number of hidden-layer neurons $n_{\max}$, the highest performing powers r in (6) are picked iteratively from 0 to $n_{\max} - 1$. Specifically, a new hidden-layer neuron forms under a given n if the MAE for that n is lower than the previous best MAE, whereas a n is bypassed and not included in the hidden-layer neurons if the MAE for that n is higher than the previous best MAE. In this manner, the WASDMC algorithm is able to maintain the lowest number of hidden-layer neurons in the NN while lowering the total MAE of the NN model. It is important to mention that $\max(X^{\odot r}) = 1$ because of the normalization in the range $[0, 1]$. As a consequence, the maximum number of hidden-layer neurons is limited to $n_{\max} = 2$. This happens as a result of the properties of (6), which lead to every matrix Z for $n > 2$ being the same as the matrix Z for $n = 2$. The diagram in Figure 2a shows the comprehensive process of the WASDMC algorithm.


Initialize
n ≤ nmax?
Yes
Calculate Z via Eq. (7) for the hidden-layer neuron powers [r;n]
Calculate W via Eq. (8), Ŷ via Eq. (9), and E(n+1) via Eq. (10)
E(n+1)<Emin?
YES
Set Emin ← E(n+1), and r ← [r;n]
Set n ← n+1
No
No
Calculate Z for r, and W
Return the optimal W, r and Emin

(**a**) The WASDMC algorithm.


Input Data
Set X the training data with the explanatory variables, and Y the target
Normalize X
Calculate W and r via WASDMC
Set V the testing data with the explanatory variables
Normalize V based on the minimum and maximum values of the normalized X
Calculate Z with r for the normalized V, and Ŷ via Eq. (9)
Return the response B(Ŷ) via Eq. (11)

(**b**) Process for modeling and predicting.

Figure 2. The WASDMC algorithm and the process for modeling and predicting with the WASDMC-based NN model.

Based on the aforementioned procedures, the following are the main steps for the complete process of training and predicting the WASDMC-based NN model of Figure 1. First, the NN model receives the standardized input X of s samples in Layer 1. Then, the WASDMC algorithm discovers the optimal number of hidden-layer neurons n^* as well as the optimal power of the activation function at each hidden-layer neuron in Layer 2. Last, $\hat{Y}$ is calculated through 9, and the following elementwise function is employed in Layer 3:

$$B(\hat{Y}_i) = \begin{cases} \max(Y) & , \hat{Y}_i > \max(Y) \\ \hat{Y}_i & , \min(Y) \leq \hat{Y}_i \leq \max(Y) \\ \min(Y) & , \hat{Y}_i < \min(Y) \end{cases}, \quad \text{for } i = 1, 2, \dots, s, \tag{11}$$

The diagram in Figure 2b shows the comprehensive process for modeling and predicting with the WASDMC-based NN model. It is important to mention that the normalization of the input data is described in the following Section 3. Additionally, since the result of (9) directly pertains to a single class, denormalization is not necessary.

3. Data Preparation

This section describes the two ISCO datasets' data preparation for use with the WASDMC-based NN. It is worth mentioning that ISCO is often built on the concepts of skill and job, i.e., the ability to do the activities and duties of a certain job, which may require some amount of formal training. A job is defined as a specific type of work performed by one person and serves as the statistical unit of ISCO-08. In other words, ISCO is an instrument for dividing jobs into a set of groups that are clearly defined in accordance with the tasks and responsibilities performed in the job, and it is designed to be used in a range of client-oriented applications as well as statistical applications. From a social, economic, and medical standpoint, this division is beneficial. Additionally, jobs can be classified into 436 4-digit unit groups, 130 3-digit minor groups, 43 2-digit sub-major groups, and ten major 1-digit groups using the ISCO four-level hierarchically organized classification system. In our approach, the classification tasks solely took into account the 4-digit unit groups.

For the performance assessment of the WASDMC-based NN, two ISCO datasets are employed. The first dataset is taken from the International Labor Organization (ILO) and can be accessed at [42]. We shall refer to this dataset as ILO-ISCO for simplicity. The second dataset is taken from [43] and includes data extracted from online sources (OS), pre-processed, and hand-annotated following the ISCO taxonomy. We shall refer to this dataset as OS-ISCO for simplicity. On the one hand, there are 436 different classes of occupations in ILO-ISCO, and each class has a variety of job titles that fit within it. On the other hand, there are 2581 different job titles in OS-ISCO, and each one is associated with one of 149 different classes of occupations.

To train and test the NN model, the input data can only be real numbers due to restrictions imposed by the WDD process. However, both of the datasets in our study contain strings; therefore, the data must be adequately processed and prepared before being input into the WASDMC-based NN model. The process that each dataset should go through is described in the following steps $\mathbf{S_1}$ to $\mathbf{S_5}$.

