T.C.

ALTINBAŞ UNIVERSITY

Electrical and Computer Engineering

Handling Categorical Data in Artificial Neural Networks

Andam Omar Anwar ANWAR

Master Thesis

Supervisor

Asst. Prof. Dr. Sefer KURNAZ

Istanbul, 2019

Handling Categorical Data in Artificial Neural Networks

by

Andam Omar Anwar ANWAR

Electrical and Computer Engineering

Submitted to the Graduate School of Science and Engineering

in partial fulfillment of the requirements for the degree of

Master of Science

ALTINBAŞ UNIVERSITY

2019

This is to certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Sefer KURNAZ

Supervisor

Examining Committee Members

Academic Title Name SURNAME Faculty, University _____

Academic Title Name SURNAME Faculty, University _____

Academic Title Name SURNAME Faculty, University _____

I certify that this thesis satisfies all the requirements as a thesis for the degree of .....

Academic Title Name SURNAME

Head of Department

Academic Title Name SURNAME

Director

Approval Date of Graduate School of Science and Engineering: ____/____/____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Andam Omar Anwar ANWAR

DEDICATION

This thesis is dedicated to my beloved parents for their love, endless support and encouragement.

Along with my wife "Khanda" who raised me, loved me and helped me in all my thesis station, and my lovely little son Andaz.

Last but not least I am dedicating this to all the learners and hard working teachers around the world.

ACKNOWLEDGMENTS

First and foremost, i have to thank my parents for their love and support throughout my life, Thank you both for giving me strength to reach for the stars and chase my dreams.

My Wife , brothers, sisters, deserve my wholehearted thanks as well !

I would like to sincerely than" my thesis adviser, Ass.Prof.Dr.Sefer Kurnaz, for his guidance and support throughout this study and specially for his confidence in me. To all my friends, thank you for your understanding and encouragement in many, many moments of crisis (our friendship makes my life a wonderful experience! cannot list all the names here, but you are always on my mind.

Thank you, Lord, for always being there for me.

This thesis is only a beginning of my journey.

ABSTRACT

Handling Categorical Data in Artificial Neural Networks

Andam Omar Anwar ANWAR,

M.Sc., Electrical and Computer Engineering, Altınbaş University

Supervisor: Asst. Prof. Dr. Sefer KURNAZ

Date: February, 2019

Pages: 49

Machine Learning (ML) techniques are being widely used to extract knowledge from real-life datasets to interact with the environments that these data are collected from. One of the widely used machine learning techniques that has shown outstanding performance, compared to other ML technique, is the Artificial Neural Networks (ANN). Similar to other ML methods, ANNs are trained using the instances in the dataset, so that, the task required from the ANN can be achieved. Real-life datasets consist of different types of values, mainly quantitative and qualitative. Handling the quantitative data is an easy task in ANN, as the inputs of these networks is numerical. However, as qualitative data may contain nominal values do not have meaningful order, encoding such values into numerical format can cause the loss of important knowledge. Thus, the selection of appropriate numerical values can significantly improve the neural networks' performance, and vice versa.

In this study, a novel method is proposed to allow ANNs produce the suitable values for the qualitative values in the dataset. As the ANN uses backpropagation to update the weights that connect the neurons in the network to each other, the proposed method produces a vector for each qualitative attribute, using One-Hot-Encoding (OHE). During training, the ANN updates the weight corresponding to each of the nominal values to reduce the error between

the predicted values and the actual values required from the ANN. Each vector is connected to a single neuron in the next layer, so that, by using the OHE, the value that appears on that neuron is equal to the weight corresponding to the position where the value is set to one in the vector.

To evaluate the ANN's performance using the proposed encoding method, different real-life datasets are used to train and evaluate the performance of the networks. Per each dataset, the predictions accuracy and loss versus the training epochs are monitored for standard ANNs with label-encoded inputs, and those using the proposed encoding method. The evaluation shows that the proposed method has improved the performance of the ANN significantly, illustrated by the higher learning rate, i.e. faster rise in accuracy and reduction in loss, as well as better predictions when the training is complete. Thus, the proposed encoding method can improve the performance of the same ANN, or produce similar performance using less-complex networks.

Keywords: Artificial Neural Network; Label Encoding; One Hot Encoding; Backpropagation.

OZET

TABLE OF CONTENTS

Page

LIST OF ABBREVIATIONS

ML : Machine Learning

ANN : Artificial Neural Network

MSE : Mean Squared Error

LE : Label-Encoding

OHE One-Hot-Encoding

CNN : Convolutional Neural Network

RNN : Recurrent Neural Network

MSE : Mean Squared Error

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

1. INTRODUCTION

Techniques that enable computers of extracting knowledge from an external environment, in order to use this knowledge to interact with that environment, are known as Machine Learning (ML) techniques [1,2]. Instead of providing static rules that define the way the computers interact with the environment, machine learning techniques allow computers to dynamically recognize these rules and make the appropriate decisions, based on the inputs collected from the environment and the rules extracted from it. The knowledge extraction phase, in which the machine learning techniques extract the knowledge from the collected sample inputs, is known as the training phase. This training phase require a set of samples collected from the environment, so that, the ML techniques detect the patterns and relations among these samples to create the model that is used to interact with the environment [3].

As the machine learning techniques require real-life data, in order to extract real knowledge that enables it to interact with the corresponding environment, the data used with the ML can be of different types. Depending of the environment that the data is collected from, the attribute's values can be quantitative or qualitive. Quantitative data can be continuous, in which an infinite number of possible values can exist in any certain range, or discrete, in which the number of values in a certain range is finite. For example, the weight of an individual is continuous, while the number of individuals in a family is discrete. Qualitive data, also known as categorical data, can be nominal, binominal or ordinal. Data are considered ordinal when each category value represents a specific range in a wider range, so that, the categories in the data can have a meaningful order, such as the baby, kid, teen and adult categories in age data [4].

The data are considered nominal when the category values do not have a specific order, such as colors. Unlike ordinal data, nominal data cannot have meaningful comparisons, where a certain color cannot be greater or smaller than another in the previous example. Moreover, when the nominal data have only two possible values, such as gender, these data are known as binomial, where the values in that data also have no possible order. However, the values of such data can still have significant knowledge to the machine learning techniques, where the existence of one categorical value can increase the possibilities of a certain output, while another value may have the opposite effect, or no significant effect at all [5, 6].

Artificial Neural Networks (ANN) has attracted significant attention in the recent years, according to their remarkable results in different machine learning applications. Similar to the humans' brains, which the ANNs are inspired from, the basic component of an ANN is the neuron, shown in Figure 1.1. The summation of the weighted inputs of a neuron is passed through an activation function, to calculate the output of the neuron. The aim of using the activation function is to provide nonlinearity to the output of the neuron, so that, more accurate boundaries can be produced by that neuron to define a more appropriate output. However, as the activation function's input is the summation of the weighted inputs, the input values still have significant effect over the output, regardless of the activation function. Moreover, to calculate the weighted summation of the inputs, it is essential that all input values are in numerical format. Thus, any alphanumerical values must be converted to numerical before being used with the ANNs [7-9].

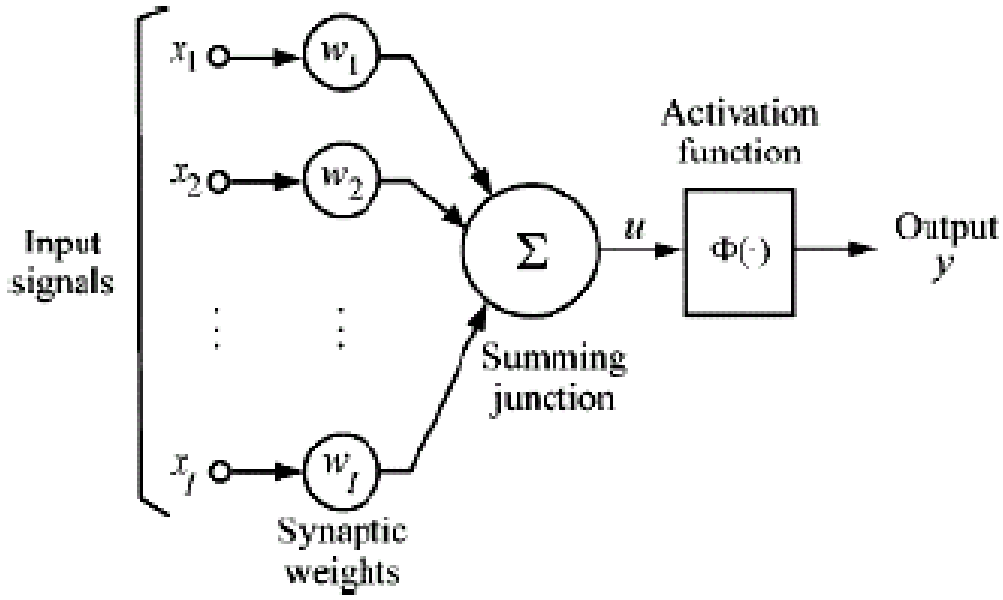

Figure 1.1: Illustration of the computations of a neuron in an artificial neural network [9].

Although there are multiple activation functions that can be used in artificial neurons, the output of these function is proportional to the value of the weighted summation. The value of the weighted summation is also proportional to an input value, directly when the weight value is positive and inversely when the weight value is negative, assuming the other inputs are constant [10]. These computations impose challenges in handling the uncertainty of

nominal values, when a certain value has more excitatory influence over the output of the neuron than another, while the numerical value assigned to the excitatory nominal value can have the opposite influence on the output [11].

In order to overcome these limitations, more complex ANNs are required, so that, the neurons at one level can detect the complex patterns in the previous outputs. Increasing the complexity of the ANN require more computer resources, to handle the more computations required by the network, and reduces the learning rate, where more training is required to update the network to achieve the required ML task. Moreover, despite the ability of increasing the complexity of the neural network, some features can still be difficult to detect, or may be neglected by the neural network for more important features, according to the limitations in the selected topology of the network [7, 12].

Encoding ordinal values into numerical can be an easy task, where each unique categorical value is assigned with a unique numerical number that defines its position in the range, so that, the relativity between the categories are maintained. However, encoding nominal values can be more challenging, as there is no meaningful order to the categorical values in the nominal data, while these values can have significant meaning on features and patterns detected in the data [13, 14]. Thus, it is important to encode these categorical values into more suitable numerical format, so that, the complexity of the ANN is maintained as low as possible, while the performance of the network is maximized.

1.1 PROBLEM STATEMENT

Machine learning has been widely used to interact with real-life datasets, while the use of artificial neural network has attracted more attention in the recent years. ANNs has shown significantly better performance, compared to other ML techniques, especially when used with larger datasets. However, as real-life datasets may contain nominal values, which are categorical values that have no meaningful order, handling the uncertainty of such values poses challenges in processing these data using ANNs [4]. This limitation is a result of the way inputs are processed in the neurons, where the summation of the weighted inputs is passed through an activation function before being outputted from the neuron. Thus,

encoding the nominal values outside the knowledge and data patterns, or features, recognized by the ANN can increase the complexity of the network required to achieve the ML task [14].

