Preference Neural Network and its Applications

by **Ayman Ahmed Elgharabawy**

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

under the supervision of Dr. Mukesh Prasad

University of Technology Sydney
School of Computer Science
Faculty of Engineering and Information Technology

December 2021

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, *Ayman Ahmed ELgharabawy* declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the *School of Computer Science*, *Faculty of Engineering and Information Technology.* at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

SIGNATURE

DATE: 6TH DECEMBER, 2021

ABSTRACT

This thesis proposes a novel label ranker network to learn the relationship between labels for classification and ranking problems. The Preference Neural Network (*PNN*) uses *spearman* correlation gradient ascent and two new activation functions,namely positive smooth staircase (*PSS*) and smooth staircase (*SS*) that accelerate the ranking by creating deterministic preference values. *PNN* is proposed in two forms, fully connected simple layers and Preference Net (*PN*), where the latter is the deep ranking form of *PNN* to learning feature selection using a novel ranker kernel to solve images classification problem. *PN* uses a new type of multiple size weighted ranker kernel to generate a feature map. *PNN* outperforms five previously proposed methods for label ranking, obtaining state-of-the-art results on label ranking, and *PN* achieves promising results on *CFAR-10* with high computational efficiency. The thesis includes different types of *PN* architecture to solve the problem of subgroup label ranking. Subgroup label ranking, which aims to rank labels in individual groups using a single ranking model, is a new problem in preference learning. This thesis also introduces the subgroup preference neural Network (*SGPNN*) that combines multiple networks that have different activation functions, learning rate, and output layer into one to discover the hidden relation between the subgroups' multi-labels. The *SGPNN* is a feedforward (*FF*), partially connected that has a single middle layer and uses stairstep (*SS*) multi-valued activation function to optimize learning and achieve better ranking performance. The novel structure of the proposed *SGPN* consists of a multi-activation function neuron (*MAFN*) in the middle layer to rank each subgroup independently. The *SGPNN* uses gradient ascent to maximize the Spearman rank correlation between the subgroups' multi-labels. Each label is represented by an output neuron that has a single *SS* function. Experiments were conducted that applied the *SGPNN* to a new synthesized dataset with subgroup label ranking achieving promising results in computational cost and performance. *PN* has experimented on image recognition benchmarks datasets, and *SG-*

PNN is applied on *EEG* motor imagery BCI competition IV dataset 2b to solve the data ambiguity. Under these varying backgrounds and scenarios, the thesis has shown that the proposed *PNN* provides a learning tool for label ranking and class classification. *PNN* outperforms label ranking state-of-the-art and gives promising results in image classification via literature explanation and empirical results.

Dedication

To my mothers´soul who died during Covid-19 pandemic.

To my wife and my kids who supported me and suffered to live without me during this research.

To my supervisors Dr.Mukesh Prasad and Prof. Ct. Lin who believed in my ability to finish this research after all these hardships.

ACKNOWLEDGMENTS

Foremost, I would like to express my sincere gratitude to my supervisors *Dr. Mukesh Prasad* and *Prof. Chin Teng Lin* for the continues support of my Ph.D study and research, for their patience, motivation, enthusiasm, and immense knowledge. Their guidance helped me in all the time of research and writing of this thesis.

I also would like to appreciate my fellow in computer science: *Dr. Mohamed Barbar* for his code review and performance optimization. and my fellow *Dr. Akram Youssry* in center of quantum for his fruitful discussion in the design of the activation function.

I also would like to thank *Prof. Nikhil Pal* in Indian Statistical Institute for his valuable suggestions in choosing the error function of Preference Neural Network to make it a unique and novel peace of network to reach the state-of-the-art in the machine learning.

additional to, I like to thank the *Dr. Nabin Sharma* in department of drones for his valuable comments in image recognition to validate the results of preference net.

Last but not the least, I would like to thank my family: my wife and my kids, for their unconditional support, both financially and emotionally throughout the whole PhD journey.

Ayman Elgharabawy

December 2021 @UTS

PUBLICATIONS

Journal Publications Related to the Thesis :

1) Preference Neural Network. **(Chapter Two)** (Under Review) (Elgharabawy et al. 2021*a*)

2) Preference Net: Image Recognition using Ranking Reduction to Classification. **(Chapter Three)** (Under Review)

3) Subgroup Preference Neural Network. **(Chapter Four)** (Published) (Elgharabawy et al. 2021*b*)

TABLE OF CONTENTS

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Statement

The task of "learning to rank" has gained much attention in recent years among many machine learning problem because it is used in many fields of human life, these fields includes expert systems, internet information retrieval by search engines, object ranking and document ordering. Label ranking is one of the ranking problems of machine learning.

This thesis focuses on four novel approaches in label ranking, Learning new types of relations between labels, Deep label ranking, classification by ranking reduction, and new image processing based on pixel ranking. The thesis also solves the problem of data ambiguity in machine learning and shows how to convert multi-class classification problems to solve them as multi-label ranking using neural networks. Also, the thesis proposes a novel network architecture to solve a new type of label preference relations. The proposed new architecture also solves ambiguous data classification and non-ground truth problems where data has class overlapping. The thesis shows how the ranker can serve the machine learning classification problems and act as a classifier with better performance using ranking reduction by using score function. In addition to computational performance, the thesis introduces nonlinear functions to accelerate the learning steps in a few epochs in order to reach a high-performance neural network.

1.2 Research Scope

The research scope covers three main area in machine learning

- Label ranking (*LR*) in Preference learning (*PL*) and coupling the LR with Artificial Neural network (*ANN*)
- Deep learning (*DL*) and *LR*and coupling the two domains by proposing Preference Net (*PN*).
- Subgroup Discovery domain (*SD*) and *LR* and coupling the two domain by introducing Subgroup Preference Neural Network (*SGPNN*). SGPNN learn multiple sub groups of labels from a conjoint data simultaneously in order to deliver a generalized model

The main scope's research areas are illustrated in Fig 1.1.

1.3 Research Questions

The following questions initiate this research:

1) Does the ranker work as a classifier with better performance?
2) Does the ranker learn any new types of preference relations not used before in machine learning, i.e., indifference and incomparable relations and labels in subgroups?

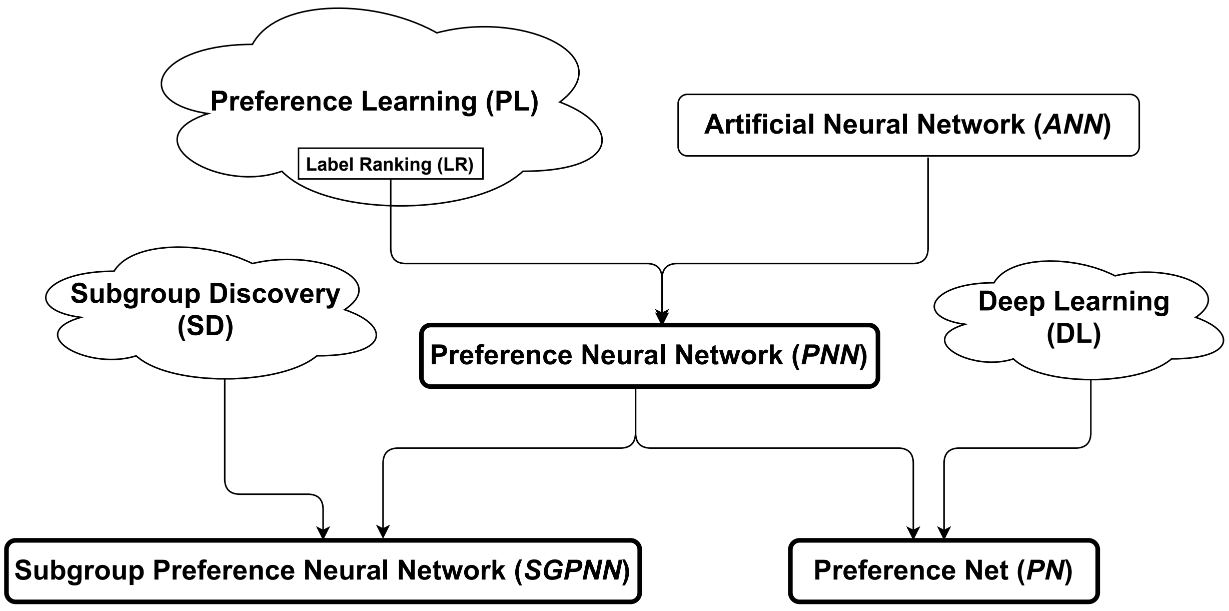

Preference Learning (PL)
Label Ranking (LR)
Artificial Neural Network (*ANN*)
Subgroup Discovery (SD)
Preference Neural Network (*PNN*)
Deep Learning (DL)
Subgroup Preference Neural Network (*SGPNN*)
Preference Net (*PN*)

Figure 1.1: Research Scope

3) Can the ranking solve the deep learning classification problems, i.e., image recognition

4) Can the ranking solve the overlapping problems, i.e., data ambiguous where classes are overlapping?

5) Can we develop a high-performance neural network based on ranking evaluation criteria to solve both ranking and classification problems?

1.4 Research Methods

This thesis proposes a new *ANN* as a semi-supervised learning where the data has training part to train the model and testing part to validate the model. The proposed *ANN* learn the different relations of label ranking by learning the relation between data instances. Each data instance is represented as input to the neural network, and the output represents the labels. The research started from the fundamental of neural networks and worked in each neural network component to be based on ranking. The research started from the activation function in order to enhance the ranking convergence performance and produce multiple output values and objective function to be suitable for ranking evaluation. The first part of the research produced fully connected neural network implemented to rank multiple labels at less computational cost and better ranking performance.

1.5 Aims and Objectives

There are four aims to this research:

1) Predicting new types of preference relations opens the road to solving new types of preference learning problems as a subgroup, indifference, and incomparable relations that handle both restricted and non-restricted ranking as illustrated in Fig. 1.2.

2) Enhance the classification accuracy of machine learning problems by converting the multi-class classification to multi-label ranking and apply ranking reduction to classification.

3) Implement new activation functions that accelerate the learning process and enhance the prediction probability.

4) Build a new ranker neural network design for preference learning, based on a native ranking structure in terms of new activation and objective functions for ranking.

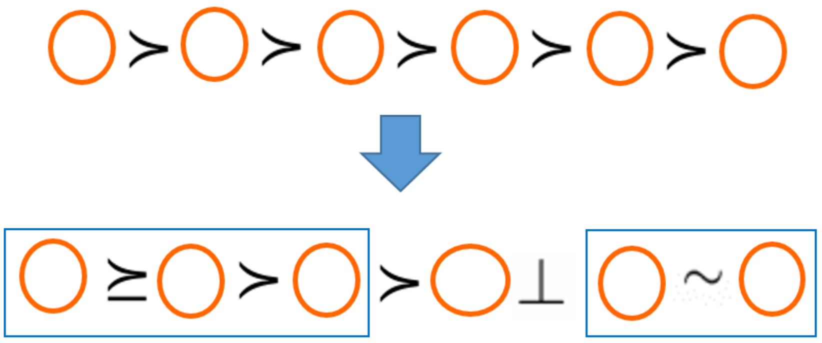


Figure 1.2: Subgroup, incomparable⊥ and indifference ∼ are New types of labels relations

1.6 Research Challenges

The thesis has main challenges to proof which can be summarized as the following

1.6.1 Combining the *PL* domain and *ANN*

The main challenge of the thesis is solving a new type of *PL* problem by introducing a new type of *ANN* to learn the preference learning rules. the *PL* and *ANN* coupling has sub-challenges that are mentioned below.

1) kendall τ error function is not differentiated or not continues for use in gradient calculation. searching for new error function for the ranking learning process.

2) Using spearman ranking correlation in conventional classification and deep learning.

3) Switching between multi-class classification problem and multi-label ranking problem.

4) Solving the problem of class overlapping, data ambiguity and sub-group label ranking.

5) Reducing the computational cost for data classification.

1.6.2 Combining the relation between *LP* domain and *DL* domain

Finding a better kernel design and computing approach to solve the deep classification problems using different sizes of the kernel and using spearman ranking correlation instead of convolution to propose a suitable alternative network compared to Convolution neural network (*CNN*) based on ranking.

1.6.3 Combining the relation between *SD* domain and *LR*

The challenge of Combining the descriptive and predictive data mining in one of the research challenges as it has sub-challenges are mentioned below

1) Integrate the steps of SD mining with a predictive machine learning approach to have a generalized learning model for several subgroups discovered by mining.

2) Combined the discovered group data in one data file and fed the network as one data has two different models.

3) The difficulties in implementation of multiple activation function neuron.

4) Dividing the network into a composite of more than one subnetwork, each network has a different learning rate, output results, and one middle layer.

1.7 Research Contributions

To address the above research issues, the research contributions in this thesis are summarized

1) Proposing a new type of activation function *SS* to enhance the predictive probability over the convention activation functions i.e, *Sigmoid*, *Relu* and *Softmax* due to the step shape that enhances the predictive probability from a range from -1 to 1 in the sigmoid to almost discrete multi-values. The new proposed activation functions accelerate the ranking convergence by reducing the number of iterations.

2) The proposed *PNN* uses gradient ascent to maximize the *spearman* ranking correlation coefficient. In contrast, other classification-based methods such as *MLP-LR* use the absolute difference of root mean square error (*RMS*) by calculating the differences between actual and predicted ranking and other *RMS* optimization, which may not give the best ranking results.

3) *PNN* network is implemented directly as a label ranker by using the new multivalued activation functions to rank all the labels together in one model. The *SS* or *PSS* functions provide multiple output values during the conversions; however, *MLP-LR* and *Ranknet* use *sigmoid* and *Relu* activation functions. These activation functions have a binary output. Thus, it ranks all the labels together in one model instead of pairwise ranking by classification.

4) *PN* uses a novel approach for image classification by learning the feature selection using pixels ranking with different sizes of weighted kernels to scan the image and generate the features map.

5) Introducing a new *ANN* architecture to rank sub group of labels, the Subgroup Preference Learning (*SGPL*), which is designed to rank multi-label incomparable/indifference subgroups by using a novel

multi activation functions Neuron (*MAFN*) that uses *SS* activation functions, including one function for each output layer.

6) Combining all the features of multi-label datasets with building a unified ranking model for different datasets from different domains. We find that the *SGPNN* has a remarkably better performance than other approaches that rank each dataset using a separate model.

1.8 Thesis Structure

The thesis is structured as follows: Chapter two gives background knowledge and related work on the main approaches that have been used in preference learning using label ranking.

Chapter three answers the research questions 1,2, and 5 where ranker can work as a classifier and solve deep learning problems. The chapter shows how the new activation functions outperform the conventional functions in ranking using the *PNN* architecture.

Chapter three introduces the preference neural network as a fully connected multilayer perceptron *MLP* and introduces the initial experiment of ranker network, which leads to the proposed network and how the spearman error enhances the error and convergence and two types of novel activation functions.

Chapter Four describes how we extended the *PNN* MLP to introduce a deep ranking approach; we name it preference net based on the proposed activation and error function of *PNN* and image preprocessing by preference learning technique using pixel ranking and averaging and introducing a new type of kernel to learn the feature. Chapter four solves the deep learning problem using ranking and answers research questions 3 and 5.

Chapter five describes the new architecture of *PNN* to solve the problem of multi-label ranking in multi groups. Subgroup preference Neural Network *SGPNN* has a new type of neuron multi activation function neuron (MAFN). Chapter five answers research question 2.

Chapter six describes the application that uses *SGPN* in the Brain Computer Interface (BCI) field. The chapter describes the Motor imagery dataset 2B used from BCI competition IV. The challenges in this dataset are complicated and have unbalanced and overlapping labels. Chapter six introduce a new algorithm to unify the input data and use a ground truth normalized to increase separability. *SGPNN* outperform most of the other deep learning approaches on this dataset. Chapter six answers research question 4.

Chapter seven concludes the thesis and outlines the scope for future work. Fig. 1.3 shows the research profile of the thesis.

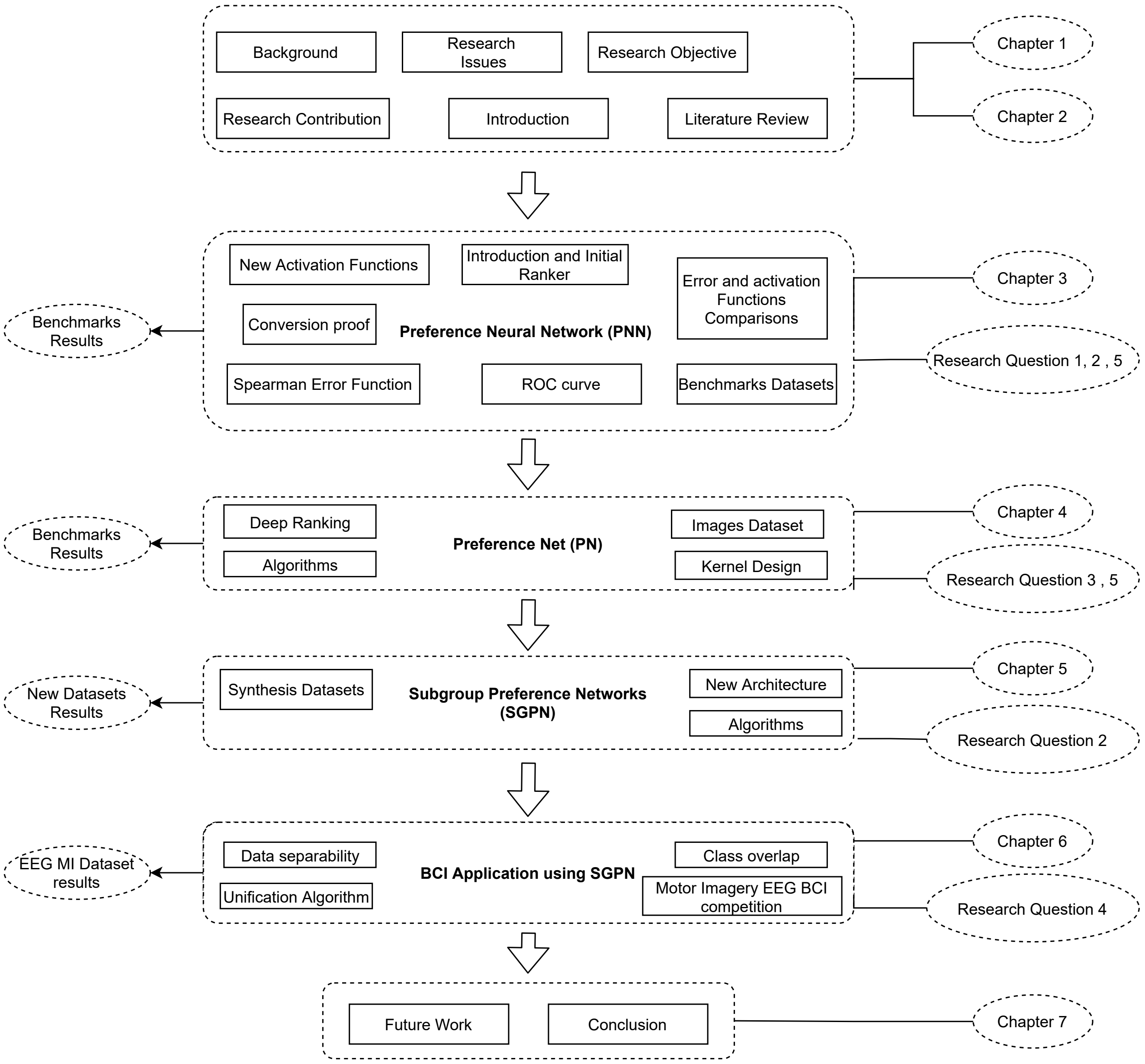


Figure 1.3: The thesis structure

CHAPTER 2

PREFERENCE LEARNING AND NEURAL NETWORK

2.1 Introduction

PL is an extended paradigm in machine learning by inducing predictive preference models from experimental data (Frnkranz & Hllermeier 2010, Brafman & Domshlak 2009, Adomavicius & Tuzhilin. 2005). *PL* has wide a scope includes data retrieval, machine learning, decision theory as shown in Fig 2.1. *PL* has applications in a variety of research areas such as knowledge discovery and recommender systems (Montaner & López 2003). *PL* has three main category fields of research problems as shown in Fig 2.2:

- Label Ranking.
- Instance Ranking.
- Object Ranking.

Of those previous research areas, label ranking (*LR*) is a challenging problem that has gained importance in information retrieval by search engines (Aiolli 2005, Crammer & Singer 2002). Unlike the common problems of multi-class and multi-label classification, multi-label ranking involves predicting the relationship between multiple labels' orders.

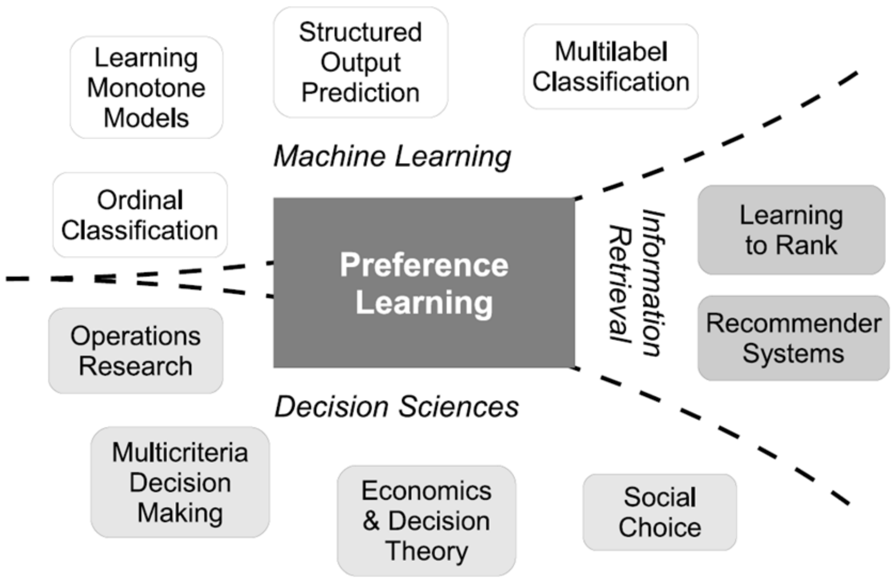


Figure 2.1: Preference Learning Domain

In case of multi-class classification, for a given instance x from the instance space $\mathbb{X}$, there is a label y associated with x, where $y \in \mathcal{L}$, where $\mathcal{L} = \{ \lambda_1,..,\lambda_n\}$, where n is the number of labels. *LR* is an extension of multi-class and multi-label classification, where each instance x is assigned an ordering of all the class labels in the set $\mathcal{L}$. This ordering gives the ranking of the labels for the given x object. This ordering can be represented by a permutation of the set $\pi = \{1, 2, \cdots, n\}$.

2.2 Label Ranking (*LR*)

Label ranking is a complex prediction task where the goal is to map instances to a total order over a finite set of predefined labels. An exciting aspect of this problem is that it contains several supervised learning

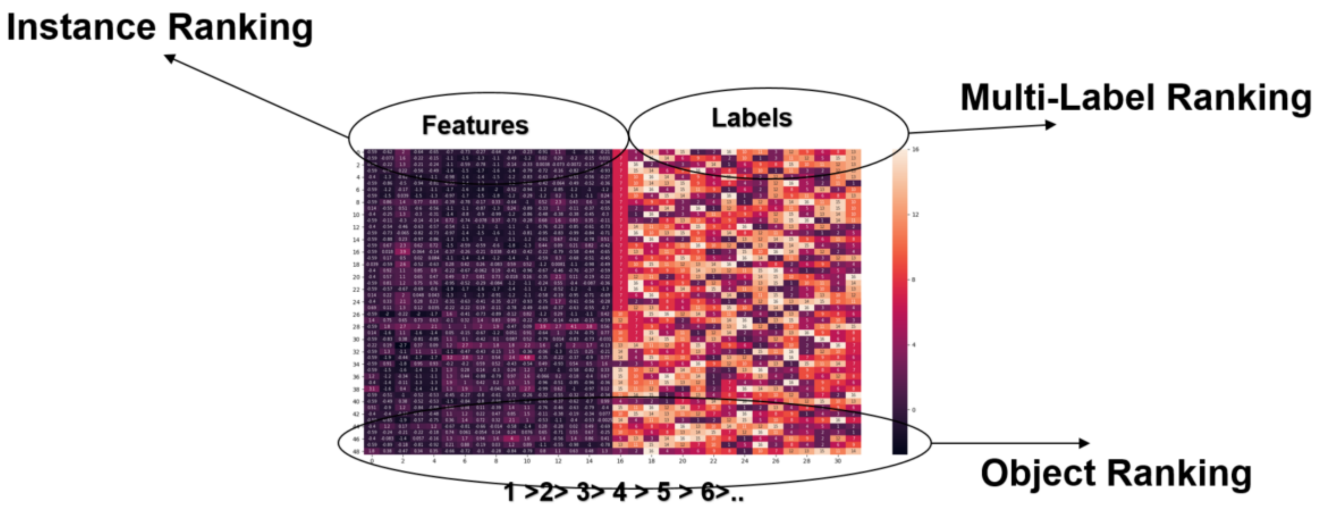


Figure 2.2: Preference Learning (*PL*) Methods

problems, including multi-class prediction, multi-label classification, and hierarchical classification. Unsurprisingly, there are many label ranking algorithms in the literature due, in part, to this versatile nature of the problem. In this chapter, we survey these algorithms.

Label preference takes one of two forms of restrictions.

- The strict label order ($\lambda_a \succ \lambda_b \succ \lambda_c \succ \lambda_d$) can be represented as $\pi = (1,2,3,4)$.
- non-restricted total order π=($\lambda_a \succ \lambda_b \simeq \lambda_c \succ \lambda_d$) can be represented as $\pi = (1,2,2,3)$, where a, b, c and d are the label indexes and λ_a, λ_b, λ_c and λ_d are the ranking values of these labels respectively.

2.2.1 Restricted Label Ranking

The label ranking association rules for restrict ranking has the following rules.

- Irreflexive where $\lambda_a \not\succ \lambda_a$
- Transitive where $(\lambda_a \succ \lambda_b) \wedge (\lambda_b \succ \lambda_c) \implies \lambda_a \succ \lambda_c$

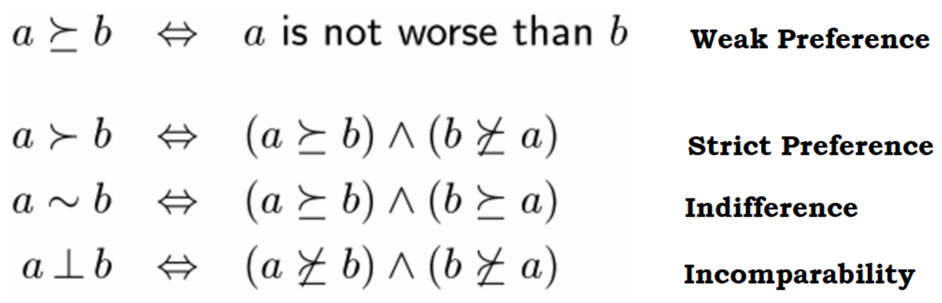
- Asymmetric $\lambda_a \succ \lambda_b$ then $\lambda_b \nsucc \lambda_a$.
- Connected : for any λ_a , λ_b in $\mathcal{L}$, either $\lambda_b \succ \lambda_a$ or $\lambda_a \succ \lambda_b$

2.2.2 Unrestricted Label Ranking

The label ranking association rules for unrestricted ranking has the following rules.

- Transitive where $\lambda_a \succeq \lambda_b \wedge \lambda_b \succeq \lambda_c \implies \lambda_a \lambda_c$.
- reflexive where $\lambda_a \succeq \lambda_a$.
- AntiSymmetric $\lambda_a \succeq \lambda_b$ then $\lambda_b \succeq \lambda_a \implies \lambda_b = \lambda_a$.
- Connected : for any $\mathcal{L} \in \{\lambda_a , \lambda_b\}$, either $\lambda_b \succeq \lambda_a$ and $\lambda_a \succeq \lambda_b$ or $\lambda_a = \lambda_b$

The meaning of association rules is mentioned in Fig. 2.3 For the non-continuous permutation space, The order is represented by the relations mentioned earlier and the $\perp$ incomparability binary relation. For example the partial order $\lambda_a \succ \lambda_b \succ \lambda_d$ can be represented as π=(1,2,0,3) where 0 represents an incomparable relation since λ_c is not comparable to (λ_a,λ_b,λ_d).


$a \succeq b \iff a$ is not worse than b **Weak Preference**

$a \succ b \iff (a \succeq b) \wedge (b \not\succeq a)$ **Strict Preference**

$a \sim b \iff (a \succeq b) \wedge (b \succeq a)$ **Indifference**

$a \perp b \iff (a \not\succeq b) \wedge (b \not\succeq a)$ **Incomparability**

Figure 2.3: Preference Relations

Various label ranking methods have been introduced in recent years (Zhou et al. 2014), such as decomposition based methods, statistical methods, similarity, and ensemble-based methods. decomposition methods include pairwise comparison (Furnkranz & Hüllermeier 2003, Fürnkranz & Hüllermeier 2010), log-linear models and constraint classification (Har-Peled et al. 2002). The pairwise approach introduced by hüllermeier E. Hüllermeier (2008) divides the multi-label ranking problem into several binary classification problems in order to predict the pairs of labels $\lambda_i \succ \lambda_j$ or $\lambda_j \prec \lambda_i$ for an input x. Statistical methods includes decision trees (Furnkranz & Hüllermeier 2011), instance-based methods (Plackett-Luce) (Cheng & Hüllermeier 2008) and Gaussian mixture model (Mihajlo et al. 2014) based approaches.For example, Mihajlo et al. (2014) uses Gaussian Mixture models to learn soft pairwise label preferences. Similarity methods minimize the distance of the labels instead of maximizing the probability of label values. For example, Aiguzhinov et al. Aiguzhinov et al. (2010) used an adapted version of the Naive Bayes algorithm, which used similarity between label rankings instead of probability. Association rules mining has also been adapted for label ranking (Jorge & J. P. da 2011). The multi-layer perceptron has also been adapted for label ranking (Ribeiro et al. 2012). For example, Ribeiro et al. (2012) proposed six approaches that use label ranking loss during backpropagation (*BP*). Cheng, Hühn & Hüllermeier (2009) used instance-based decision tree to rank the labels based on predictive probability models of a decision tree Cheng et al. (2009). Hüllermeier combined both a decision tree and supervised clustering in two approaches for label ranking by mapping between instances and multi-labels ranking space Grbovic et al. (2013).

Zhou & Qiu (2018) provided a scalable decision tree structure by implementing a random forest with a parallel computational architecture for extreme label ranking (Zhou & Qiu 2018). (de Sá et al. 2017) introduced label random forest (*LRF*) as an ensemble method of ranking (de Sá et al.

2017). *LRF* was based on the best approach result of previous ranking decision trees using entropy-based ranking. Jung and Tewari proposed an approach for label ranking based on the voting of the best learners and scoring the labels for ranking (Jung & Tewari 2018). Song and Huang proposed a framework to solve the vulnerability of multi-label deep learning models (Song et al. 2018). Yan and Wang proposed a long short term memory (*LSTM*) based multi-label ranking model for document classification (Yan et al. 2017). This method uses a two-step process - the first step learns the document representation while the second step ranks the labels. Guo and Hou introduced low-rank multi-label classification with missing labels (*LRML*), which recovered the missing labels via Laplacian manifold regularization derived from the feature space by utilizing the low-rank mapping(Guo et al. 2018). In our study, we shall use staircase activation functions. In this context, it is worth noting that.

Preference mining (*PM*) is a new domain of preference learning (*PL*), which aims to discover the local patterns and deviations of subsets of data (Holland et al. 2003, de S*á* & Duivesteijn 2018). *PL* aims to build a predictive model that can predict a multi-label ranking problem based on preference relations over a permutation space ω where each member of a group of k labels has a preference λ value, $\mathcal{L} = \{\lambda_1, \lambda_2, ..., \lambda_k\}$, where the differences of λ value represents preference relations ($\succ, \succeq, \nsucc, \nsucceq, \sim, \prec, \preceq$) (Frnkranz & Hllermeier 2010, Chankong & Haimes 2008). However, real-world data can be ambiguous and often lack preference relations between two or more labels, and the missing relations can be mapped to an indifference $\sim$, or incomparability $\perp$, relation (Brinker & Hüllermeier 2007, Chiclana et al. 2009). These two relations create a partial order on the ω space where $\lambda_a \perp \lambda_b$ or $\lambda_a \sim \lambda_b$. The partial relations were solved in terms of the relation between labels in one ω space in (Vembu & Gärtner

2010, Henzgen & Hüllermeier 2014). For example, $\pi = (\lambda_a \succ \lambda_b \sim \lambda_c \succ \lambda_d)$ is mapped to $\pi = (1,2,2,3)$ and $\pi = (\lambda_a \succ \lambda_b \succ \lambda_c \perp \lambda_d)$ is mapped to $\pi = (1,2,3,0)$. However, sometimes the data collected from the likes of recommender systems, elections, and surveys deviate from the population and in such cases label ranking cannot be predicted using the same learning model. Such a deviation is addressed by extracting patterns to identify the subgroup of data for the interesting targets using subgroup discovery (*SD*) approaches (Klösgen & Zytkow 2002). These interesting targets could be combined with subgroups for all data to learn these targets using a predictive machine learning approach. However, The proposed subgroup labeling is a preference learning approach (Belfodil et al. 2019, Lucas et al. 2018, Liu et al. 2020). Subgroup discovery ranking is preference mining was proposed by Rueping (2009) using a ranking support vector machine (SVM) to rank subgroups of data according to users' interests. Many approaches were introduced for restricted and non-restricted label ranking (Carranza-Alarcon et al. 2020). Cheng et al. (2009) proposed the instance-based decision tree to rank the labels based on predictive probability models of a decision tree. Grbovic et al. (2013) combined both a decision tree and supervised clustering in two approaches for label ranking by mapping between instances and label ranking space.