$\mathbf{S_1}$: It is necessary to develop a primary vocabulary for each class. To accomplish this, the different job titles that all fall under one class must be grouped together, and then they must be tokenized according to rules based on Unicode Standard Annex #29 [44], with all letters converted to lowercase and punctuation removed. As a result, given that there are m classes, we create the structure vector $V = [v_1, v_2, \dots, v_m]$, where $v_j, j = 1, \dots, m$ is the vocabulary of class j.

$\mathbf{S_2}$: Consider that the NN model will be tested using s in number samples, each of which contains one job title. It is necessary to develop a vocabulary for each job title. To accomplish this, each job title must be tokenized according to rules based on Unicode Standard Annex #29 [44], with all letters converted to lowercase and punctuation removed. As a result, we create the structure vector $G = [g_1, g_2, \dots, g_s]$, where $g_i, i = 1, \dots, s$ is the vocabulary of some class.

$\mathbf{S_3}$: Considering that some words of a vocabulary in G may belong to several different class vocabularies in V, we create an input matrix X for training and testing the model that includes the percentages of similarity between the vocabularies in G and the vocabularies in V. Particularly, the following process is taken into consideration to create the input matrix $X \in \mathbb{R}^{s\times m}$. Assume the aforementioned structure vectors V and G, where the vocabularies $v_j, j = 1, \dots, m$ and $g_i, i = 1, \dots, s$ have k_j and h_i amount of words, respectively, and that strcmp$(\cdot)$ is a function that returns 1 (true) if the input strings are identical and 0 (false) otherwise (see [45]). Then, the percentage of the vocabulary v_j that matches the vocabulary g_i is

$$P_v(i,j) = \frac{1}{k_j}\sum_{q=1}^{k_j} \mathrm{strcmp}(g_i, v_j(q)), \tag{12}$$

where $v_j(q)$ refers to the qth element (word) of the vocabulary v_j, and the percentage of the vocabulary g_i that matches the vocabulary v_j is

$$P_g(i,j) = \frac{1}{h_i}\sum_{q=1}^{k_j} \mathrm{strcmp}(g_i, v_j(q)). \tag{13}$$

This led us to determine that the ij element of X should represent the average of (12) and (13) as follows:

$$X(i,j) = \frac{1}{2}(P_v(i,j) + P_g(i,j)), \tag{14}$$

which takes values in the range $[0,1]$.

S$_4$: To train the NN model, we create the input matrix X_{tr}. Given that there are m distinct classes, setting $G = V$ in step **S$_3$** will create an input matrix $X_{tr} \in \mathbb{R}^{m \times m}$. As a result, X_{tr} will include the percentages of similarity between the vocabularies in V. Additionally, we create the target vector $Y_{tr} = [1, 2, \ldots, m]^{\mathrm{T}} \in \mathbb{R}^{m}$, where each number from 1 to m corresponds to a single class.

S$_5$: To test the NN model, we create the input matrix X_{te}. Given that there are s in number samples, step **S$_3$** will create an input matrix $X_{te} \in \mathbb{R}^{s \times m}$. As a result, X_{te} will include the percentages of similarity between the vocabularies in G and the vocabularies in V. Additionally, the numbers that the classes have taken in Y_{tr} should be used to create the target vector $Y_{te} \in \mathbb{R}^{s}$.

It should be emphasized that steps **S$_1$** to **S$_5$** are a heuristic approach that accomplishes the problem's goals, namely converting texts to numbers and standardizing the input matrix. By employing the steps **S$_1$** to **S$_5$**, the following input matrices and target vectors will be created for training and testing the models for each dataset. In the case of ILO-ISCO, we will create an input matrix $X_{tr} \in \mathbb{R}^{436 \times 436}$ and a target vector $Y_{tr} \in \mathbb{R}^{436}$ since there are 436 different classes of occupations. Additionally, we will use the first and the last job titles of each class as test data. As a result, we will create an input matrix $X_{te} \in \mathbb{R}^{872 \times 436}$ and a target vector $Y_{te} \in \mathbb{R}^{872}$. In this way, there are 1308 samples in total, of which 33.3% are used in the training set and 66.7% in the testing set. In the case of OS-ISCO, we will create an input matrix $X_{tr} \in \mathbb{R}^{149 \times 149}$ and a target vector $Y_{tr} \in \mathbb{R}^{149}$ since there are 149 different classes of occupations. Additionally, we will use all the job titles of each class as test data. As a result, we will create an input matrix $X_{te} \in \mathbb{R}^{2581 \times 149}$ and a target vector $Y_{te} \in \mathbb{R}^{2581}$ since there are 2581 different job titles. In this way, there are 2730 samples in total, of which 5.5% are used in the training set and 94.5% in the testing set. It is important to mention that the number of samples in the training set must always equal the number of different classes of occupations, whereas the number of samples in the testing set can be chosen at the user's discretion. In light of this, the user may decide on the training-to-testing dataset ratio.