1.2 AIM OF THE STUDY

To improve the performance of the ANNs, when processing nominal data, while maintaining the complexity of the network to minimum, this study aims to proposed a novel method that allows neural networks of encoding the nominal data according to their needs. The proposed method allows the network to assign any numerical values, depending on the recognized patterns or features and how the nominal value participate in them, so that, maximum knowledge is extracted from the dataset. This approach is expected to improve the learning rate of the neural network, so that, knowledge is extracted faster and more accurately. As the weights of the inputs are adjusted during the training of the neural network, the proposed method distributes the nominal values over multiple weights, one weight per each value, so that, the value assigned to that weight represents the value that the input is encoded into. The performance of the neural networks is evaluated, and compared, when the proposed method is used with the existing encoding method, by monitoring the accuracy and error of these networks during training.

1.3 THESIS LAYOUT

The remaining chapters of this thesis are organized as follows:

- Chapter two reviews the encoding techniques being used to transform nominal data into numerical values and the architecture of artificial neural network, their types and applications.
- Chapter three describes the topology proposed to handle the uncertainty of nominal values with ANNs, so that, the numerical value per each categorical value is assigned by the neural network itself, instead of using predefined encoding values.
- Chapter four describes the environment used to perform the experiments used to evaluate the performance of the neural network and the improvement in the performance when the proposed method is used. This chapter also describes the dataset used for the evaluation and their structure.

- Chapter five discusses the results of the conducted experiments, in order to illustrate the improvement in the performance of the ANNs when the proposed topology is used.
- Chapter six concludes the thesis by illustrating the significance of the research and summarizes the results of the conducted experiments and the findings of the study.

2. LITERATURE REVIEW

2.1 INTRODUCTION

The neural networks' structure and the distribution of the neurons in these networks are first illustrated in this chapter. Then, the computations employed to calculate the outputs of the neurons, hence the output of the neural network, and computations required to update the weights in the neural network, to achieve the required ML task are described in details. Moreover, the data types exist in real-life datasets and the characteristics of each type are described in details, alongside with the method used to encode the nominal values into numerical format, are also reviewed in this chapter.

2.2 ARTIFICAL NEURAL NETWORKS

Inspired from humans' brains, ANNs use mathematical representations to simulate the electrical signals and the way they are processed and communicated among the biological neurons in these brains. As shown in Figure 2.1, inputs of a biological neuron are collected by the dendrites and delivered to the cell body [15]. However, before collecting these inputs, by the dendrites, the electrical impulses received by the neuron, which are the inputs of the neuron, are passed through the synapses. These synapses contain electrochemical conductors that adjust the effect of the received input on the cell body, by controlling the conductivity of the electrochemical solution [16, 17].

Based on the magnitudes of the electrical impulses delivered by the dendrites to the cell body, after being adjusted by the synapses, the cell body makes the decision, whether to output an impulse or not. This output is then carried out by the axon to another neuron or to the organ corresponding to the decision made by the neuron to execute the required task. Inputs that excite the cell body to output an impulse are known as excitatory inputs, while inputs that inhibit the impulse outputted by the cell body are known as inhibitory inputs. Moreover, regardless of the type of effect the input has on the decision made by the cell body, the magnitude of that effect is controlled by the synapses, so that, some inputs can have significantly higher effect on the decision that others. Different decision that are made by

different humans for same inputs is a result of the different connections between the neurons in the brains and the different conductivities in their synapses [18].

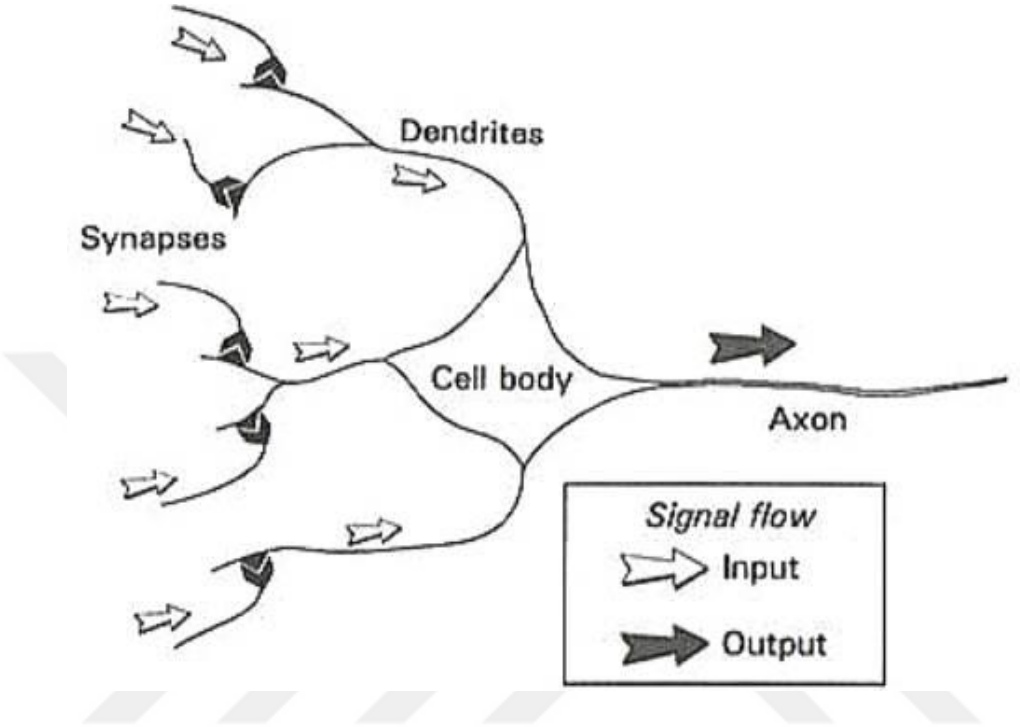

Figure 2.1: Biological neuron's structure in humans' brains [18].

Similarly, a neuron in an artificial neural network, shown in Figure 2.2, collects the inputs, which can be collected from the external environment or outputs or other neurons, in order to come up with the appropriate output. The weights (w) in the figure simulate the synapses of the biological neuron. The absolute value of the weight, i.e. the magnitude, represents the conductivity of the synapsis, where higher magnitudes indicate higher effects of the corresponding inputs, while the sign of the weight, positive or negative, represents the type of the effect, excitatory of inhibitory respectively. To increase the flexibility of the computations in the neuron, according to its needs to detect a certain pattern or feature in the dataset, a bias value is also included in the summation, which is a static value that is updated during the training phase of the ANN. Then, the weighted values of the inputs are summed up, with the bias value, in the neuron and passed through an activations function [7, 8].

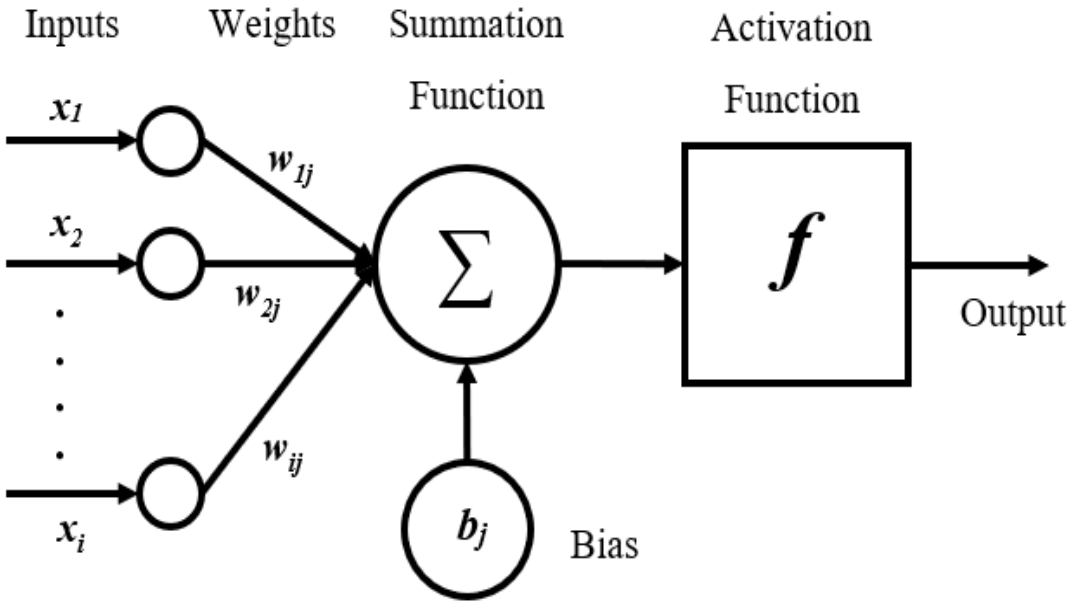

Figure 2.2: Computations of a neuron in an artificial neural network [8].

The number of neurons in ANNs can be different from one network to another, depending on the complexity of the features, or patterns, in the input data and the number of inputs provided to the neural network, as well as the number of outputs required from it. These neurons are distributed in different levels, known as layers, which can be of three types, based on their position in the neural network. The first type of layers is the input layers, which collects the inputs from the environment that the neural network is interacting with. The number of neurons in this type of layers is determined by the count of inputs collected from the external environment that the network is interacting with. The other type of layers in neural networks is the output layer, which delivers the network's outputs to the external environment. Hence, the number of required outputs determines the number of neurons in this layer [19]. Each neural network typically has only one input and one output layers, which poses the need of a third type of layers in order to allow more flexibility in determining the number of neurons in the ANN and the complexity of the features and patterns the network can recognize. Thus, a third type of layers, known as hidden layers, are used in these networks, so that, any number of neurons can be used in these layers, with any number of hidden layers in the neural network. These layers are denoted as hidden layers as they are

hidden from the external environment, unlike the input and output layers, which are connected to it [20].

2.2.1 Activation Functions

As mentioned earlier, activation functions are used in neurons to allow nonlinear outputs from the neuron, depending on the calculated summation of weighted inputs. Although each neuron in the neural network can be assigned with a different type of activations function, neurons in the same layer are normally assigned with the same activation function, as the tasks of neurons in the same layer are identical. Rectified Linear Unit (ReLU), shown in Equation 2.1, is widely used as an activation function for neurons in the hidden layers of ANNs, according to the improvement in the performance of these networks, when this activation function is used. However, the task assigned to the neural networks and the outputs required from are used to select the appropriate activation function. Most of the widely used activation functions in the neurons of the output layer are [21, 22]:

- **Sigmoid:** The output of this activation function, shown in Equation 2.2, is limited in the range (0,1). This function is normally used when the ANN is required to measure the probability of the input to belong to one of only two possible outputs. By comparing the calculated probability to a threshold value, the output can be converted from linear to binary decision. ANNs that achieve such tasks normally have a single neuron in the output layer. However, it is possible to have multiple neurons in the output layer with Sigmoid activation function, when the input may have multiple labels at once, so that, the output of each neuron represents the probability of assigning the corresponding label to the input.
- **Soft-Max:** Using this activation function, shown in Equation 2.3, the summation of the outputs of the output layer's neurons is always one, while the output value per each neuron is within the range (0,1). As the summation of the output is guaranteed to be equal to one, and each output value represents the probability of assigning the corresponding label to the input, this function is used when the input can have only one of the possible labels, so that, the label correspondent to the neuron with the highest output is assigned to the input.