2.3 Classification-Based Label Ranking

Ranking problems are first considered as a classification problem. Classification is a kind of supervised learning problem in which the target variable that one is trying to predict is discrete

2.3.1 Pairwise Label Ranking

Pairwise Label ranking is the basic deterministic approach for label ranking. This approach is called all pairs, 1-vs-1, or round-robin learning. Here, it is used as a special binarization technique, that is, to decompose a polytomous classification problem into a set of pairwise problems, thereby making multi-class problems amenable to binary classification methods. Motivated by its successful use for classification as well as its intuitive appeal from a preference and decision-making perspective, the Label Pairwise Classification (*LPC*) approach has been extended to different types of preference learning and ranking problems in recent years. The purpose of this chapter is to give an overview of existing work and recent developments in this line of research. the idea is to transform K class into involving classes $y = y_1, y_2, \cdots y_k$ into $k(k-1)/2$ binary problems

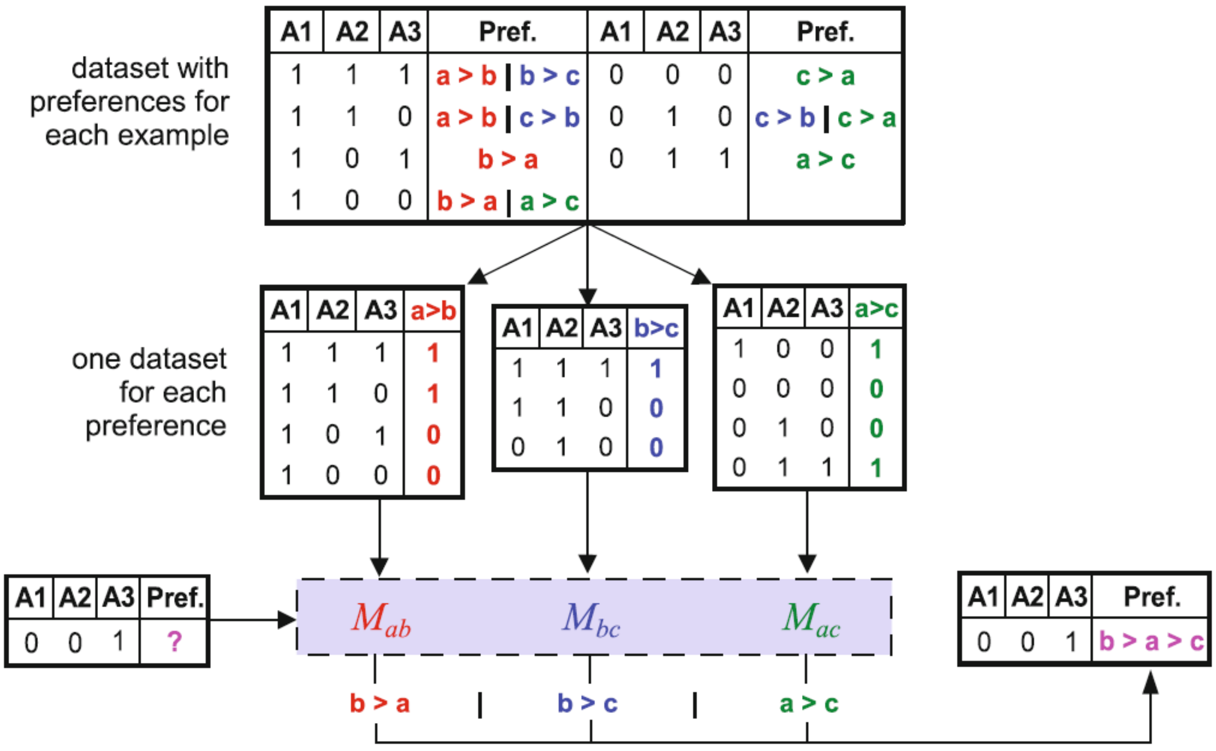
The Fig 2.4 illustrates the entire process. First, the original training set is transformed into three two-class training sets, one for each possible pair of labels, containing only those training examples for which the relation between these two labels is known. Then three binary models, M_{ab}, M_{bc}, and M_{ac} are trained. For example, the result could be simple rules like the following:

$$M_{ab} : a > b \quad if \quad A_2 = 1 \tag{2.1}$$

$$M_{bc} : b > c \quad if \quad A_3 = 1 \tag{2.2}$$

$$M_{ac} : a > c \quad if \quad A_1 = 1 \quad and \quad A_3 = 1 \tag{2.3}$$

where A1, A2, A3 are label values.


dataset with preferences for each example
one dataset for each preference
M_{ab}
M_{bc}
M_{ac}
b > a | b > c | a > c

A1	A2	A3	Pref.	A1	A2	A3	Pref.
1	1	1	a > b | b > c	0	0	0	c > a
1	1	0	a > b | c > b	0	1	0	c > b | c > a
1	0	1	b > a	0	1	1	a > c
1	0	0	b > a | a > c				

A1	A2	A3	a>b
1	1	1	1
1	1	0	1
1	0	1	0
1	0	0	0

A1	A2	A3	b>c
1	1	1	1
1	1	0	0
0	1	0	0

A1	A2	A3	a>c
1	0	0	1
0	0	0	0
0	1	0	0
0	1	1	1

A1	A2	A3	Pref.
0	0	1	?

A1	A2	A3	Pref.
0	0	1	b > a > c

Figure 2.4: *LPC* approach

2.3.2 Probability-Based label ranking

Probability approaches do not map the existence of a deterministic $x- > \omega$ mapping. Instead, every instance is associated with a probability distribution over ω. This means that, for each datapoint Xx, there exists a probability distribution $P(\mathring{u}|x)$ such that, for every $\pi \in \omega, \pi_{\pi|x}$ is the probability that $\pi|x = \pi$. (Note that if rankings are interpreted as qualitative probabilities, then $P(\mathring{u}|x)$ is a probability over probability distributions, i.e., a second-order probability.)

2.3.3 Label Ranking based on the Plackett-Luce Model

Mallows model (Mallows, 1957), a distance-based probability model belonging to the family of exponential distributions. The standard Mallows model is determined by two parameters:

$$P_{(\pi|\theta,\pi_0)} = \frac{exp(-\theta D(\pi,\pi_0))}{\phi(\theta)} \tag{2.4}$$

The ranking π is the location parameter (mode, center ranking) and θ is a spread parameter. Moreover, D is a distance measure on rankings, and the constant $\phi = \phi(\theta)$ is a normalization factor that depends on the spread.

Some of the methods mentioned above and their variants have some issues that can be broadly categorized into two types:

1) Drawbacks of classification-based models: When the ranking model is constructed using binary classification models of an associated higher dimensional label space, such a method cannot consider the interaction between labels. In this case, such rankings based on minimizing pairwise classification errors are not necessarily equivalent to maximizing the label ranking's performance considering all labels. Thus, The ranking methods do not learn the preference relation between all labels.

   The ranking methods learn both unrestricted and restricted ranking labels using the same learning approach.

2) Drawbacks of probability-based models: Some such methods use probabilistic scores for individual classes for ranking the labels. Such methods cannot capture dependencies between labels and do not take into account the distance between labels.

2.4 Classification by Ranking Reduction

Using ranking to minimize classification loss was introduced by (Kotlowski et al. 2011) by measuring the regret function of the classifier and the ranker where regret is the difference between the loss of learning compared to the best alternative method. Ailon & Mohri (2007) confirmed that it is hard to reach a faultless ranking of all preference labels. Also (Balcan et al. 2008, Abdulrahman et al. 2019) proposed different robust approaches to reduce the ranking for better classification. Brinker Brinker & Hüllermeier (2019) introduced (Label Ranking to Multi-class Classification *(LR2MC)* using decomposition and aggregation. The decomposition is achieved by dividing the ranking space into multiple partial ranking sets $L \in \mathcal{L}$. Aggregation by minimizing the sum of partial ranking sets' loss values as shown in Eq. 2.5

$$f(x) = \underset{\pi \in \mathcal{L}^*}{\operatorname{argmax}} \sum_{L \in \mathcal{L}} \ell_{01}(\pi^L, f_L(x)) \tag{2.5}$$

where

$$\ell_{01}(\rho, \rho') = \begin{cases} 0, & for \quad \rho = \rho' \\ 1, & for \quad \rho \neq \rho' \end{cases} \tag{2.6}$$

where for ρ, $\rho' \in$ L denotes the 0/1-loss for two label rankings. f_L is loss function.

2.5 Artificial Neural Network (ANN)

Artificial Neural Network (ANN) is an interconnected group of artificial neurons that uses a mathematical model for information processing based on a connection. ANN is an approach to computation and an adaptive system that changes its output based on external information that flows through the network (Kwok & Yeung 1997, Skarding et al. 2021, Chen et al. 2019, Liu et al. 2021).

2.5.1 Feedforward *(FF)* ANN

Artificial neural network wherein connections between the nodes do not form a cycle (Zell 1994) As such, it is different from its descendant: recurrent neural networks.

The feedforward neural network was the first and simplest type of artificial neural network devised (Schmidhuber 2009). In this network, the information moves in only one directionforwardfrom the input nodes, through the hidden nodes, and to the output nodes where no cycles in the network.

2.5.2 Back Propagation (*BP*)

Backpropagation (*BP*) is a simple training method that uses the evaluated gradients of a neural network to adjust the weights of the neural network. This is a form of gradient descent, as we are descending the gradients to lower values. As these weights are adjusted, the neural network should produce a more desirable output. There are mainly two types of *BP*:

- Online training implies modifying the weights after every training set element. The gradient changes are applied to the weights. Training progresses to the next training set element and calculates an update to the neural network. This training continues until every training set element has been used. One iteration, or epoch, of training, has been completed.

- Batch training also makes use of every training set element. However, the weights are not updated for every training set element. Rather, the gradients for each training set element are summed. Once every training set element has been used, the neural network weights can be updated. At this point, the iteration is considered complete (Paeedeh & Ghiasi-Shirazi 2021).

2.5.3 Activation functions

For decades convention *ANN* activation functions have been used for neural networks. A function that defines the output of a node in artificial neuronal networks according to the given input. The functions are linear and non-linear and have the common feature as it is differentiated, monotonic, and continues to produce the values in the gradient descent process, Sigmoid, Tanh except for *Relu* that is used mainly in pooling.

A multi-valued activation function has been proposed by (Aizenberg et al. 2000) using convex shape to support multi-values and complex numbers neural networks. In addition, Moraga & Heider (1999) introduced a similar function to design networks for realizing any multivalued function; however, Moraga & Heider (1999) used exponential function derivative which did not give promising results in the *PNN* implementation using the ranking objective function in *FF* and backpropagation (*BP*) steps.

2.5.4 Dropout Regularization

Dropout introduces regularization within the network, which ultimately improves generalization by randomly skipping some units or connections with a certain probability. In NNs, multiple connections that learn a non-linear relation are sometimes co-adapted, which causes overfitting. (Hinton et al. 2012b). This random dropping of some connections or units produces several thinned network architectures, and finally, one representative network is selected with small weights. This selected architecture is then considered as an approximation of all the proposed networks (Srivastava et al. 2014).

2.5.5 *ANN* for Ranking

ANN for ranking was first introduced as (RankNet) by Chris Burges (2005) to solve the problem of object ranking for sorting web documents by search engine. Rank-net uses gradient descent and probabilistic ranking cost function for each object pair. Multi-layer perceptron (*MLP-LR*) for label ranking (Ribeiro et al. 2012) employs *NN* architecture using a sigmoid activation function to calculate the error between the actual and expected values of the output labels. However, it uses a local approach to minimize the individual error per each output neuron by subtracting actual - predicted value and using Kendall error as a global approach. Both approaches do not use a ranking objective function in BP and learning steps.

2.6 Deep Learning

Deep Learning (*DL*) is the learning to extract high-level and complex abstractions of data through a hierarchical learning process. *DL* succeeded in big data fields such as image recognition and natural language processing (*NLP*) by learning the selection of the best features Sornam et al. (2017), Vedantam (2021). While many studies have successfully used deep learning for classification problems, the primary learning challenge is choosing the network architecture and structure of nodes' numbers and hidden layers. There are various network architectures used as deep learning models, as shown below.

- Autoencoder (Vahdat & Kautz 2020)
- Convolutional Deep Belief Network(*CDBN*) (Zhong & Fang 2020)
- Convolutional Neural Network (*CNN*)

- Deep Belief Network (*DBN*) (Hinton 2009)
- Deep Boltzmann Machine (*DBM*) (Salakhutdinov & Hinton 2009)
- Long short-term Memory Network (*LSTM*) (Yan et al. 2017)
- Recurrent Neural Network (*RNN*) (Kumar et al. 2020)
- Restricted Boltzmann Machine (*RBM*) (Alphonse et al. 2021)

The thesis proposes Preference Net (*PN*) as a new *ANN* different from *CNN* in terms of architecture, layers, and functionality to solve image recognition problems.

2.6.1 Convolution Neural Network (*CNN*)

Nowadays, *CNN*s are considered as the most widely used algorithms inspired by Artificial Intelligence (*AI*) techniques. innovations in *CNN* architectures into seven different categories. These seven categories are based on spatial exploitation, depth, multi-path, width, feature-map exploitation, channel boosting, and attention. Additionally, the elementary understanding of CNN components, current challenges, and applications of *CNN* are also provided.

*CNN*s have been applied to visual tasks since the late 1980s. In 1989, (Lecun et al. 1998) proposed the first multilayered *CNN* named ConvNet, whose origin was rooted in (Fukushima et al. 1983, Fukushima 1988) . Lecun et al. (1998) proposed a supervised training of ConvNet using the backpropagation algorithm in comparison to the unsupervised reinforcement learning scheme used by its predecessor Neocognitron (Linnainmaa 1970; Lecun et al. (1998). This ConvNet showed successful results for handwritten digit and zip code recognition-related problems Zhang et al. (2015). In 1998, Lecun et al. (1998) proposed an improved version of ConvNet, which was famously known as LeNet-5, and it started the use of

CNN in classifying characters in document recognition related applications. The availability of extensive training data and hardware advancements are the factors that contributed to the advancement in *CNN* research. But the main driving forces that have accelerated the research and give rise to the use of *CNN*s in image classification and recognition was brought by AlexNet, which showed exemplary performance in 2012-ILSVRC (reduced error rate from 25.8 to 16.4) as compared to conventional CV techniques Krizhevsky et al. (2012*a*). The taxonomy of *CNN* architectures details of the state-of-the-art *CNN* models, their parameters, and performance on benchmark datasets are summarized in Table 2.1 For the convolution learning step, some hyperparameters are defined in order to produce the feature map that include (Emmert-Streib et al. 2020):

- Kernel size: each kernel has a square window size. The kernel performs the convolution process with region matching
- Moving step: the number of pixels that the kernel will move to the next position
- Zero padding: The padding is the parameter used to specify the number of zeros to pad around the input border. This parameter is to preserve the dimension of an input image.

Table 2.1: Performance comparison of the recent architectures of different categories. Top 5 error rate is reported for all architectures.

Architecture Name	**Year**	**Error Rate**	**Depth**	**Category**
LeNet (Lecun et al. 1998)	1998	MNIST: 0.95	5	Spatial Exploitation
AlexNet (Krizhevsky et al. 2012*b*)	2012	ImageNet: 16.4	8	Spatial Exploitation
ZfNet (Zeiler & Fergus 2014)	2014	ImageNet: 11.7	8	Spatial Exploitation
VGG Krizhevsky et al. (2012*c*)	2014	ImageNet: 7.3	19	Spatial Exploitation
GoogLeNet (Szegedy et al. 2015)	2015	ImageNet: 6.7	22	Spatial Exploitation
Inception-V3 (Szegedy et al. 2017)	2015	ImageNet: 3.5	159	Depth + Width
Highway Networks (Srivastava et al. 2015)	2015	CIFAR-10: 7.76	19	Depth + Multi-Path
Inception-V4 (Szegedy et al. 2017)	2016	ImageNet: 4.01	70	Depth +Width
InceptionResNet (Szegedy et al. 2017)	2016	ImageNet: 3.52	572	Depth + Width + Multi-Path
ResNet (He et al. 2016*a*)	2016	ImageNet: 3.6	152	Depth + Multi-Path
DelugeNet (Kuen et al. 2017)	2016	CIFAR-10: 3.76	146	Multi-path
FractalNet (Larsson et al. 2017)	2016	CIFAR-10: 7.27	20	Multi-Path
WideResNet (Zagoruyko & Komodakis 2016*a*)	2016	CIFAR-10: 3.89	5	Width
Xception Chollet (2017)	2017	ImageNet: 0.055	28	Width
Residual Attention Wang et al. (2017)	2017	CIFAR-10: 3.90	126	Attention
ResNeXt (Xie et al. 2020)	2017	CIFAR-10: 3.58	452	Width
Squeeze/Excitation Hu et al. (2020)	2017	ImageNet: 2.3	29	Feature-Map Exploitation
DenseNet (Huang et al. 2018)	2017	CIFAR-10+: 3.46	152	Multi-Path
PolyNet (Zhang et al. 2017)	2017	ImageNet:Single:4.25	190	Width
PyramidalNet (Han et al. 2017)	2017	ImageNet: 4.7	190	Width
ResNeXt101(32x4d) (Woo et al. 2018)	2018	ImageNet: 5.59	250	Attention
Concurrent Spatial (Roy et al. 2018)	2018	MALC: 0.12	250	Attention
Channel Boosted (Chouhan et al. 2019)	2018	-	5	Channel Boosting
Squeeze Excitation (Hu et al. 2018)	2018	CIFAR-10: 3.58	152	Feature-Map Exploitation

2.6.2 Pooling Layer

Pooling layers aim at reducing the dimension of the input with some pre-specified pooling methods. There are many types of pooling methods, e.g., averaging-pooling, min-pooling, and some advanced pooling methods, such as fractional max-pooling and stochastic pooling. The most common pooling method used is max-pooling, as it has been shown to be better in dealing with images by capturing (Scherer et al. 2010).

2.7 Human Visualization and Frame Variation

Visual saliency detection using the Markov chain model is one of the approaches that simulates the human visual system by highlighting the most important area in an image and calculating superpixels as absorbing nodes. However, this approach needs a saliency optimization on the results and have calculation cost (Jiang et al. 2020),(Gupta et al. 2020).

Particle Swarm Optimization (PSO) in movement detection is based on the concept of variation and inter-frame difference for feature selection. PSO algorithms are mainly used in human motion detection in sports, and it is used based on probabilistic optimization algorithm (Lei et al. 2021) and CNN (Zhang 2022).

2.8 Subgroup Discovery (*SD*)

Subgroup Discovery (*SD*) is a descriptive induction data mining technique that discovers interesting associations among different variables with respect to a property of interest. It was first introduced by Klösgen (1996) and Wrobel (1997). Existing *SD* methods utilize different strategies for searching, pruning, and ranking subgroups. Selecting features of a *SD* algorithm is crucial for generating quality subgroups. *SD* aims to extract relations among different variables with respect to the property of interest. These relations are represented in the form of rules represented as follows:

$$\text{Condition} \rightarrow \text{Target}$$

2.8.1 *SD* Methodology

subgroup discovery algorithm consists of three major phases, candidate subgroup generation, pruning, and post-processing as illustrated in Fig 2.5.

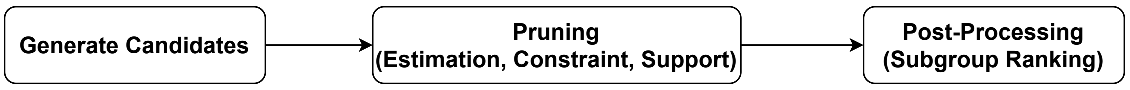

Generate Candidates
Pruning (Estimation, Constraint, Support)
Post-Processing (Subgroup Ranking)

Figure 2.5: SD Methodology

2.8.1.1 Generating Candidates

Each *SD* algorithm uses a specific strategy to search for the candidate subgroups. The search strategy is essential for extracting subgroups as the volume of search space is exponential with respect to the number of attributes and their values. The search space is traversed by starting with simple descriptions and processing them in a more general to the specific manner by adding up more attribute-value pairs. Different search strategies have been employed so far for subgroup discovery; among them, the most widely used strategies are exhaustive search, beam search, and genetic algorithm (GA) based search.

2.8.1.2 Pruning

The pruning scheme selects only the significant candidates. Different methods use a number of pruning strategies. The major types include minimum support or coverage pruning. Several popular methods implement this pruning technique (Gamberger & Lavrac 2002),(Kavsek et al. 2003).

2.8.1.3 Post-Processing

The final phase of the subgroup discovery algorithm implements a quality measure for the purpose of ranking subgroups. These measures are very vital for evaluating subgroups as the interest attained directly relies on them.

Preference mining (*PM*) is an extended domain of *PL* and *SD*, which aims to discover the local patterns and deviations of subsets of data (Holland et al. 2003, de S*á* & Duivesteijn 2018). Using a conjoint model based on the fusion of a different group of data sensors has been introduced in emotion recognition by Pandeya, Y. Pandeya et al. (2021). It uses deep learning to classify emotions Pandeya et al. (2021) from audio and video information. Rueping (2009) proposed subgroup ranking using the support vector machine (SVM) to rank subgroups concerning the user's concept of interestingness.

2.9 Potential Applications

The thesis proposes three neural network types for preference learning.

2.9.1 *PNN* Applications

PNN is proposed for label ranking, which can be part of decision making and recommending systems.

2.9.2 *PN* Applications

PN is proposed for image recognition applications, including face recognition, object recognition, and remote sensing.

2.9.3 Subgroup Preference Neural Network*SGPNN* Applications

SGPNN could be used in many potential applications, i.e., brain-computer interface (*BCI*) applications where EEG data may have ambiguity, be complicated, and unbalanced. Another medical application is where data fusion is collected from different sensors, i.e., the study of human emotions recognition. *SGPNN* could be part of an expert system to build an accumulated learning model for judgment, elections, medical diagnosing from different conjoint historical data.

CHAPTER 3

Preference Neural Network

3.1 Introduction

The proposed Preference Neural Network (*PNN*) is based on an initial experiment to implement a computationally efficient label ranker network based on the Kendall τ error function and *sigmoid* activation function using simple structure as illustrated in section Fig. 3.1. The ranker network is a fully connected, three-layer net. The input represents one instance of data with three inputs, and there are six neurons in the hidden layer and three output neurons representing the labels' index. Each neuron represents the ranking value. A small toy data set is used in this experiment. The ranker uses *RMS* gradient descent as an error function to measure the difference between the predicted and actual ranking values. The ranker has Kendall τ as a stopping criterion. The same *ANN* structure, number of neurons and learning rate using *SS* activation function , and *spearman* error function and gradient ascent of ρ will be discussed in section IV. The ranking convergence reaches to $\tau \simeq 1$ after 160 epochs using the *Sigmoid* function Elgharabawy (2020*c*). The *sigmoid* and *ReLU* shapes have a slightly high rate of change in y, and it produces a more extensive output range of data. Therefore, we consider ranking performance as one of the disadvantages of *sigmoid* function in the ranker network.

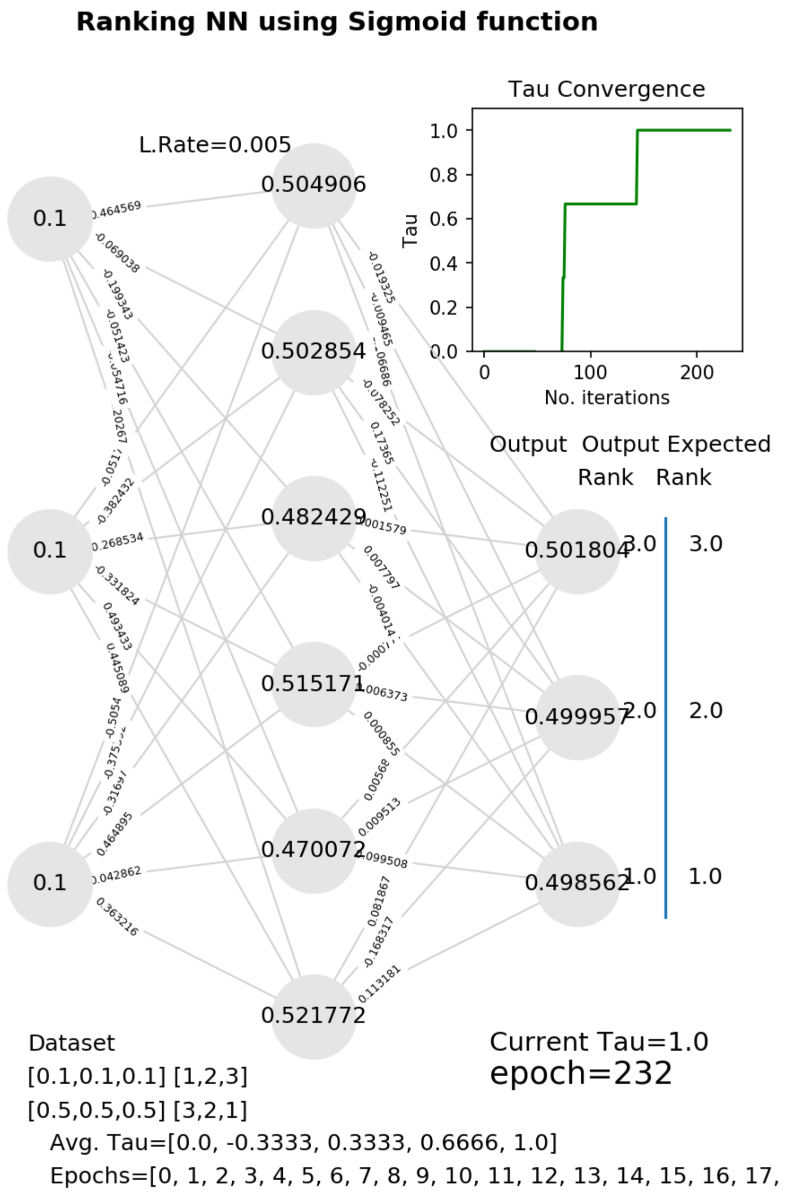


Figure 3.1: Ranker *NN* Sample output image of video file demonstrates the evaluation of *NN* using Sigmoid Activation for ranking.

The ranker network has two main problems.

1) The ranker uses two different error functions, RMS for learning and Kendall τ for stopping criteria. Kendall τ is not used for learning because it is not continuous or differentiable. Both functions are not consistent as stopping criteria measure the relative ranking, and *RMS* does not, which may lead to incorrect stopping criteria. Enhancing the *RMS* may not also increase the error performance, as illustrated in Fig. 4.7 in a comparison between the ranker network. evaluation using both τ and *RMS* and *PNN* ranking evaluation using ρ and *RMS*.

2) The convergence performance takes numbers of iterations to reach the ranking $\tau \simeq 1$ based on the shape of *sigmoid* or *Relu* functions and learning rate as shown in the experiment video link (Elgharabawy 2020*c*) due to the slope shape between -1 or 0 and 1. The prediction probability is almost equal to the values from -1 or 0 to 1.

3.2 Problem Formulation

For multi-class and multi-label problems, learning the data's preference relation predicts the classification and label ranking. i.e. data instance $\mathcal{D} \in \{x_1, x_2, \ldots, x_n\}$. the output labels are predicted as ranked set labels that have preference relations $\mathcal{L} = \{\lambda_{y_1}, \ldots, \lambda_{y_n}\}$. *PNN* creates a model that learns from an input set of ranked data to predict a set of new ranked data. The following section presents the initial experiment to rank labels using the usual network structure.

3.3 *PNN* Components

3.3.1 Activation Functions

The usual *ANN* activation functions have a binary output or range of values based on a threshold. However, these functions do not produce multiple deterministic values on the y-axis. This chapter proposes new functions to slow the differential rate around ranking values on the y-axis to solve ranking instability. The proposed functions are designed to be non-linear, monotonic, continuous, and differentiable using a polynomial of the *tanh* function. The step width maintains the stability of the ranking during the forward and backward processes. Moraga & Heider (1999) introduced a similar multi-valued function. However, the proposed exponential derivative was not applied to an *ANN* implementation. Moraga's exponential function is geometrically similar to the step function in Bologna (2000). However, the newly proposed functions consist of *tanh* polynomial instead of exponential due to the difficulty in implementation. The new functions detect consecutive integer values, and the transition from low to high rank (or vice versa) is fast and does not interfere with threshold detection.

3.3.1.1 Positive Smooth Staircase (*PSS*)

As a non-linear and monotonic activation function, positive smooth staircase (PSS) is represented as a bounded smooth staircase function start from x=0 to ∞. Thus, it is not geometrically symmetrical around the y-axis as shown in Fig.3.2. *PSS* is a polynomial of multiple *tanh* functions and is therefore differentiable and continuous. The function squashes the output neurons values during the *FF* into finite multiple integer values. These values represents the preference values from {*0* to *n*} where 0 represent the incomparable relation $\perp$ and values from 1 to n represent the label ranking. The activation function is given in Eq. 3.1. *PSS* is scaled by increasing the step width w.

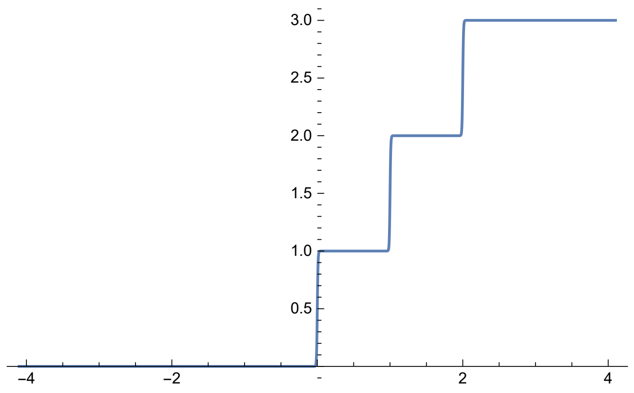


Figure 3.2: *PSS* activation function where $n = 3$ and step width $w = 1$

$$y = \sum_{i=0}^{n} \left(-\frac{1}{2} \tanh(-100(x - wi)) \right) + \frac{n}{2} \tag{3.1}$$

where n is number of output labels, w is the step width.

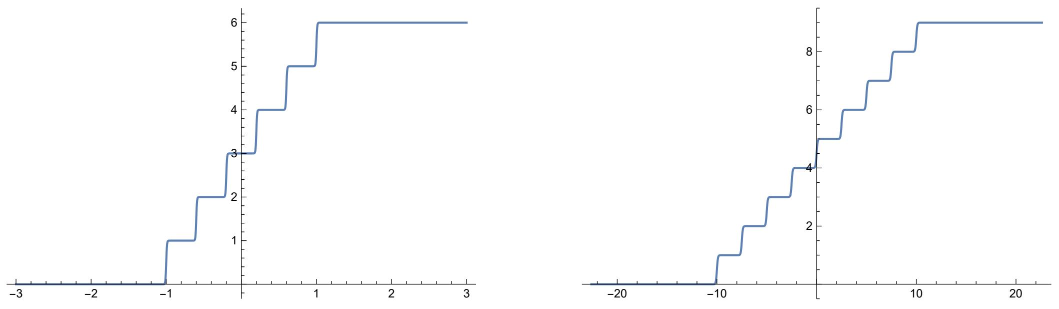
3.3.1.2 Smooth Staircase (*SS*)

The proposed *SS* represents a staircase similar to (*PSS*). However, *SS* has a variable boundary value used as a hyperparameter in the learning process. The derivative of the activation function is discussed in section III and the performance comparison between *SS* and *PSS* is mentioned in section v.

The activation function is given in Eq. 3.2.

$$y = -\frac{1}{2} \left(\sum_{i=0}^{n} \tanh(\frac{-100}{b} x + c(1 - \frac{2i}{n-1})) \right) + \frac{n}{2} \tag{3.2}$$

where $c = 100$, n = number of ranked labels, b is the boundary value, and (*SS*) lies between $-b$ and b. The (*SS*) function has the shape of smooth stair steps, where each step represents an integer number of label ranking on the y-axis from *0* to ∞ as shown in Fig. 1, The *SS* step is not flat,



(a) **(b)**

Figure 3.3: *SS* activation function where $n = 6$ and 9 and boundary $b = 1$ and 10 in (a) and (b) respectively.

but it has a differential slope. The function boundary value on x-axis is from -b to b Therefore, input values must be scaled from -b to b. The step width is 1 when n$\simeq 2b$. The convergence rate is based on the step width. However, it may take less time to converge based on network hyperparameters. Fig. 3.3 (a) and (b) shows the activation functions to rank 6 and 9 labels, respectively. The *SS* is scaled by increasing the boundary value b

3.3.2 Ranking Loss Function

Two main error functions have been used for label ranking, namely; Kendall τ Kendall (1948) and *spearman* ρ Spearman (1961). However, the Kendall τ function lacks continuity and differentiability. Therefore, the *spearman* ρ correlation coefficient is used to measure the ranking between output labels. *spearman* ρ error derivative is used as a gradient ascent process for *BP*, and correlation is used as a ranking evaluation function for convergence stopping criteria. τ_{Avg} is the average τ per label divided by the number of instances m, as shown in line 8 of Algorithm 1. *spearman* ρ measures the relative ranking correlation between actual and expected values instead of using the absolute difference of root means square error (*RMS*) because gradient descent of *RMS* may not reduce the ranking error. For example, $\pi_1 = (1, 2.1, 2.2)$ and $\pi_2 = (1, 2.2, 2.1)$, have a low *RMS* $= 0.081$ but a low ranking correlation $\rho = 0.5$ and $\tau = 0.3$.