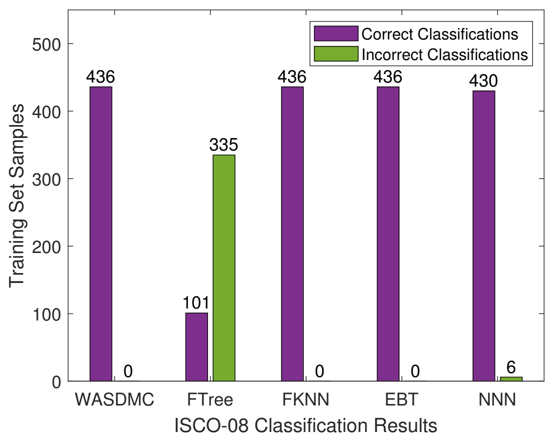
4. NNs Performance Assessment

In this section, the performance of the WASDMC-based NN is examined using the multiclass classification task of classifying occupations based on ISCO-08. Using the ILO-ISCO and OS-ISCO datasets described in Section 3, the WASDMC-based NN performance is compared to some of the best-performing classification models that the MATLAB classification learner app has to offer. These classification models are fine tree (FTree), fine K-nearest neighbors (FKNNs), the ensemble bagged trees (EBTs), and the narrow neural network (NNN). More information on the algorithms that these classification models employ and how they are implemented can be found in [46]. Additionally, to demonstrate how the proposed activation function performs differently from existing activation functions, the power activation function (PAF) from [34] is also employed in place of the power maxout (6). The following elementwise function is the PAF:

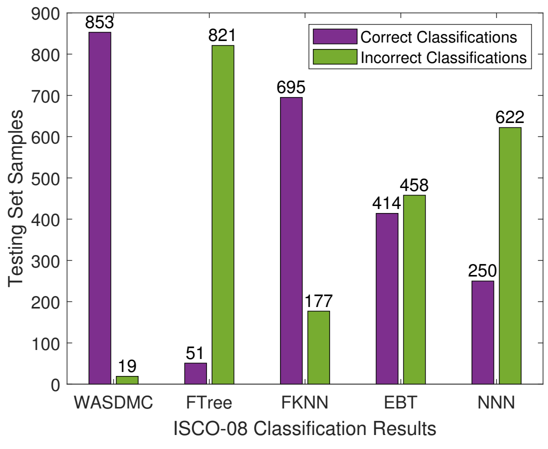
$$R_r(X_i) = X_i^r, \quad \text{for } i = 1, 2, \ldots, m. \tag{15}$$

In this section, the WASDMC-PAF is the abbreviation for the WASDMC-based NN with the PAF, while the WASDMC is the abbreviation for the WASDMC-based NN with the power maxout activation function. Additionally, the WASDMC-PAF maximum number of hidden-layer neurons is set to $n_{\max} = 20$, while the FTree, FKNN, EBT, and NNN models' training parameters are kept at their default configuration in the MATLAB classification learner app. It is worth mentioning that you can download the full development and implementation of the computational methods described in this paper from GitHub at [47]. The concepts and algorithms described in Sections 2 and 3 were used in this link to construct a MATLAB package that can handle the multiclass classification tasks required by this section. Additionally, the MATLAB package offers extensive installation guidelines and thorough implementation.

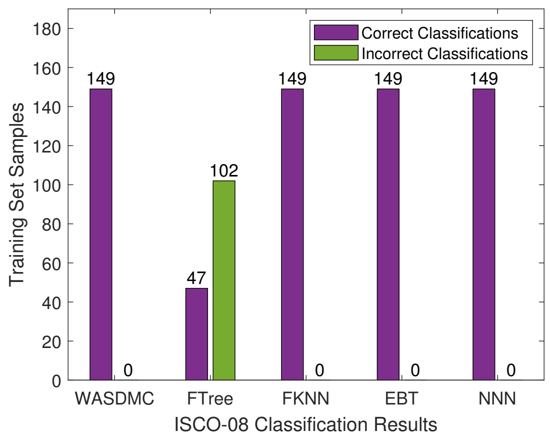
In the case of ILO-ISCO, the training set's input matrix $X_{tr} \in \mathbb{R}^{436\times436}$ and the target vector $Y_{tr} \in \mathbb{R}^{436}$ from Section 3 were used for training the classification models. Their results on the training set are presented in Figure 3a, where we observe that the WASDMC, FKNN and EBT all have perfect classification scores. That is, these models classified correctly all the 436 samples of the training set. The WASDMC-PAF performs the worst and the FTree performs the second-worst, while both models feature more incorrect classifications than correct ones. Furthermore, the testing set's input matrix $X_{te} \in \mathbb{R}^{872\times436}$ and the target vector $Y_{te} \in \mathbb{R}^{872}$ from Section 3 were used for testing the classification models. Their results on the 872 samples of the testing set are presented in Figure 3b, where we observe that the WASDMC has the best performance, with only 19 incorrect classifications, and FKNN has the second-best performance, with 177 incorrect classifications. The WASDMC-PAF performs the worst, with 852 incorrect classifications, the FTree performs the second worst, with 821 incorrect classifications, the NNN performs the third worst, with 622 incorrect classifications, and the EBT performs the fourth worst, with 458 incorrect classifications. Notice that the WASDMC-PAF, FTree, NNN and EBT have more incorrect classifications than correct ones.