- **Linear:** When the summation of the weighted inputs is outputted directly to the external environment, without passing it through a nonlinear activation function, a linear activation function is considered to be exist, where the value of the output is multiplied by the constant value of one. This activation function is used in regression problems, where the output can by any value, including negative values, based on the characteristics of the inputs.

$$ReLU(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \quad (2.1)$$

$$Sigmoid(x) = \frac{1}{1 + e^{-x}} \quad (2.2)$$

$$SoftMax(x) = \frac{e^{x_i}}{\sum_{j=0}^{k} x_j} \quad (2.3)$$

where *k* is the neurons count in the output layer and *i* is the position of the neuron the output is calculated for.

2.2.2 Forward Pass

The computations executed on the input values, by the ANN, in order to calculate the output of the network are known as the forward pass. In a fully connected network, the inputs of the neurons in the hidden and output layer are collected from the neurons in the previous layer, as shown in the sample network in Figure 2.3. An array that holds the values of the weights, between each two neurons in the sequential layers, is used to multiply the outputs from the previous layer, before calculating the summation of the weighted sums in the neuron. The use of such topology allows the combination of patterns detected by neurons in a specific layer in the following one. Moreover, to increase the number of patterns, or features, that can be detected at a certain level of complexity can be increased by increasing the number of neurons in the corresponding layer [23, 24]. The output of neuron *i*, with *J* inputs *x* from the previous layer, using the weights array *w*, is calculated using Equation 2.4.

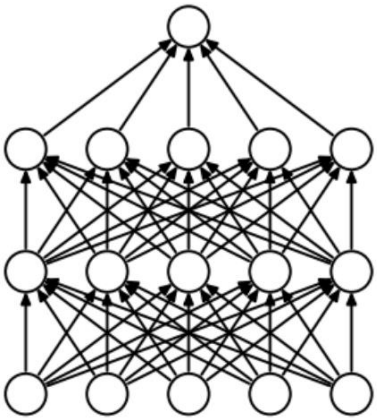

Figure 2.3: A sample fully connected artificial neural network.

$$Output_i = \sum_{j=1}^{J} x_j w_{ij} + b_i \tag{2.4}$$

where b_i is the bias value for that neuron.

2.2.3 Reverse Pass (Backpropagation)

Similar to humans' brains, where the topology of the biological neural network and the conductivities of the synapses define the decisions made by the brain, ANNs also rely of the distribution of the neurons and the weights among them to make the required decision. Two identical neural networks can be used in completely different task, in which different decisions are made, by using different weights values among their neurons. The value of a weight between two neurons defines the type of the effect, the output of the neuron in the previous layer over the neuron in the next one, as well as the significance of that effect on the output of the neuron in the later layer [25].

Backpropagation has a key-role in the popularity of neural networks, as the performance of these networks is significantly improved when this technique is used to update the value of the network's weights. In order to update the weights of the ANN, backpropagation requires three values, as shown in Equation 2.5, which are the rate of change of the network's output, with respect to the weight being updated $\frac{\partial o}{\partial w}$, the error E between the output of the network and the one actually required from it and the learning rate L [26].

$$\widehat{w} = w - \frac{\partial O}{\partial w} \times E \times L \tag{2.5}$$

Regardless of the type of error function used by the neural network, such as the cross-entropy and Mean Squared Error (MSE) functions, these functions calculate a single value that represents the difference between the output of the neural network, using the current weights values, and the values required from the network. The output of the neural network is collected by processing a batch of sample inputs, from the training dataset, using the forward pass of the neural network, while the actual outputs are collected directly from the training dataset, or by processing the inputs using predefined functions. The calculated error value is then used by in the backpropagation. However, as large error values can produce large delta values, for weights updates, a learning rate is used to control the delta values in lower ranges. This control of the delta values ensures the avoidance of exploding weight values, so that, the weights values that produce the minimum error can be discovered [27].

By calculating the rate of change of the output error, with respect to the weight values, three possible values can be produces [20, 28], which are:

- A positive value, which indicate that increasing that weight value increases the error. Thus, the weight value must be decreased by the calculate delta value, in order to decrease the difference between the outputs of the neural network and the required ones.
- A negative value that indicates that the error is decreased by increasing the value of that weight. Thus, the current weight value must be increased by the calculated delta value, in order to reduce the error value and produce more accurate outputs.
- A zero value, which indicates that no change is required to the current value of the weight.

According to these possible values and by using the formula shown in Equation 2.5, the values of the weights in the neural network can be updated in order to reduce the difference between the predictions of the neural network and the actual output that is required to achieve the task of the ANN. However, according to the need of learning rate, to reduce the delta value used to update the weight values, the optimal performance of the neural network,

produced by minimizing the error through updating the weights, calculating the optimal weights values require multiple iterations, i.e. epochs [7].

2.3 EMPLOYMENT OF ARITIFICIAL NEURAL NETWORK IN ML

Depending on the information used in the training of the ML techniques, two main categories of ML exist, unsupervised and supervised [3]. Training an unsupervised ML technique require only the samples collected from the environment, where the aim of such techniques is to find similarities and relation among these inputs [29]. In contrast, training a supervised machine learning technique requires additional information about each of the collected sample, which is added by a human expert. The aim of supervised ML techniques is to find the features and patterns that join the characteristic of each data instance and the information added to these sample inputs [30].

Artificial neural networks have been used in both types of machine learning, in where these networks have been able to achieve remarkable performances, compared to other techniques used for the same tasks. Moreover, depending on the shape of the inputs of the ANN and the nature of the environment these samples are collected from, different types of ANNs are being used. Some of the most popular existing types of artificial neural networks are:

- **Feed-Forward ANN:** The output of each neuron at a certain layer in this type of ANNs is weighted and forwarded as the input to all the neurons in the following layer. The neurons in the input layer are distributed in a one-dimensional formation, so that, a neuron is assigned for each input. Thus, this type of neural networks is used when each data instance is characterized using a one-dimensional vector [31, 32].
- **Convolutional Neural Networks (CNN):** The inputs of each neuron in a convolutional layer are collected using a two-dimensional window, known as filter, which is convoluted through the entire input. Per each convolution, the output of the neuron is placed in a two-dimensional array corresponding to the position of the filter. This results in a three-dimensional array, where the size of the third dimension equals to the number of neurons in the layer. Such approach allows the detection of local two-dimensional features in the input, which makes this type of ANN have better performance with two- and three-dimensional inputs, such as images [33, 34].

- **Recurrent Neural Networks (RNN):** This type of neural networks also processes two-dimensional inputs but the inputs are fed in one-dimensional vectors, one at a time. However, as the output of a neuron is feedback to the same neuron, the output at a certain time instance, i.e. an input vector at a certain position in the input, is affected by the computations from the previous time instance. Thus, this type of ANNs is widely used to process timeseries, as the inputs are time-relative to each other [35, 36].

One the important supervised applications that employs ANNs, regardless of its type, is classification, is classification, where the inputs are categorized based on their characteristics into categories. The characteristics of inputs in each category are extracted by the neural network during the training, so that, the characteristics of future inputs are investigated to predict their categories. A neuron in the output layer is assigned per each possible category in the dataset, except when two categories exist in the network, where a single-neuron output layer is used to determine which of the labels is assigned to the input, based on the output value of the neuron. The Soft-Max function is used for activation in the output-layer's neurons when each input can be assigned with a single category, while the Sigmoid function is used when an input can be assigned with more than one category [37, 38].

In unsupervised ML, artificial neural networks are widely employed for autoencoding, where the dimensionality of an input is reduced using these networks [39]. Although a single ANN is used for auto encoding, splitting the layers into encoding and decoding layer, sharing the same encoded values, as shown in Figure 2.4, can simplifies the operation of such networks. Autoencoders are trained to output values similar to the inputs of the network. After being trained, the encoder can be used to output the code calculated for the input, which can be used to recalculate the input using the decoding part of the network. Thus, the same value, or quite similar ones, of the input can be retrieved using a smaller descriptor, which is the code value that connects the encoder with the decoder. As the error is calculated by measuring the similarity between the input and output of the network, no extra information is required with the dataset for training [39, 40].

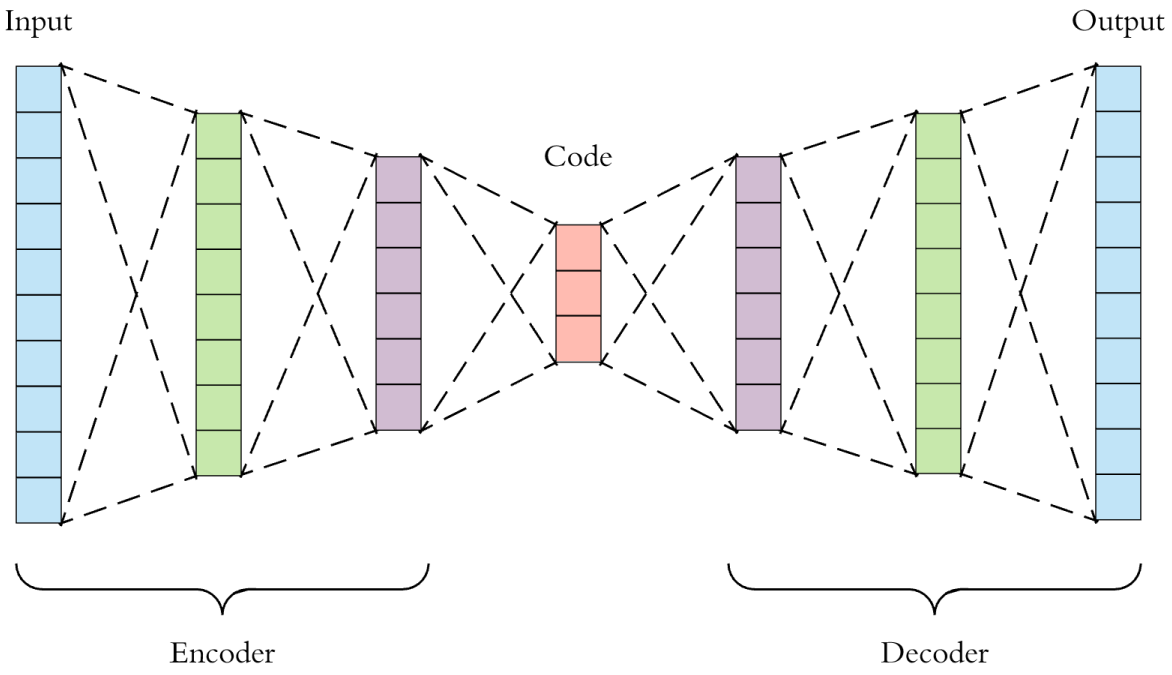

Figure 2.4: Illustration of an autoencoding artificial neural network [40].