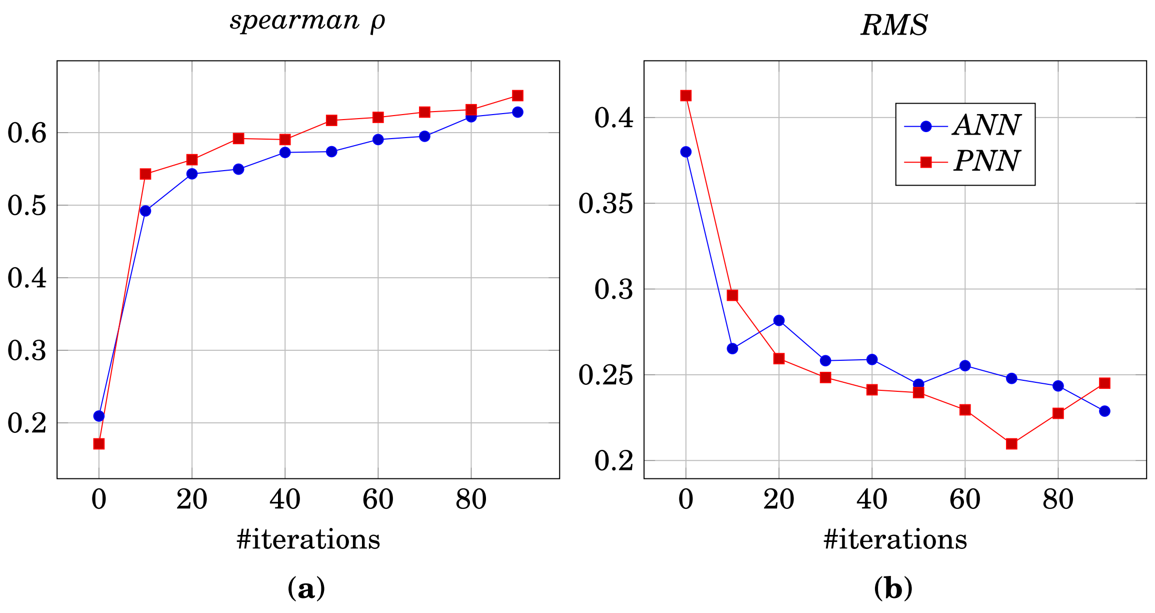


Figure 3.4: Ranker network and *PNN* evaluation in terms of *RMS* and *spearman* correlation error functions using iris DS in (a) and (b) respectively.

Fig 3.4 shows the comparison between the initial ranker network and *PNN*; the ranker network uses Kendall τ in which has lower performance as a stopping criterion compared to *PNN spearman* because the stopping criteria are based on the *RMS* per iteration; however, *PNN* uses *spearman* for both ranking step and stopping criteria.

The *spearman* error function is represented by Eq.3.3

$$\rho = 1 - \frac{6\sum_{i=1}^{m}(y_i - yt_i)^2}{m(m^2-1)} \tag{3.3}$$

where y_i, yt_i, i and m represent rank output value, expected rank value, label index and number of instances, respectively.

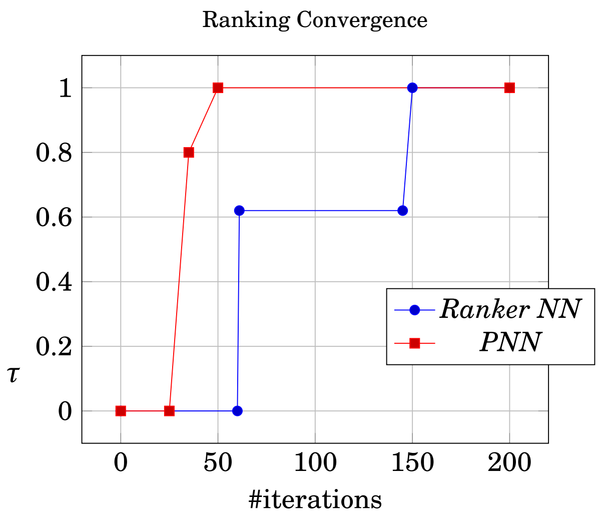


Figure 3.5: The structure used in both ranker *ANN* and *PNN* where $\varphi_{n=3}$, $f_{in} = 3$ and $\lambda_{out} = 3$, per $\langle x_1, \pi_1 \rangle$, $\mathcal{L} \in \{\lambda_a, \lambda_b, \lambda_c\}$ where $\pi_1 = \{1,2,3\}$. and comparison of the convergence for both *NN*'s. The demo video of convergence of two *NN* in the link Elgharabawy (2020*c*).

3.3.3 *PNN* Structure

3.3.3.1 One middle layer

The *ANN* has multiple hidden layers. However, we propose *PNN* with a single middle layer instead of multi-hidden layers because ranking performance is not enhanced by increasing the number of hidden layers due to fixed multi-valued neuron output, as shown in Fig. 3.6; *SS* function is experimented using Seven benchmark data sets Cheng et al. (2009) by changing the number of hidden layers with the following hyperparameters; learning rate (l.r.)=0.05, and each layer has neuron $i = 100$ and $b = 10$). We found that by increasing the number of hidden layers, the ranking performance decreases, and more iteration is required to reach $\rho \simeq 1$. The low performance because of the shape of *SS* produces multiple deterministic values, which decrease the arbitrarily complex decision regions and degrees of freedom per extra hidden layer.

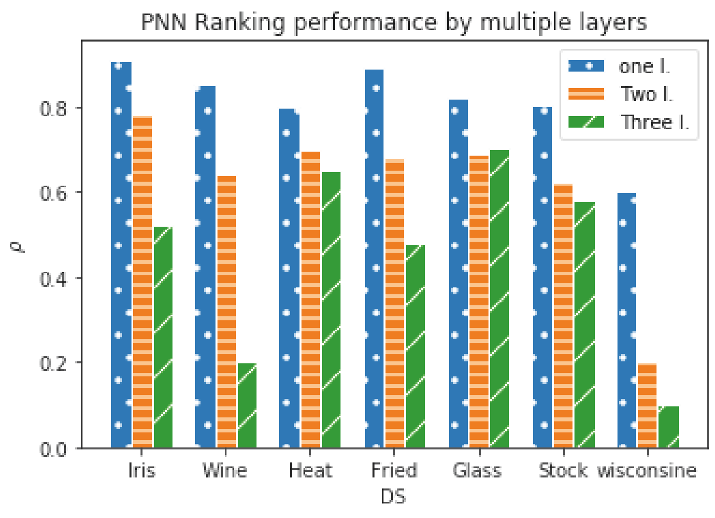


Figure 3.6: Multiple layer label ranking comparison of benchmark data sets Cheng et al. (2009) results using the *PNN* and *SS* functions after 100 epochs and learning rate = 0.007.

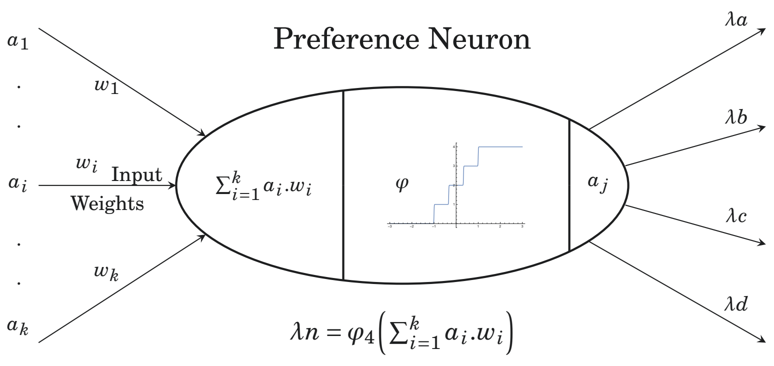


Figure 3.7: The structure of preference neuron where $\varphi_{n=4}$.

3.3.3.2 Using Sigmoid/Relu with SS

Multiple hidden layer is experimented using Relu and Sigmoid functions and the output layer has SS function, However, the results didnt outperform the middle layer approach with SS, and it increased the number of iteration to reach the same result, Thus using *SS* with other activation function does not add a value in terms of accuracy or performance.

3.3.3.3 Preference Neuron

Preference Neuron is a multi-valued neuron use a *PSS* or *SS* as an activation function. Each function has a single output; however, *PN* output is graphically drawn by n number of arrows links that represent the multi-deterministic values. The *PN* in the middle layer connects to only n output neurons $stp = n+1$; where stp is the number of *SS* steps. The *PN* in output layer represents the preference value. The middle and output *PN*s produce a preference value from 0 to ∞ as illustrated in Fig. 3.7.

The *PNN* is fully connected to multiple-valued neurons and a single-hidden layer *ANN*. The input layer represents the number of features per data instance. The hidden neurons are equal to or greater than the number of output neurons, $H_n \geq \mathcal{L}_n$, to reach error convergence after a finite number of iterations. The output layer represents the label indexes as neurons, where the labels are displayed in a fixed order, as shown in Fig. 3.8.

The *ANN* is scaled up by increasing the hidden layers and neurons; however, increasing the hidden layers in *PNN* does not enhance the ranking correlation because it does not arbitrarily increase complex decision regions and degrees of freedom to solve more complex ranking problems. This limitation is due to the multi-semi discrete-valued activation function, which limits the output data variation. Therefore, instead of increasing the hidden layer, *PNN* is scaling up by increasing the number of neurons in the middle layer and scaling input data boundary value and increasing the *PSS* step width and *SS* boundaries which are equal to the input data scaling value, which leads to increased data separability.

PNN outperform initial ranking ranker network by 24 epochs comparing to the initial ranker that achieve $\rho \simeq 1$ after 200 iterations, The video link demonstrates the ranking convergence as shown in Fig. 3.5

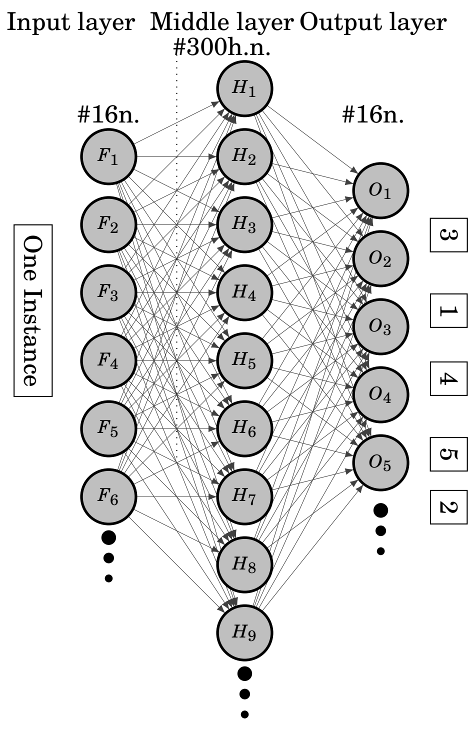


Figure 3.8: *PNN* where $\varphi_{n=16}$, $f_{in} = 16$ and $\lambda_{out} = 16$, per $\langle x_1, \pi_1 \rangle$, $\mathcal{L} \in \{\lambda_a, \lambda_b, \lambda_c, \lambda_d\}$ where $\pi_1 = \{1, 2, 3, 4, \ldots, 16\}$.

and video demo Elgharabawy (2020*c*), and a summary of the two networks are presented in Table 3.1.

The output labels represent the ranking values. The differential *PSS* and *SS* functions to accelerate the convergence after a few iterations due to the staircase shape, which achieves stability in learning. *PNN* simplifies the calculation of *FF* and *BP*, and updates weights into two steps due to single middle layer architecture. Therefore, the batch weight updating technique is not used in *PNN*, and pattern update is used in one step. The network bias is low due to the limited *PN* output variation, so it is not calculated. Each neuron uses the activation function in *FF* step, and calculates the preference number from 1 to n, where n is the number of label classes. During *BP*. The processes of *FF* and *BP* are executed in two steps until $\rho_{Avg} \simeq 1$ or the number of iterations reaches (10^6) as mentioned in the algorithm section.

The *SS* step width decreases by increasing the number of labels; thus, we increase function boundary b to increase the step width to $\simeq 1$ to make the ranking convergence; In addition, a few complex data sets may need more data separability to enhance the ranking. Therefore, we use the b value as a hyperparameter to keep the stair width $>= 1$ and normalize input data from $-b$ to b. The boundaries are important for non-normalized data and the function is configured by the boundaries values. and the step width is the merit of *PS* function because the function is configured by the step width.

Table 3.1 shows a brief comparison between Ranker *ANN* and *PNN*.

Table 3.1: *ANN* types used in initial experiment.

Type	**Ranker *ANN***	***PNN***
activation fun.	*ReLU, Sigmoid*	PSS, SS
gradient	descent	ascent
objective fun.	*rms*	ρ
stopping criteria.	τ	ρ

3.3.4 Baseline Algorithm

Algorithm 1 represents the three functions of the network learning process; feed-forward (*FF*), *BP*, and updating weights (*UW*). Algorithm 1 represents the learning flow of *PN*. Algorithm 2 represents the simplified BP function in two steps.

3.4 Network Evaluation

This section evaluates the *PNN* against different activation functions and architectures. All weights are initialized = 0 to compare activation functions and A and B have the same initialized random weights to evaluate the structure.

```
Algorithm 1: PNN learning flow
   Data: $\mathcal{D} \in \{x_1, x_2, \ldots, x_d\}$
   Result: $\pi \in \{\lambda_{y_1}, \ldots, \lambda_{y_n}\}$
 1 Randomly initialize weights $\omega_{i,j} \in \{-0.05, 0.05\}$
 2 repeat
 3     forall $\langle x_i, \pi_i \rangle \in \mathcal{D}$ do
 4         $a_i|_{l-1} = \sum_{i=1}^{m} \varphi(a_i \cdot \omega_i)|_n$ // FF
 5         PNN BP()
 6         $\omega_{i new} = \omega_{i old} - \eta \cdot \delta_i$ //UW
 7 until $\rho_{Avg}$ = 1 or #iterations $\geq 10^6$;
```

```
Algorithm 2: PNN BP
 8 Step 1: for each $pn_i$ in Output layer do
 9     $Err_i = \rho = -6 \cdot \frac{(2yt_i - y_i)}{m(m^2-1)}$ //spearman error
10     $\delta_i = Err \cdot \varphi\prime$
11 Step 2: for each $pn_i$ in middle layer do
12     $Err_i = \sum_{k=0}^{m} \omega_k \cdot \delta_k$
13     $\delta_i = Err \cdot \varphi\prime$
```

3.4.1 Activation Functions Evaluation

PNN is tested on iris and stock data sets using four activation functions. *SS*, *PSS*, *ReLU*, *sigmoid*, and *tanh*. *PNN* has one middle layer and the number of hidden neuron (h.n.) is 50, while l.r.= 0.05. Fig. 3.9 shows the convergence after 500 iterations using four activation functions (*SS*, *PSS*, *sigmoid*, *ReLU* and *tanh*) respectively. We noticed that *PSS* and *SS* has a stable rate of ranking convergence comparing to *sigmoid*, *tanh*, and *ReLU*. This stability is due to the stairstep width, which leads each point to reach the correct ranking during *FF* and *BP* in fewer epochs.

Also *SS* and sigmoid functions are experimented according to convergence rate and number of epoch as shown in Fig 3.10

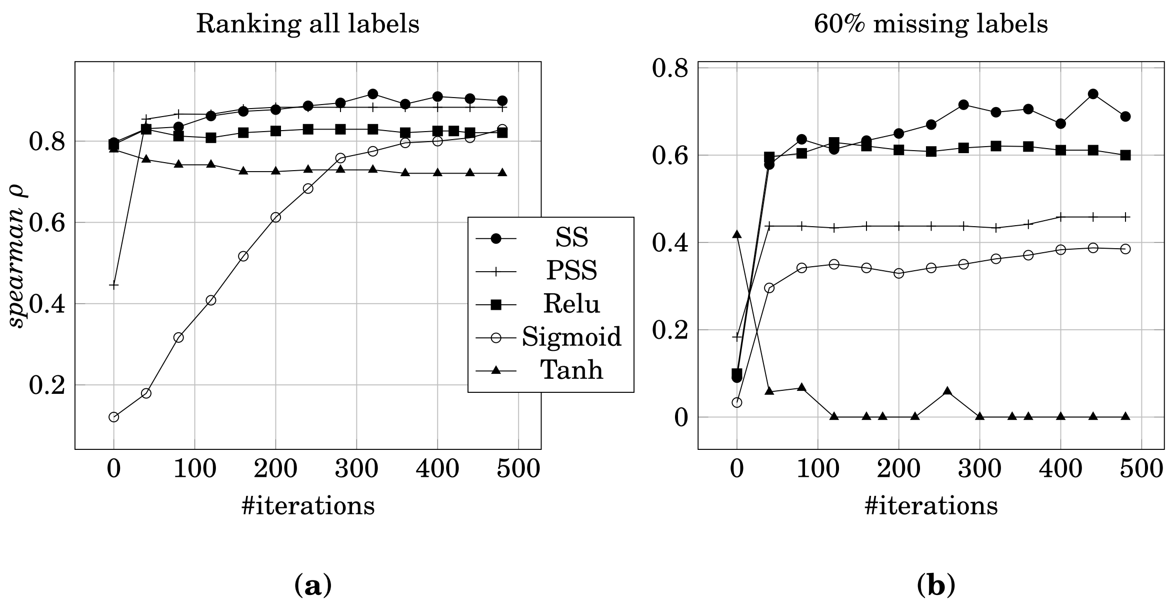


Figure 3.9: *PNN* activation function comparison using complete labels and 60% missing labels in (a) and (b), respectively.

3.4.1.1 *PSS* and *SS* Evaluation

As shown in Fig 3.9, *PSS* reaches convergence and remains stable for a long number of iterations compared to *SS*. However, *SS* has better ρ than *PSS*. This good performance of *SS* is due to the reason:

- The symmetry of *SS* function on the x axis. The *SS* shape handles both positive and negative normalized data. It reduces the number of iterations to reach the correct ranking values.

To have the same performance for *SS* and *PSS*, the input data should be scaled from 0 to step width the number of steps and from -b to b for *PSS* and *SS* respectively.

3.4.1.2 Missing Labels Evaluation

Activation functions are evaluated by removing a random number of labels per instance. *PNN* marked the missing label as -1; *PNN* neglects er-

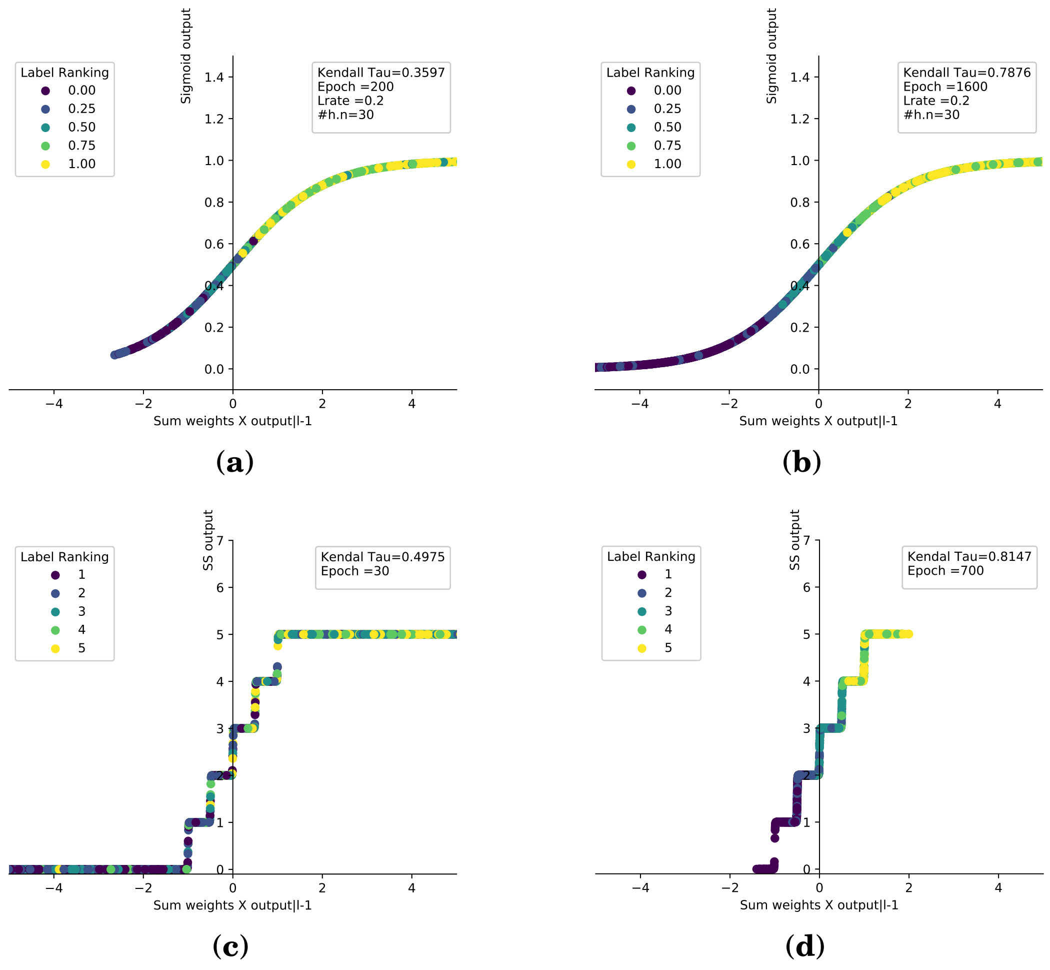


Figure 3.10: The graphical comparison between convergence of *Sigmoid* and *SS* functions to rank stock dataset, (**a**) *Sigmoid* has $\tau = 0.3597$ and epoch = 200. (**b**) *Sigmoid* has $\tau = 0.7876$ and epoch = 1600. (**c**) *SS* has $\tau = 0.4975$ and epoch = 30. (**d**) *SS* has $\tau = 0.8147$ and epoch = 700.

ror calculation during *BP*, $\delta = 0$. Thus, the missing label weights remain constants per learning iteration. The missing label approach is applied to the data set by using 20% and 60% of the training data. The ranking performance decreases when the number of missing labels increases. However, *SS* and *PSS* have more stable convergence than other functions. This evaluation is performed on the iris data set, as shown in Fig. 3.9.

3.4.1.3 Statistical Test

The *PNN* results were evaluated using receiver operating characteristic (*ROC*) curves. The true positive and negative for each rank are evaluated per label as shown in Fig. 3.11. The confusion matrix on wine and glass DS are shown in Fig. 3.12 where τ = 0.947, 0.84, Accuracy = 0.935 and 0.8 in (a) and (b) respectively.

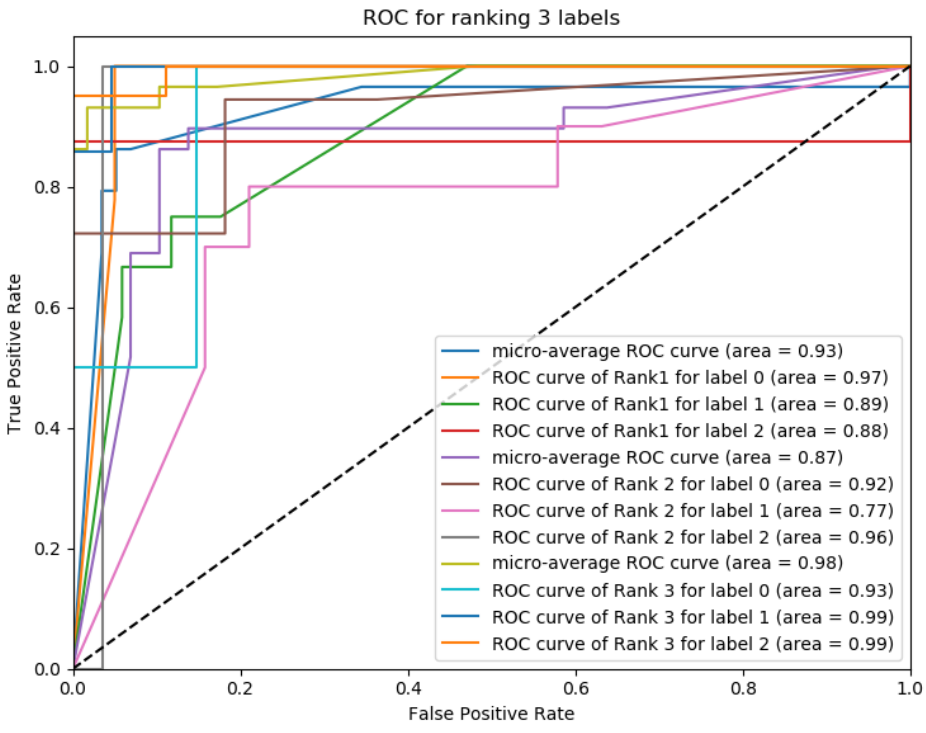
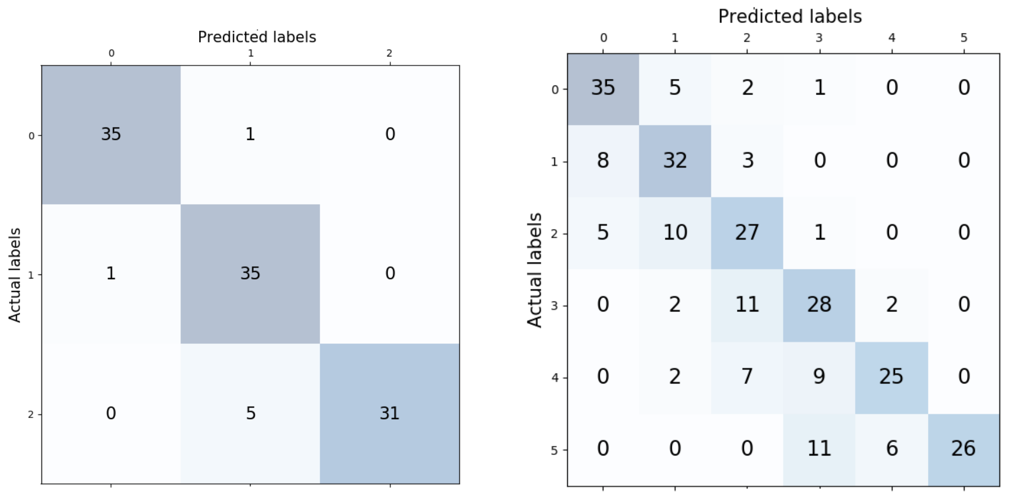


Figure 3.11: ROC of three label ranking on the wine data set using *PNN* h.n=100 and 50 epochs

3.4.1.4 *PNN* Dropout

Dropout is applied as a regularization approach to enhance the *PNN* ranking stability by reducing over-fitting. We drop out the weights that have a probability of less than 0.5. these dropped weights are removed



(a) **(b)**

Figure 3.12: The confusion matrix of testing data for wine, glass DS where τ = 0.947, 0.84, Accuracy = 0.935 and 0.8 in (a) and (b) respectively.

from FF, BP, and updating weight steps. The comparison between dropout and non-dropout of *PNN* are shown in Fig. 3.13. The gap between the training model and ten-fold cross-validation curves has been reduced using dropout regularization of type *A* using hyperparameters (l.r.=0.05, h.n.=100) on the iris data set. The dropout technique is used with all the data ranking results in the next section.

The following section is the evaluation of ranking experiments using label benchmark data sets.

3.5 Experiments

This section describes the classification and label ranking benchmark data sets, and the results using *PNN* and a comparison with existing classification and ranking methods.

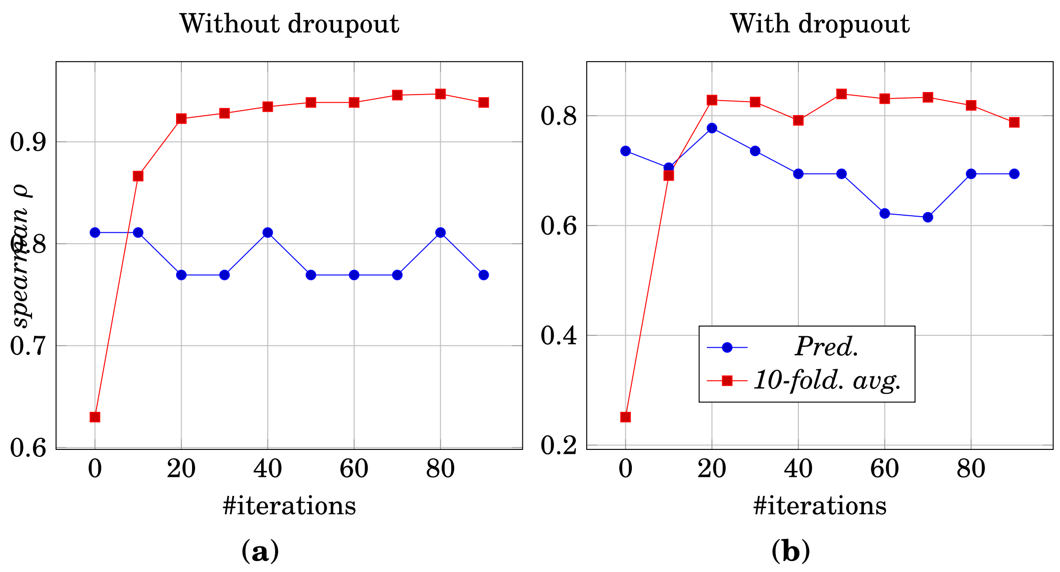


Figure 3.13: Training and validation performance without and with dropout regularization approach in (a) and (b) respectively.

3.5.1 Data sets

3.5.1.1 Label Ranking Data

PNN is experimented with using three different types of benchmark data sets to evaluate the multi-label ranking performance. The first type of data set focuses on exceptions preference mining (de S*á* & Duivesteijn 2018), and the 'algae' data set is the first type that highlights the indifference preferences problem, where labels have repeated preference value (Cláudio 2018). German elections 2005, 2009, and modified sushi are considered new and restricted preference data sets. The second type is real-world data related to biological science (E. Hüllermeier 2008). The third type of data set is semi-synthetic (*SS*) taken from the *KEBI* Data Repository at the Philipps University of Marburg (Cheng et al. 2009). All data sets do not have ranking ground truth, and all labels have a continuous permutation space of relations between labels. Table 3.3 summarizes the main characteristics of the data sets.

Table 3.2: Benchmark data sets for label ranking; preference mining Cláudio (2018), real-world data sets Grbovic et al. (2013) and semi-synthetic (*s-s*) Cheng et al. (2009).

Type	DS	Cat.	#Inst.	#Attr.	#lbl.
Mining	algae	chemical stat.	317	11	7
	german.2005	user pref.	413	31	5
	german.2009	user pref.	413	31	5
	sushi	user pref.	5000	13	7
	top7movies	user pref.	602	7	7
Real	cold	biology	2,465	24	4
	diau	biology	2,465	24	7
	dtt	biology	2,465	24	4
	heat	biology	2,465	24	6
	spo	biology	2,465	24	11
Semi-Synthesized	authorship	A	841	70	4
	bodyfat	B	252	7	7
	calhousing	B	20,640	4	4
	cpu-small	B	8192	6	5
	elevators	B	16,599	9	9
	fried	B	40,769	9	5
	glass	A	214	9	6
	housing	B	506	6	6
	iris	A	150	4	3
	pendigits	A	10,992	16	10
	segment	A	2310	18	7
	stock	B	950	5	5
	vehicle	A	846	18	4
	vowel	A	528	10	11
	wine	A	178	13	3
	wisconsin	B	194	16	16

Table 3.3: Benchmark data sets for label ranking; preference mining Cláudio (2018), semi-synthetic (*SS*) Cheng et al. (2009) and real-world data sets

3.5.2 Results

3.5.2.1 Label Ranking Results

PNN is evaluated by restricted and non-restricted label ranking data sets. The results are derived using *spearman* ρ and converted to *Kendall* τ coefficient for comparison with other approaches. For data validation, we used 10-fold cross-validation. To avoid the over-fitting problem, We varied these hyperparameters, i.e. l.r.= (0.005, 0.05, 0.1) hidden neuron = no.inputs (5, 10, 50, 100, 200) neurons and scaling boundaries from 1 to 250) are chosen within each cross-validation fold by using the best l.r. on each fold and calculating the average τ of ten folds. Grid searching is used to obtain the best hyperparameter values. For type B, we use three output groups with l.r.=0.001 and $w_b = 0.01$.

3.5.2.2 Benchmark Results

Table 3.4 summarizes *PNN* ranking performance over 16 strict label ranking data sets by using l.r. and m.n. The results are compared with four methods for label ranking; supervised clustering (Grbovic et al. 2013), supervised decision tree (Cheng et al. 2009), *MLP* label ranking (Ribeiro et al. 2012), and label ranking tree forest (*LRT*) (de Sá et al. 2017). Each method's results are generated by ten-fold cross-validation. The comparison selects only the best approach for each method.

During the experiment, it was found that the ranking performance increased by increasing the number of central neurons. All the results are held using a single hidden layer with various hidden neurons (50 to 300) and *SS* activation function. The Kendall τ error converges and reaches close to 1 after 2000 iterations, as shown in Fig. 3.14.

Table 3.4 compares *PNN* with the similar approaches used for label ranking. These approaches include; Decision trees (Grbovic et al. 2013), *MLP-LR* (Ribeiro et al. 2012) and label ranking trees forest *LRT* (de Sá et al. 2017). In this comparison, we choose the method that has the best results for each approach.

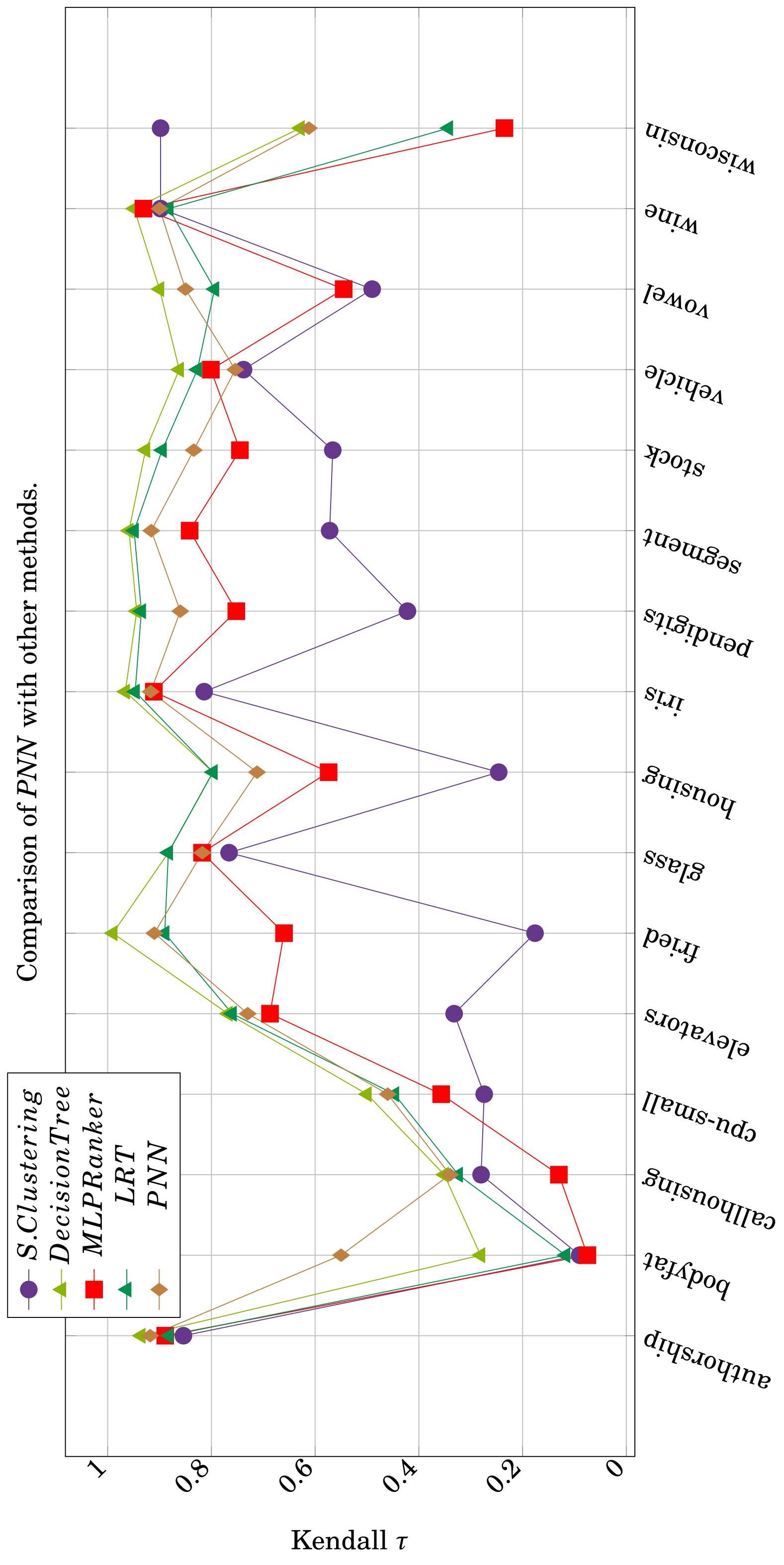


Figure 3.14: Ranking performance comparison of *PNN* with other approaches.