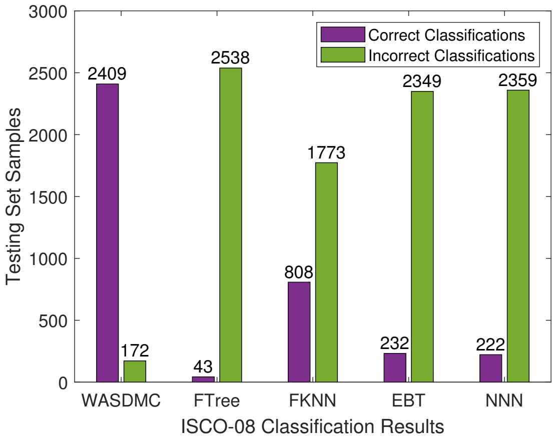
(**a**) Classification of ILO-ISCO's training set.



(**b**) Classification of ILO-ISCO's testing set.



(**c**) Classification of OS-ISCO's training set.



(**d**) Classification of OS-ISCO's testing set.

Figure 3. The classification of the ILO-ISCO and OS-ISCO training and testing sets by NNs.

In the case of OS-ISCO, the training set's input matrix $X_{tr} \in \mathbb{R}^{149\times149}$ and the target vector $Y_{tr} \in \mathbb{R}^{149}$ from Section 3 were used for training the classification models. Their results on the training set are presented in Figure 3c, where we observe that the WASDMC, FKNN, EBT and NNN all have perfect classification scores. That is, these models classified correctly all the 149 samples of the training set. The WASDMC-PAF performs the worst and the FTree performs the second worst, while both models feature more incorrect classifications than correct ones. Furthermore, the testing set's input matrix $X_{te} \in \mathbb{R}^{2581 \times 149}$ and the target vector $Y_{te} \in \mathbb{R}^{2581}$ from Section 3 were used for testing the classification models. Their results on the 2581 samples of the testing set are presented in Figure 3d, where we observe that the WASDMC has the best performance, with only 172 incorrect classifications, while the rest of the models have more incorrect classifications than correct ones. Particularly, the FKNN has the second-best performance, with 1773 incorrect classifications, the EBT has the third-best performance, with 2349 incorrect classifications, the NNN has the fourth-best performance, with 2359 incorrect classifications, the FTree has the second-worst performance, with 2538 incorrect classifications, and the WASDMC-PAF has the worst performance, with 2566 incorrect classifications.

To statistically assess the classification models' performance, their outputs on the testing sets of ILO-ISCO and OS-ISCO are subjected to a number of performance metrics in Tables 1 and 2, respectively. These performance metrics are the MAE, accuracy, sensitivity, specificity, precision, false positive rate (FPR), F-score, Matthews correlation coefficient (MCC), and Cohen's κ. In particular, the accuracy is the proportion of samples that were correctly classified to the total number of samples, while the specificity is the proportion of correctly classified negative samples, and the sensitivity (or recall) is the proportion of correctly classified positive samples. As a result, sensitivity and specificity can be thought of as two different types of accuracy, with the first being relevant to actual positive samples and the second to actual negative samples. Moreover, precision shows the proportion of correctly classified positive samples to all positively predicted samples, sensitivity shows the correlation between observed and predicted classifications, MCC shows the proportion of correctly classified negative samples, FPR shows the proportion of incorrectly classified negative samples, and Cohen's κ is used to measure inter-rater reliability for categorical items. More information and greater analysis about these metrics are provided in [48,49]. Last, the training accuracy (TA) as well as the average time consumption (TC) and the memory usage (MU) for training the NN models are also included in these tables.

Table 1. The NN models' statistics on the testing set of ILO-ISCO.

	Neural Network Models					
ILO-ISCO	**WASDMC**	**WASDMC-PAF**	**FTree**	**FKNN**	**EBT**	**NNN**
TC	0.1882 s	1.0407 s	4.0399 sec	1.0738 s	14.8563 s	42.8813 s
MU	4.6925 MB	4.6925 MB	7.5769 MB	6.2226 MB	92.3990 MB	6.3485 MB
TA	1	0.0229	0.2317	1	1	0.9794
MAE	1.2580	86.1238	162.0447	10.4610	31.7133	81.8543
Accuracy	0.9782	0.0229	0.0584	0.7970	0.4782	0.2637
Sensitivity	0.9858	0.4417	0.9160	0.9285	0.7833	0.6370
Specificity	0.9999	0.9977	0.9978	0.9995	0.9988	0.9983
Precision	0.9782	0.0229	0.0584	0.7970	0.4782	0.2637
FPR	5×10^{-5}	0.0022	0.0021	4×10^{-4}	0.0011	0.0016
F-score	0.9756	0.5678	0.0727	0.7892	0.4900	0.4103
MCC	0.9821	0.4462	0.9165	0.8942	0.7719	0.6568
Cohen's κ	0.9782	0.0207	0.0563	0.7966	0.4770	0.2621