2.4 DATA TYPES

A dataset consists of a data instances, each instance is defined by a set of attributes' values. Data collected from real-life environments can have different types in their attributes, mainly, quantitative and qualitative [41]. Quantitative attributes use numerical values to characterize the instance. As shown in Figure 2.5, a continuous quantitative value can have any, of infinite number of possible, values in a certain range, while a discrete value can have only one of a finite number of possible values. However, in both cases, continuous or discrete, the similarity between two data instances, regarding the characteristic that the attribute represents, can be measured by measuring the difference between their quantitative values, so that, data instances with more similar values are more similar in that characteristic when the difference between their values is low [42].

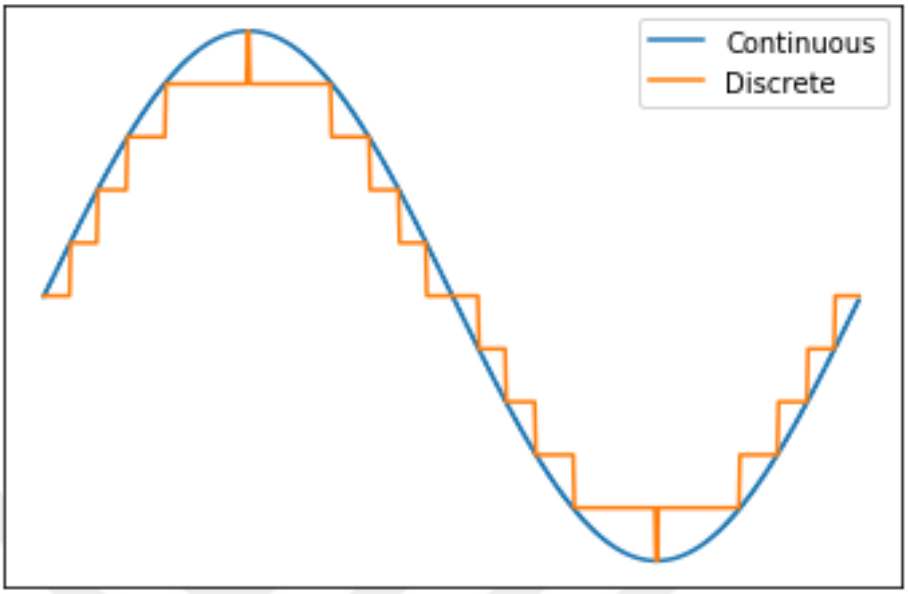

Figure 2.5: Illustration of continuous and discrete quantitative values.

Qualitative data, also known as categorical, is the other type of data that can be found in a real-life dataset. This type of data can be divided into three main categories, nominal, binominal and ordinal, where the binominal type of qualitative data is similar to nominal but has only two possible categorical values. The values of this type of data is always discrete, and can be described using numerical or alphanumerical values. The uncertainty in categorical values is the main challenge against handling this type of data in machine learning, where the position of the value in a category cannot be recognized [43, 44]. For example, measuring the similarity between two data instances, with teen value in the age attribute, is quite difficult, as the individual with this value can be actually aged anywhere between being one day away from being a child to one day away from being an adult. Thus, it is possible that the individual is more similar to another data instance with the child value, or senior, in the age attribute than another with the teen value, regarding the age similarity. Additionally, as the categorical values in a nominal attribute cannot have a reasonable order, it is impossible to measure the similarity between these values, directly from the dataset. However, these values are still of significant importance to machine learning, as they contain important information that can assist the ML technique to achieve the required task [45].

2.5 CATEGORICAL DATA ENCODING

Most of machine learning techniques, especially ANNs, can only handle numerical data, while many of the real-life datasets contain categorical attributes. When the categorical data is ordinal, it is possible to use a human's experience to convert it into numerical format, so that, categories are assigned with numbers that reflect their position in the overall range of the attribute [46]. However, this process requires additional knowledge and manual processing of the dataset, which is undesired operation in machine learning as it aims to automate data processing and knowledge extraction. Moreover, this kind of conversion limits the performance of the ML technique, as it is restricted to the knowledge imposed by the expert [47]. Additionally, as the aim of machine learning is to recognize and extract hidden features and patterns, in the dataset, that are difficult to be recognized by humans, the conversion of nominal values into numerical format can mislead the ML technique, so that, the knowledge extraction becomes quite difficult or inaccurate knowledge is extracted [48].

2.5.1 Label Encoding

As artificial neural networks can only handle numerical data, alphanumerical values in a qualitative attribute are converted to numerical format using Label-Encoding (LE). Per each categorical attribute, with alphanumerical values, all distinct values are collected from that attribute and ordered, alphabetically if no other order is specified [49]. Thus, for the sample alphanumerical values, shown in Table 2.1, the label-encoded value per each of them is shown. However, even if a reasonable order is selected, from an expert's point of view, the encoded values are not guaranteed to satisfy the order that can assist the ANN to extract the knowledge, accurately, using simpler topology and less resources consumption.

Table 2.1: Sample qualitative alphanumerical values and their coressponding encoded values.

Alphanumerical Values	Label-Encoded Values
D	3
A	0
C	2
B	1
A	0
D	3
C	2

2.5.2 One-Hot-Encoding

Except when a binominal, or continuous, output is required from the ANN, the number of neurons in the output layer is equal to the number of distinct values. The probability of an output to be assigned to the input is reflected by the neuron assigned to that output. In order to train the ANN, it is important to convert the shape of the actual outputs, from the dataset, into a format that allows calculating the error between the predicted and actual values. This type of encoding is known as One-Hot-Encoding (OHE), where each value is converted into a vector. The length of the vector is equal to the number of distinct values in the categorical attribute that holds the output values. All the values in the vector are set to zero, except the value placed in the position corresponding to the required output is set to one, which is the reason behind the OHE denotation [50, 51]. As the number of distinct values, in alphanumerical format, shown in Table 2.2 is four, the size of the vector created per each value is four, where only one of the values in the vector is set to one, among the zeros. This indicates that there a probability of 100% that this label is assigned to the input, which is used to measure the error for the neural network during training.

Table 2.2: Sample qualitative alphanumerical values and their coressponding OHE values.

Alphanumerical Values	OHE Vectors
D	[0, 0, 0, 1]
A	[1, 0, 0, 0]
C	[0, 0, 1, 0]
B	[0, 1, 0, 0]
A	[1, 0, 0, 0]
D	[0, 0, 0, 1]
C	[0, 0, 1, 0]

3. METHODOLOGY

Despite the popularity of using label and manual encoding to convert nominal data into numerical values, the values selected using both methods are not guaranteed to be suitable for the computations implemented in the ANNs, to achieve the best performance. Even when the values are encoded manually by an expert, the values that may seem reasonable for an expert may not match the needs of the neural network, especially that all ML techniques aim to extract knowledge that is hidden in the inputs. Thus, such encoding techniques limit the performance of the neural network, so that, the knowledge extraction becomes more difficult for these networks, and more complex networks become required to achieve the same tasks.

As the neural networks already have the ability to select the weights values that can maximize its performance, the proposed method aims to allow these networks to adjust the numerical value that represents each categorical, or nominal, value in an attribute. This adjustment is achieved during the training of the neural network, using backpropagation, so that, each categorical value is assigned with the suitable numerical value that achieves the best performance in the network. However, as the real-life datasets may contain mixed types of data, it is still important for the proposed method to handle quantitative values the same way they currently do, so that, the performance of the neural network is maximized when any type of data is used.

To handle any type of data, the proposed method uses two layers at the inputs side of the neural network. The number of neurons in the second layer is selected based on the number of attributes in the dataset, so that, a neuron is assigned per each attribute, regardless of the type of the data in that attribute. However, the number of neurons in the first layer, which directly receives the inputs from the environment is calculated depending on the type of the data in each attribute and the number of distinct values in attributes with nominal values. The connection among the neurons in these layers are defined based on the type of the data that the corresponding attribute have.

As shown in Figure 3.1, an attribute with quantitative values is assigned with a single neuron, in the first layer, which is directly connected to the corresponding neuron in the second layer. However, the number of neurons assigned to a nominal attribute is equal to the number of distinct values in that attribute, in the first layer. Neurons assigned for a certain nominal

attribute in the first layer are connected to a single neuron, which is assigned for that input, in the second layer. These neurons are not connected to any other neurons in the second layer.

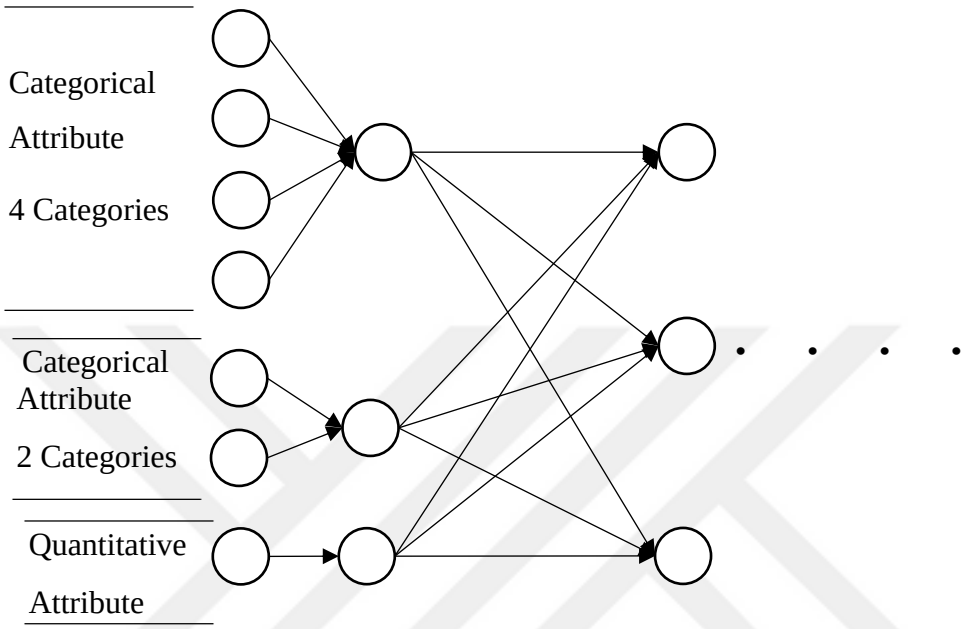

Figure 3.1: Illustration of the topology in the two layers using the proposed method.

To allow assigning the suitable value per each categorical value in the dataset, categorical inputs are encoded using OHE, so that, per each input, the set of neurons in the first layer that are connected to a single neuron in the second layer can have only a single input with the value one. As the value received by the neurons in the first layer are zeros, except one, the value received by the neuron in the second layer is equal to the value of the weight between the neuron in the second layer and the one that represent the nominal value in that attribute. Thus, by updating the weights values between the neurons in the first and second layers, the values of each nominal value, received by the neuron in the second layer, can be adjusted according to the needs of the neural network.