Table 3.4: *PNN* performance comparison with various approaches: supervised clustering (Grbovic et al. 2013), supervised decision tree (Cheng et al. 2009), *MLP* label ranking (Ribeiro et al. 2012), Kernel Ridge Regression (K.R.R)(Korba et al. 2018) and label ranking tree forest (*LRT*) (de Sá et al. 2017)

DS	**S.Clust.**	DT	MLP-LR	LRT	**K.R.R**	PNN
authorship	0.854	0.936(IBLR)	0.889(LA)	0.882	0.93	0.918
bodyfat	0.09	0.281(CC)	0.075(CA)	0.117	-	0.5591
calhousing	0.28	0.351(IBLR)	0.130(SSGA)	0.324	-	0.34
cpu-small	0.274	0.50(IBLR)	0.357(CA)	0.447	-	0.46
elevators	0.332	0.768(CC)	0.687(LA)	0.760	-	0.73
fried	0.176	0.99(CC)	0.660(CA)	0.890	-	0.91
glass	0.766	0.883(LRT)	0.818(LA)	0.883	0.83	0.8175
housing	0.246	0.797(LRT)	0.574(CA)	0.797	-	0.712
iris	0.814	0.966(IBLR)	0.911(LA)	0.947	0.97	0.917
pendigits	0.422	0.944(IBLR)	0.752(CA)	0.935	-	0.86
segment	0.572	0.959(IBLR)	0.842(CA)	0.949	-	0.916
stock	0.566	0.927(IBLR)	0.745(CA)	0.895	-	0.834
vehicle	0.738	0.862(IBLR)	0.801(LA)	0.827	0.89	0.754
vowel	0.49	0.90(IBLR)	0.545(CA)	0.794	0.88	0.85
wine	0.898	0.949(IBLR)	0.931(LA)	0.882	0.89	0.90
wisconsin	0.09	0.629(CC)	0.235(CA)	0.343	-	0.612
Average	0.475	0.79	0.621	0.730	N.A.	0.755

3.5.2.3 Preference Mining Results

The ranking performance of the new preference mining data set is represented in table 3.5. Two hundred fifty hidden neurons are used to enhance the ranking performance of the algae data set's repeated label values. However, restricted labels ranking data sets of the same type, i.e., (German elections and sushi), did not require a high number of hidden neurons and incurred less computational cost.

Experiments on the biological real-world data set were conducted using supervised clustering (*SC*) (Grbovic et al. 2013), Table 3.6 presents the comparison between *PNN* and supervised clustering on biological real-world data in terms of $Loss_{LR}$ as given in Eq. 3.4.

Table 3.5: Preference mining ranking performance in terms of the Kendall τ coefficient and learning step and number of hidden neurons.

Preference Mining Data			
DS	**A**vg.τ	**l**.step	#m.n.
algae	0.751	0.005	100
german2005	0.89	0.005	20
german2009	0.78	0.005	20
sushi	0.69	0.005	300
top7 movies	0.602	0.005	20

$$\tau = 1 - 2 \cdot Loss_{LR} \tag{3.4}$$

where τ is Kendall τ ranking error and $Loss_{LR}$ is the ranking loss function.

SS function with 16 steps is used to rank the Wisconsin data set with 16 labels. By increasing the number of steps in the interval and scaling up the features between -100 and 100, The step width is small. In order to enhance ranking performance, the data set has many labels. The number of hidden neurons is increased in order to exceed $\tau = 0.5$.

Table 3.6: Comparison between *PNN* and supervised clustered on biological real world data in terms of $Loss_{LR}$

Biological real world data		
DS	**S.Clustering**	***PNN***
cold	0.198	0.11
diau	0.304	0.255
dtt	0.124	0.01
heat	0.072	0.013
spo	0.118	0.014
Average	0.1632	0.0804

Table 3.7: *PNN* ranking performance in terms of τ coefficient, learning step and number of hidden neurons.

Type	**DS**	**Avg.** τ	**#m.n.**	**l.r.**	**#Iterations.**	**Dropout**	**Scaling.**	**Training t.**	**Testing t.**
Real	cold	0.4	10	0.0008	2000	yes	-4:4	2.8h	1.2s
	diau	0.466	400	0.0005	2500	yes	-2:2	2.9h	4s
	dtt	0.60	400	0.0001	5000	yes	-4:4	5.7h	1.88s
	heat	0.876	450	0.0005	5000	yes	-2:2	6.2h	1.18s
	spo	0.8	300	0.0005	5000	yes	-4:4	7.4h	0.98s
	German2005	0.8	300	0.0005	1000	no	-4:4	35.15m	0.0879s
	German2009	0.67	300	0.0005	500	no	-4:4	7.087m	0.105s
Semi-Synthesized	authorship	0.931	200	0.0008	200	no	-4:4	3.82m	0.34s
	bodyfat	0.559	100	0.0005	2500	yes	-2:2	16.92m	0.44s
	calhousing	0.34	200	0.0007	1000	no	-2:2	5.03h	4.127s
	cpu-small	0.46	200	0.005	1000	no	-2:2	2.089h	1.717
	elevators	0.73	20	0.003	100	no	-2:2	27.03m	3.7s
	fried	0.89	100	0.005	100	no	-2:2	1.02h	8.45s
	glass	0.948	100	0.005	100	no	-3:3	14.8s	0.04s
	housing	0.7615	25	0.005	100	no	-3:3	37.21s	0.1s
	iris	0.956	100	0.005	100	no	-3:3	29.39s	0.066s
	pendigits	0.86	100	0.005	100	no	-3:3	34.6m	5.69s
	segment	0.956	20	0.007	100	no	-3:3	440.8s	0.94s
	stock	0.868	100	0.005	100	no	-3:3	142.48s	0.87s
	vehicle	0.869	100	0.005	100	no	-3:3	91s	0.2s
	vowel	0.85	100	0.005	100	no	-3:3	88.37s	0.312s
	wine	0.90	100	0.005	100	no	-3:3	19.19s	0.063s
	wisconsin	0.61	300	0.0005	2500	yes	-4:4	13.56m	0.1332s

3.5.3 Computational Platform

PNN and *PN* are implemented from scratch without using the Tensorflow API and developed using Numba API to speed its execution on the GPU. It used Cuda 10.1 and Tensorflow-GPU 2.3 for GPU execution and executed on the University of Technology Sydney High-Performance Computing cluster based on Linux RedHat 7.7, which has an NVIDIA Quadro GV100 and memory of 32 G.B. For non-GPU version of *PNN* is located at Github Repository Elgharabawy (2022).

3.5.4 Discussion

It can be noticed from Table 3.7 that *PNN* outperforms on S_S data sets with $\tau_{Avg} = 0.8$, whereas other methods such as, supervised clustering, decision tree, *MLP-ranker* and *LRT*, have results $\tau_{Avg} = 0.79, 0.73, 0.62, 0.475$, respectively. Also, the performance of *PNN* is almost 50% better than supervised clustering in terms of ranking loss function $Loss_{LR}$ on real-world biological data set, as shown in table 3.6. The superiority of *PNN* is used for classification and ranking problems. The ranking is used in input data as a feature selection criteria is a novel approach for deep learning.

The proposed *PNN* has multi-output neuron which gives multiple numeric values and rank the output labels simultaneously in one model is an advanced step over pairwise label ranking based on classification. *PNN* could be used to solve new preference mining problems. One of these problems is incomparability between labels, where Label ranking has incomparable relation $\perp$, i.e., ranking space $(\lambda_a \succ \lambda_b \perp \lambda_c)$ is encoded to (1, 2, -1) and $(\lambda_a \succ \lambda_b) \perp (\lambda_c \succ \lambda_d)$ is encoded to (1, 2, -1, -2). *PNN* could be used to solve new problem of non-strict partial orders ranking, i.e., ranking space $(\lambda_a \succ \lambda_b \succeq \lambda_c)$ is encoded to (1, 2, 3) or (1, 2, 2). Future research may focus on modifying *PNN* architecture by adding bias and solving problems of extreme multi-label ranking.

3.6 Conclusion

This chapter proposes a novel method to rank a complete multi-label space in output labels and features extraction in both simple and deep learning. *PNN* are native ranker networks for image classification and label ranking problems that uses *SS* or *PSS* to rank the multi-label per instance. This neural network's novelty is a new kernel mechanism, activation, and objective functions. This approach takes less computational time with a single middle layer. It is indexing multi-labels as output neurons with preference values. The neuron output structure can be mapped to integer ranking value; thus, *PNN* accelerates the ranking learning by assigning the rank value to more than one output layer to reinforce updating the random weights. *PNN* is implemented using python programming language 3.6, and activation functions are modeled using wolframe Mathematica software (Inc. n.d.).

Chapter 4

Preference Net : Classification by Ranking Reduction

4.1 Introduction

Deep Learning (*DL*) is the learning to extract high-level and complex abstractions of data through a hierarchical learning process. *DL* succeeded in big data fields such as image recognition and natural language processing (*NLP*) by learning the selection of the best features Sornam et al. (2017), Vedantam (2021). While many studies have successfully used deep learning for classification problems, the main learning challenge is choosing the network architecture and structure in terms of nodes' numbers and hidden layers. (Autoencoder Vahdat & Kautz (2020), Convolutional Deep Belief Network (*CDBN*) Zhong & Fang (2020), Convolutional Neural Network (*CNN*), Deep Belief Network (*DBN*) Hinton (2009), Deep Boltzmann Machine (*DBM*) Salakhutdinov & Hinton (2009), Long short-term Memory Network (*LSTM*) Yan et al. (2017), Zhao et al. (2017), Recurrent Neural Network (*RNN*) Kumar et al. (2020), and Restricted Boltzmann Machine (*RBM*) Alphonse et al. (2021)) are the main deep learning approaches used in deep learning.

In computer vision, the convolution architecture is dominant in *DL* by using a variation of CNN-like architectures (Lecun et al. 1998, Krizhevsky et al. 2012*d*, He et al. 2016*b*, Wang et al. 2017).Some architectures replaced the convolutions entirely Kumar et al. (2020), Yan et al. (2017), Zhao et al. (2017). However, these models have succeeded in image classification; they have not yet been scaled effectively on large-size images and use specialized attention patterns. Therefore, in large-scale image recognition, classic *ResNet* like architectures are still state-of-the -art Mahajan2019CategoricalIC, Farooq2020COVIDResNetAD. Therefore, kernel computation and scaling are still fixed and specialized to certain images type.

Preference neural network (*PNN*) was introduced by (Elgharabawy et al. 2021*b*) as the first *ANN* for ranking using spearman objective function and new smooth staircase (*SS*) activation function designed to accelerate the ranking by producing multi preference values (Elgharabawy et al. 2021*b*). The deep neural network (*DNN*) is introduced for object ranking to solve document retrieval problems Wang et al. (2020). RankNet Chris Burges (2005), RankBoost Freund et al. (2003), and Lambda MART Wu et al. (2010), and deep pairwise label ranking models Jian et al. (2019), are convolution neural Network (*CNN*) approaches for the vector representation of the query and document. *CNN* is used for image retrieval in Li et al. (2020) and label classification Ji et al. (2020).

The *CNN* mentioned above, and their variants exhibit several issues that can be broadly summarized as:

1) Partial detection as *CNN* kernel detects small size features such as edges with kernels that occupy only tens or hundreds of pixels.

Thus, it ignores the relationship between different parts of the whole image in large images. For example, *CNN* detects the image edges in the human face by combining features (the mouth, two eyes, the face oval, and a nose) with a high probability to classify the subject without learning the relationship between these features of CNN with several layers

2) Slow in computational performance due to large number of *CNN* several layers.

3) Challenges in detecting object under different angels, backgrounds, lighting conditions.

The proposed *PN* has several advantages over existing *CNN* classification approaches.

1) Simplifying the calculation based on the difference of pixel values of greyscale images.

2) Enhancing the predictive probability and accelerate the ranking convergence rate by using new *PS* activation function over existing *sigmoid*, *Relu* and *Softmax* due to the step shape to produce almost discrete multi-values from 0 to n where n is the number of ranked labels.

3) Speeding the computational calculation of single epoch due to *PN* 5 layers.

4) Boosting the accuracy, sensitivity, and image classification results by pixels' ranking and reducing the ranking output to classification using score function.

Section II explains the *PN* components, namely Activation functions, Objective function, and network structure.

4.2 *PN* Components

4.2.1 Multiple *PNN*

PN components consists of multiple PNN network, each PNN learn one kernel size.

The *spearman* error function is represented by Eq.4.1

$$\rho = 1 - \frac{6\sum_{i=1}^{m}(y_i - yt_i)^2}{m(m^2-1)} \tag{4.1}$$

where y_i, yt_i, i and m represent rank output value, expected rank value, label index and number of instances, respectively.

4.2.2 Preference Neuron

thw PNN preference Neuron is used with *PSS* function

4.2.3 Preference Neural Network

The *PNN* is fully connected to multiple-valued neurons and a single-hidden layer proposed by ELgharabawy Elgharabawy et al. (2021*b*,*a*). The input layer represents the number of features per data instance. The hidden neurons are equal to or greater than the number of output neurons, $H_n \geq \mathcal{L}_n$, to reach error convergence after a finite number of iterations. The output layer represents the label indexes as neurons, where the labels are displayed in a fixed order.

Section 4.3 describes the data preprocessing steps, feature selections.

4.3 *PN* Structure and Processing

This approach converts the multi-class classification problem into a multi-label ranking problem in two steps as shown in Fig 4.1.

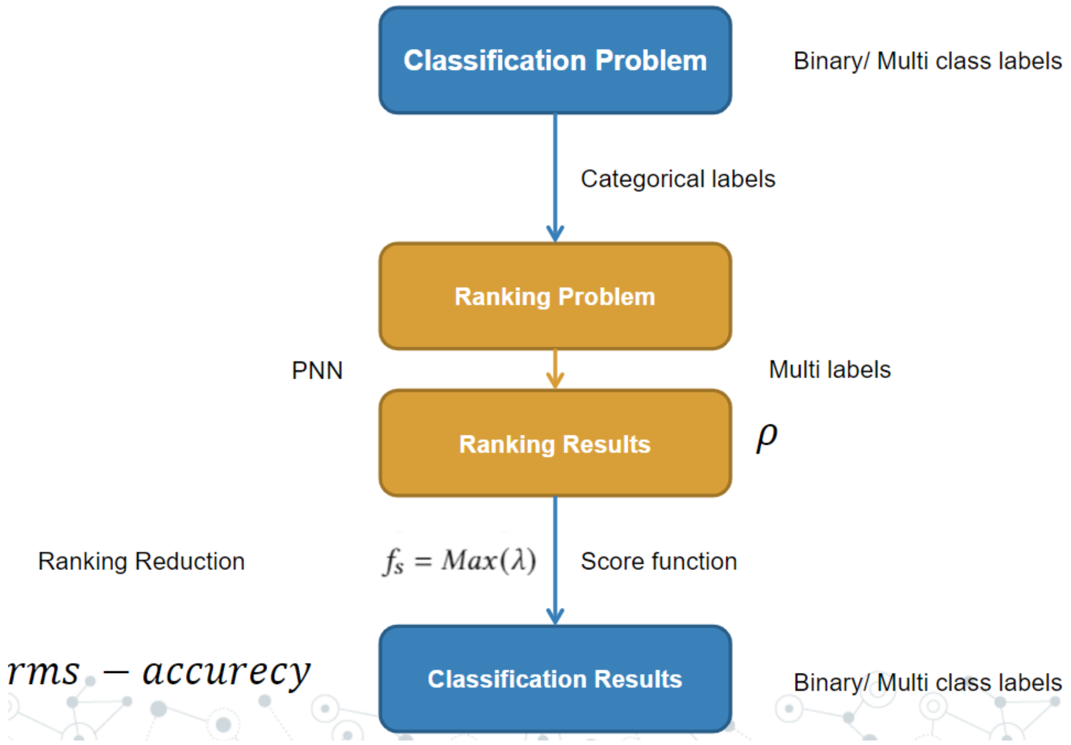


Figure 4.1: Classification by ranking reduction flow

1) Converting Label Classification into Label Ranking:

   By converting the multi class vector to binary categorical class labels i.e: for class 1,2,3,..,10 and 4 are presented as ranked labels (2, 1, 1, .. , 1) for class label 1 and (1, 1, .. , 1, 2) for class label 10.

2) Converting Ranking Results into Binary Classification: using ranking reduction into classification by scorer function s: $\mathcal{L} \to \mathbb{R}$. For output labels to choose the max value $f_s = Max(\lambda)$ for the binary classification that has c : $\mathcal{X} \to \{\pm 1\}$. *PN* reranked the final results using the score function by choosing the highest preference value as the class label. The binary ranking is used to calculate the image classification accuracy, sensitivity and specificity. For example, The final *PN* results of 10 labels are (2, 1.5, 1, 1.7, 1, 1, 0, 1, 0) which has ρ =0.5, then the binary ranking is (2, 1, .. , 1) which increase the

$\rho = 1$ correlated to class categorical output (1, 0, 0, 0, 0, 0, 0, 0, 0, 0).

4.3.1 Image Prepossessing

4.3.1.1 Greyscale Conversion

Data scaling as red, green and blue (*RGB*) colours is not considered for ranking because *PN* measures the preference values between pixels. Thus, the image is converted from *RGB* colour to Greyscale.

4.3.1.2 Pixels' Ranking

The image pixels are ranked using Spearman correlation by flattening the image into one numeric vector and ranking the vector which represents the pixels values from 0-255 to 1-156 as illustrated in Fig. 4.2. From (a) to (b) in this step we rank subset of the data by ranking window part of pixels' image by ranking the image from $\pi = \{\lambda_1, .., \lambda_m\}$ to $\pi = \{\lambda_1, .., \lambda_k\}$ where the maximum greyscale value $\lambda_m = 255$ and λ_k is the maximum ranked pixel value as illustrated in Fig. 4.2 (c).

4.3.1.3 Pixel Averaging

Ranking image pixels have an almost low-ranking correlation due to noise, scaling, light, and object movement; therefore, window averaging is proposed by calculating the mean of pixel values of the small flattened window size of 2x2 of 4 pixels as shown in Fig. 4.3. The overall image ρ of pixels increased from 0.216 to 0.79 in (a) and (b), from 0.137 to 0.75 for noisy images in (c) and (d), and scaled images from -0.18 to 0.71 in (e) and (f).

Pixel ranking and Averaging are two approaches have been tested on two sample images of remote sensing and faces images to detect the similarity. The experiment shows high ranking correlations using different

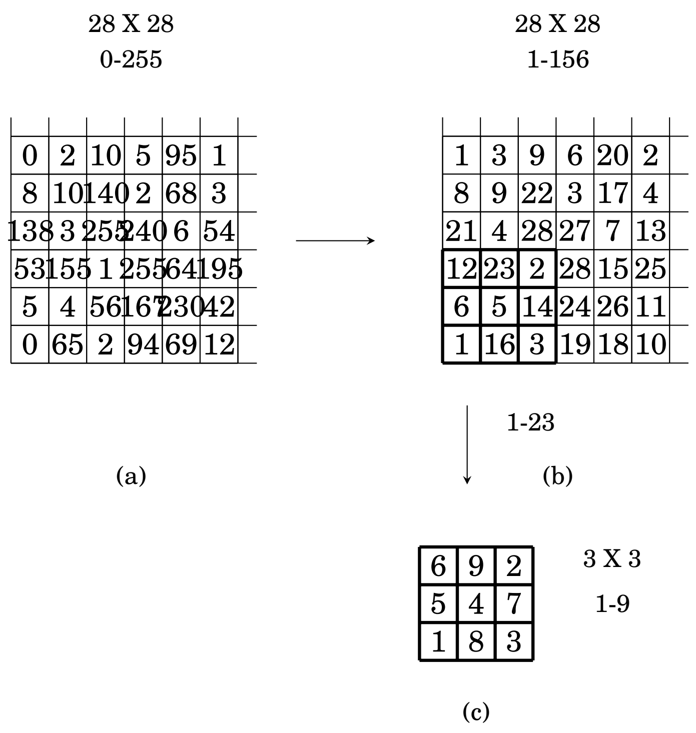


Figure 4.2: Image pixel ranking for each flattened window.

window size as shown in Fig 4.4. It detects the high correlation by starting from the large window size to be set as the image size. It reduces the size and scan until it reaches the highest correlation.

4.3.2 Feature Extraction

This chapter proposes a new approach for feature selection based on data preference values by ranking the pixels instead of *CNN* convolution. The features are based on ranking computational space. Therefore, the kernel window size is considered a factor for feature selection.

4.3.2.1 Window Pixels' Ranking

For each scanned window in the image, the flatten ranked vector is ranked before measuring the ρ with the ranker kernel. the Fig. 4.2 shows the window size 3X3 range from $\lambda_{k_1} = 23$ to $\lambda_{k_2} = 9$.

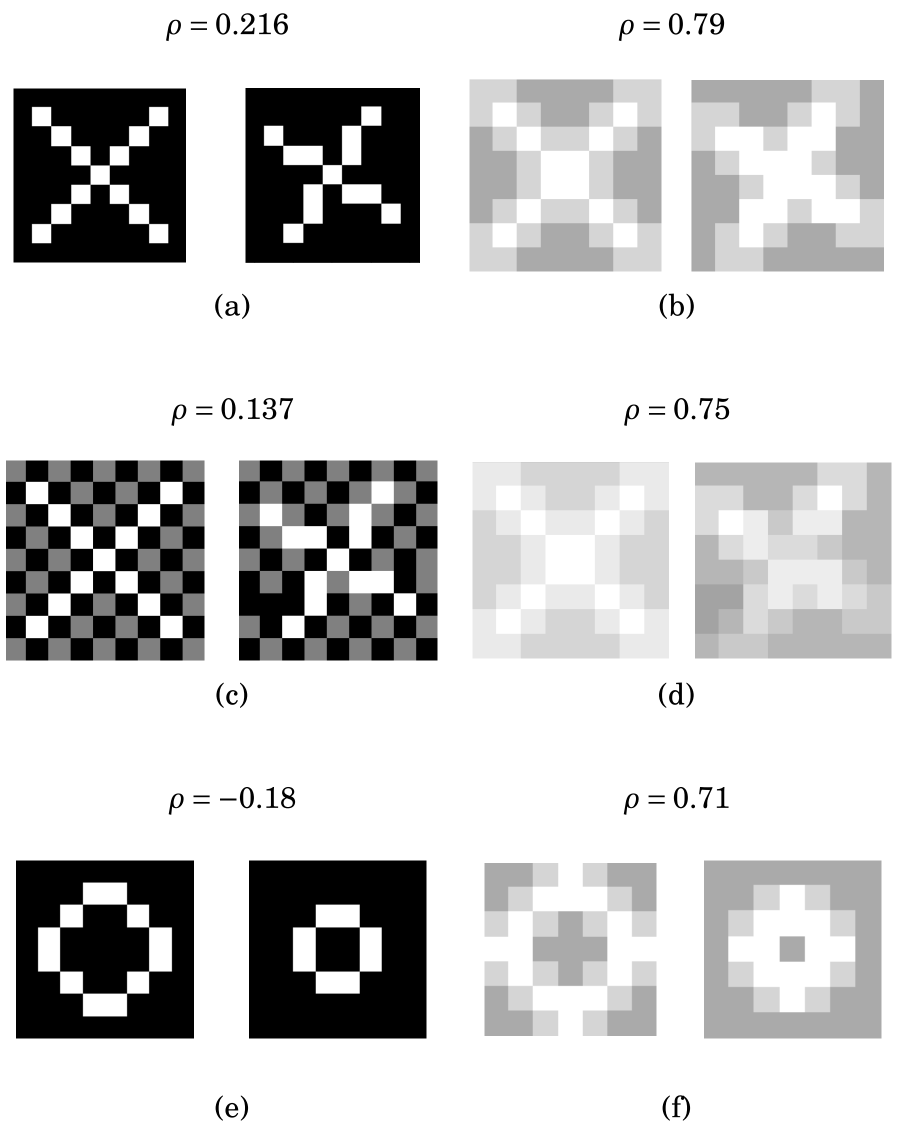


Figure 4.3: Sample of moving objects in (a) and (b) without and with averaging by window 2x2. The ranking of two flattened images are $\rho = 0.216$ and 0.79 in (a) and (b), respectively. Sample of moving noisy object in (c) and (d) without and with image averaging by a window of 2x2. The ranking of two flattened images are $\rho = 0.137$ and 0.75 in (c) and (d) respectively. and $\rho = -0.18$ and 0.71 of scaled circle in (e) and (f), respectively.

Ranking the pixel reduces the data margin, so it reduces the computational complexity.

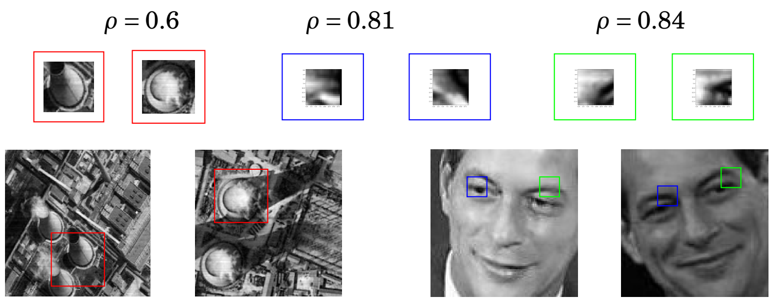


Figure 4.4: Detecting the similarity in remote sensing and face recognition by ranking the image pixels after averaging the pixels using a 2x2 window.

4.3.2.2 Weighted Ranker Kernel

The kernel weights are randomly initialized from -0.05 to 0.05 learns the features by BP the weights. The partial change in the kernel dKw is by differentiating the *spearman* correlation as in Eq. 4.2

$$dKw = 2 \cdot Img - d\rho \cdot \frac{n^3 - n}{-6} \tag{4.2}$$

Where Img is the original image matrix, and $d\rho$ is the differentiating the *spearman*.

Different kernel sizes could be used. However, we propose multiple kernels for big images' size. We use three different kernels to capture the relations between different features in the image.

4.3.2.3 Max Pooling

PN uses the max. pooling approach to reduce the features map's size and select the highest correlation values to feed to the *PNN*.

4.3.3 Network Structure

PN is the deep learning structure of *PNN* for image classification. It consists of five layers; a ranking features map, a max. pooling and three *PNN* layers. *PN* has one or multiple different sizes of *PNN*s connected by one output layer. Five layers are the minumum layers for the proposed deep learning classification. Many different structures can be applied for the future work. Each *PNN* has *PSS* where $\varphi_{n=2}$ for binary ranking to map the classification. The number of output neurons is the number of classes. *PN* have one or more ranker kernels with different sizes; Each kernel has one corresponding *PNN* as shown in Fig 4.5. *PN* uses the weighted kernel ranking to scan the image and extract the features map of *spearman* correlation values of the kernel with the scanned ranked image window as $\rho(\pi_k, \pi_w)$ where π_k is the kernel preference values and π_w is the scanned window image preference values. Each kernel scans the image by one step and creates a *spearman* features list. Max. Pooling is used to minimize the feature map used as input to *PNN*. Small size kernels are used for an image that does not have observed similarity as CFAR-10, where one object has different shapes. i.e., truck. The three kernels (8, 10, 20) are used in classification of *Mnist* data set LeCun & Cortes (2010). Three kernels with sizes (6, 7, 8) are used for *CFAR-10* Krizhevsky (2009*a*). The main advantage of *PN* is choosing the kernel size by eye observation as it is based on the size of meaningful feature (letter, eye, etc) comparing to convolution kernel. The calculation of correlation based on adding and subtracting and doesn't use multiplication as convolution. In Addition to, miminum number of layers comparing to CNN. Table 4.1 represents a brief comparison between *CNN* and *PN* in terms of components.

**conventional fun.: relu, logistic, sigmoid, tanh, gaussian, softmax, maxout.*

Table 4.1: Comparison between *CNN* and *PN*.

Type	***CNN***	***PN***
Activation Fun.	fun. *	PSS
Kernel	convolution	spearman ρ
Pooling	avg.,max.	max
Layer	multiple	single
Gradient	Descent	Ascent
Objective Fun.	rms	ρ
Stopping Criteria.	rms	ρ

4.4 Algorithms

4.4.1 Kernel size

The ranker kernel window size is chosen according to the highest correlation between two images of the same class. The kernel scans the two images and calculates ρ, and the number of flattening windows correlation exceeds 0.8 and 0.6. the number of kernels is from 3 to 5 kernels. The Mnist dataset kernel size is chosen according to the table 4.2

4.4.2 Baseline Algorithm

Algorithm 1 represents the three functions of the network learning process: feed-forward (*FF*), *BP*, and updating weights (*UW*). Algorithm 3 represents the learning flow of *PN*.Algorithm 4 represents the *PN* learning cycle. Algorithm 5 represents the simplified BP function in two steps.

4.4.3 Complexity Analysis

4.4.3.1 Time Complexity

- *FF* time complexity corresponds to *FF* of middle and output layers, and m and n are number of nodes in the middle and output layers. W_m and W_o are weighted matrix and SS_t is the activation function

Algorithm 3: *PNN* learning flow

```
   Data: $\mathcal{D} \in \{x_1, x_2, \ldots, x_d\}$
   Result: $\pi \in \{\lambda_{y_1}, \ldots, \lambda_{y_n}\}$
14 Randomly initialize weights $\omega_{i,j} \in \{-0.05, 0.05\}$
15 repeat
16     forall $\langle x_i, \pi_i \rangle \in \mathcal{D}$ do
17         $a_i|_{l-1} = \sum_{i=1}^{m} \varphi(a_i \cdot \omega_i)|_n$ // FF
18         PNN BP()
19         $\omega_{i_{new}} = \omega_{i_{old}} - \eta \cdot \delta_i$ //UW
20 until $\rho_{Avg}$ = 1 or $\#iterations \geq 10^6$;
```

Algorithm 4: *PN* Learning flow

```
21 Converting image to greyscale
22 Flattening image
23 Image pixel ranking
24 2D Image
25 Pixel averaging by a 2X2 window
26 Flattening image
27 Select one/more kernel sizes.
28 Random init. Kernel $K\omega_{x,y} \in \{-0.05, 0.05\}$
29 Random init. PNN $\omega_{i,j} \in \{-0.05, 0.05\}$
30 repeat
31     2D Image
32     Scanned window pixel ranking $Img_w$
33     Compute $\rho(Img_w, Kw)$ feature map
34     Max. Pooling.
35     Flattening image
36     PNN FF()
37     PNN BP()
38     PNN UW()
39     Max. Pooling BP()
40     Ranker kernel BP and UW()
41 until $\rho_{Avg}$ = 1 or $\#iterations \geq 10^6$;
```

of number of instances t. The time complexity in Eq. 4.3

$$\mathcal{O}(m \cdot o \cdot t) \tag{4.3}$$

- *BB* starts with calculating the error of output layer $E_{ot} = \rho_o' \, Delta_o =$

Algorithm 5: *PNN* BP

```
42 Step 1: for each $pn_i$ in Output layer do
43     $Err_i = \rho = -6 \cdot \frac{(2yt_i - y_i)}{m(m^2-1)}$ //spearman error
44     $\delta_i = Err \cdot \varphi\prime$
45 Step 2: for each $pn_i$ in middle layer do
46     $Err_i = \sum_{k=0}^{m} \omega_k \cdot \delta_k$
47     $\delta_i = Err \cdot \varphi\prime$
```

Table 4.2: The number of window correlated of two images in Fig. 4.6 (a) of class 3 of *Mnist* dataset by using image averaging 2x2 and 4x4 window using different window size from 3x3 to 28x28.

Win.Size		4	5	6	7	8	9	10	11	12	13	14	15	16
# win.	**ρ>0.95(2)**	737	419	233	134	79	46	22	9	4	1	0	0	0
	ρ>0.95(4)	2625	1997	1276	872	503	286	163	78	38	21	7	3	0
	ρ>0.6(2)	20176	18982	16706	14003	11424	9388	7681	6100	4681	3538	2673	118	19

$E_{ot} \cdot SS'$ and $Delta_m = E_{mt} \cdot SS'$ then UW

$$W_m = W_m - Delta_m \tag{4.4}$$

This time complexity is then multiplied by the number of epochs n

$$\mathcal{O}(\mathrm{n \cdot m \cdot o \cdot t}) \tag{4.5}$$

4.4.3.2 Input Neurons

The number of *PN* input neurons is represented by Eq. 4.6

$$\#Input = (Img_W - K_W + 1) \cdot (Img_H - K_H + 1) \tag{4.6}$$

where K_W is kernel width and K_H is kernel height.