The NN models' statistics on the testing set of ILO-ISCO are presented in Table 1. Therein, we observe that the WASDMC performs better than the WASDMC-PAF, FTree, FKNN, EBT and NNN models. Particularly, the WASDMC is about 6 times faster than the WASDMC-PAF, the second-fastest model, and about 230 times faster than the NNN, the slowest model. The MUs of WASDMC and WASDMC-PAF are identical, and they are each about 1.3 times lower than the MU of FKNN, which is the third-lowest MU, and about 20 times lower than the MU of EBT, which is the highest MU. The TAs of WASDMC, FKNN and EBT are identical, and they are each about 43 times higher than the TA of WASDMC-PAF, which is the lowest TA. The MAE of WASDMC is about 8 times lower than the MAE of FKNN, which is the second-lowest MAE, and about 130 times lower than the MAE of FTree, which is the highest MAE. The accuracy of WASDMC is about 1.2 times higher than the accuracy of FKNN, which is the second-highest accuracy, and about 42 times higher than the accuracy of WASDMC-PAF, which is the lowest accuracy. Furthermore, the WASDMC has the highest sensitivity, specificity, precision, F-score and Cohen's κ values, while the FKNN has the second-highest values in these metrics. The WASDMC-PAF has the lowest sensitivity, specificity, precision and Cohen's κ values, and the FTree has the lowest F-score value. The FPR of WASDMC is about 8 times lower than the FPR of FKNN, which is the second-lowest FPR, and about 44 times lower than the FPR of WASDMC-PAF, which is the highest FPR. Last, the WASDMC has the highest MCC, while the FKNN has the second-highest, and the WASDMC-PAF has the lowest.

Table 2. The NN models' statistics on the testing set of OS-ISCO.

	Neural Network Models					
OS-ISCO	**WASDMC**	**WASDMC-PAF**	**FTree**	**FKNN**	**EBT**	**NNN**
TC	0.0138 s	0.0846 s	0.5327 s	0.2275 s	2.6160 s	2.040 s
MU	3.3539 MB	3.3539 MB	3.7661 MB	3.5534 MB	14.8418 MB	3.6103 MB
TA	1	0.0403	0.3154	1	1	0.9933
MAE	2.0987	68.7791	56.5947	19.5877	36.8442	38.7617
Accuracy	0.9333	0.0058	0.0166	0.3130	0.0840	0.0867
Sensitivity	0.9170	0.6779	0.7612	0.5792	0.3425	0.2863
Specificity	0.9995	0.9933	0.9934	0.9953	0.9938	0.9938
Precision	0.9633	0.0402	0.1556	0.7699	0.4661	0.3967
FPR	4×10^{-4}	0.0067	0.0066	0.0046	0.0061	0.0061
F-score	0.9222	0.3490	0.1143	0.5045	0.3281	0.4134
MCC	0.9293	0.6798	0.7796	0.5751	0.3347	0.3095
Cohen's κ	0.9278	0.0023	0.0145	0.3094	0.0818	0.0810

The NN models' statistics on the testing set of OS-ISCO are presented in Table 2. Therein, we observe that the WASDMC performs better than the WASDMC-PAF, FTree, FKNN, EBT and NNN models. In particular, the WASDMC is about 6 times faster than the WASDMC-PAF, the second-fastest model, and about 190 times faster than the EBT, the slowest model. The MUs of WASDMC and WASDMC-PAF are identical, and they are each slightly lower than the MU of FKNN, which is the third-lowest MU, and about 4.4 times lower than the MU of EBT, which is the highest MU. The TAs of WASDMC, FKNN and EBT are identical, and they are each about 25 times higher than the TA of WASDMC-PAF, which is the lowest TA. The MAE of WASDMC is about 9 times lower than the MAE of FKNN, which is the second-lowest MAE, and about 32 times lower than the MAE of WASDMC-PAF, which is the highest MAE. The accuracy of WASDMC is about 3 times higher than the accuracy of FKNN, which is the second-highest accuracy, and about 160 times higher than the accuracy of WASDMC-PAF, which is the lowest accuracy. Furthermore, the WASDMC has the highest specificity, precision, F-score and Cohen's κ values, while the FKNN has the second-highest values in these metrics. The WASDMC-PAF has the lowest specificity, precision and Cohen's κ values, and the FTree has the lowest F-score value. The WASDMC has the highest sensitivity, while the FTree has the second-highest sensitivity, and the NNN has the lowest sensitivity. The FPR of WASDMC is about 11 times lower than the FPR of FKNN, which is the second-lowest FPR, and about 16 times lower than the FPR of WASDMC-PAF, which is the highest FPR. Last, the WASDMC has the highest MCC, while the FKNN has the second highest, and the WASDMC-PAF has the lowest.