For the popular example of tennis play prediction based on the weather's condition, shown in Table 3.1, the data consists of four categorical attributes. Although it is possible to manually encode the two ordinal attributes, humidity and temperature, it is difficult to find reasonably encoded values for the two nominal attributes, wind and outlook. Moreover, the suitability of the manually encoded values for the computations in the neural network is questionable, as the way the neural network builds its decision is unknown. The existing

method to process this dataset using an artificial neural network requires converting these values into numerical format, using LE, as shown in Table 3.2, and use the ANN shown in Figure 3.2 to process this dataset.

Table 3.1: Weather conditions for tennis play forecast data table.

Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

Table 3.2: Output of LE the weather conditions.

Outlook	Temperature	Humidity	Windy	Play
2	1	0	0	0
2	1	0	1	0
0	1	0	0	1
1	2	0	0	1
1	0	1	0	1
1	0	1	1	0
0	0	1	1	1
2	2	0	0	0
2	0	1	0	1
1	2	1	0	1
2	2	1	1	1
0	2	0	1	1
0	1	1	0	1
1	2	0	1	0

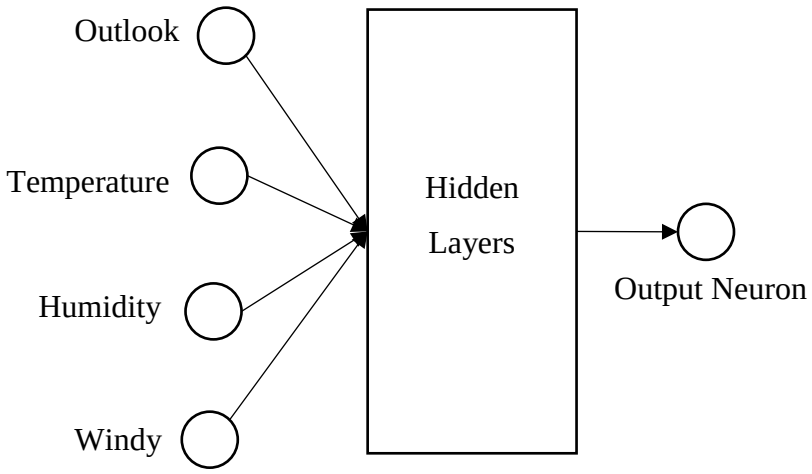

Figure 3.2: A sample neural network to process the tennis playing dataset using the existing method.

Using the proposed method, the data is converted into numerical values, shown in Table 3.3, using OHE, and the ANN shown in Figure 3.3 is used to process the encoded data.

Table 3.3: The OHE values of the weather conditions to be processed using the proposed ANN topology

Outlook			Temperature			Humidity		Windy		
Overcast	Rainy	Sunny	Cool	Hot	Mild	High	Low	False	True	Play
0	0	1	0	1	0	1	0	1	0	0
0	0	1	0	1	0	1	0	0	1	0
1	0	0	0	1	0	1	0	1	0	1
0	1	0	0	0	1	1	0	1	0	1
0	1	0	1	0	0	0	1	1	0	1
0	1	0	1	0	0	0	1	0	1	0
1	0	0	1	0	0	0	1	0	1	1
0	0	1	0	0	1	1	0	1	0	0
0	0	1	1	0	0	0	1	1	0	1
0	1	0	0	0	1	0	1	1	0	1
0	0	1	0	0	1	0	1	0	1	1
1	0	0	0	0	1	1	0	0	1	1
1	0	0	0	1	0	0	1	1	0	1
0	1	0	1	0	1	1	0	0	1	0

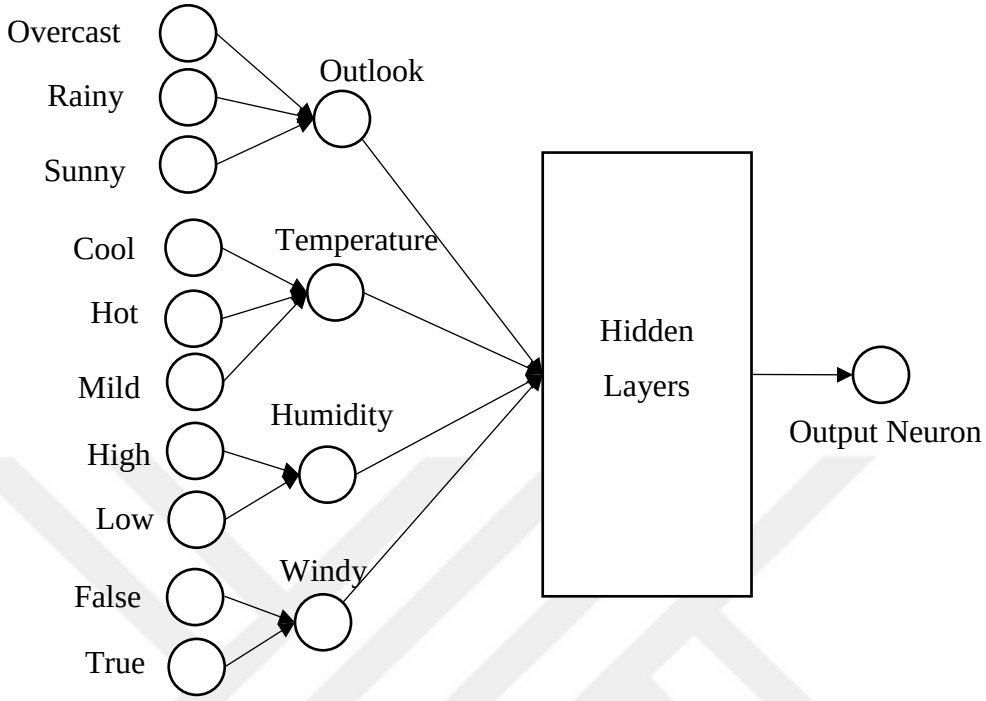

Figure 3.3: The topology of the ANN required to process the weather conditions dataset using the proposed method.

4. EXPERIMENTAL RESULTS

4.1 INTRODUCTION

As the aim of the proposed method is to improve the performance of ANNs when the input data include categorical values, this chapter illustrates the results of the experiments conducted to illustrate the difference in the performance when the propose topology is used. Each dataset is used to evaluate both the existing and the proposed topologies of artificial neural networks, so that, the improvements in the learning rate and overall performance of these networks can be measured and comparted.

4.2 EVALUATION DATASETS

Three UCI [52] datasets are used for the evaluation, which are the KDD 99, census income and nursery datasets. The KDD 99 dataset, which contains information collected from a network traffic that contains normal and intrusion access to the network, consists of 494,021 instances. Each instance is characterized using 41 attributes, where three of these attributes are categorical, which represents 7.32% of attributes. The census income dataset consists of 199,522 instances, each characterized using 41 attributes. The total number of categorical attributes in this dataset is 28, which represents 68.29% of the total attributes in the dataset. Additionally, the nursery dataset consists of 18,959 instances characterized using eight categorical attributes, i.e. 100% of the attributes are categorical. Table 4.1 summarizes the descriptions of these dataset. As the ratio of categorical attributes in a real-life dataset may vary, depending on the characteristics of the environment that the data is collected from, theses dataset are selected in order to evaluate the performance of the proposed method under different scenarios.

Table 4.1: Summary of the datasets used in the evaluation procedure.

Dataset	Instances Count	Attributes Count	Categorical Attributes Count	Percentage of Categorical Attributes
KDD99	494,021	41	3	7.32%
Census Income	199,522	41	28	68.29%
Nursery	18,959	8	8	100%

4.3 PERFORMANCE EVALUATION MEASURES

Backpropagation relies of the difference between the outputs predicted by the neural network and the actual outputs, acquired from the dataset, that the neural network is required to output. However, according to the use of the learning rate to avoid huge delta values to update the weights values, the neural network requires multiple iterations, epochs, to find the optimal value per each weight. The difference between the outputs of the neural network and the required ones is known the loss, where many functions are used to calculate that loss. Mean Squared Error (MSE), is one of the widely used function to calculate the error of the neural network. As shown in Equation 4.1, the MSE is calculated by calculating the squared difference between each output and its corresponding required value. Then, the mean of these values is calculated, by summing them up and dividing them by the number of outputs.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(P_i - A_i)^2 \tag{4.1}$$

where n is the number of classes in the dataset, P is the predicted value and A is the actual value of a data instance.

Another important measure is the accuracy of the predictions provided by the neural network. As the datasets used for the evaluation contain a class for each data instance, the task required from the neural network is to predict the correct class for each of the instances. The output with the highest value is used as the predicted label for that input. Thus, the accuracy of the predictions can be defined as the ratio between the total number of correctly predicted labels to the data instances' count, as shown in Equation 4.2.

$$Accuracy = \frac{Correctly\ predicted\ labels}{Data\ instances'count} \quad (4.2)$$

To illustrate the learning rate of the neural network, the values of the loss and accuracy are monitored through all the training epochs, so that, faster improvement in these values illustrate faster learning rate. The loss value is considered to be improved when it is reduced throughout the training, while the accuracy is improved when its value is increased during the training. Moreover, for accurate evaluation, the instances in the dataset are distributed in five bins, using stratified cross-validation. Stratified indicates that the ratios of each class in the bins is maintained as similar to its ratio in the original dataset as possible. The use of such approach eliminates any chance to produce biased splits, which can be more suitable for one of the networks than another. The overall performance of the neural network is illustrated by calculating the average loss and accuracy of that network per each epoch, for all the bins.

4.4 EXPERIMENT A – THE KDD 99 DATASET

The KDD 99 dataset, which has three, out of the 41, attributes with categorical values, as shown in Table 4.2, is used in this experiment. These categorical attributes are transformed into numerical values using both LE and the proposed encoding methods. Each formation is fed to a neural network with a suitable topology and the average loss and accuracy per each epoch for the five bins are monitored during the training. The neural network that relies on the LE value has an accuracy of 98.93% after being trained for 100 epochs, compared to an accuracy of 99.96% when the proposed topology is used. Other than the additional layer added to handle the OHE inputs, the topologies of these networks are identical, with 41, 64, 32, 16 and 32 neurons in each layer, respectively.

Table 4.2: Description of the attributes in the KDD99 dataset.

Attribute #	Data Type	No. of distinct values
1	Quantitative	-
2	Qualitative	3
3	Qualitative	66
4	Qualitative	11
5	Quantitative	-
6	Quantitative	-
7	Quantitative	-
8	Quantitative	-
9	Quantitative	-
10	Quantitative	-
11	Quantitative	-
12	Quantitative	-
13	Quantitative	-
14	Quantitative	-
15	Quantitative	-
16	Quantitative	-
17	Quantitative	-
18	Quantitative	-
19	Quantitative	-
20	Quantitative	-
21	Quantitative	-
22	Quantitative	-
23	Quantitative	-
24	Quantitative	-
25	Quantitative	-
26	Quantitative	-
27	Quantitative	-
28	Quantitative	-
29	Quantitative	-
30	Quantitative	-
31	Quantitative	-
32	Quantitative	-
33	Quantitative	-
34	Quantitative	-
35	Quantitative	-
36	Quantitative	-
37	Quantitative	-
38	Quantitative	-
39	Quantitative	-
40	Quantitative	-
41	Quantitative	-

The number of classes in this dataset is 32, which is the reason behind setting the number of neurons in the output layer to that number and the use of Soft-Max activation function. However, the 1.03% improvement in the performance, despite the existence of only 7.32% categorical attributes, reflects the importance of these attributes and the significant knowledge they contain, which has been able to improve the performance of the neural network when the suitable values are assigned per each of the nominal values in these attributes. Figure 4.1 also shows that the use of the proposed method has been able to improve the learning rate of the neural network, as the loss value has decreased, and accuracy increased, rapidly, compared to the use of the standard method.