4.4.4 Choosing The Kernels

Choosing the size of kernels is based on the image size and the minimum window size of meaningful observed feature. For example the for Mnist dataset where the image has size 28X28, The meaningful features are 10x10, 15x15, 20x20 and 25x25. The number of kernels for 28 images may not exceed the number of meaningful feature sizes of 4 kernels.

4.4.5 Color Image Compensation

PN have alternative approach to classify RGB color by separate the image into three R,G and B images and process the three images using multiple ranking kernel.

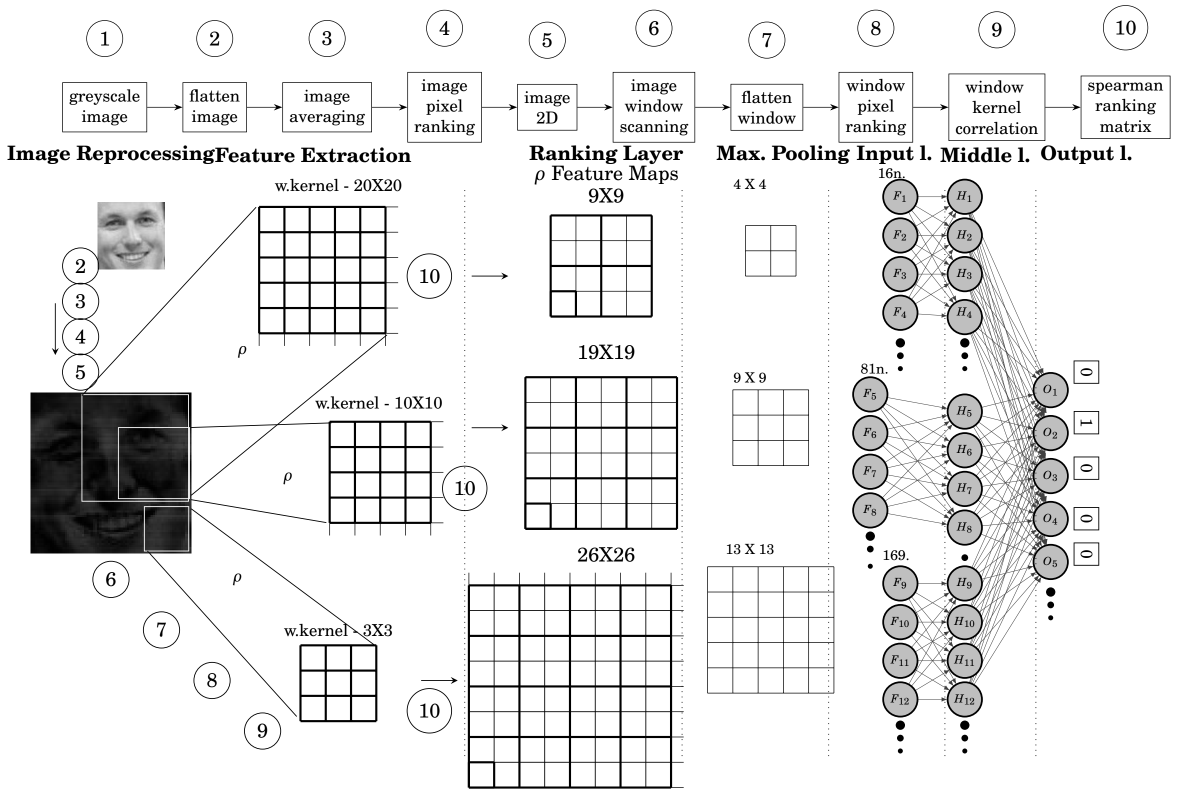


Figure 4.5: The *PN* structure has three kernels and three *PNN*s where $\varphi_{n=2}$, $f_{1in} = 16, f_{2in} = 81, f_{3in} = 169$ and $\lambda_{out} = 15$, per $\langle x_1, \pi_1 \rangle, \pi \in \{\lambda_1, \lambda_2, \lambda_3 \cdots, \lambda_{15}\}$.

4.5 Network Evaluation

This section evaluates the *PNN* against different activation functions and architectures. All weights are initialized = 0 to compare activation functions and *A* and *B* have the same initialized random weights to evaluate the structure.

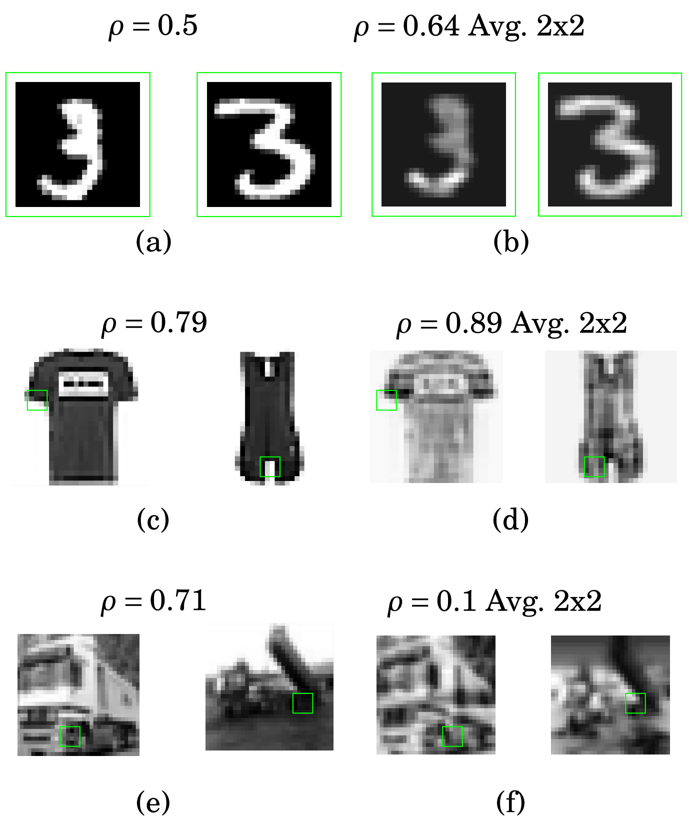


Figure 4.6: Image averaging using image width window size applied in MNIST , MNIST-Fashion and Cifar-10 datasets in (a),(c), and (e) and the images after averaging using 2X2 in (b), (d), (f) in order to choose the best kernel windows by searching for the best correlation for all window sizes

4.5.1 Multiple Kernel Evaluation

Increasing the number of ranker kernel increase the rate of convergence and reaches up to three kernels to a stable rate as shown in Fig. 4.7.

4.5.2 Dropout Regularization

Dropout is applied as a regularization approach to enhance the *PNN* ranking stability by reducing over-fitting. We drop out the weights that have a probability of less than 0.5. these dropped weights are removed from FF, BP, and updating weight steps. The gap between the training model and ten-fold cross-validation curves has been reduced using dropout regularization using hyperparameters (l.r.=0.05, h.n.=100) on the Mnist data set. The dropout technique is used with all the data ranking results in the next section.

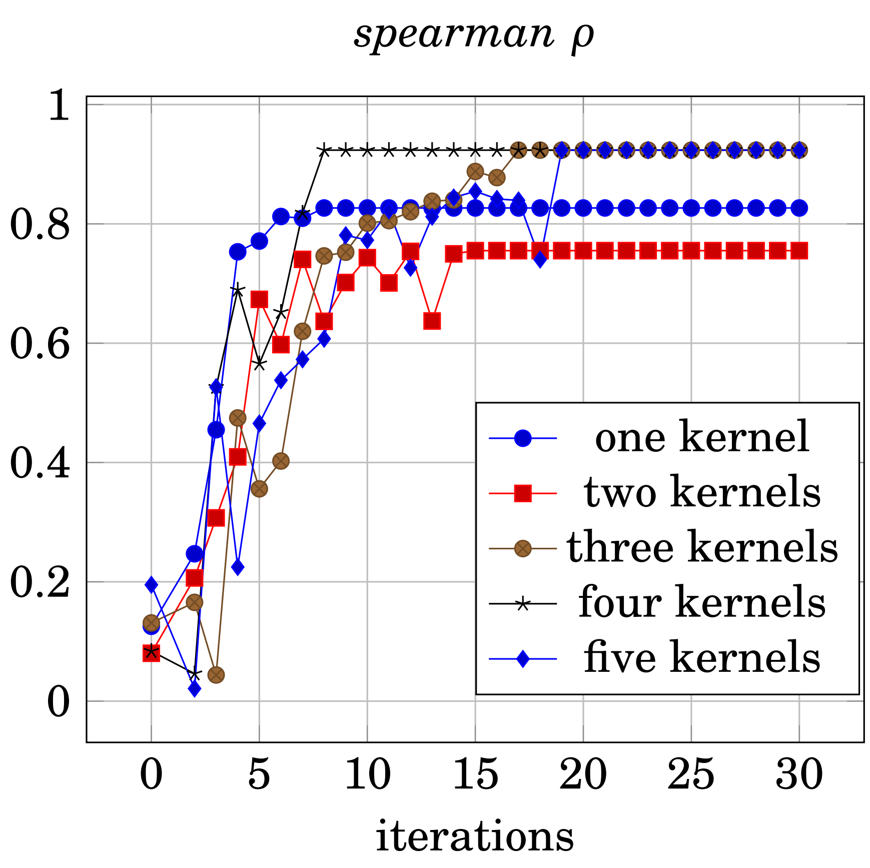


Figure 4.7: Comparison between the number of ranker kernels used in PN; one kernel (15X15) , two kernels (15X15 and 20X20), three kernels (15X15, 20X20 and 25X25),four kernels (10X10, 15X15, 20X20 and 25X25), five kernels (5X5, 10X10, 15X15, 20X20 and 25X25) for training 10 images of Mnist dataset where image size is 28X28. as shown by increasing number of kernels the performance is reached to a stable convergence rate doesn't reach faultless ranking $\rho = 1.0$ as mentioned by Ailon Ailon & Mohri (2007).

The following section is the evaluation of ranking experiments using image recognition benchmark data sets.

4.6 Experiments

This section describes the classification benchmark data sets, the results using *PN* , and a comparison with existing classification methods.

4.6.1 Data sets

PN is evaluated using *Mnist* LeCun et al. (2010), Fashion-Mnist Xiao et al. (2017*a*), and *CIFAR-10* Krizhevsky (2009*a*) data sets.

4.6.1.1 MNIST

It consists of hand-written digits and is the most used dataset within the deep learning community. The dataset is trivial to learn and simple to reach good performance LeCun et al. (2010). It is included in the experiment for algorithm completeness of benchmarks.

4.6.1.2 MNIST-Fashion

is a rather new dataset with different classes of clothing and is a drop-in replacement for MNIST Xiao et al. (2017*b*). It is harder but has the same size, input dimension, and number of classes as Mnist.

4.6.1.3 CIFAR-10

The CIFAR-10 dataset contains 60,000 32x32 color images in 10 different classes. (Krizhevsky 2009*b*). The 10 different classes represent airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 6,000 images in each class.

4.6.2 Results

4.6.2.1 Ensemble weighted models

The results are extracted using ensemble-weighted average models. This approach is executed by dividing the training dataset (50,000) images into a small chunk of data 1000 images and running a parallel process for each chunk on *PN* and getting the model and validation results for each mode as shown in Fig. 4.8. According to the validation result, the weight is determined, then testing data of 20,000 images are executed on fifty models, the average weight is taken of the 50 models to determine the final result.

4.6.2.2 Output Ranking Results

In terms of ranking, the output of *PN* barely reaches ρ=0.85; however, applying binary ranking on the final results increases the classification accuracy and sensitivity before and after using cost function in (a) and (b), respectively, in MNIST, MNIST-Fashion, and CIFAR-10 as shown in table 4.3.

4.6.2.3 Image Classification Results

PN is tested on the *CFAR-10* Krizhevsky (2009*a*), MNIST-Fashion data set Xiao et al. (2017*a*) using 3 kernel sizes and MNIST as shown in Table 4.3. Table IV shows the results compared to other convolutions networks.

4.6.2.4 Statistical Test

The *PN* results were evaluated using receiver operating characteristic (*ROC*) curves. The true positive and negative for each rank are evaluated per label as shown in Fig. 4.9 which represents a comparison between

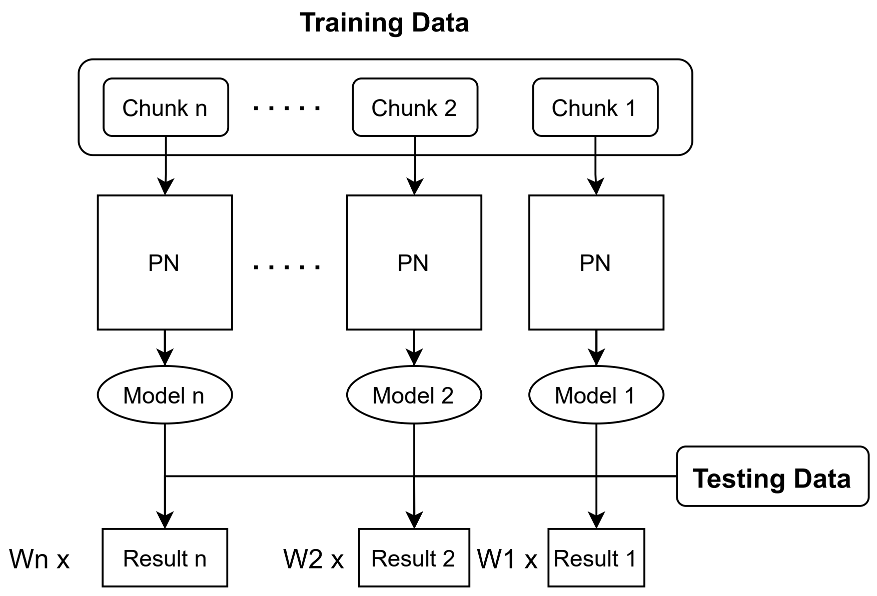


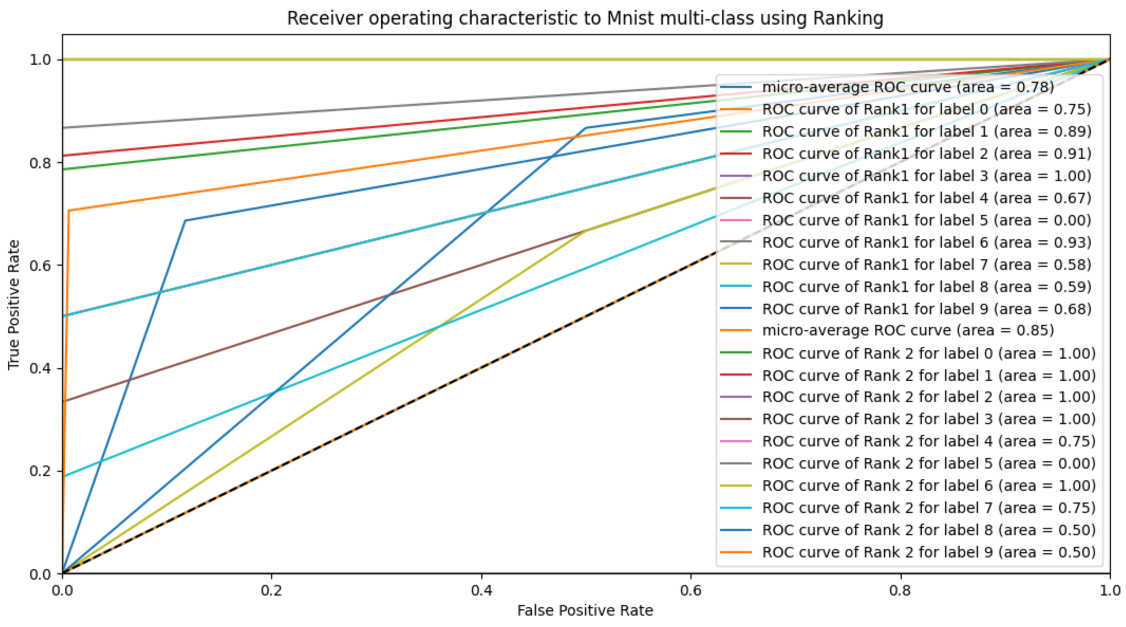
Figure 4.8: weighted ensemble average models flow of the image classification using *PN*s parallel processing for each chunk of training data.

ranking and classification after reduction using *ROC* in Fig. 4.9 (a) and (b) respectively.

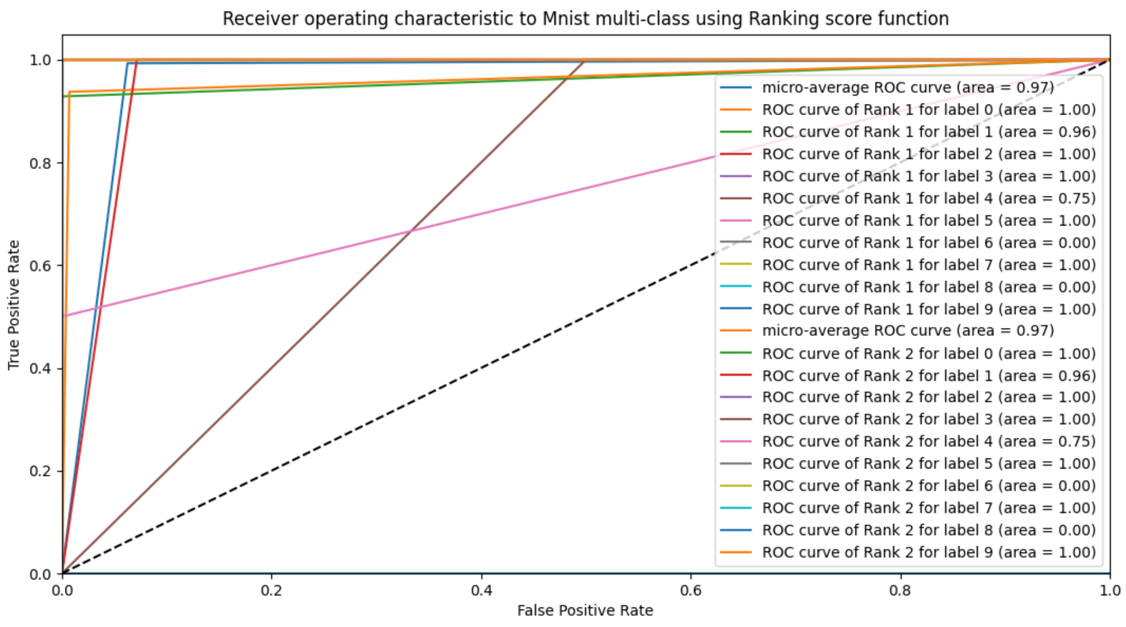
Table 4.3: *PN* results on datasets(Mnist, Mnist-Fashion, and CIFAR-10) where The accuracy and sensitivity before and after applying score function.

DS	**Mnist**	**Mnist-Fashion**	**CIFAR-10**
window avg.	2x2	2x2	-
kernels	8,10,20	6,8,10	6,7,8
h.n.	100,100,100	100,100,100	150,150,150
Epochs	200	500	1000
Acc.-before-score	0.75	0.71	0.632
Acc.-after-score	0.974	0.9316	0.9083
Sens.-before-score	0.654	0.71	0.782
Sens.-after-score	0.954	0.912	0.906

The *PN* are implemented from scratch without the Tensorflow API at the University of Technology Sydney High-Performance Computing cluster based on Linux RedHat 7.7.



(a)



(b)

Figure 4.9: *ROC* evaluation of 500 images of Mnist in the first 50 epochs using *PN* output ranking in (a). The output ranking after ranking score applied for ranking reduction in (b) where ranking 1 and 2 using the score and inverse of score function $f_{s2} = Max(\lambda)$ and $f_{s1} = notMax(\lambda)$

Table 4.4: Comparison of classification on CIFAR-10 Krizhevsky (2009*a*) and Fashion-Mnist data set Xiao et al. (2017*a*) Data sets using different convolution models

DS	**M**odel	**B**aseline
MNIST	CapsNet Mazzia et al. (2021)	97.22
	MCDNN Ciresan et al. (2012)	99.26
	SpinalNet Kabir et al. (2020)	98.73
	PrefNet	97.4
Fashion-MNIST	DARTS Tanveer et al. (2021)	0.96
	SAM Foret et al. (2021)	0.951
	Wide-ResNet Zagoruyko & Komodakis (2016*b*)	0.955
	PrefNet	0.9316
CIFAR-10	ResNet He et al. (2016*a*)	92.22
	WRN Zagoruyko & Komodakis (2016*c*)	96.26
	Dense Huang et al. (2017)	93.73
	PrefNet	92.41
CIFAR-100	ResNet He et al. (2016*a*)	72.22
	WRN Zagoruyko & Komodakis (2016*c*)	78.26
	Dense Huang et al. (2017)	81.73
	EfficientNetV2-M Tan & Le (2021)	92.2
	EffNet-L2 (SAM) Ridnik et al. (2021)	96.08
-	CvT Wu et al. (2021)	94.39
-	PrefNet	90.83

4.6.3 Discussion and Future Work

This chapter introduces a new classification approach by ranking reduction. This approacg is achieved by initial ranking step to reach a stable results in validation and testing, then using cost function to boost the accuracy to outperform some other approaches. It can be noticed from table 4.3 that *PN* is performing better than *CapsNet* Mazzia et al. (2021) and. Different types of architectures of *PN* could be used to enhance the results to reach state-of-the-art in terms of image recognition.

The superiority of *PN* is using a new type of weighted kernel in pixels' ranking correlation and creating spearman correlation features matrix as a new feature selection to be a novel approach for deep label ranking for image recognition.

4.7 Conclusion

This chapter proposed a novel method to rank a complete multi-label space in output labels and features extraction in both simple and deep learning. *PN* are native ranker networks for image classification and label ranking problems that use *PSS* to rank the multi-label per instance. This neural network's novelty uses a new kernel mechanism based on correlation instead of convolution. However, the ranking results hardly reach 0.9, which requires the score function to increase the accuracy by reducing the ranking to classification. This approach takes less computational time with a single middle layer. It is indexing multi-labels as output neurons with preference values. The neuron output structure can be mapped to an integer ranking value. *PN* is implemented using python programming language, and activation functions are modeled using Wolframe Mathematica software Inc. (n.d.).

CHAPTER 5

SUBGROUP PREFERENCE NEURAL NETWORK

5.1 Introduction

Subgroup label ranking is proposed to rank multiple group of labels using a single learning model, which is a new problem faced in preference learning. This chapter introduces the Subgroup Preference Neural Network (*SGPNN*) that combines multiple networks have different activation function, learning rate, and output layer into one artificial neural network (*ANN*) to discover the hidden relation between the subgroups' multi-labels. The *SGPNN* is a *FF*, partially connected network that has a single middle layer and uses stairstep (*SS*) multi-valued activation function to enhance the prediction's probability and accelerate the ranking convergence. The novel structure of the proposed *SGPNN* consists of a multi-activation function neuron (*MAFN*) in the middle layer to rank each subgroup independently. The *SGPNN* uses gradient ascent to maximize the Spearman ranking correlation between the groups of labels. Each label is represented by an output neuron that has a single *SS* function. The proposed *SGPNN* using conjoint dataset outperforms the other label ranking methods which uses each dataset individually. The proposed *SGPNN* achieves an average accuracy of 91.4% using the conjoint dataset compared to supervised clustering, decision tree, multilayer perceptron label ranking and label ranking forests that achieve an average accuracy of 60%, 84.8%, 69.2% and 73%, respectively, using the individual dataset.

This chapter is providing the tool to support the third phase of *SD* to rank the discovered subgroup. Also, the chapter proposes an approach to convert unrestricted label ranking to restricted label ranking data. The Subgroup Preference Networks (*SGPNN*) built upon preference neural network (*PNN*) to rank multi-label subgroups data $\mathcal{D} \in \{\langle x_n, (\pi_{n1} \perp \pi_{n2} \ldots \perp \pi_{nm})\rangle\}$ where π_n is a group of labels and m = number of subgroups. The primary motivation of this work is to build a unified predictive ranking data model to support multiple exception mining analysis instead of having different models for different subgroup discovery problems. The multi-label subgroup is employed in the following scenarios:

1. Real customer data often explicitly rate different categories of products and services as multi-label subgroups, e.g., restaurant rating based on food quality and customer services (Vargas-Govea et al. 2011).

2. Multi-label ranking of related datasets collected in different time periods, e.g, German elections in 2005 and 2009 (Rebelo 2018*b*,*a*).

3. Multi-label data that have unrestricted preference relations between labels are converted into connected subgroups that have restricted relations. This can be seen in the sushi dataset (de Sá 2018, Kamishima 2003) where $\lambda_a \succ (\lambda_b, \lambda_c)$ is solved by 2 subgroups using the indifference $\sim$ or incomparable $\perp$ relations as $(\lambda_a \succ \lambda_b \succ \lambda_c) \sim (\lambda_a \succ \lambda_c \succ \lambda_b)$

or $(\lambda_a \succ \lambda_b \succ \lambda_c) \perp (\lambda_a \succ \lambda_c \succ \lambda_b)$. Another example of no ground-truth data where one data record has two labels $\pi_x = (\lambda_a \succ \lambda_b)$ and $\pi_x = (\lambda_b \succ \lambda_a)$ which are mapped to $\pi_x = (\lambda_a \succ \lambda_b) \perp (\lambda_b \succ \lambda_a)$.

The current challenge of the proposed *SGPNN* is synthesizing the data to create a group of labels per each data instance from multiple datasets or subgroup discovery processes.

To sum up, the key contributions in this chapter are:

- New ranker *SGPNN* has a novel design to rank multi-label incomparable/indifference subgroups, where no relation or no difference between groups. *SGPNN* has a novel multi activation function Neuron (*MAFN*) that uses *SS* activation function, including one function for each output layer.

- The conjoint features of multi-label datasets build a unified *NN* ranking model for different datasets from different domains. We find that the *SGPNN* has a remarkably better performance than other approaches that rank each dataset using a separate model.

- Discovering the hidden relation between different datasets by learning them together in one model is a novel approach to build an accumulative learning approach.

- Solving the ambiguous data by labeling the class overlap data with label groups.

5.2 Subgroup Preference Neural Network (*SGPNN*)

This chapter proposes an novel approach to convert the ambiguous and unrestricted label ranking to restricted label ranking data and a new

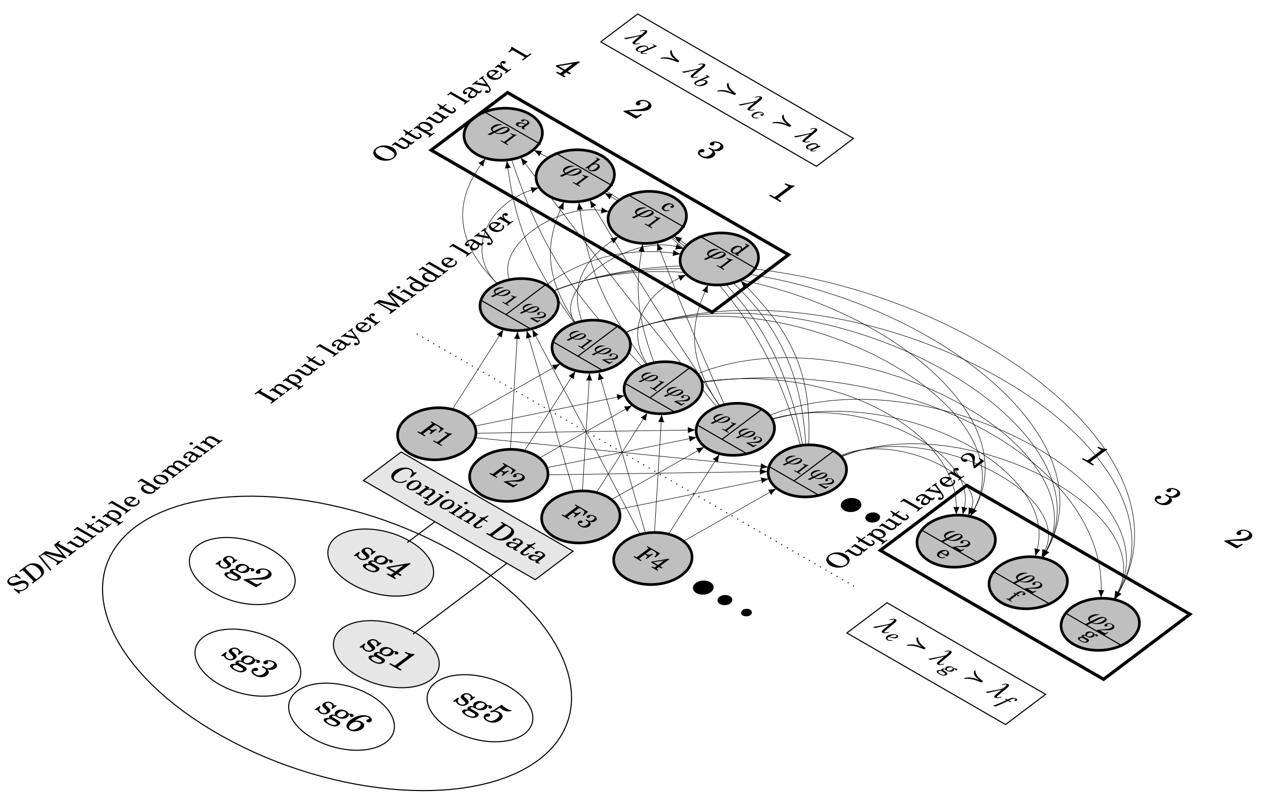


Figure 5.1: An example of two subgroups architecture of *SGPNN* to rank conjoint data from two subgroups data, each group has 4 and 3 labels respectively, where $\varphi 1_n$=4, $\varphi 2_n$=1, $f_{in} = 4$. A video demo of 2-subgroup architecture is available in (Elgharabawy 2020c)

ranker *ANN* to rank a new type of data has group of label. The Subgroup Preference Networks (*SGPNN*) built upon preference neural network (*PNN*) to rank multi-label subgroups data $\mathcal{D} \in \{\langle x_n, (\pi_{n1} \perp \pi_{n2} ... \perp \pi_{nm})\rangle\}$ where π_n is a group of labels and m = number of subgroups. The thesis introduces *SGPNN* as a complementary machine learning predictive step to *SD* methodology that follows the post-processing to classify and rank the discovered subgroups. The primary motivation of *SGPNN* is to build a unified predictive ranking data model to support multiple exception mining analysis instead of having different models for different subgroup discovery problems.

The next section describes the components and architecture of *SG-*

PNN that is based on *PNN* architecture to rank label subgroups.

5.2.1 Multi Activation Function Neuron (*MAFN*)

The *SGPNN* introduces the multi activation function Neuron (*MAFN*) to address the architecture limitation of the *PNN* to rank different lengths of output layers. The *MAFN* contains the same number of inputs because they share the same w_m weights with input neurons where w_m is the weight of middle layer, $y_{in} = \sum a_i \cdot w_i$. *MAFN* contains k number of φ activation function and lr learning rate, $k = n$, where n is the number of output layer. For example, Fig. 5.2 shows a *MAFN* which has two φ, where each function has a single output; It is graphically represented by multiple #n output links because *PN* connects only to n number of output neurons where $S = n + 1$ and s is φ step number.

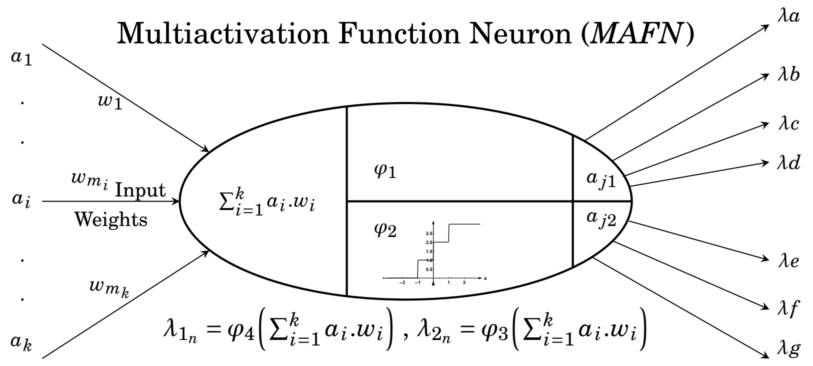


Figure 5.2: The structure of the *MAFN* where $\varphi_{1_{|n=4}}$ and $\varphi_{2_{|n=3}}$.

As shown in Fig. 5.2, $\varphi_{1_{|n=4}}$ and $\varphi_{2_{|n=3}}$ of the *MAFN* is connected to 2 output groups of 4 and 3 neurons respectively.

In a conventional *ANN*, the sufficient number of hidden neurons to achieve convergence is determined by the Cao and Mirchandani theorem (Mirchandani & Cao 1989). In an n dimensional space, the maximum number of regions that are linearly separable into M regions using h

hidden nodes is

$$M(h,n) = \sum_{k=0}^{n} \begin{bmatrix} h \\ k \end{bmatrix} \quad where \begin{bmatrix} h \\ k \end{bmatrix} = 0 \quad when \quad h < k \tag{5.1}$$

However, the *SGPNN* has multiple Euclidean n-space for each output layer. Therefore, $m \cdot n < k_{mafn}$, where n is the n-dimensional Euclidean space and m is the number of spaces per each output layer.