In general, the WASDMC model functioned brilliantly in handling multiclass classification tasks, while its performance in contrast to WASDMC-PAF, FTree, FKNN, EBT and NNN models is superior. This is in keeping with the information shown in Tabs. in Tables 1 and 2. Furthermore, FKNN, which employs the k-nearest neighbors algorithm, has the second best overall performance, whereas the NNN, a NN classifier with one fully connected layer, has the second-worst overall performance. When FTree and EBT are compared, EBT performs marginally better than FTree because it integrates multiple decision trees rather than just one to generate higher predictive performance. It is important to note that despite having the weakest performance, the WASDMC-PAF had a higher F-score than FTree in both datasets and a lower MAE than FTree in the ILO-ISCO dataset. However, the F-scores do not account for true negatives, giving precision and FPR equal weight. As a consequence, measures such as the MCC and Cohen's κ are preferred to evaluate performance [48,49]. Therefore, it is evident that the FTree outperforms the WASDMC-PAF based on MCC and Cohen's κ values. When WASDMC and WASDMC-PAF are compared, their performance results differ significantly. It is important to acknowledge that the proposed power maxout activation function in (6) is superior to the PAF in (15). Aside from sharing the same MU and having the lowest TCs due to using the same WASD algorithm, the WASDMC had the best results across all measures, whereas the WASDMC-PAF had the worst results when compared to the FTree, FKNN, EBT, and NNN models. It is worth mentioning that the WDD process imposes a constraint on the WASDMC algorithm that the input data must only be real numbers. Additionally, the WASDMC algorithm's disadvantage is that it is only suitable for multiclass classification tasks. However, the WASDMC algorithm's advantages include low computational complexity, minimal use of hidden-layer neurons, low training time consumption, and high accuracy.

5. Conclusions

This paper presented the WASDMC-based NN model, which is trained by a novel WASDMC algorithm and can handle multiclass classification tasks. Applications on two publicly accessible ISCO datasets revealed that the WASDMC model handles multiclass classification tasks brilliantly, outperforming some of the best classification models available in the MATLAB classification learner app. Particularly, in both datasets, the WASDMC model had the lowest MU and was about 6 times faster than the second-fastest model and more than 190 times faster than the slowest model. In the remaining statistical metrics, the WASDMC model outperformed the other models by a factor of 1.2 to 160. Additionally, when used with the WASDMC to address classification tasks, the proposed power maxout activation function demonstrated to be much superior to the PAF.

There are certain limitations on this work as well as some recommendations.

1. The availability of the data, which constrained the extent of our analysis, was a study limitation. That is, more testing could be done, more results would be obtained, and the conclusions would be strengthened even further if more ISCO classification datasets were made accessible to the public.
2. Another option to assess and enhance the effectiveness and precision of a WASD algorithm is to incorporate a cluster validity method, comparable to the one used in conventional k-means clustering. With this method, a WASD algorithm can avoid identifying patterns in seemingly random data, which will enhance the performance of the WASD-based NN models.
3. Applications in numerous scientific fields could show how the WASDMC-based NN model is more reliable than conventional NN models.

Author Contributions: All authors (D.L., S.D.M., P.Z. and G.T.) contributed equally. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Ministry of Science and Higher Education of the Russian Federation (Grant No. 075-15-2022-1121).

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.