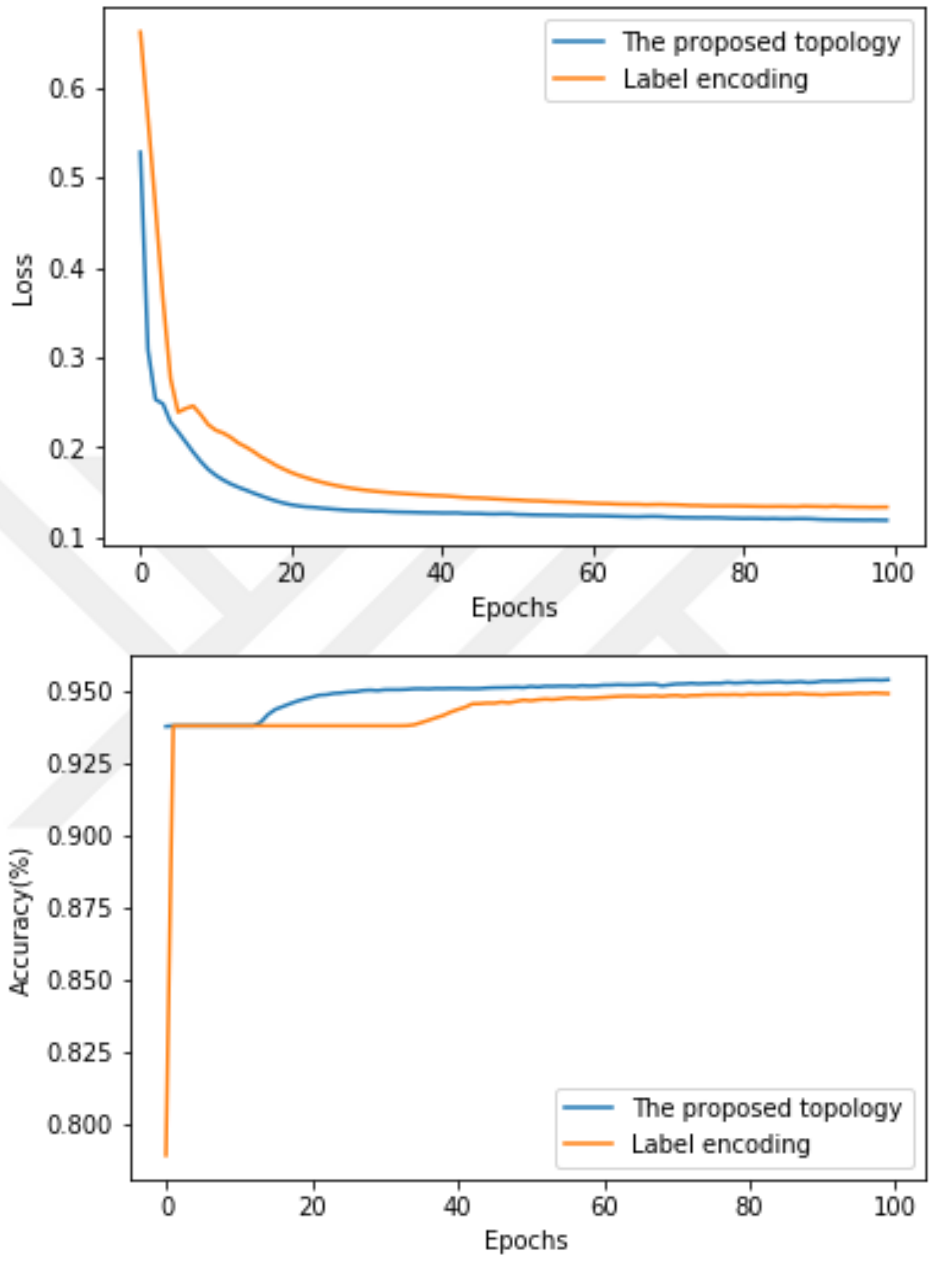

Figure 4.1: Illustration of the loss and accuracy versus training epochs using the KDD 99 dataset. Top: Loss versus Epochs; Bottom: Accuracy versus Epochs.

4.5 EXPERIMENT B – THE CENSUS INCOME DATASET

The categorical attributes in this dataset represent 68.29% of the total number of attributes, i.e. 28 out of 41 attributes, that are used to characterize the instances in the dataset, as

summarized in table 4.3. The same neural networks described in the previous experiment are used to process the dataset, except the output layer, which consists of a single neuron, with Sigmoid activation function, as there are only two possible labels per each instance. After training these networks for 100 epochs, the neural network that uses LE has shown a predictions accuracy of 94.91%, while the neural network implemented using the proposed method has shown an accuracy of 95.39%. Thus, the proposed method has only been able to achieve 0.48% better accuracy.

Despite the expectations of larger difference as the ratio of categorical attributes in the dataset is increased, it is possible that the values set by the LE are suitable for the computations of the neural network, or the selected topology is complex enough, for the required task, so that, it has been able to handle the values selected by the LE. However, as shown in Figure 4.2, the proposed method has shown significantly better learning rate, where the performance of the neural network has been able to achieve high accuracy, and low loss, within few epochs. Moreover, the figure shows that the neural network that uses the existing method has faced some struggle before it has been able to adopt the values imposed by LE.

Table 4.3: Description of the attributes in the census income dataset.

Attribute #	Data Type	No. of distinct values
1	Quantitative	-
2	Qualitative	9
3	Quantitative	-
4	Quantitative	-
5	Qualitative	17
6	Quantitative	-
7	Qualitative	3
8	Qualitative	7
9	Qualitative	24
10	Qualitative	15
11	Qualitative	5
12	Qualitative	10
13	Qualitative	2
14	Qualitative	3
15	Qualitative	6
16	Qualitative	8
17	Quantitative	-
18	Quantitative	-
19	Quantitative	-
20	Qualitative	6
21	Qualitative	6
22	Qualitative	51
23	Qualitative	38
24	Qualitative	8
25	Quantitative	-
26	Qualitative	10
27	Qualitative	9
28	Qualitative	10
29	Qualitative	3
30	Qualitative	4
31	Quantitative	-
32	Qualitative	5
33	Qualitative	43
34	Qualitative	43
35	Qualitative	43
36	Qualitative	5
37	Quantitative	-
38	Qualitative	3
39	Quantitative	-
40	Quantitative	-
41	Quantitative	-

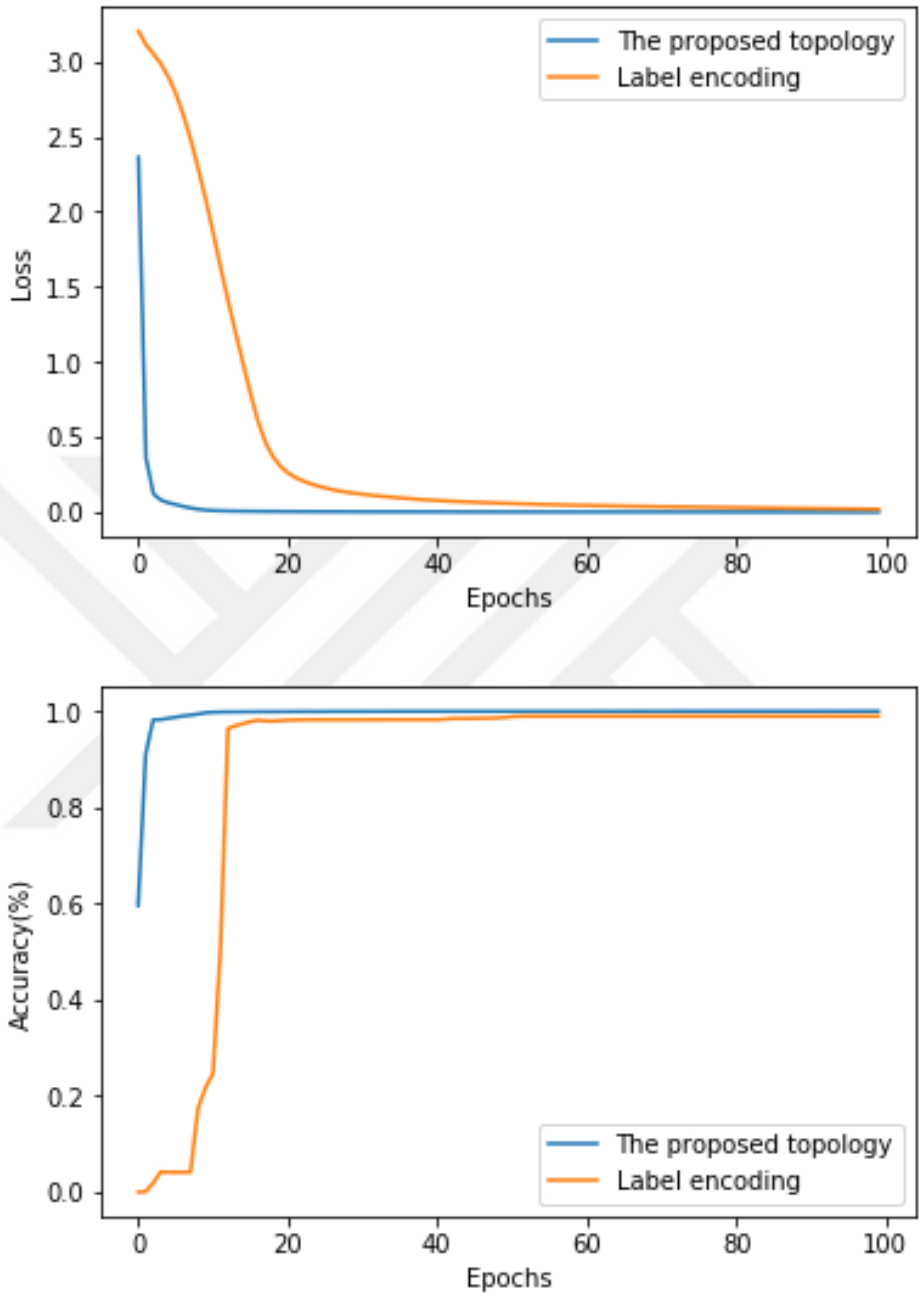

Figure 4.2: Illustration of the loss and accuracy versus training epochs using the census income dataset. Top: Loss versus Epochs; Bottom: Accuracy versus Epochs.