5.2.2 *SGPNN* Architecture

The *SGPNN* is designed to address the architectural shortcoming of *PNN*s not being extendable by ranking label's groups separately. The *SGPNN* ranks different sizes of output layers while maintaining the single middle layer design. It has two types of neurons, *PN* and *MAFN*, which are used in the output and middle layers, respectively. The input layer represents one instance of data features. The middle layer has multiple *MAFN*s that use a separate learning rate and φ activation function for each output layer. The *SGPNN* is geometrically fully connected; however, The *FF*, the *BP*, and the *UW* are functionally separated for each wo output layers' weights as illustrated in Fig. 6.4. The weights of the *MAFN* are updated by the summation of all the δ errors learning rate, $\sum_{i=1}^{k}(lr_i \cdot \delta_{mi})$. Each output layer is a group of *PN*s that represent the ranked labels. The *SGPNN* scales up by increasing the number of *MAFN*s. Fig. 5.4 illustrates examples of three subgroups architecture used for ranking emotions dataset where the first, second, and third group has 3, 1, and 4 labels respectively, to solve the problem $\pi = (h \succ p \succ q) \perp (e) \perp (a \succ b \succ c \succ d)$. The second subgroup has one label e that has three ranking values (1, 2, 3), which represent the preference values $(\succ, \perp, \prec)$ between the two subgroups. The learning of the ranking process is executed in three steps; the *FF*, the *BP*, and the *UW*. The learning stops after 20,000 epochs or

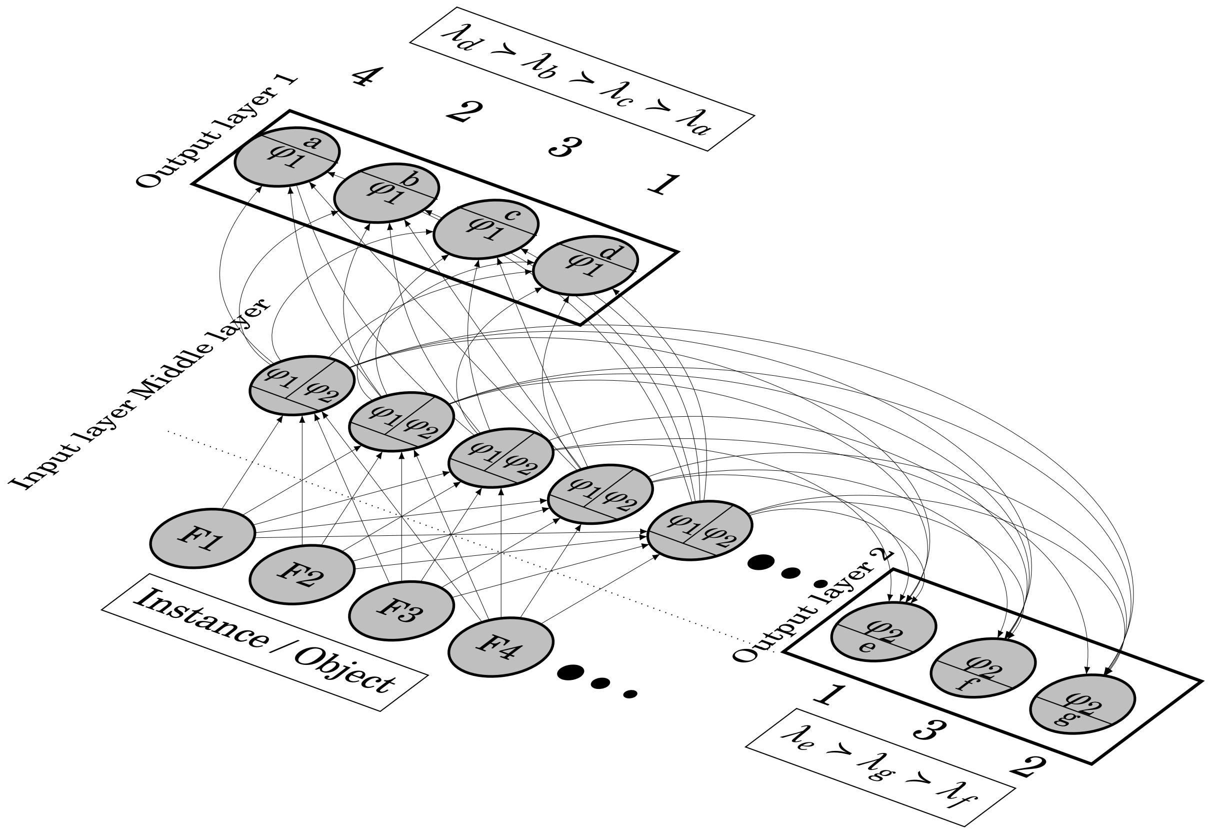


Figure 5.3: Two subgroups architecture of *SGPNN*, each group has 4 and 3 labels respectively, where $\varphi 1_{n=4}$, $\varphi 2_{n=1}$, $f_{in} = 4$. A video demo of 2-subgroup architecture is available in (Elgharabawy 2020*c*)

Spearman's ρ reaches 1. A video demo that shows the ranking learning process using simple toy data is available at (Elgharabawy 2020*c*).

5.3 Data Preparation and Learning Algorithm

This section describes the ranking unification preprocessing and *SGPNN* learning steps (*FF*, *BP* and *UW*).

5.3.1 Conjoint Data

The conjoint dataset that is used in *SGPNN* is synthesized by concatenating the features for each real subgroup per each data point as shown in Eq 5.2, 5.3

$$Fi_{sum} = \sum_{i=1}^{ns} Fi_i \tag{5.2}$$

$$Di_{sum} = \prod_{i=1}^{ns} Di_i \tag{5.3}$$

where Di is data instance, Fi is the number of feature Fi_{sum} is the sum of of features of each SG data, ns is the number of dataset. and Di_{sum} is the total number of data instance for each SG data.

5.3.2 Ranking Unification

We introduce a new method for creating label ranking ground truth by converting the unrestricted ranking to restricted ranking by unifying the data instances and adding subgroups to the labels. The percentage of a unique ranking is measured using Eq.5.4

$$U_\pi = \frac{\#distinct\ rankings}{n} \tag{5.4}$$

The number of subgroups is determined by the maximum number of repeated records using Eq.5.5

$$\#sg = Max(x_r) \tag{5.5}$$

where $\#sg$ is the number of subgroups and x_r is the number of duplicated data records. This chapter applies algorithm 6 to convert the data from non-restricted rankings with no ground truth to restricted label ranking by removing duplicated data instances and accumulating the corresponding labels in a subgroup. The algorithm removes the duplication and assigns the corresponding labels as a subgroup to one unique data record. For non-repeated records, the additional subgroup has values of zero.

Algorithm 6: Ranking Unification.

```
Data: $\mathcal{D} \in \{\langle x_n, \pi_n \rangle\}$
while $x_i$ do
    if $x_i \notin \mathcal{D}[i]$ then
        Dist $\leftarrow x_i$ ;
    else
        while $j \in \mathcal{D}$ do
            if $x_i$ in $D[j]$ then
                m++ # of subgroups;
                D $\leftarrow$ $(j, \pi_i)$ ;
        end
    end
end
Result: $\mathcal{D} \in \{\langle x_n, (\pi_{n1} \perp \pi_{n2} \perp \pi_{n3} ... \perp \pi_{nm}) \rangle\}$
```

5.3.3 *SGPNN* Learning Steps

This section shows the *FF*, *BP* and updating weights (*UW*) processes in the middle and output layer of the *SGPNN*.

5.3.3.1 Middle layer *FF*

The output Y of single a *MAFN* per subgroup j is shown in Eq. 5.6

$$Y_j = \varphi_j \sum_{k=1}^{d} x_k \cdot wm_k \Bigg|_{j=0}^{g} \tag{5.6}$$

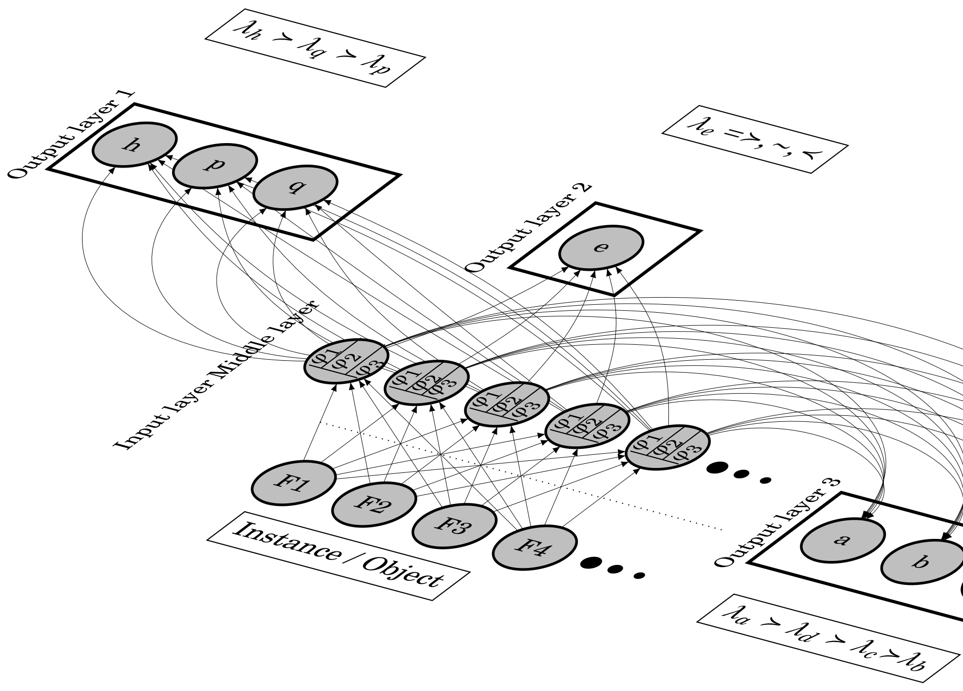
where g is the number of subgroups, d is the number of input features, and φ_j is the activation function per subgroup.

5.3.3.2 Output layer *FF*

The output of single neurons is shown in Eq. 5.7

$$Y_j = \varphi_j \sum_{k=1}^{m} x_{jk} \cdot wo_{jk} \Bigg|_{j=0}^{g} \tag{5.7}$$

where m is the number of *MAFN*s.



$\lambda_h \succ \lambda_q \succ \lambda_p$
Output layer 1
h
p
q
$\lambda_e = \succ, \sim, \prec$
Output layer 2
e
Input layer Middle layer
$\varphi 1$
$\varphi 2$
$\varphi 3$
F1
F2
F3
F4
Instance / Object
Output layer 3
a
b
c
d
$\lambda_a \succ \lambda_d \succ \lambda_c \succ \lambda_b$

Figure 5.4: Three subgroups architecture *SGPNN* used in ranking emotions dataset where $\varphi 1_{n=4}, \varphi 2_{n=3}, \varphi 3_{n=3}$, and $f_{in} = 4$. the second subgroup is represented by one node that has 3 values (1, 2, and 3) mapped to preference relations $\lambda_e = \succ, \sim, \prec$.

5.3.3.3 Output layer *BP*

Output δ_j error of a single output neuron is given in Eq. 5.8, and 5.9 the error per subgroup j is

$$Err_j = \rho_j\prime = \frac{-6 \cdot \sum_{k=1}^{o}(2yt_k - y_k)}{m(m^2-1)} \quad , \quad \delta_j = \rho_j\prime \cdot \varphi_i\prime \tag{5.8}$$

$$\varphi_j = -\frac{1}{2} \cdot \Big(\sum_{s=0}^{n} \tanh(\frac{-100}{b} \cdot y_o + c(1 - \frac{2s}{n-1})) \Big) + \frac{n}{2} \tag{5.9}$$

The δ in Eq. 5.10 is obtained by differentiating of Eq. 5.9 and substituting the result into Eq. 5.8

$$\delta_j = \big(\frac{-6 \cdot \sum_{j=1}^{o}(2yt_j - y_i)}{m(m^2-1)} \big) \cdot \varphi_j\prime \Big|_{j=0}^{g} \tag{5.10}$$

$$\begin{aligned}\varphi_{j\prime} = \Big(-\frac{1}{2}\cdot\Big(\sum_{s=0}^{n} 1-\tanh(\frac{-100}{b}\cdot y_o + c(1-\frac{2s}{n-1}))^2\cdot\frac{-100}{b}\cdot y_o \\ +(\frac{-100}{b}\cdot\tanh(\frac{-100}{b}\cdot y_o + c(1-\frac{2s}{n-1})))\Big)\Big)\end{aligned} \tag{5.11}$$

5.3.3.4 Middle layer *BP*

After updating δo, The output error is calculated by summing the δ of the *MAFN* using Eq. 5.12.

$$Err_j = \sum_{i=0}^{o} wo_{ij}\cdot\delta_{ij}\Bigg|_{j=0}^{g} \quad , \quad \delta_{mfn} = Err_j\cdot\varphi_{j\prime}\Big|_{j=0}^{g} \tag{5.12}$$

The δ_m *MAFN*'s error in Eq. 5.13.

$$\begin{aligned}\delta_{mfn} = Err_{ij}\cdot -\frac{1}{2}\cdot\Big(\sum_{i=0}^{n} 1-\tanh(\frac{-100}{b}\cdot x + c(1-\frac{2i}{n-1}))^2 \\ \cdot\frac{-100}{b}\cdot x + (\frac{-100}{b}\cdot\tanh(\frac{-100}{b}\cdot x + c(1-\frac{2i}{n-1})))\Big)\end{aligned} \tag{5.13}$$

5.3.3.5 Output layer UW

The process for updating the weights using gradient ascent with sums of *PN* δ is shown in Eq. 5.14

$$\sum_{i=1}^{m} wo_{ij|new} = wo_{ij|old} + (lr_i\cdot\delta_{i_j}\cdot y_{ij})\Bigg|_{j=0}^{g} \tag{5.14}$$

5.3.3.6 Middle layer UW

Updating the weights of the middle layer by summing of the *MAFN*'s δ values is shown in Eq. 5.15

$$\sum_{i=1}^{d} (w_{m_{ij|new}} = w_{mij|old} + lr_i\cdot\delta_i\cdot y_{ij})\Bigg|_{j=0}^{g} \tag{5.15}$$

5.3.4 Dropout Regularization

We apply dropout as a regularization approach to enhance the *SGPNN* validation performance to reduce over-fitting using 50% probability technique. The process assigns a random number from -0.9 to 0.9 and shuts down the weights with less than 0.5 of the random value per iteration for wo and wm.

5.4 Experiments

5.4.1 Datasets

The *SGPNN* is experimented on both real-world and semi-synthesized (*S-S*) datasets. The real data have multi-label subgroups for one set of features, e.g., restaurant-food-services. The *S-S* data are collected from different domains. The features from the same domain have small variations, e.g., the German elections dataset has examples of a relevant subgroup where features are collected from the same context. We examined the data uncertainty by measuring the percentage of U_π unique multi-label ranking. Given that d is the amount of the data, The description is presented in Table 5.1.

Table 5.1: Datasets (rest-food-services (Vargas-Govea et al. 2011),german-2005/9 (Rebelo 2018*a*,*b*),emotions (Trohidis & Tsoumakas 2011), sushi (de Sá 2018), iris, wine, and stock (Cheng et al. 2009)) used for *SGPNN* evaluation.

Dataset	Category	Domain	Type	Sub.Rel.	Inst.	Attr.	Sub.	Labels	U_π
					92	13	2	5,5,	87.7%
rest-food-services	user rating	single	real	$\sim$	100	13	2	10,10	76.9%
					176	13	2	20,20	57.7%
german-2005/9	elections			$\sim$	412	31	2	5,5	100%
emotions	music	single	s-s	$\succ,\sim,\prec$	392	72	3	4,2	100%
sushi	user rating			$\perp$	4825	10	3	10,10,10	95%
iris-wine	bio.-chem.				26700	17	2	3,3	99.7%
iris-stock	bio.-trades	multi.	s-s	$\perp$	142500	9	2	3,5	99.8%
wine-stock	chem.-trades				169100	18	2	3,5	100%
iris-wine-stock	bio.-chem.-trades				25365000	22	3	3,3,5	99.9%

5.4.1.1 Restaurants and Consumers Rating

The restaurant-food-services dataset is built using real restaurant and consumer reviews from the recommender systems domain (Vargas-Govea et al. 2011) and contains multi-label subgroups. The features of this dataset are customer profiles and geographical location. The two subgroups are food quality and restaurant service, and each subgroup has 130 multi-label, which represents the number of restaurants. To simplify the calculation, we use part of the data containing 5, 10, and 20 restaurants for the two groups in 3 small datasets and select the corresponding features records of users' profiles who rated these restaurants.

5.4.1.2 German Elections in 2005 and 2009

The german-2005/9 is an *S-S* dataset combined from two real datasets based on German elections in 2005 and 2009 (Rebelo 2018*a*,*b*). The multi-label of the two separate datasets is grouped into two subgroups for 2005 and 2009. We use only the 2009 dataset features to rank both 2005 and 2009 labels because 2009 features have historical data and user profiles for the 2005 elections.

5.4.1.3 Emotions

The emotions dataset is used for subgroup preference relations($\succ,\sim,\prec$). The original Emotion dataset is used to detect six types of emotions where an emotion belongs to many to one or many emotion types. The original dataset has six classes (amazed/surprised, happy/pleased, relaxing/calm, quiet/still, sad/lonely, angry/fearful). The data are modified by creating two subgroups. Positive feelings for (amazed-surprised, happy-pleased, relaxing-calm, quiet-still) and the music reflects Negative feelings for (sad-lonely, angry-fearful) (Trohidis & Tsoumakas 2011). Table 5.2 shows the heuristic rules applied for the preference relation between positive and negative feeling subgroups based on the subgroup labels' ranking. The ranking of sub-labels starts from 1 to 3. * represents the ranked value from 1 to 3.

Table 5.2: Positive and negative Emotional feelings subgroups relations according to the ranking of the labels for each group. labels per each group are extracted from emotion dataset (Trohidis & Tsoumakas 2011).

Sub1.			Sub2.	Sub3.	
Positive feeling sub.			Rel.	Negative feeling sub.	
amazed surprised	happy pleased	relaxing calm		sad lonely	angry aggressive
1	*	*	$\succ$	*	*
1	*	*	$\sim$	*	1
2 or 3	*	*	$\prec$	*	1
2 or 3	*	*	$\sim$	*	*
2 or 3	1	*	$\prec$	1	*
2 or 3	2 or 3	*	$\succ$	1	1

5.4.1.4 Irrelevant Subgroups Data

We create a new hypothetical conjoint dataset from three different domains (biology, chemistry, and stock exchange) for preference mining analysis to study data similarity and measure the *SGPNN* performance against a state-of-the-art ranking approach. The conjoint data is collected from the benchmark and well-known multi-label ranking datasets from different domains, specifically; Iris, Wine, and Stock (Cheng et al. 2009) to compare the performance of these data as subgroups with previous approaches that experimented with those datasets as a single problem.

5.4.1.5 Label Ranking benchmark dataset

Most user rating data has no ground truth due to different user preferences. The sushi (de Sá 2018, Kamishima 2003) is one of the multi-label datasets that have an unrestricted multi-label ranking as some identical data features have different multi-label rankings due to different user ratings. The unrestricted ranking is converted into a restricted subgroup of multi-label for each instance of data by removing the duplicated features and assign the labels for each repeated instance as a subgroup to a unique feature. Creating unique instances reduces the number f instances from 5,000 to 4,825 instances. Therefore, the maximum number of repeated instances is three, which means that the dataset has three subgroups. The instances that have unique second or third subgroups have zeros values.

5.5 SGPNN Results

All data are evaluated by dividing the data into 20% for testing set and 80% for training and validation. Ten-fold cross-validation is applied to the remaining 80% of the data. We use the best hyperparameters, e.g., data normalization using a scale from $-b$ to b, where b is the *SS* boundary value, and the number of iterations is 20,000 epochs. This configuration is used for evaluating both the *PNN* and the *SGPNN*. The results are presented in Table 5.3. The results of the 2005 and 2009 German elections data are illustrated in Fig. 5.5 (a). The figure shows the ranking performance of combined data using *SGPNN* outperforms the ranking of 2005 and 2009 datasets separately using *PNN*. The testing results of the validated models are mentioned in Table 5.3. Fig. 5.5 shows the training model by *PNN* and *SGPNN*; The figure shows that the subgroup gives a better ranking performance compared to the single ranking per dataset. The testing results of the models after 20,000 epochs is compared the single ranking *PNN*, and *SGPNN* with other multi-label ranking approaches in terms of Kendall's τ in Table 5.4. The *SGPNN* results are the

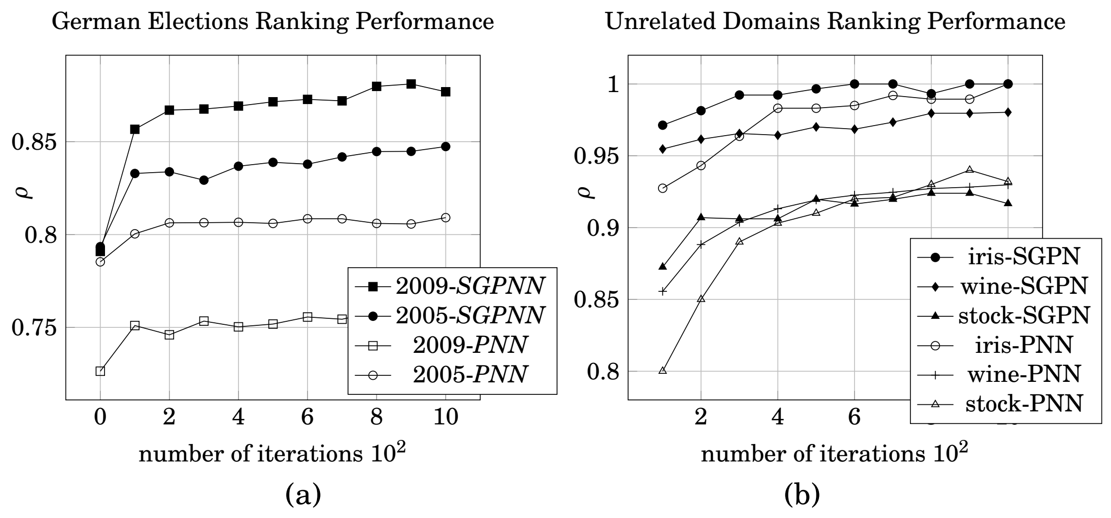


Figure 5.5: Training convergence of german election and iris,wine and stock using *PNN* and *SGPNN* in (a) and (b) respectively.

ranking of each dataset as a subgroup with the other two datasets.

5.5.1 Relevant Subgroup Data

The results of the 2005 and 2009 German elections are illustrated in Fig. 5.5 by subgroup and separate datasets, where the training model ranks convergence in terms of Spearman's ρ and the number of iterations. It is noticed that *SGPNN* outperforms both separate ranks of the 2005 and 2009 datasets using the *PNN*. The validated models' testing results after 20,000 epochs and the best epoch's hyper-parameters are displayed in Table 5.3.

5.5.2 Non-Relevant Subgroup Data

The results of the training data of conjoint iris, wine, and stock are illustrated in Fig. 5.5 by *SGPNN* comparing to ranking them separately using *PNN*, in additional to the state-of-the-art methods of testing data as shown in Table 5.4. It is noticed that *SGPNN* outperforms the other label ranking methods; supervised clustering (Grbovic et al. 2013), su-

Table 5.3: *SGPNN* and *PNN* ranking testing data performance comparison of subgroup datasets.

Dataset	**S.Group**	**Scale**	**#MAFN**	**L.r.**	***PNN***	***SGPNN***
rest-food-serv.	food quality	-1:1	100	0.06	0.814	0.912
	customer service			0.07	0.898	0.902
german election	year 2005	-20:20	100	0.05	0.8125	0.897
	year 2007			0.06	0.762	0.821
emotions	positive feeling	-10:10	100	0.05	0.616	0.87
	negative feeling				0.56	0.82
sushi	unique user pref. 1	-20:20	100	0.05	0.741	0.851
	unique user pref. 2					0.813
	unique user pref. 3					0.92
iris-wine	biology (iris)	-10:10	200	0.0007	0.917	0.933
	chemistry (wine)				0.901	0.804
iris-stock	biology (iris)	-10:10	200	0.0007	0.917	0.91
	trades (stock)				0.834	0.75
wine-stock	chemistry (wine)	-10:10	200	0.0007	0.901	0.911
	trades (stock)				0.834	0.732
iris-wine-stock	biology (iris)	-10:10	200	0.0007	0.917	0.912
	chemistry (wine)				0.901	0.856
	trades (stock)				0.834	0.956
Average					0.82	0.865

Table 5.4: *SGPNN* and *PNN* performance comparison with state-of-the-art approaches: supervised clustering (Grbovic et al. 2013), supervised decision tree (Cheng et al. 2009), multi-layer perceptron label ranking (Ribeiro et al. 2012), and label ranking tree forest (*LRT*) (de Sá et al. 2017).

Multi Label Ranking Methods						
Dataset	**S. Clust.**	DT	MLP-LR	LRT	PNN	SGPNN (iris-wine-stock)
iris	0.814	0.966 (IBLR)	0.925 (LA)	0.947	0.917	0.921
wine	0.898	0.949 (IBLR)	0.931 (LA)	0.882	0.901	0.865
stock	0.566	0.927 (IBLR)	0.745 (CA)	0.895	0.834	0.956
Average	0.6	0.848	0.692	0.730	0.884	0.914

pervised decision tree Cheng et al. (2009), multi-layer perceptron label ranking (Ribeiro et al. 2012), and label ranking tree forest (*LRT*) (de Sá et al. 2017).

5.6 Discussion

The results show that learning the labels as a subgroup from a relevant domain enhances each group's ranking compared to ranking them separately. This enhancement in ranking is almost due to sharing the network weights of two or more problems. The sharing weights accelerate the convergence, similar to reinforcement learning. This chapter proposes a novel learning method, The *SGPNN*, to rank multi-label subgroups to support the analysis of exception preference mining and *SD*. This approach is a part of the broader sphere of reinforcement learning to learn from multiple data sources and build a conjoint unified learning model. The computation time may increase by increasing the number of subgroups and higher rank accuracy; however, *SGPNN* reaches a high convergence rate in fewer iterations. The challenging future work is building a deep ranking *SGPNN* network to learn from big data. *SGPNN* can integrate with *SD* to work on a fewer number of groups to find the hidden relation between a group of data.

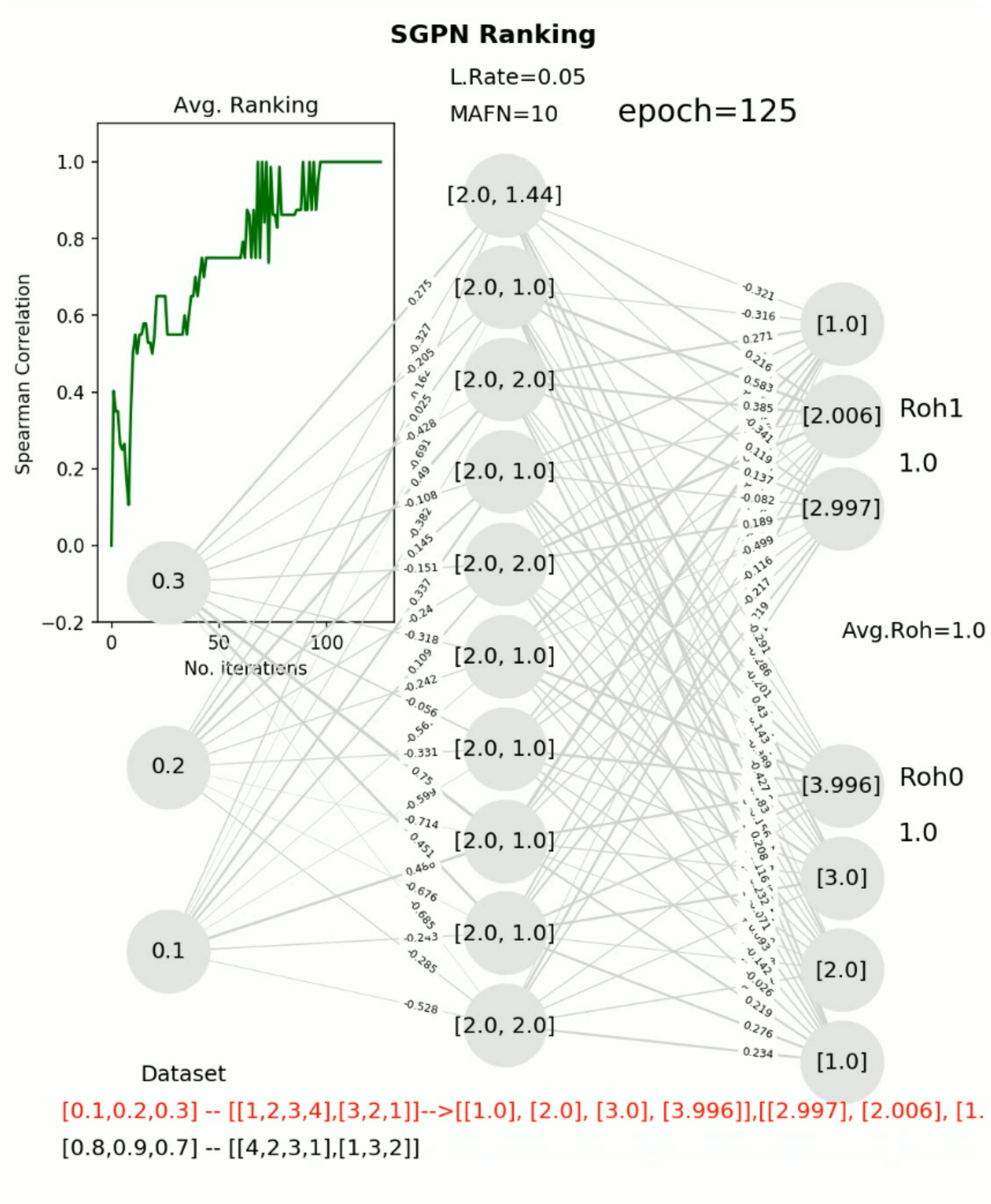
5.6.1 Convergence Fluctuation

The data set wine-stock and iris-stock take a longer time for convergence due to data separability and complexity; thus, convergence for each group of labels is not linear and have many fluctuations more than the ranking of single label group, these fluctuations are not related to the gradient error in ranking, but it is the average ranking between two subgroups as each subgroup tend to increase the ranking it updates its weights which reflect on the common weights which may reduce the convergence of the second group. The fluctuation is shown in the video link of convergence of two groups using toy dataset (Elgharabawy 2020*c*) and shown in Fig 5.6. The convergence fluctuations are not noticed when we use three subgroups together as in the iris-wine-stock dataset using the same hyper-parameters of two subgroups *SGPNN*.

5.7 Conclusions and Future Works

The *SGPNN* is a new step in preference learning to predict the subgroups from conjoint data by proposing a simple three layers *FF* network that has different outputs to build the conjoint model from a different group of data. This chapter proposes a simple network with one middle layer and a new activation function to speed up the learning to rank using the new Spearman objective function. This chapter introduces the novel *MAFN* to serve more than one group of labels. In addition, creating conjoint data from multiple datasets reinforce the learning to rank and enhance accuracy. The proposed network with one middle layer simplifies the process of *FF*, *BB* and *UW* in three steps for middle and output layer comparing to the conventional *ANN*.

The future work of *SGPNN* is to coupling the relation with different *SD* methodologies to rank the subgroup. The data used in the experiment are relatively tiny; thus *SGPNN* opens a road to develop a deep learning network based on *MAFN*, *PNN*, Spearman error function, and *SS* function to accelerate the learning to build a more complicated conjoint model. The *SGPNN* integrates with *SD* to study the relations, similarity, and separability from different domains to have a shared learning model.



[0.1,0.2,0.3] -- [[1,2,3,4],[3,2,1]]-->[[1.0], [2.0], [3.0], [3.996]],[[2.997], [2.006], [1.

[0.8,0.9,0.7] -- [[4,2,3,1],[1,3,2]]

Figure 5.6: video image of *SGPNN* convergence using toy dataset

CHAPTER 6

Preference networks in BCI Application

6.1 Introduction

MOTOR Imagery brain-computer interface (MI-BCI) learns the EEG brain signal imagery movement activities to control and execute machine by commands (Ince et al. 2009). Imagery movement signal task requests the patient to imagine certain mental tasks and learn the response EEG signal by signal processing and machine learning and convert it to commands. Many applications have been proposed for self-based BCIs, such as P300 and motor imagery. Motor imagery (*MI*) is a mental image of a motor act of any human body peripherals without real movement. Many studies and approaches were introduced for classification multi-class *MI*[]. Many features selections in the time and frequency domain and classification methods have been introduced for motor imagery signals such as SVM and decision tree (Polat & Güne 2007). Most BCI *MI* methods depend on human knowledge in the neuroscience field and personal experience. But, due to the variation of human EEG behavior and limited human knowledge. the best features may not be extracted well for each subject. Recently, the conventional EEG signal feature recognition has been used by different Machine learning classifiers. Most of the classifiers use time, frequency, or both domains (Ince et al. 2005). Another MI approach uses wavelet packets then dynamic frequency feature selection (DFFS) to select the feature with the highest classification accuracy for each experimental object for feature extraction (Li et al. 2017)

Recently, deep learning *DL* gained much attention in computer vision, speech recognition, and recommendation systems. *DL* has achieved great success. Because the deep learning method is learning the input signal features, it solves the problem of the manual design of features extraction (Xiao & Fang 2021),(Chu et al. 2018)., *CNN* which is one of *DL* algorithms, has become the most the method used in the motor imagination EEG classification algorithm because of its better feature extraction capabilities.Lawhern et al. (2018) proposed a *CNN* structure that can be applied to a variety of popular brain-computer interface paradigms (including motor imagination, P300 (Lawhern et al. 2018). Class overlapping is a common problem in MI-EEG data where data point has more than one opposite class. which is called data ambiguity as shown in Fig. **??**

From the previous approaches, MI classification has the following drawbacks

1) Most of the classifiers does not solve the problem of data ambiguity where the data has class overlapping.

1) *CNN* uses a fixed kernel window shape that is designed for 2D image classification because the good image feature is determined by the image edges and curves in x and y axes. However, the EEG Sig-

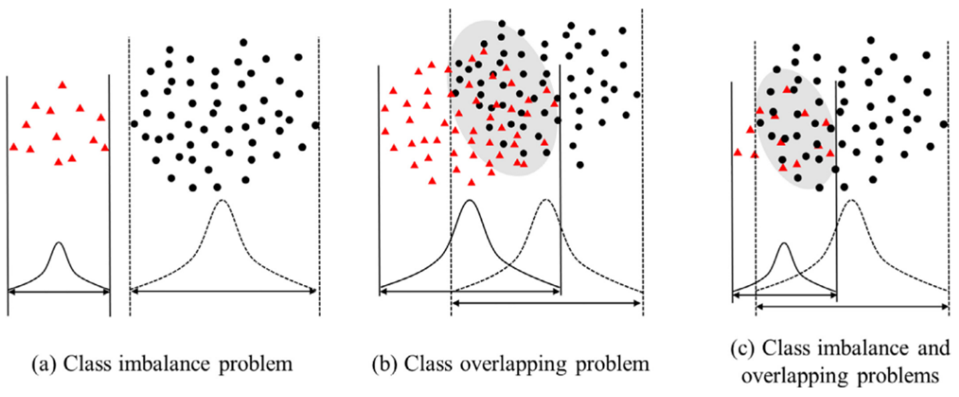

(a) Class imbalance problem
(b) Class overlapping problem
(c) Class imbalance and overlapping problems

Figure 6.1: the MI-EEG has BCI competion IV 2B has two problems, class overlap, unbalance data and both in (a), (b), (c) respectively.

nal feature is a spectrum part in 1-D data in time and in a certain frequency range.