References

1. Simos, T.E.; Katsikis, V.N.; Mourtas, S.D. A multi-input with multi-function activated weights and structure determination neuronet for classification problems and applications in firm fraud and loan approval. *Appl. Soft Comput.* **2022**, *127*, 109351. [CrossRef]
2. Raj, R.J.S.; Shobana, S.J.; Pustokhina, I.V.; Pustokhin, D.A.; Gupta, D.; Shankar, K. Optimal feature selection-based medical image classification using deep learning model in internet of medical things. *IEEE Access* **2020**, *8*, 58006–58017. [CrossRef]
3. Rahim, T.; Usman, M.A.; Shin, S.Y. A survey on contemporary computer-aided tumor, polyp, and ulcer detection methods in wireless capsule endoscopy imaging. *Comput. Med, Imaging Graph.* **2020**, *85*, 101767. [CrossRef] [PubMed]
4. Bigdeli, B.; Pahlavani, P.; Amirkolaee, H.A. An ensemble deep learning method as data fusion system for remote sensing multisensor classification. *Appl. Soft Comput.* **2021**, *110*, 107563. [CrossRef]
5. Rácz, A.; Bajusz, D.; Héberger, K. Effect of dataset size and train/test split ratios in QSAR/QSPR multiclass classification. *Molecules* **2021**, *26*, 1111. [CrossRef]
6. Venkatesan, R.; Er, M.J. A novel progressive learning technique for multi-class classification. *Neurocomputing* **2016**, *207*, 310–321. [CrossRef]
7. Mourtas, S.D.; Katsikis, V.N. Exploiting the Black-Litterman framework through error-correction neural networks. *Neurocomputing* **2022**, *498*, 43–58. [CrossRef]
8. Mourtas, S.D. A weights direct determination neuronet for time-series with applications in the industrial indices of the federal reserve bank of St. Louis. *J. Forecast.* **2022**, *14*, 1512–1524. [CrossRef]
9. Mourtas, S.D.; Katsikis, V.N.; Drakonakis, E.; Kotsios, S. Stabilization of stochastic exchange rate dynamics under central bank intervention using neuronets. *Int. J. Inf. Technol. Decis.* **2022**, 1–29. [CrossRef]
10. Simos, T.E.; Katsikis, V.N.; Mourtas, S.D. Multi-input bio-inspired weights and structure determination neuronet with applications in European Central Bank publications. *Math. Comput. Simul.* **2022**, *193*, 451–465. [CrossRef]
11. Simos, T.E.; Katsikis, V.N.; Mourtas, S.D. A fuzzy WASD neuronet with application in breast cancer prediction. *Neural Comput. Appl.* **2021**, *34*, 3019–3031. [CrossRef]
12. Daliri, M.R. A hybrid automatic system for the diagnosis of lung cancer based on genetic algorithm and fuzzy extreme learning machines. *J. Med. Syst.* **2012**, *36*, 1001–1005. [CrossRef]
13. Chen, L.; Huang, Z.; Li, Y.; Zeng, N.; Liu, M.; Peng, A.; Jin, L. Weight and structure determination neural network aided with double pseudoinversion for diagnosis of flat foot. *IEEE Access* **2019**, *7*, 33001–33008. [CrossRef]
14. Gayathri, S.; Krishna, A.K.; Gopi, V.P.; Palanisamy, P. Automated binary and multiclass classification of diabetic retinopathy using Haralick and multiresolution features. *IEEE Access* **2020**, *8*, 57497–57504. [CrossRef]
15. Mourtas, S.; Katsikis, V.; Kasimis, C. Feedback control systems stabilization using a bio-inspired neural network. *Eai Endorsed Trans. Robot.* **2022**, *1*, 1–13. [CrossRef]
16. Huang, H.; Fu, D.; Xiao, X.; Ning, Y.; Wang, H.; Jin, L.; Liao, S. Modified Newton integration neural algorithm for dynamic complex-valued matrix pseudoinversion applied to mobile object localization. *IEEE Trans. Ind. Infor.* **2020**, *17*, 2432–2442. [CrossRef]
17. Premalatha, N.; Arasu, A.V. Prediction of solar radiation for solar systems by using ANN models with different back propagation algorithms. *J. Appl. Res. Technol.* **2016**, *14*, 206–214. [CrossRef]
18. Huang, C.; Jia, X.; Zhang, Z. A modified back propagation artificial neural network model based on genetic algorithm to predict the flow behavior of 5754 aluminum alloy. *Materials* **2018**, *11*, 855. [CrossRef]
19. Groes, F.; Kircher, P.; Manovskii, I. The U-shapes of occupational mobility. *Rev. Econ. Stud.* **2015**, *82*, 659–692. [CrossRef]
20. Khalis, M.; Charbotel, B.; Fort, E.; Chajes, V.; Charaka, H.; Rhazi, K.E. Occupation and female breast cancer: A case-control study in Morocco. *Rev. Épidémiol. Santé Publique* **2018**, *66*, S302. [CrossRef]
21. Heinesen, E.; Imai, S.; Maruyama, S. Employment, job skills and occupational mobility of cancer survivors. *J. Health Econ.* **2018**, *58*, 151–175. [CrossRef] [PubMed]
22. Generalao, I.N. Measuring the telework potential of jobs: Evidence from the international standard classification of occupations. *Philipp. Rev. Econ.* **2021**, *58*, 92–127. [CrossRef]
23. Choi, S.B.; Yoon, J.H.; Lee, W. The modified international standard classification of occupations defined by the clustering of occupational characteristics in the Korean working conditions survey. *Ind. Health* **2020**, *58*, 132–141. [CrossRef] [PubMed]
24. Züll, C. The coding of occupations. In *GESIS Survey Guidelines*; GESIS–Leibniz Institute for the Social Sciences: Mannheim, Germany, 2016. [CrossRef]
25. Marc, D.T.; Dua, P.; Fenton, S.H.; Lalani, K.; Butler-Henderson, K. *The Health Information Workforce*; Health Informatics; Chapter Occupational Classifications in the Health Information Disciplines; Springer: Cham, Switzerland, 2021; pp. 71–78. [CrossRef]