4.6 EXPERIMENT C – THE NURSERY DATASET

The instances in this dataset are characterized using eight categorical attributes, i.e. 100% of attributes are categorical, as shown in Table 4.4. The same neural networks are trained to predict the five possible classes, one per each instance, which imposes the use of five neurons in the output layer, with Soft-Max activation function. The average loss and accuracy, for the five folds, are also monitored during the 100 training epochs, to illustrate the learning rate of these networks, as shown in Figure 4.3. The neural networks that uses the existing LE-based method has produced an accuracy of 98.71%, while the neural network that uses the proposed method has an accuracy of 99.99%. The 1.26% better performance of the neural network that uses the proposed method, as well as the almost perfect predictions, with only one misclassified data instance, illustrate the remarkable performance of the proposed method, as this dataset consists of only categorical data. On the other hand, the existing method, using LE, has produces 167 misclassified data instances, which illustrate the difficulties that the neural network faces to adopt the values imposed by the LE.

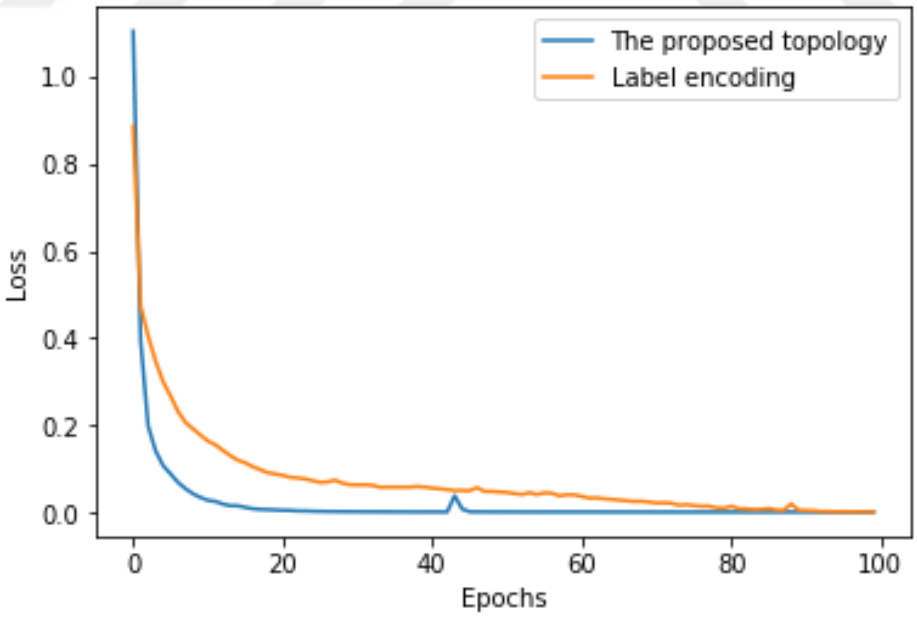

Figure 4.3.a: Illustration of the loss and accuracy versus training epochs using the nursery dataset. Top: Loss versus Epochs; Bottom: Accuracy versus Epochs.

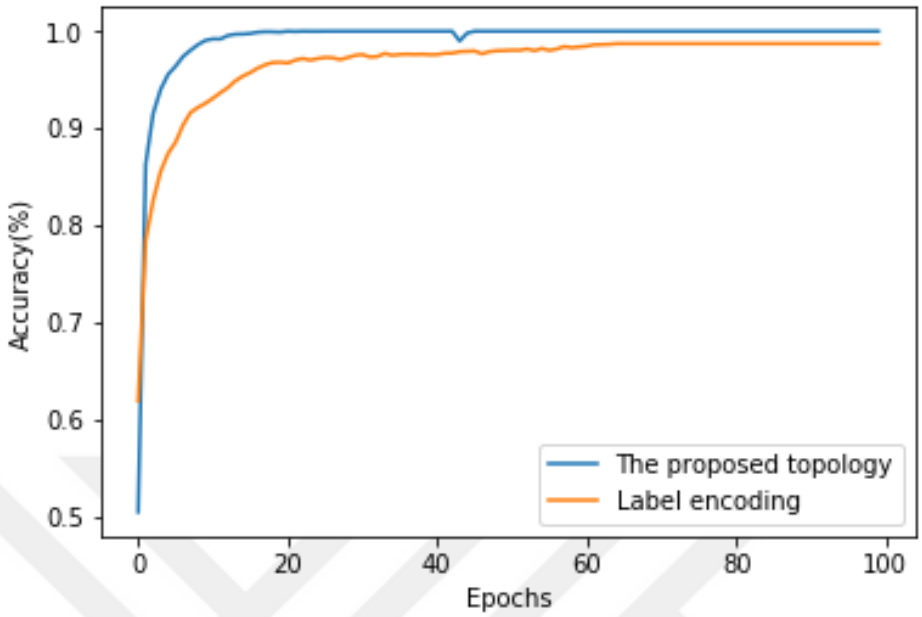

Figure 4.4.b: Illustration of the loss and accuracy versus training epochs using the nursery dataset. Top: Loss versus Epochs; Bottom: Accuracy versus Epochs.

Table 4.4: Description of the attributes in the nursery dataset.

Attribute #	**Data Type**	**No. of distinct values**
1	Qualitative	3
2	Qualitative	5
3	Qualitative	4
4	Qualitative	4
5	Qualitative	3
6	Qualitative	2
7	Qualitative	3
8	Qualitative	3

Another important difference between the proposed method and the LE-based method is the how fast these networks have been able to learn from the dataset, represented by the change in the loss and accuracy values versus training epochs. The use of the proposed method has been able to produce a significantly better start, as well as quick improvement in its performance. In contrast, the neural network that uses the existing method requires more training epochs to recognize the more complex patterns in the dataset, as the unsuitable values produces by the label encoder increased this complexity. Moreover, Figure 4.3 shows that the neural network that uses the LE method has not been able to improve its performance despite it has reached a high accuracy before the end of the training. This behavior shows that the values produced by the label encoder have caused the loss of knowledge in the dataset, while such behavior does not exist in the proposed method.

5. DISCUSSION

The results acquired from the experiment conducted to evaluate the performance of the existing and proposed methods, summarized in Table 5.1 and Figure 5.1, show that the proposed method has been able to achieve higher accuracy in all the conducted experiments. Although the improvement in the performance is expected to increase as the percentage of categorical attributes in the dataset increases, the improvement in the second experiment has been less than the one in the first experiment. Such anomaly reflects the effect of label encoded values over the performance of the neural network, as such similar performance between both methods is a result of the possible suitable values produced by the label encoder, or a complex neural network with respect to the features and patterns in that dataset. However, the faster learning of the neural network that uses the proposed method, shown by the faster improvement with respect to the training epochs, show that the values selected by the neural network itself are more suitable than those selected by the label encoder.

Table 5.1: Summary of the results collected from the conducted experiments.

Dataset	Percentage of Categorical Attributes	LE method Accuracy	Proposed Method Accuracy	Improvement
KDD99	7.32%	98.93%	99.96%	1.03%
Census Income	68.29%	94.91%	95.39%	0.48%
Nursery	100%	98.71%	99.99%	1.26%
Average	**58.54%**	**97.52%**	**98.45%**	**0.92%**

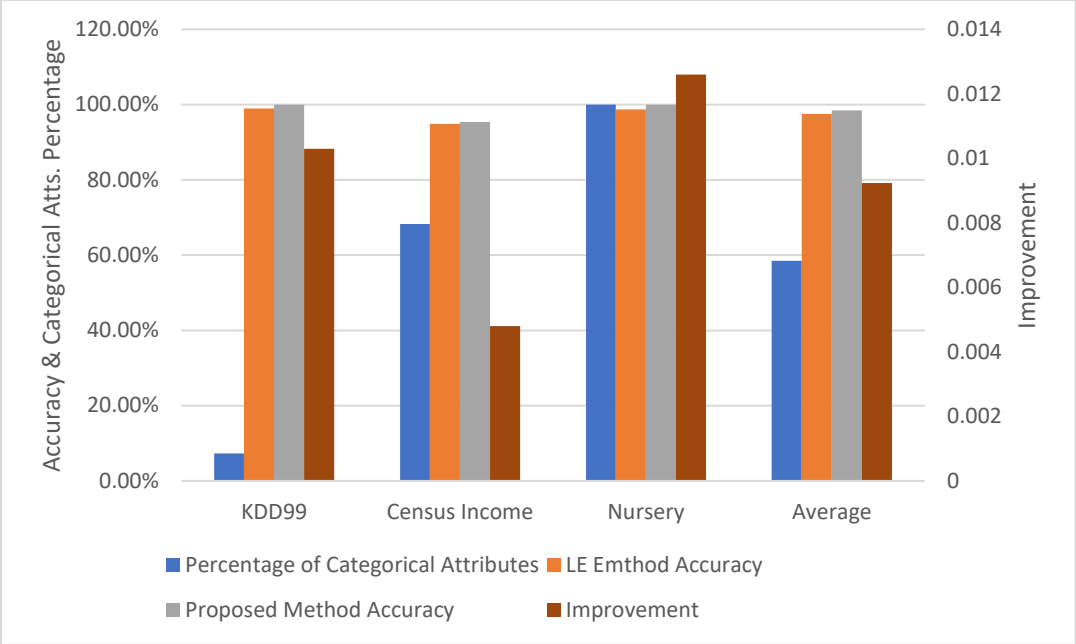

Figure 5.1: Illustration of the results collected from the conducted experiments.

Additionally, the remarkable performance of 99.99% using the proposed method, with the nursery dataset, which has 100% categorical attributes, illustrates the significance of the proposed method. As only one data instance is misclassified, compared to 167 using the LE-based method, out of the 18,959 data instances in the dataset, show that the maximum possible knowledge is extracted from the dataset using the proposed method. However, using the existing method, the accuracy has not been able to improve during the training, despite that the accuracy has reached to the maximum level before the training is completed. This behavior shows that the values produced by the LE has caused hiding some of the knowledge in the dataset, so that, the neural network has not been able to recognize despite the available epochs for such discovery.

As many of the real-life dataset may contain qualitative data, depending on the environment the data are collected from, the use of the proposed method can improve the performance of the ANNs employed in different applications. Some of the recent applications implemented using these networks, to process real-life data with qualitative data, such as [53-55], can improve their performance by employing the proposed method. Such improvement can be achieved in two aspects, the accuracy and simplicity of the neural network. As the knowledge extraction is significantly improved, simpler neural networks can be used to achieve the

required tasks, so that, faster and more accurate decisions using less computing-resources can be produced. Thus, the performance of these methods can be significantly improved, where better and rapid predictions can be made.

6. CONCLUSION

The data collected from a real-life environment can contain quantitative and qualitative values. Reasoning values of quantitative type is a simple task, as the actual value reflects its position in the range of data for that attribute. Qualitative data can be nominal or ordinal, where the approximate position of an ordinal value can be recognized in the range of data for an attribute. However, the precise position is sill uncertain, which imposes the challenge of handling such uncertainties in these values, even if the ordinal values are encoded by an expert. Moreover, the position of a nominal value in a range is unknown, as the order of these values and a defined range cannot be recognized. As this type of data in the dataset still contain important knowledge from the ML technique, it is important to convert these values into a numerical format with reasonable values, from the ML technique's point of view.