2) Most classification approaches does not consider the relation between features per instance as ranked.

this chapter proposes a motor imagery EEG signal recognition approach based on a *SGPN* network. The main contributions in this research are as follows:

1) Proposing a new algorithm to convert non-ground truth and class overlap to unique data by creating a group of labels for each ambiguous instance.

2) *SGPNN* uses a ranking approach and the smooth staircase *SS* as an activation function that enhances the predictive probability over the *sigmoid* and *Softmax* due to the step shape that enhances the predictive probability from a range from -1 to 1 in the sigmoid to almost discrete multi-values.

3) Identify a threshold for data separability and ambiguity.

6.2 Material and Methods

6.2.1 Converting Classification into Ranking

The proposed approach is based on converting the multi-class classification problem into a multi-label ranking problem by two points:

1) Converting the class vector to binary class matrix and the binary digits are the output neurons values i.e: for class 1,2,3 and 4 are presented as ranked labels (2,1,1,1),(1,2,1,1),(1,1,2,1),(1,1,1,2) respectively.

2) the input of the *SGPN* is all the feature vector of one data instance. not feature row as in convention *ANN*.

6.3 Data Preparation

This section describes the ranking unification preprocessing and *SGPN* learning steps (*FF*, *BP* and *UW*).

6.3.1 SGPN to solve the problem of class overlap

The idea of the proposed *SGPN* is based on solving the low classification accuracy due to non-ground truth data because of class overlapping.

$$S_i = \{f_i, li\} \quad , \quad S_j = \{f_j, lj\} \tag{6.1}$$

$$\theta_{over_i} \le abs\Big(\sum_{x=0,1,..}^{m} f_{xi} - f_{xj}\Big) \tag{6.2}$$

$$\theta_{sep_i} \geq abs\left(\sum_{x=0,1,..}^{m} f_{xi} - f_{xj}\right) \tag{6.3}$$

The threshold of class overlap of any two data instances by θ_i is equal or less than the sum of the absolute difference of two instances i, j of all # features m. Therefore, for two duplicate instances $\theta = 0$.

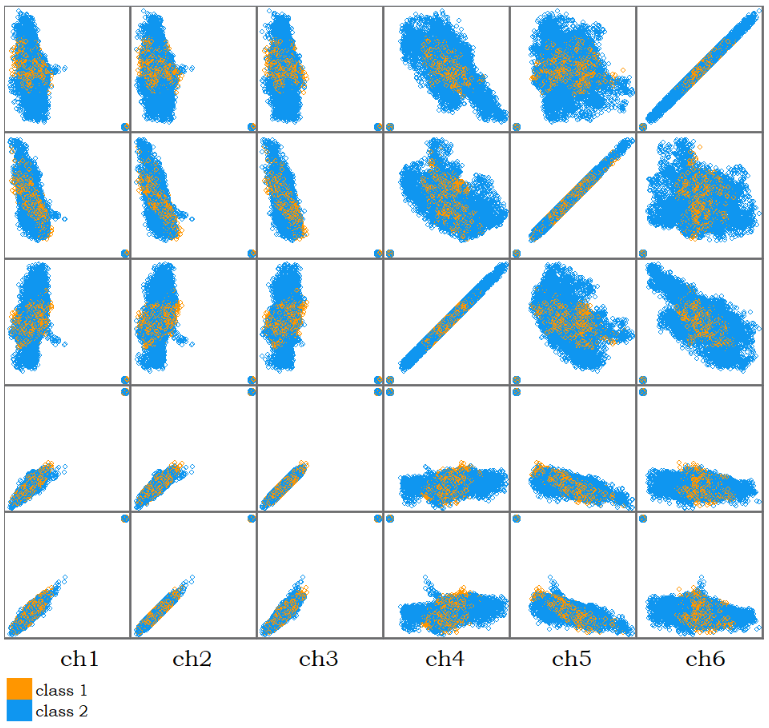


Figure 6.2: Matrix plot of the dataset 2b where data are unbalanced and separability is low

6.3.1.1 Ranking Unification

We introduce a new method for creating label ranking ground truth by converting the unrestricted ranking to restricted ranking by unifying the data instances and adding subgroups to the labels. The percentage of a unique ranking is measured using Eq.6.4

$$U_\pi = \frac{\#distinct\ rankings}{n} \tag{6.4}$$

The number of subgroups is determined by the maximum number of repeated records with different class labels using Eq.6.5

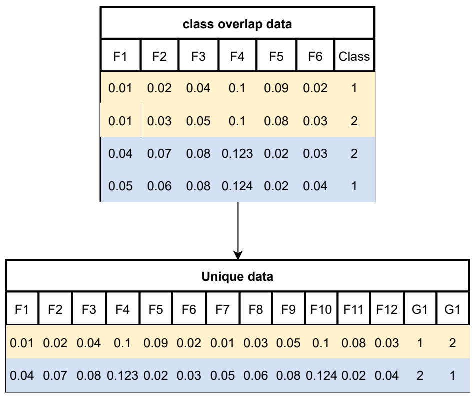
$$\#sg = Max(x_r) \tag{6.5}$$

where $\#sg$ is the number of subgroups and x_r is the number of duplicated data records for different classes.

This chapter applies algorithm 7 to convert the data from non-restricted rankings with no ground truth to restricted label ranking by removing duplicated data instances and accumulating the corresponding labels in a subgroup. The algorithm removes the duplication and assigns the corresponding labels as a subgroup to one unique data record. For non-repeated records, the additional subgroup has values of zero.Fig. 6.3 (a) illustrate the duplicate records has opposite class and separate the label in two subgroups. and using *SGPNN* and *PNN* for the overlapping data in Fig. 6.3(b).

Algorithm 7: Class overlap Unification.

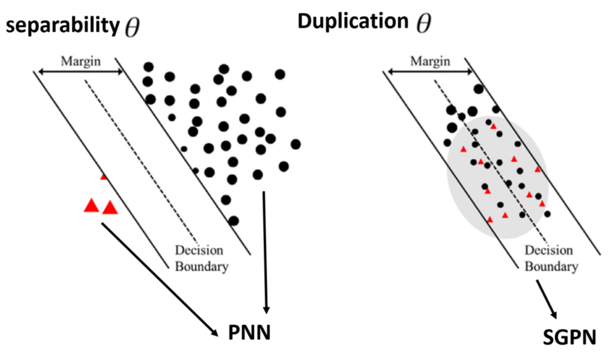
```
Data: $\mathcal{D} \in \{\langle x_n, \pi_n \rangle\}$
while $x_i$ do
    if $x_i \notin D[i]$ then
        Dist $\leftarrow x_i$ ;
    else
        while $j \in \mathcal{D}$ do
            if $x_i$ in $\mathcal{D}[j]$ then
                m++ # of subgroups;
                D $\leftarrow (j, \pi_i)$ ;
        end
    end
end
Result: $\mathcal{D} \in \{\langle x_n, (\pi_{n1} \perp \pi_{n2} \perp \pi_{n3} ... \perp \pi_{nm}) \rangle\}$
```



class overlap data						
F1	F2	F3	F4	F5	F6	Class
0.01	0.02	0.04	0.1	0.09	0.02	1
0.01	0.03	0.05	0.1	0.08	0.03	2
0.04	0.07	0.08	0.123	0.02	0.03	2
0.05	0.06	0.08	0.124	0.02	0.04	1

Unique data													
F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	G1	G1
0.01	0.02	0.04	0.1	0.09	0.02	0.01	0.03	0.05	0.1	0.08	0.03	1	2
0.04	0.07	0.08	0.123	0.02	0.03	0.05	0.06	0.08	0.124	0.02	0.04	2	1

(a)



(b)

Figure 6.3: Example of class overlapping unification in (a), Using *SGPNN* and *PNN* to learn the classoverlap and separable data in (b)

6.3.2 *SGPNN* Learning Steps

This section shows the *FF*, *BP* and updating weights (*UW*) processes in the middle and output layer of the *SGPNN*.

6.3.2.1 Middle layer *FF*

the output Y of single a *MAFN* per subgroup j is shown in Eq. 6.6

$$Y_j = \varphi_j \sum_{k=1}^{d} x_k \cdot wm_k \Bigg|_{j=0}^{g} \tag{6.6}$$

where g is the number of subgroups, d is the number of input features, and φ_j is the activation function per subgroup.

6.3.2.2 Output layer *FF*

The output of single neurons is shown in Eq. 6.7

$$Y_j = \varphi_j \sum_{k=1}^{m} x_{jk} \cdot wo_{jk} \Bigg|_{j=0}^{g} \tag{6.7}$$

where m is the number of *MAFN*s.

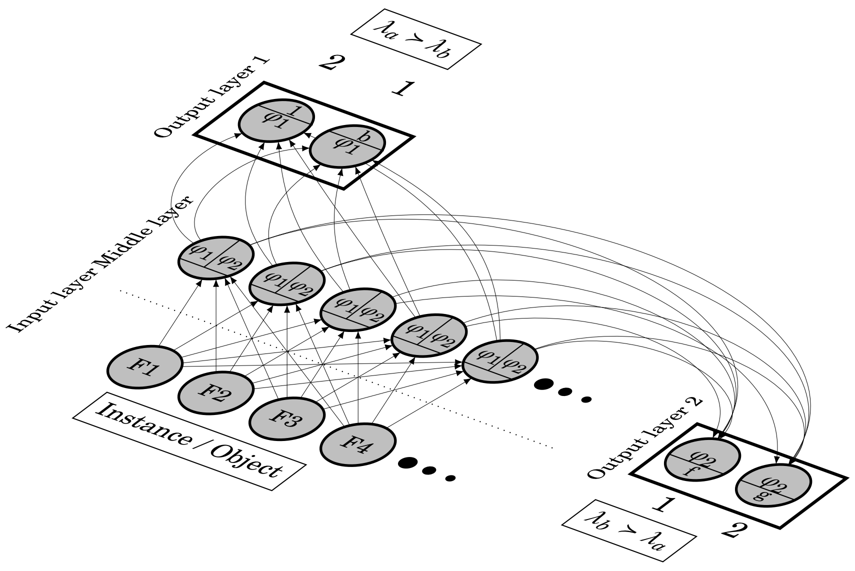


Figure 6.4: SGPNN architecture used for classification BCI competition VI dataset 2B , each group has 2 labels represents the opposite class classification, where $\varphi 1_n$=2, $\varphi 2_n$=2, $f_{in} = 6$.

6.3.3 Architecture and Data Processing

We propose a new PN to classify EEG signals according to ranking the data. The architecture of *SGPNN* is shown in Fig. 6.4. Where two subgroups. Each group is for one class as shown in the steps are shown in Fig **??**

1) Data Preprocessing: The EEG signal of 6 features is processed using SDFT to generate power spectrum in 2D with width and height 129 X 33 for all frequency range per each data instance. The spectrum is flattened to 4257 features.

6.4 *MI-EEG* Dataset

The dataset is used in this research is BCI Competition IV Dataset 2b for motor imagery classification (Tangermann, Müller, Aertsen, Birbaumer, Braun, Brunner, Leeb, Mehring, Miller, Mueller-Putz, Nolte, Pfurtscheller, Preissl, Schalk, Schlögl, Vidaurre, Waldert & Blankertz 2012). The next section describes the data.

6.4.1 Dataset 2B

The data has total six channels (0.5-100Hz; notch filtered) , 250Hz sampling rate, two classes represent left and right-hand imagery, and nine subjects. Five sessions were recorded from each subject. Three sessions for training and two for testing. (Tangermann, Müller, Aertsen, Birbaumer, Braun, Brunner, Leeb, Mehring, Miller, Mueller-Putz, Nolte, Pfurtscheller, Preissl, Schalk, Schlögl, Vidaurre, Waldert & Blankertz 2012) The datasets are collected from the Graz University of Technology repository (Tangermann, Müller, Aertsen, Birbaumer, Braun, Brunner, Leeb, Mehring, Miller, Mueller-Putz, Nolte, Pfurtscheller, Preissl, Schalk, Schlögl, Vidaurre, Waldert & Blankertz 2012).

6.5 Experiment Flow

Dataset 2b has six-channel and three sessions. The three sessions are accumulated in one file in order to build a generalized model as shown in step 1 in Fig. **??** similarly, dataset 2b has three sessions for training and two sessions for testing. Each session file has a 3X6X604803 data bin. The data is flattened, which leads to an increase in the total number of features for the data used by *SGPNN* as mentioned in step 3. The data is divided into 80% for training the model and 20% for data validation. Five-fold cross-validation to build the model due to large data size and high variance between training and testing results. And for each fold, we validate the model. as in step 5. after each tenfold, we change the hyperparameters; the parameters are learning rate, number of hidden neurons, the scaling value of input data, and *SS* boundaries. Steps 5, 6, and 7 are repeated sequentially for all the hyperparameters. Moreover, generate the results. The best results of the model and hyperparameters are used for testing the unseen testing data of the competition in step 8 in Fig. **??**.

6.6 Classification Results

Dataset 2b is challenging in classification due to high biased and data separability is low, then the classes are binaries for multi-label ranking. In the experiment, we use 80% of the data for training and 20% for testing; then we apply five-fold cross-validation on the training data to build the network model. We use sequential search to use the best hyperparameters for the training, the hyperparameter are the θ_{dub}, θ_{sep} # hidden neuron, scaling input data and *SS* function boundaries and learning rate. This section describes the comparison between other recent approaches on the same dataset as shown in Table 5.1. As shown in Fig., the training is highly biased starting from the first epoch. However, validation is too low due to the large data size, and MI-EEG is highly biased. Therefore we increased the number of folds to 20. The accurecy is measured by ranking reduction to classification using score function by converting the output ranking to binary classification using scorer function s: $\mathcal{L} \rightarrow \mathbb{R}$. For output labels to choose the max value $f_s = Max(\lambda)$ for the binary classification that has c : $\mathcal{X} \rightarrow \{\pm 1\}$.

Table 6.1 shows the classification per each subject

6.7 Discussion

Table 6.1 it shows that one subgroup of labels gives positive results compared to the second group. which means *SGPNN* discover two different

Table 6.1: subjects correlation ρ and accuracy using *PN EEG* classification *BCI* Competition *IV* Dataset *2B*

Subject	Avr. three Sessions					
	FBCSP	CNN.	CNN-SAE	CNN-VAE	SGPN	
					G1	G2
1	0.546	0.488	0.517	0.522	-0.9	0.8928
2	0.208	0.289	0.324	0.346	-0.2	0.5235
3	0.244	0.427	0.494	0.436	-0.9	0.3421
4	0.888	0.888	0.905	0.908	-0.16	0.821
5	0.692	0.593	0.655	0.646	-0.61	0.7452
6	0.534	0.495	0.579	0.642	0.6	0.99
7	0.409	0.409	0.488	0.550	-0.3	0.642
8	0.413	0.443	0.494	0.506	-0.2	0.875
9	0.583	0.415	0.463	0.518	0.389	0.712
Avrg.	0.583	0.415	0.463	0.518	-0.25	0.7271

Table 6.2: subjects correlation ρ and accuracy using *PN EEG* classification *BCI* Competition *IV* Dataset *2B* and data separability and duplication θ

Subject	ρ	Acc.	dup.θ	sep.θ
01	-0.9	0.8928	0.01	0.5
02	-0.2	0.7235	0.001	0.1
03	-0.9	0.3421	0.01	1
04	-0.16	0.821	0.001	0.7
05	-0.61	0.7452	0.01	0.2
06	0.6	0.99	0.001	0.1
07	-0.3	0.642	0.01	0.3
08	-0.2	0.875	0.001	0.2
09	0.389	0.712	0.01	0.5

models, one model is the correct classification of the ambiguous data instances. Table 6.2 show the θ as a hyperparameter is chosen to get the best accuracy. The hyperparameters are , θ_{dup} for overlapped data and θ_{sep} for separability.

6.8 Conclusion

Class overlapping and unbalanced data are challenging problems in machine learning. The proposed algorithm of sub-grouping the labels shows a significant enhancement in the classification by converting binary class labels into two groups. One of the two groups shows significant high ranking results comparing to the other. This sub-grouping approach could be using in different datasets has class overlapping and unbalance data.

Chapter 7

Conclusion and Future Work

The superiority of *PNN* is solving both classification and ranking problems. The ranking is used in input data as a feature selection criterion is a novel approach for deep learning.

Encoding the labels preference relation to numeric values and rank the output labels simultaneously in one model is an advanced step over pairwise label ranking based on classification. *PNN* could be used to solve new preference mining problems. One of these problems is incomparability between labels, where Label ranking has incomparable relation $\perp$, i.e., ranking space $(\lambda_a \succ \lambda_b \perp \lambda_c)$ is encoded to (1, 2, -1) and $(\lambda_a \succ \lambda_b) \perp (\lambda_c \succ \lambda_d)$ is encoded to (1, 2, -1, -2). *PNN* could be used to solve new problem of non-strict partial orders ranking, i.e., ranking space $(\lambda_a \succ \lambda_b \succeq \lambda_c)$ is encoded to (1, 2, 3) or (1, 2, 2). Future research may focus on modifying *PNN* architecture by adding bias and solving problems of extreme multi-label ranking.

This thesis proposed a novel method to rank a complete multi-label space in output labels and features extraction in both simple and deep learning. *PNN* and *PN* are native ranker networks for image classification and label ranking problems that uses *SS* or *PSS* to rank the multi-label per instance. This neural network's novelty is a new kernel mechanism, activation, and objective functions. This approach takes less computational time with a single middle layer. It is indexing multi-labels as output neurons with preference values. The neuron output structure can be mapped to integer ranking value; thus, *PNN* accelerates the ranking learning by assigning the rank value to more than one output layer to reinforce updating the random weights. *PNN* is implemented using python programming language 3.6, and activation functions are modeled using wolfram Mathematica software Inc. (n.d.). A video demo that shows the ranking learning process using toy data is available to download Elgharabawy (2020*c*). This thesis also proposes a new novel architecture of the neural network to support sub-group label ranking by introducing a new type of ranking problem and dataset. Also propose a novel solution to solve the class overlap problem of complex classification datasets, i.e., EEG.

SGPNN is a new step in preference learning to predict the subgroups from conjoint data by proposing a three layers *FF* network that has different outputs to build the conjoint model from a different group of data. However, the data used in the experiment is relatively tiny; thus, *SGPNN* opens a road to develop a deep learning network based on *MAFN*, *PNN*, spearman error function, and *SS* function to accelerate the learning for big data deep ranking to build a more complicated conjoint model. The future work of *SGPNN* is coupling the relation with different *SD* methodologies to rank the subgroup. The future work on the *SGPNN* will be ranking an excessive number of labels and subgroups of labels, which may be challenging in terms of computational time. *SGPNN* could be used in many potential applications, i.e., brain-computer interface

(*BCI*) applications where EEG data are ambiguous, complicated, and have class overlap. Another medical application where data fusion is collected from different sensors, i.e., the study of human emotions recognition. *SGPNN* could be part of an expert system to build accumulated models for judgment, elections, medical diagnosing from different conjoint historical data. Also, the future work of the study of the data similarity, separability of the data from different domains to have a shared learning model, i.e., iris, wine, and stock datasets.

Acknowledgement

This work was supported in part by the Australian Research Council (ARC) under discovery grant DP180100656 and DP210101093. Research was also sponsored in part by US Office of Naval Research Global under Cooperative Agreement Number ONRG - NICOP - N62909-19-1-2058 and AFOSR DST Australian Autonomy Initiative agreement ID10134. We also thank the NSW Defence Innovation Network and NSW State Government of Australia for financial support in part of this research through grant PP21-22.03.02.

APPENDIX A

A.1 *PNN* Proof of Convergence

The output of preference neuron a is obtained from the activation function φ given in Eq. (37)

$$a_j = \varphi(\omega^T.a_i) \tag{A.1}$$

where a_i and a_j are neuron input and output, respectively. Therefore, the neural output behavior is shown in Eq. (38)

$$\{a_1, t_1\}, \{a_2, t_2\}, .., \{a_j, t_q\} \tag{A.2}$$

where each target output t_q is the preference value equal to 1, 2, 3,.., n.

The total inputs to the preference neuron is calculated as the following after neglecting the bias.

$$a_j = \omega^T.a_i = \omega^T.z \tag{A.3}$$

The weighted vector is given by Eq. (40).

$$\omega_{new} = \omega_{old} + E_{err}.z \tag{A.4}$$

where E_{err} is the ranking error value from 0 to n.

After k iterations for which the weight changes, the learning process is shown in Eq. (41).

$$\omega(k) = \omega(k-1) + z'(k-1) \tag{A.5}$$

The solution weighted vector ω_s ranks all the input Q correctly. $z'(k-1)$ is the appropriate member of the set as shown in Eq. (41)

$$z_1, z_2, z_3, .., z_Q. \tag{A.6}$$

To get preference value for 1, 2 and 3, then t_q=1, 2 and 3 as given in Eqs. (43)-(45).

$$\omega_s^T z_1 > \delta > 0 \tag{A.7}$$

$$\omega_s^T z_2 > \delta > 1 \tag{A.8}$$

$$\omega_s^T z_3 > \delta > 2 \tag{A.9}$$

The objective of the proof of convergence is to find the upper and lower bounds on the length of the weighted vector. After k iterations, it can be represented as Eq. (46)

$$\omega(k) = z'(0) + z'(1) + .. + z'(k-1) \tag{A.10}$$

By taking the inner product of the solution weighted vector ω_s with the weight vector of k iteration, we can obtain Eq. (47)

$$\omega_s^T.x(k) = \omega_s^T.z'(0) + \omega_s^T.z'(1) + .. + \omega_s^T.z'(k-1) \tag{A.11}$$

Eqs. (42) and (43) are substitutes as in Eq. (48)

$$\omega_s^T.z'(i) > \delta \tag{A.12}$$

Therefore,

$$\omega_s^T.\omega(k) > k\delta \tag{A.13}$$

From the Cauchy-Schwarts inequality Bergelson (2008)

$$(\omega_s^T.\omega(k))^2 \leq \| \omega_s \|^2 \| \omega(k) \|^2 \tag{A.14}$$

where

$$\| \omega \|^2 = \omega^T \omega \tag{A.15}$$

From Eq. (49) we can put the lower bound on the squared length at iteration k :

$$\| \omega(k) \|^2 \geq \frac{(\omega_s^T \omega(k))^2}{\| \omega_s \|^2} > \frac{(k\delta)^2}{\| \omega_s \|^2} \tag{A.16}$$

To find an upper bound for the length of weight vector, the change in the length at iteration k is given in Eq. (53)

$$\| \omega(k) \|^2 = \omega^T(k).\omega(k) \tag{A.17}$$

$$\| \omega(k) \|^2 = [\omega(k-1) + z'(k-1)]^T [\omega(k-1) + z'(k-1)] \tag{A.18}$$

$$\| \omega(k) \|^2 = \omega^T(k-1)\omega(k-1) + 2\omega^T(k-1)z'(k-1) + z'^T(k-1)z'(k-1) \tag{A.19}$$

Eq. (A17) can be simplified as

$$\| \omega(k) \|^2 \leq \| \omega(k-1) \|^2 + \| z'(k-1) \|^2 \tag{A.20}$$

Eq. (A17) can be repeated for $\| \omega(k-1) \|^2$, $\| \omega(k-2) \|^2$, to obtain

$$\| \omega(k) \|^2 \leq \| z'(0) \|^2 + \| z'(1) \|^2 + ... + \| z'(k-1) \|^2 \tag{A.21}$$

If $\Pi = max\{\| z'(i) \|\}$, this upper bound can be simplified to Eq. (A22).

$$\| \omega(k) \|^2 \leq K\Pi \tag{A.22}$$

The weights only change to a finite number of times because k has an upper bound. Therefore, the *NN* learning converges after a finite number of iterations.

A.2 Supplemental Material

A.3 Video Files

Ranker NN Convergence	(Elgharabawy 2021*a*)
PNN Convergence	(Elgharabawy 2021*b*)
SGPNN Convergence	(Elgharabawy 2021*d*)
SGPNN log file	(Elgharabawy 2021*c*)

A.4 Dataset Files

A.4.1 Synthesized data used for SGPNN

Restaurant food and services surveys	(Elgharabawy 2020*d*)
German Election 2005 and 2009	(Elgharabawy 2020*a*)
Emotions	(Elgharabawy 2020*b*)
Sushi	(Elgharabawy 2020*e*)

Appendix B

B.1 Symbols and Abbreviations

PNN	preference neural network
PN	preference net
SGPNN	subgroup preference neural network
FF	feed forward
BP	back propagation
UW	updating weights
$\mathcal{D} \in \{x_1, x_2, \ldots, x_d\}$	single data instance
$\mathbb{X}$	all instances space
$\pi \in \{\lambda_{y_1}, \ldots, \lambda_{y_n}\}$	labels permutation space
$L \in \{\lambda_a, \lambda_b, \lambda_c\}$	all labels space
ρ	spearman ranking correlation
τ	Kendall ranking correlation
π	permutation label space
φ_n	activation Function for n label ranking

λ_a	preference value of label a
ps	positive stair Activation Function
ss	stair step activation function
w	ss function step width
f_{in}	number of features
rms	root mean square
n	number of labels = number of stair steps-1
$K_{w,h}$	ranker kernel has width and height
$\perp$	incomparable preference relation
$\succ$	strict preference relation
$\not\prec$	weak preference relation
$\simeq$	indifference preference relation

APPENDIX C

C.1 Video Snapshoot of *PNN* Ranking Convergence.

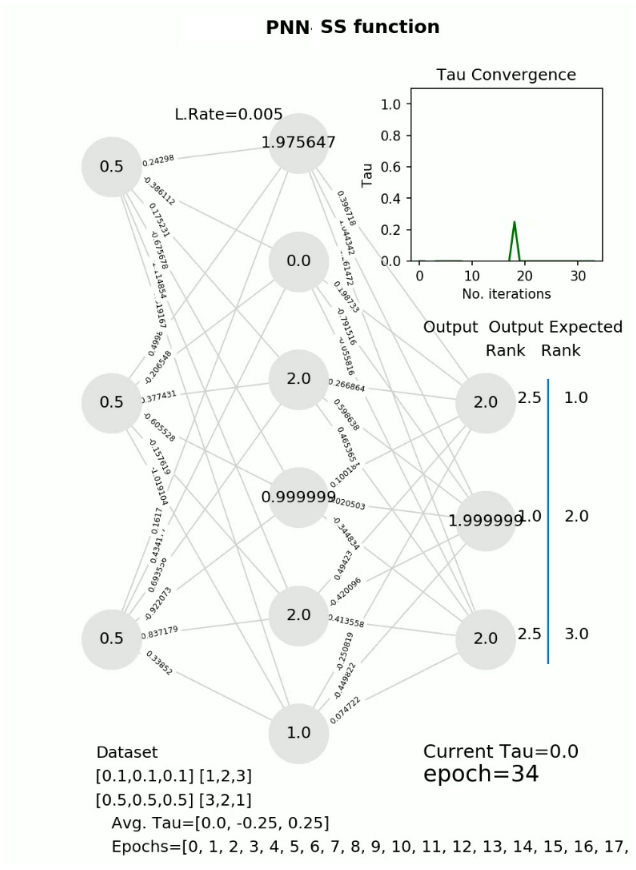


Figure C.1: *PNN* at epoch 34

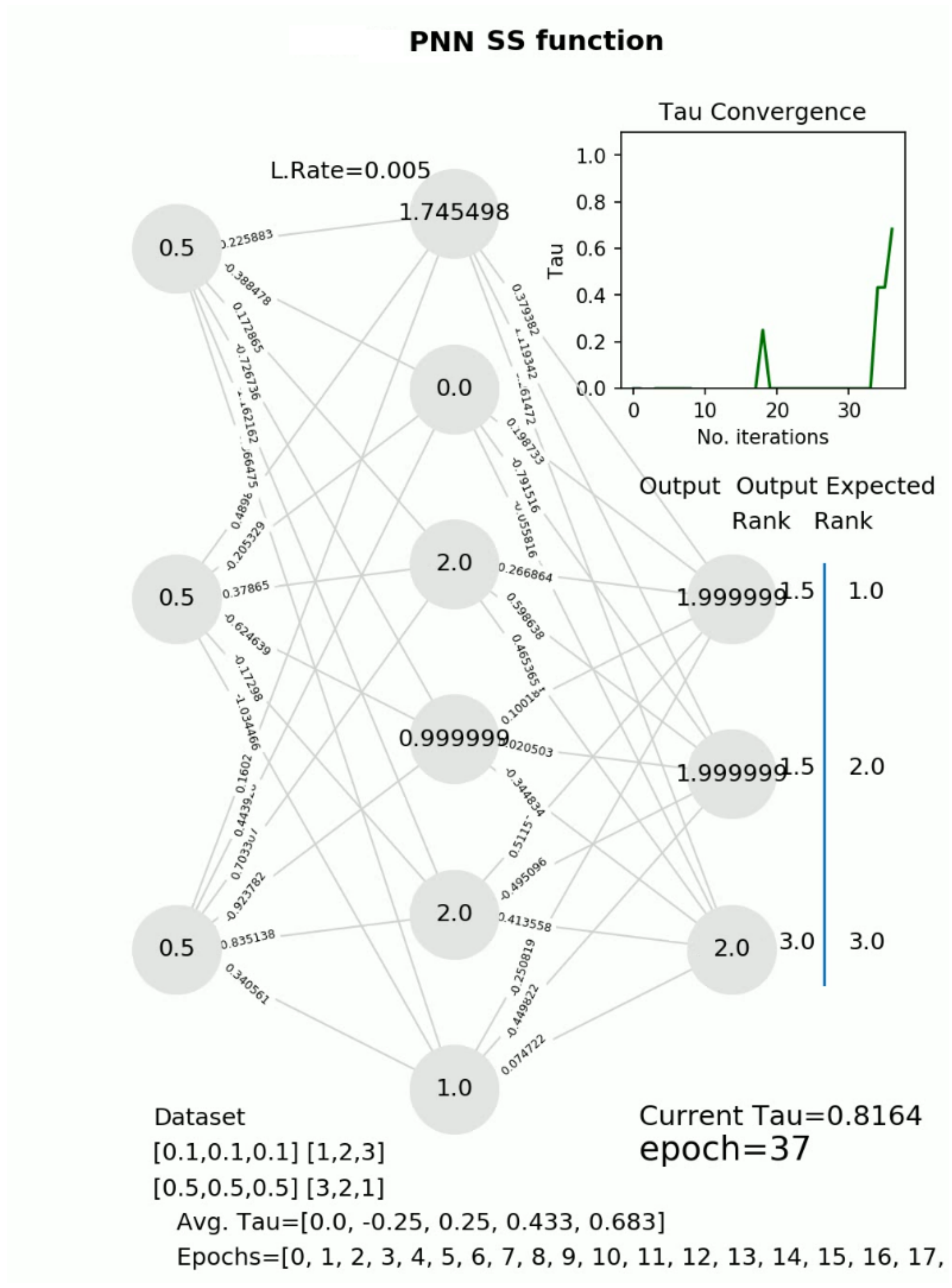
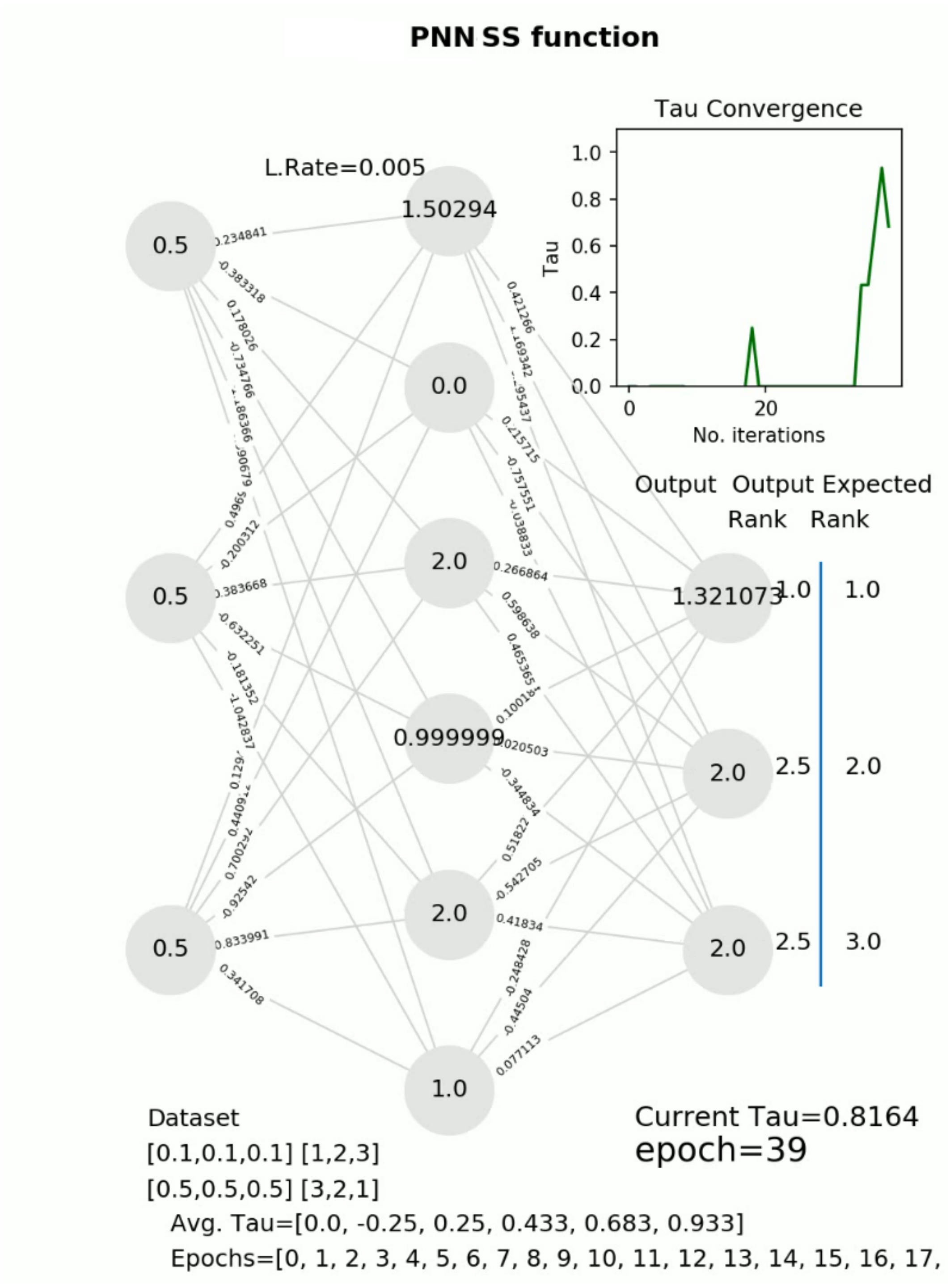