26. Uter, W. *Kanerva's Occupational Dermatology*; Chapter Classification of Occupations; Springer: Berlin/Heidelberg, Germany, 2012. [CrossRef]
27. Aalto-Korte, K.; Koskela, K.; Pesonen, M. 12-year data on skin diseases in the Finnish register of occupational diseases II: Risk occupations with special reference to allergic contact dermatitis. *Contact Dermat.* **2020**, *82*, 343–349. [CrossRef]
28. Trzaskawka, P. Names of music professions–a linguistic case study. *Analele Univ. Ovidius Constanţa. Ser. Filol.* **2020**, *31*, 90–105.
29. Kuodytė, V.; Petkevičius, L. Education-to-skill mapping using hierarchical classification and transformer neural network. *Appl. Sci.* **2021**, *11*, 5868. [CrossRef]
30. Lovaglio, P.G.; Cesarini, M.; Mercorio, F.; Mezzanzanica, M. Skills in demand for ICT and statistical occupations: Evidence from web-based job vacancies. *Stat. Anal. Data Min.* **2018**, *11*, 78–91. [CrossRef]
31. Liu, J.; Ng, Y.C.; Gui, Z.; Singhal, T.; Blessing, L.T.M.; Wood, K.L.; Lim, K.H. Title2Vec: A contextual job title embedding for occupational named entity recognition and other applications. *J. Big Data* **2022**, *9*, 99. [CrossRef]
32. Matbouli, Y.T.; Alghamdi, S.M. Statistical machine learning regression models for salary prediction featuring economy wide activities and occupations. *Information* **2022**, *13*, 495. [CrossRef]
33. Malandri, L.; Mercorio, F.; Mezzanzanica, M.; Nobani, N. MEET-LM: A method for embeddings evaluation for taxonomic data in the labour market. *Comput. Ind.* **2021**, *124*, 103341. [CrossRef]
34. Zhang, Y.; Chen, D.; Ye, C. *Deep Neural Networks: WASD Neuronet Models, Algorithms, and Applications*; CRC Press: Boca Raton, FL, USA, 2019.
35. Zhang, Y.; Yu, X.; Xiao, L.; Li, W.; Fan, Z.; Zhang, W. Weights and structure determination of articial neuronets. In *Self-Organization: Theories and Methods*; Nova Science: New York, NY, USA, 2013.
36. Zhang, Y.; Yin, Y.; Guo, D.; Yu, X.; Xiao, L. Cross-validation based weights and structure determination of Chebyshev-polynomial neural networks for pattern classification. *Pattern Recognit.* **2014**, *47*, 3414–3428. [CrossRef]
37. Zhang, Y.; Wang, R.; Lao, W.; Deng, J. Signum-function-activated WASD neuronet and its XOR application. *Acta Sci. Nat. Univ. Sunyatseni* **2014**, *53*, 1–7.
38. Zhang, Y.; Guo, D.; Luo, Z.; Zhai, K.; Tan, H. CP-activated WASD neuronet approach to Asian population prediction with abundant experimental verification. *Neurocomputing* **2016**, *198*, 48–57. [CrossRef]
39. Zeng, T.; Zhang, Y.; Li, Z.; Qiu, B.; Ye, C. Predictions of USA presidential parties from 2021 to 2037 using historical data through square wave-activated WASD neural network. *IEEE Access* **2020**, *8*, 56630–56640. [CrossRef]
40. Zhang, Y.; Lao, W.; Jin, L.; Chen, T.; Liu, J. Growing-type WASD for power-activation neuronet to model and forecast monthly time series. In Proceedings of the 2013 10th IEEE International Conference on Control and Automation (ICCA), Hangzhou, China, 12–14 June 2013; pp. 1312–1317. [CrossRef]
41. Goodfellow, I.J.; Warde-Farley, D.; Mirza, M.; Courville, A.; Bengio, Y. Maxout networks. *arXiv* **2013**, arXiv:1302.4389.
42. ISCO-08 Structure and Definitions. Available online: https://www.ilo.org/ilostat-files/ISCO/newdocs-08-2021/ISCO-08/ISCO-08\EN\Structure\and\definitions.xlsx (accessed on 18 December 2022).
43. Varelas, G.; Lagios, D.; Ntouroukis, S.; Zervas, P.; Parsons, K.; Tzimas, G. Employing natural language processing techniques for online job vacancies classification. In *Artificial Intelligence Applications and Innovations. AIAI 2022 IFIP WG 12.5 International Workshops. AIAI 2022*; IFIP Advances in Information and Communication Technology; Maglogiannis, I., Iliadis, L., Macintyre, J., Cortez, P., Eds.; Springer: Cham, Switzerland, 2022; Volume 652, pp. 333–344. [CrossRef]
44. Davis, M.; Iancu, L. Unicode text segmentation. *Unicode Stand. Annex.* **2018**, *29*, 65.
45. Gupta, A.K. *Numerical Methods Using MATLAB*; MATLAB Solutions Series, Berkley; Springer Press: New York, NY, USA, 2014.
46. Kim, P. *MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence*; Apress: Berkeley, CA, USA, 2017. [CrossRef]
47. WASD-Based Neural Network for Multiclass Classification. Available online: https://github.com/SDMourtas/WASDMC (accessed on 21 December 2022).
48. Tharwat, A. Classification assessment methods. *Appl. Comput. Inf.* **2020**, *17*, 168–192. [CrossRef]
49. McHugh, M.L. Interrater reliability: the kappa statistic. *Biochem. Med.* **2012**, *22*, 276–282. [CrossRef]