The existing method that converts qualitative values into numerical requires a predefined ordering rule, alphabetically by default, to convert the values in a qualitative attribute, in a dataset, to numerical format is known as label encoding. However, the values produced by this method are not guaranteed to be suitable for the ML techniques to extract the maximum knowledge in the dataset. Even if the ordering rule is defined by an expert in the domain that the data are collected from, these rules may conflict with the ML technique's mathematical computation used to extract the knowledge. Thus, allowing the ML technique to set a numerical value per each qualitative value is the optimal solution to allow the extraction of the maximum knowledge from the dataset.

Artificial neural networks are one of the ML techniques that are being widely used in different ML applications. The mathematical implementations in these networks are inspired by the signals processing and communications in a human's brain. A value, known as weight, is multiplied by the output of a neuron before being inputted to another neuron, to adjust the effect of source neuron on the computation in the destination one. To extract the knowledge from the dataset, during training, so that, the neural network becomes capable of achieving the intended task, two passes are executed in the network, forward and reverse. Calculating the output values of the neural network based on the input values is the forward pass, as the computation occur from the input to the output layers. In the reverse pass, backpropagation calculated the difference between the output of the network and the actual outputs, collected

from the dataset, in order to update the weight values. Thus, two identical neural networks can be employed to execute different tasks, by providing different training dataset to produce different weights values.

According to the ability of these networks to calculate the weight values suitable for their task, a novel method is proposed in this study to make use of the backpropagation to select the values suitable to encode qualitative attributes. A vector is created per each categorical attribute, where the length of the vector is equal to the number of distinct values in that attribute, and set all the values in that vector to zero, except the value located in a position corresponding to the categorical value of that data instance in the attribute. An additional layer is added to the neural network that has no role in patterns and features recognition but collects the inputs of each vector and connects them to a single neuron in the next layer. Thus, the number of neurons in the second layer is equal to the number of attributes in the original dataset, while the number of neurons in the first layer depends on the number of attributes, as well as the number of distinct values in each of the qualitative attributes. During backpropagation, the suitable value for each distinct value is selected, by adjusting the weight that connects this value to the neuron responsible of that attribute in the second layer. As the input can only be zero or one, the value that appear on the neuron corresponding to the attribute is equal to the weight value assigned to that categorical value.

The performance of the proposed method is evaluated using three UCI datasets, which has different ratio of qualitative attributes, and compared to the use of the existing LE-based method. The results of these experiments show that the performance of the ANN is improves when the proposed method is used with dataset that have categorical data. In addition to the improved accuracy, the proposed method has shown faster learning rate, reflected by the rapid decrement in the loss and rapid increment in the accuracy against the training epochs. Moreover, the results show that the use of the LE-based method has not been able to extract the maximum knowledge, as the value produced by the label encoder has hidden some of the patterns in the dataset. Thus, the proposed method can improve the performance of many application implemented based on ANNs to process real-life dataset that contain categorical attributes.

In future work, the employment of the proposed methods in different application is going to be evaluated. Some of the important applications that can make use of the proposed method can be, but not limited to, cyber security, medical data processing and natural language processing. According to the definite existence of categorical data in these applications, the use of the proposed method can improve the accuracy of the ANN and reduce the complexity of their topologies, so that, better decision can be produces using less computer resources, or faster decisions using the same resources.

REFERENCES

[1] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," Science, vol. 349, pp. 255-260, 2015.

[2] N. M. Nasrabadi, "Pattern recognition and machine learning," *Journal of electronic imaging*, vol. 16, p. 049901, 2007.

[3] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical machine learning tools and techniques*: Morgan Kaufmann, 2016.

[4] D. A. Cook, A. Kuper, R. Hatala, and S. Ginsburg, "When assessment data are words: validity evidence for qualitative educational assessments," *Academic Medicine*, vol. 91, pp. 1359-1369, 2016.

[5] J. Wang, J. Li, and Q. Su, "Monitoring categorical processes by integrating ordinal information," in *Industrial Engineering and Engineering Management (IEEM), 2016 IEEE International Conference on*, 2016, pp. 224-227.

[6] O. Moschidis and T. Chadjipadelis, "A Method for Transforming Ordinal Variables," in *Data Science*, ed: Springer, 2017, pp. 285-294.

[7] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85-117, 2015.

[8] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, pp. 31-44, 1996.

[9] C. Gershenson, "Artificial neural networks for beginners," *arXiv preprint cs/0308031*, 2003.

[10] Z. Zhang, "Artificial neural network," in *Multivariate Time Series Analysis in Climate and Environmental Research*, ed: Springer, 2018, pp. 1-35.

[11] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 427-436.

[12] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, *et al.*, "Mastering the game of Go with deep neural networks and tree search," *nature*, vol. 529, p. 484, 2016.

[13] W. Zhang, T. Du, and J. Wang, "Deep learning over multi-field categorical data," in *European conference on information retrieval*, 2016, pp. 45-57.

[14] R. K. Brouwer, "A feed-forward network for input that is both categorical and quantitative," *Neural Networks*, vol. 15, pp. 881-890, 2002.

[15] J. R. Rabuñal, *Artificial neural networks in real-life applications*: IGI Global, 2005.

[16] R. Beale and T. Jackson, *Neural Computing-an introduction*: CRC Press, 1990.

[17] R. E. Uhrig, "Introduction to artificial neural networks," in *Industrial Electronics, Control, and Instrumentation, 1995., Proceedings of the 1995 IEEE IECON 21st International Conference on*, 1995, pp. 33-37.

[18] K. Gurney, *An introduction to neural networks*: CRC press, 2014.

[19] S. Samarasinghe, *Neural networks for applied sciences and engineering: from fundamentals to complex pattern recognition*: Auerbach publications, 2016.

[20] M. Cilimkovic, "Neural networks and back propagation algorithm," *Institute of Technology Blanchardstown, Blanchardstown Road North Dublin*, vol. 15, 2015.

[21] X. Jin, C. Xu, J. Feng, Y. Wei, J. Xiong, and S. Yan, "Deep Learning with S-Shaped Rectified Linear Activation Units," in *AAAI*, 2016, p. 3.2.

[22] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, pp. 563-575, 2017.

[23] D. E. Rumelhart, B. Widrow, and M. A. Lehr, "The basic ideas in neural networks," *Communications of the ACM*, vol. 37, pp. 87-93, 1994.

[24] J. Fulcher, "Computational intelligence: an introduction," in *Computational intelligence: a compendium*, ed: Springer, 2008, pp. 3-78.

[25] I. K. Sethi and A. K. Jain, *Artificial neural networks and statistical pattern recognition: old and new connections* vol. 11: Elsevier, 2014.

[26] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural networks for perception*, ed: Elsevier, 1992, pp. 65-93.

[27] R. Miikkulainen, J. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, *et al.*, "Evolving deep neural networks," in *Artificial Intelligence in the Age of Neural Networks and Brain Computing*, ed: Elsevier, 2019, pp. 293-312.

[28] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, p. 436, 2015.

[29] T. Hastie, R. Tibshirani, and J. Friedman, "Unsupervised learning," in *The elements of statistical learning*, ed: Springer, 2009, pp. 485-585.

[30] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, pp. 3-24, 2007.

[31] D. Svozil, V. Kvasnicka, and J. Pospichal, "Introduction to multi-layer feed-forward neural networks," *Chemometrics and intelligent laboratory systems*, vol. 39, pp. 43-62, 1997.

[32] R. Eldan and O. Shamir, "The power of depth for feedforward neural networks," in *Conference on Learning Theory*, 2016, pp. 907-940.

[33] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *null*, 2003, p. 958.

[34] T. N. Sainath, A.-r. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Acoustics, speech and signal processing (ICASSP), 2013 IEEE international conference on*, 2013, pp. 8614-8618.

[35] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth annual conference of the international speech communication association*, 2014.

[36] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Gated feedback recurrent neural networks," in *International Conference on Machine Learning*, 2015, pp. 2067-2075.

[37] M. Tkáč and R. Verner, "Artificial neural networks in business: Two decades of research," *Applied Soft Computing*, vol. 38, pp. 788-804, 2016.

[38] D. R. Amancio, C. H. Comin, D. Casanova, G. Travieso, O. M. Bruno, F. A. Rodrigues, *et al.*, "A systematic comparison of supervised classifiers," *PloS one*, vol. 9, p. e94137, 2014.

[39] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," *arXiv preprint arXiv:1512.09300*, 2015.

[40] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, pp. 504-507, 2006.

[41] A. Bryman, "Quantitative and qualitative research: further reflections on their integration," in *Mixing methods: Qualitative and quantitative research*, ed: Routledge, 2017, pp. 57-78.

[42] G. B. Rossman and B. L. Wilson, "Numbers and words: Combining quantitative and qualitative methods in a single large-scale evaluation study," *Evaluation review*, vol. 9, pp. 627-643, 1985.

[43] D. P. Acharjya and K. Ahmed, "A survey on big data analytics: challenges, open research issues and tools," *Int. J. Adv. Comput. Sci. Appl*, vol. 7, pp. 1-11, 2016.

[44] S. Singh, C. Mayfield, S. Prabhakar, R. Shah, and S. Hambrusch, "Indexing uncertain categorical data," in *2007 IEEE 23rd International Conference on Data Engineering*, 2007, pp. 616-625.

[45] Y. Xia and B. Xi, "Conceptual clustering categorical data with uncertainty," in *Tools with Artificial Intelligence, 2007. ICTAI 2007. 19th IEEE International Conference on*, 2007, pp. 329-336.

[46] D. W. van der Palm, L. A. van der Ark, and J. K. Vermunt, "A comparison of incomplete-data methods for categorical data," *Statistical methods in medical research*, vol. 25, pp. 754-774, 2016.

[47] D. Ienco and R. G. Pensa, "Positive and unlabeled learning in categorical data," *Neurocomputing*, vol. 196, pp. 113-124, 2016.

[48] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs," *Proceedings of the IEEE*, vol. 104, pp. 11-33, 2016.

[49] Z. Qu and J. Hullman, "Evaluating visualization sets: Trade-offs between local effectiveness and global consistency," in *Proceedings of the Sixth Workshop on Beyond Time and Errors on Novel Evaluation Methods for Visualization*, 2016, pp. 44-52.

[50] B. G. Horne and C. L. Giles, "An experimental comparison of recurrent neural networks," in *Advances in neural information processing systems*, 1995, pp. 697-704.

[51] S. Marinai, M. Gori, and G. Soda, "Artificial neural networks for document analysis and recognition," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, pp. 23-35, 2005.

[52] C. Blake and C. Merz, "UCI repository of machine learning databases [http://www. ics. uci. edu/∼ mlearn/MLRepository. html], Department of Information and Computer Science," *University of California, Irvine, CA*, vol. 55, 1998.

[53] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, 2016, pp. 258-263.

[54] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *Ieee Access*, vol. 5, pp. 21954-21961, 2017.

[55] E. Hodo, X. Bellekens, A. Hamilton, P.-L. Dubouilh, E. Iorkyase, C. Tachtatzis, *et al.*, "Threat analysis of IoT networks using artificial neural network intrusion detection system," in *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, 2016, pp. 1-6.