Figure C.2: *PNN* at epoch 37



Dataset
[0.1,0.1,0.1] [1,2,3]
[0.5,0.5,0.5] [3,2,1]
Avg. Tau=[0.0, -0.25, 0.25, 0.433, 0.683, 0.933]
Epochs=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,

Figure C.3: *PNN* at epoch 39

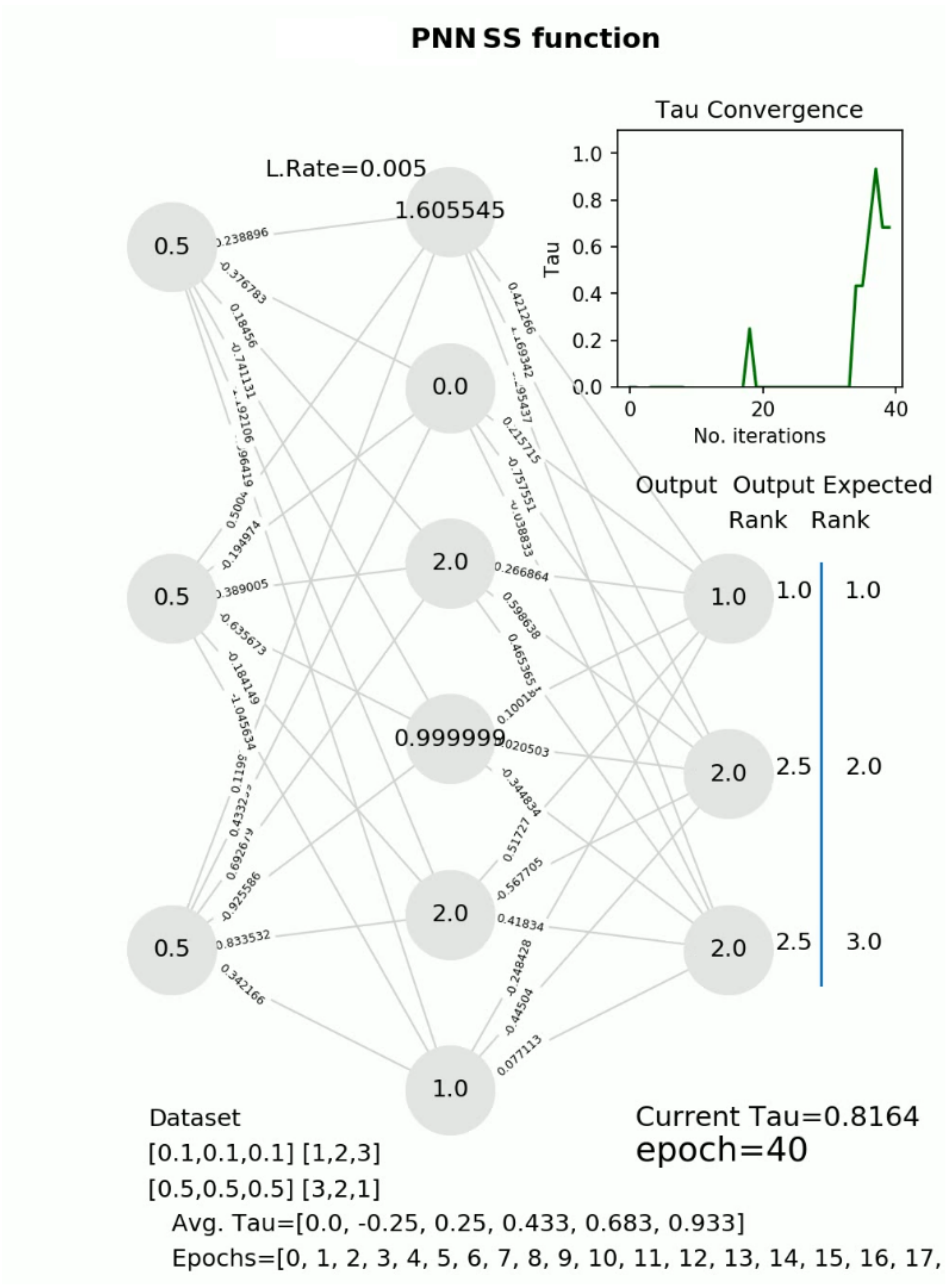


Figure C.4: *PNN* at epoch 40

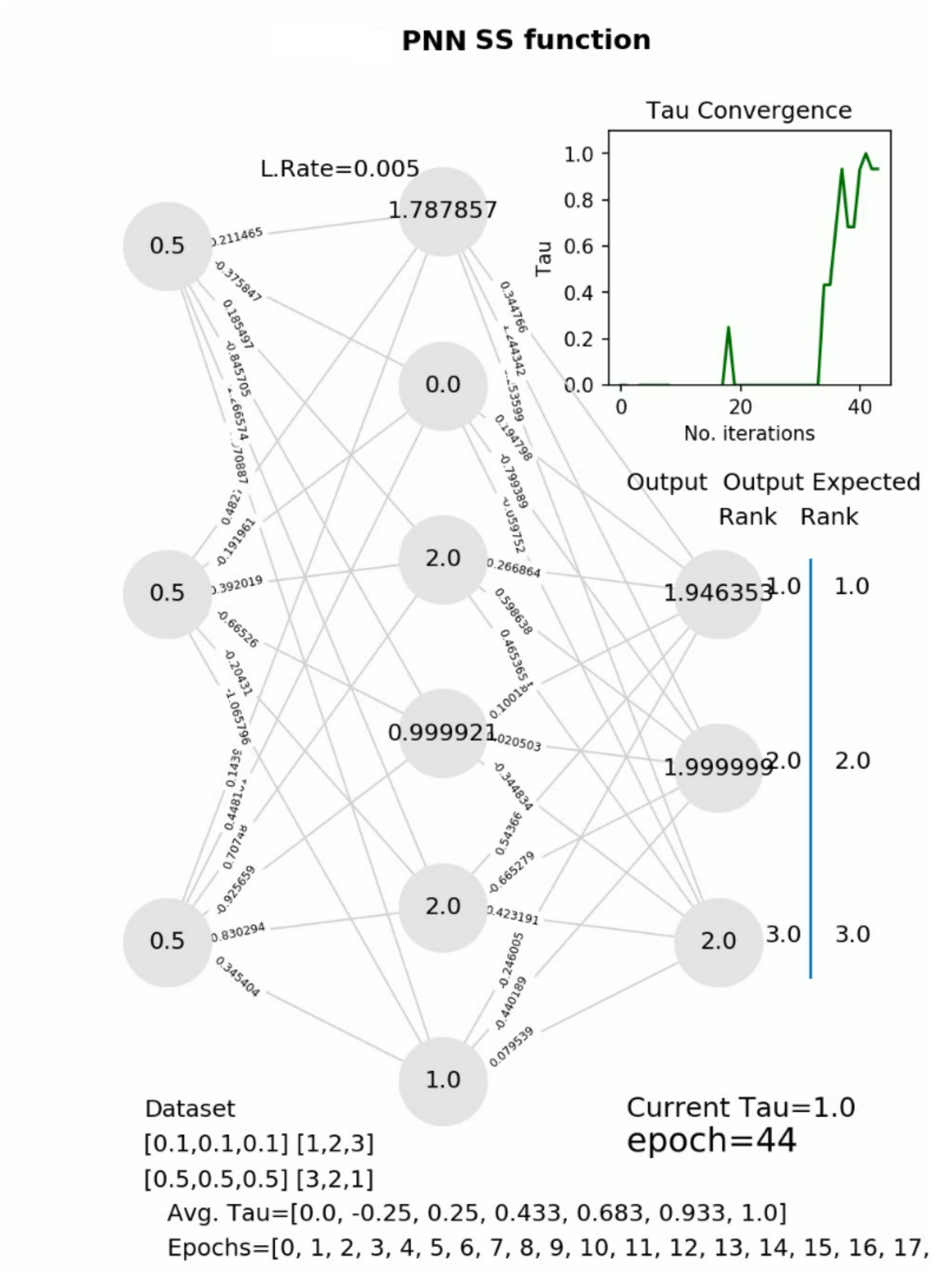
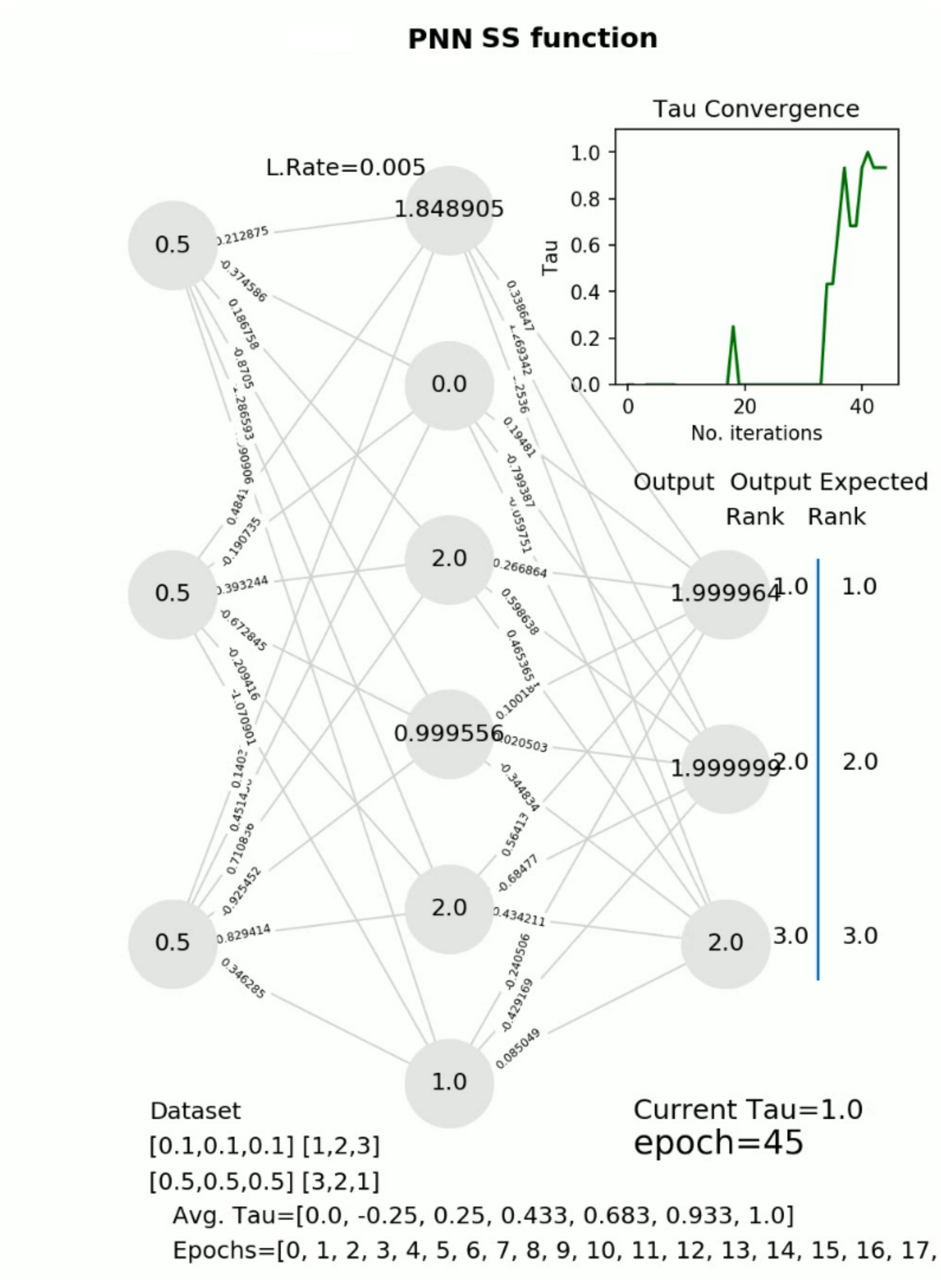
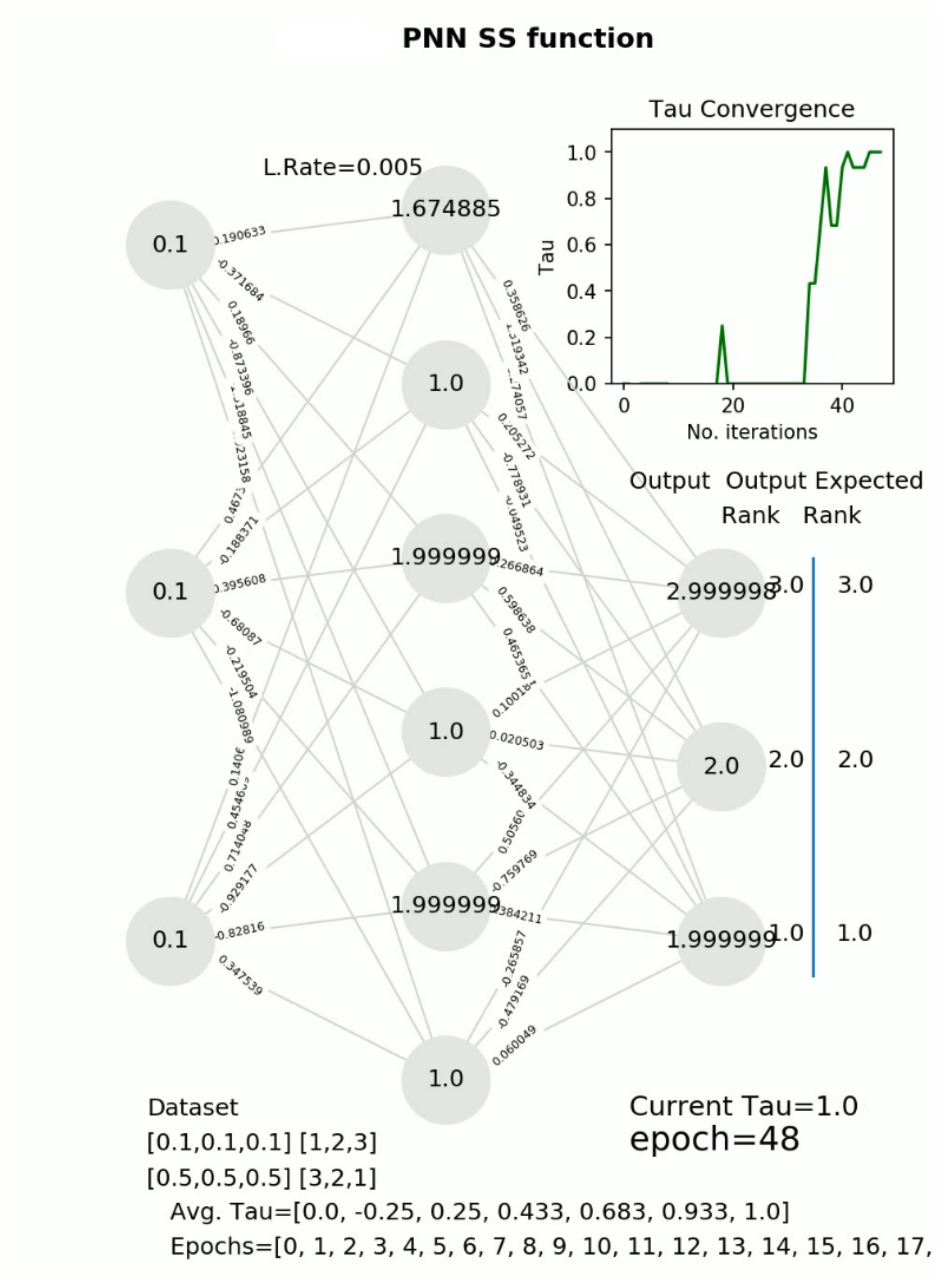


Figure C.5: *PNN* at epoch 44



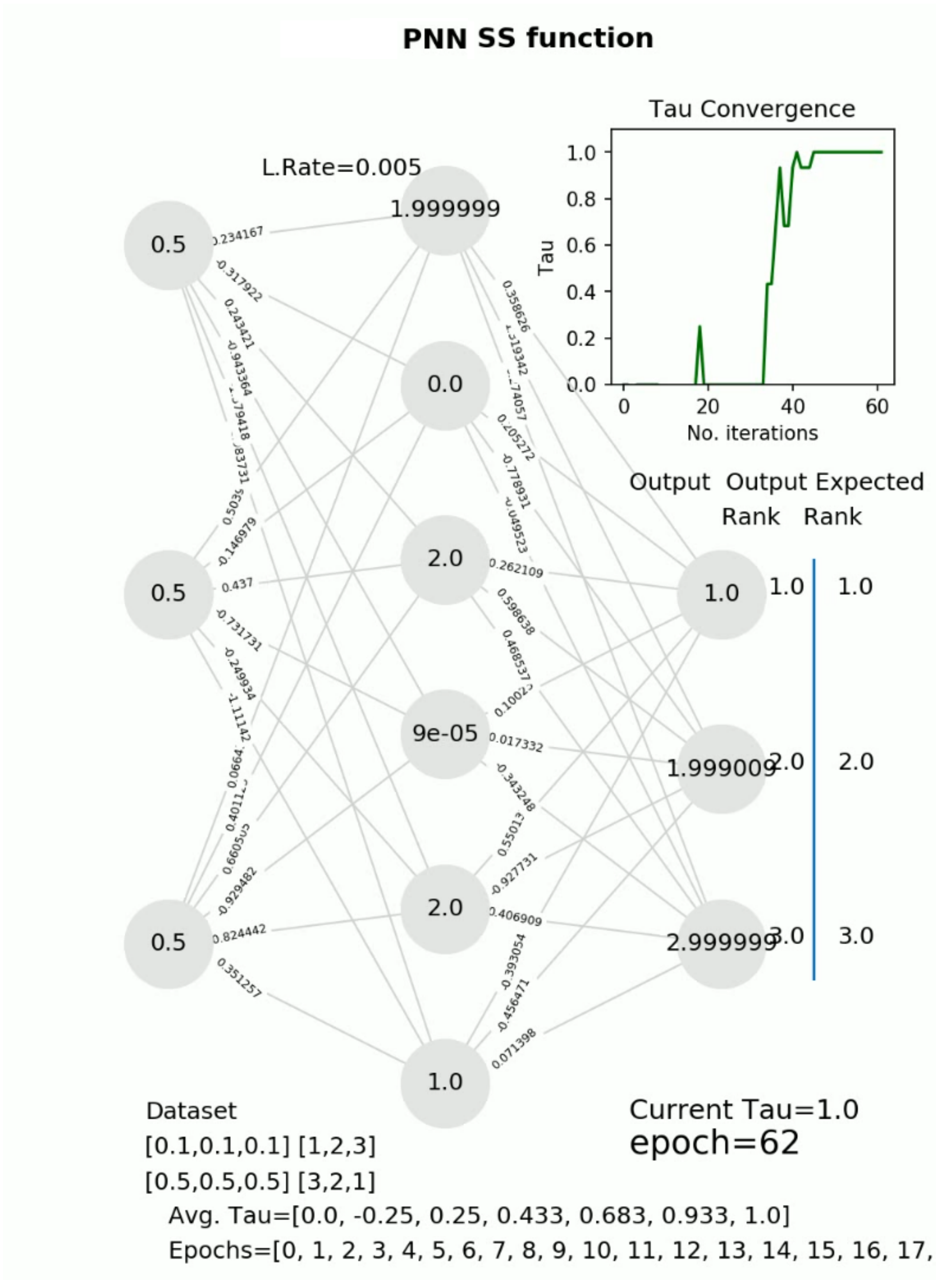
```
Dataset
[0.1,0.1,0.1] [1,2,3]
[0.5,0.5,0.5] [3,2,1]
   Avg. Tau=[0.0, -0.25, 0.25, 0.433, 0.683, 0.933, 1.0]
   Epochs=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
```

Figure C.6: *PNN* at epoch 45



Dataset
[0.1,0.1,0.1] [1,2,3]
[0.5,0.5,0.5] [3,2,1]
Avg. Tau=[0.0, -0.25, 0.25, 0.433, 0.683, 0.933, 1.0]
Epochs=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,

Figure C.7: *PNN* at epoch 48



Dataset
[0.1,0.1,0.1] [1,2,3]
[0.5,0.5,0.5] [3,2,1]
Avg. Tau=[0.0, -0.25, 0.25, 0.433, 0.683, 0.933, 1.0]
Epochs=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,

Figure C.8: *PNN* at epoch 62

APPENDIX D

D.1 Snapshoots of *SGPNN* Ranking Convergence.

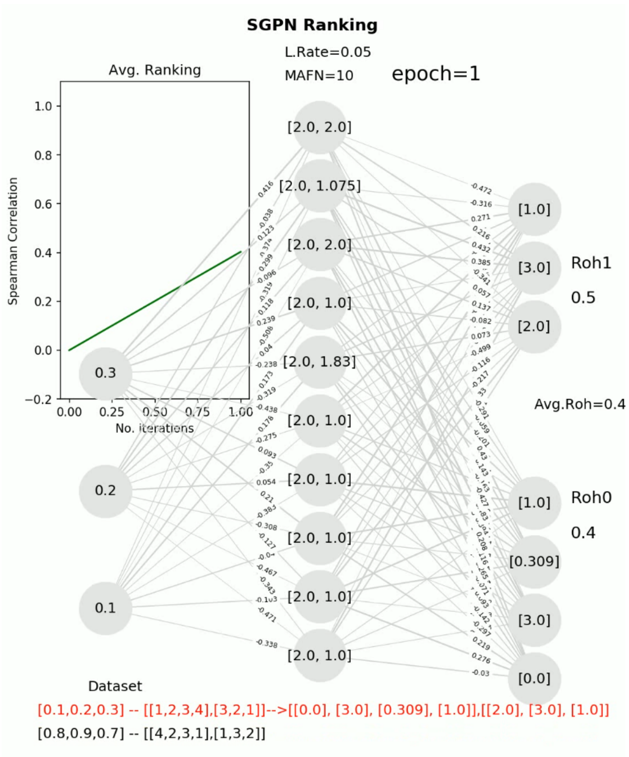


Figure D.1: *SGPNN* at epoch 1

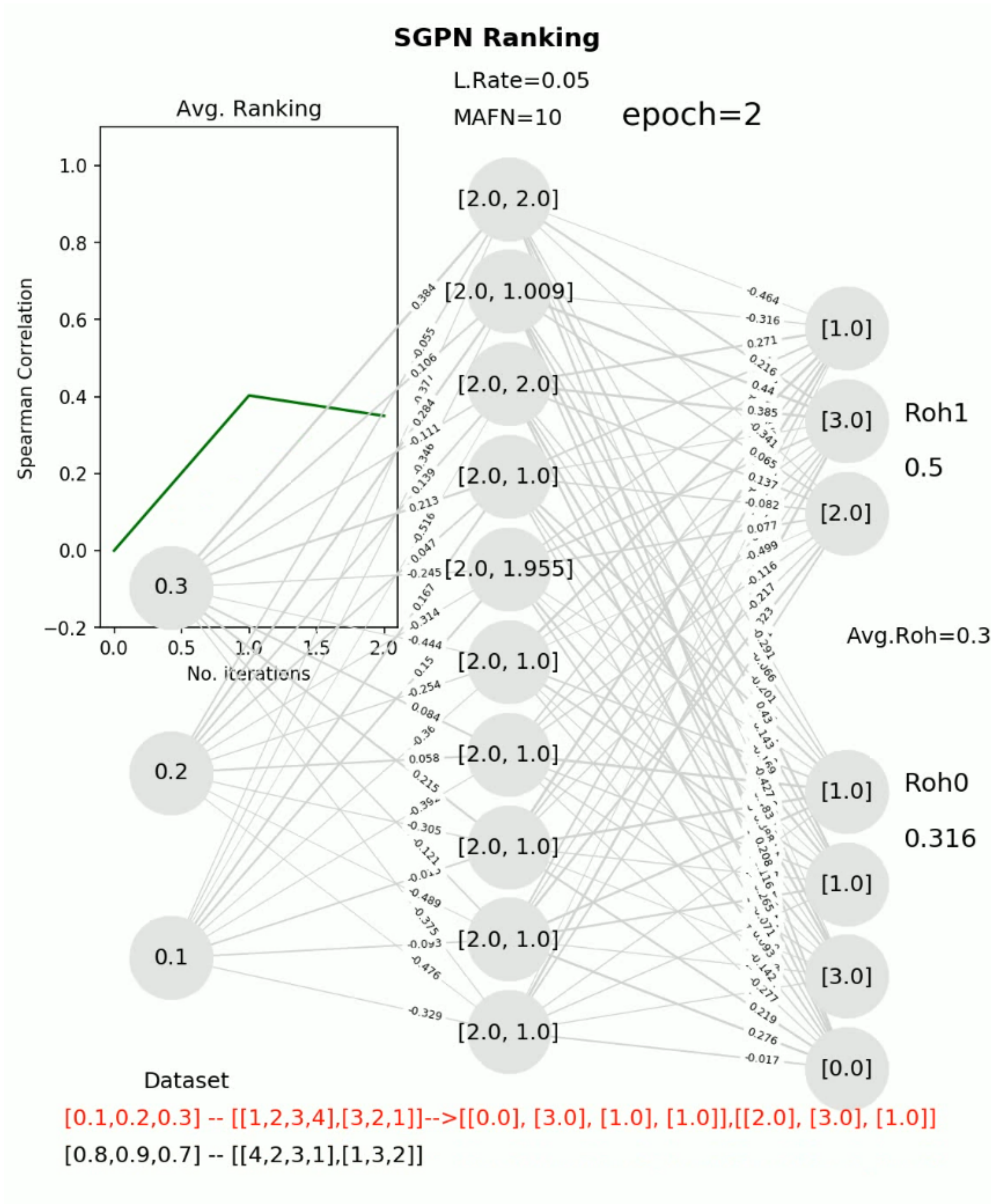


Figure D.2: *SGPNN* at epoch 2

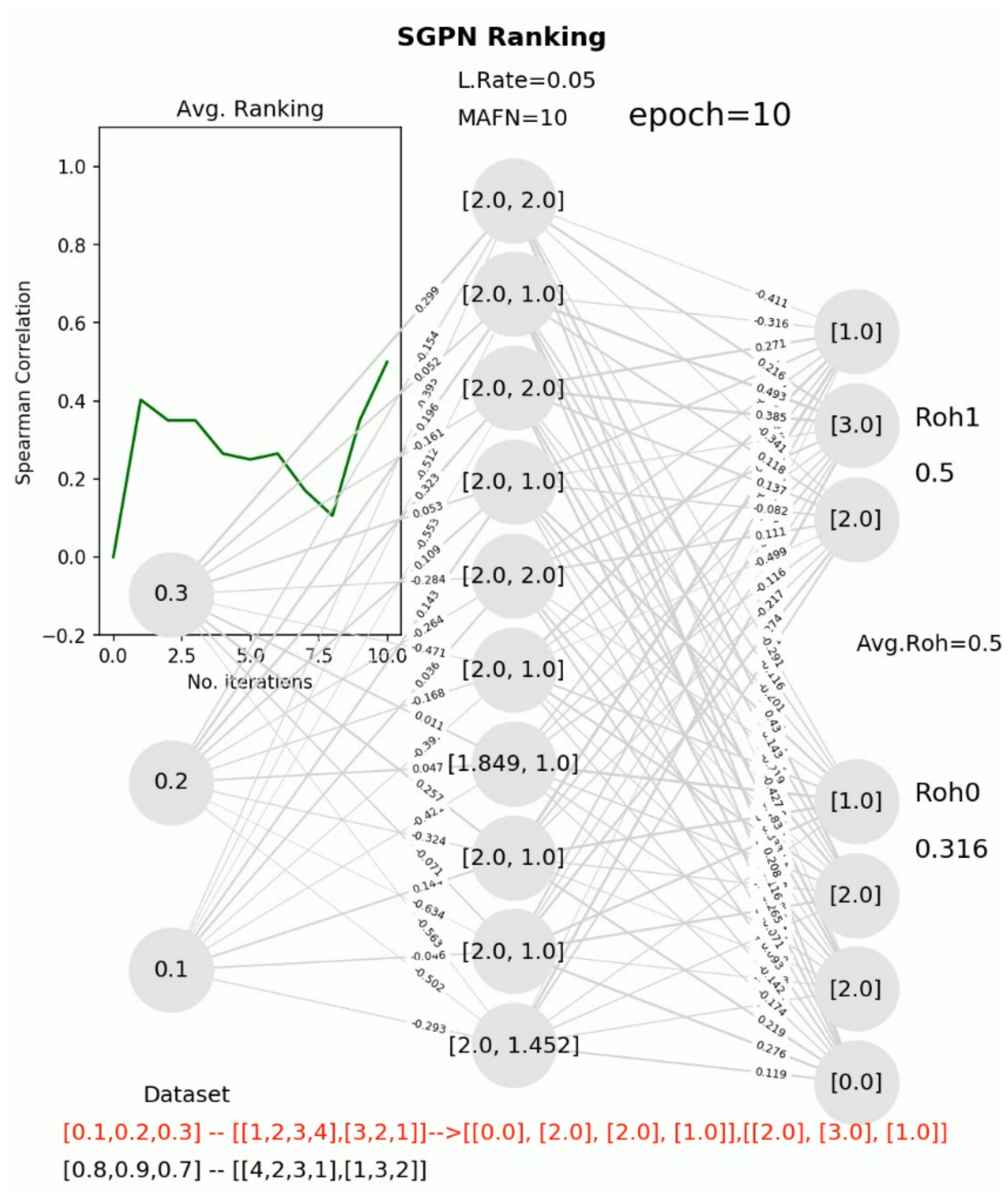


Figure D.3: *SGPNN* at epoch 10

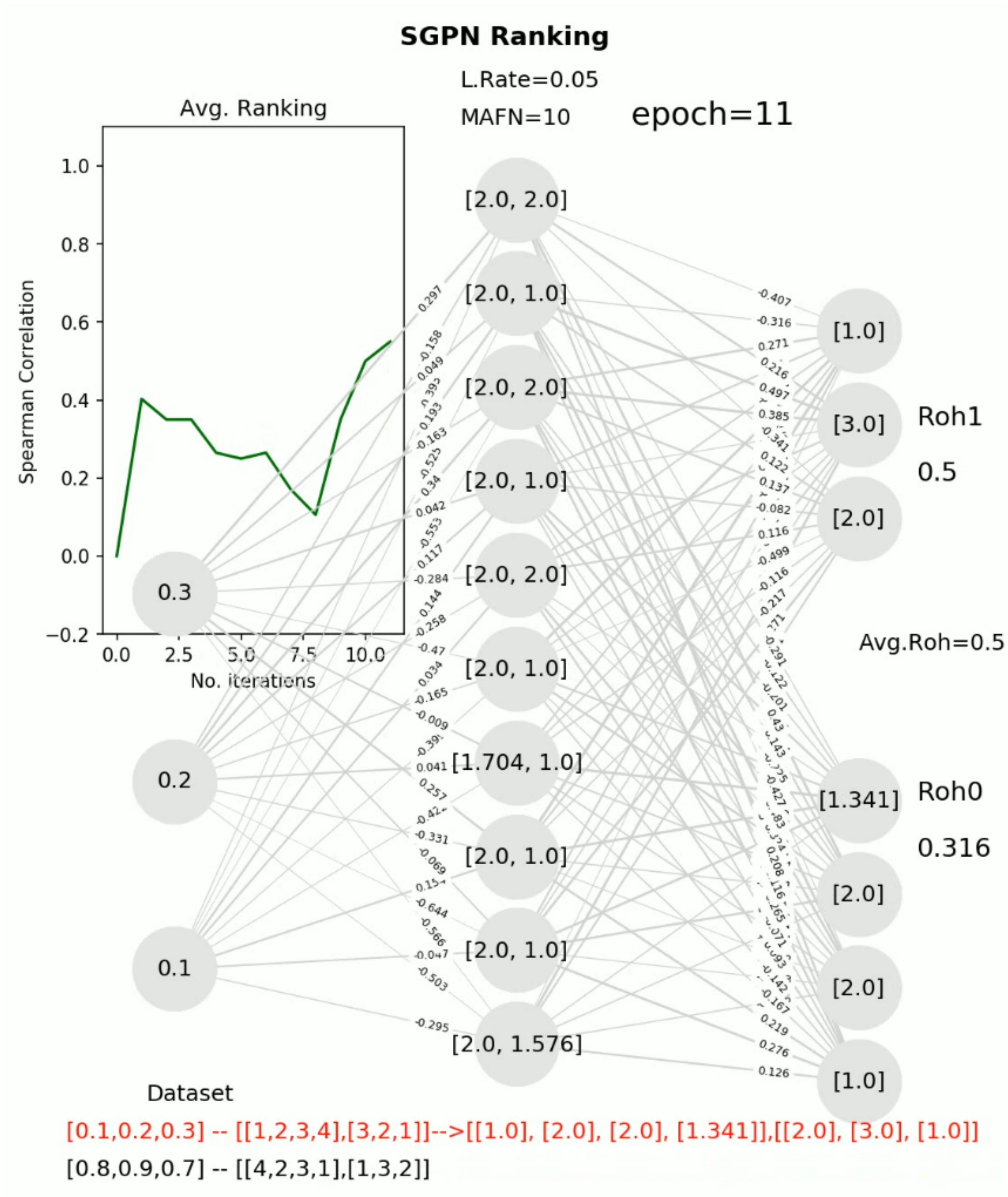


Figure D.4: *SGPNN* at epoch 11

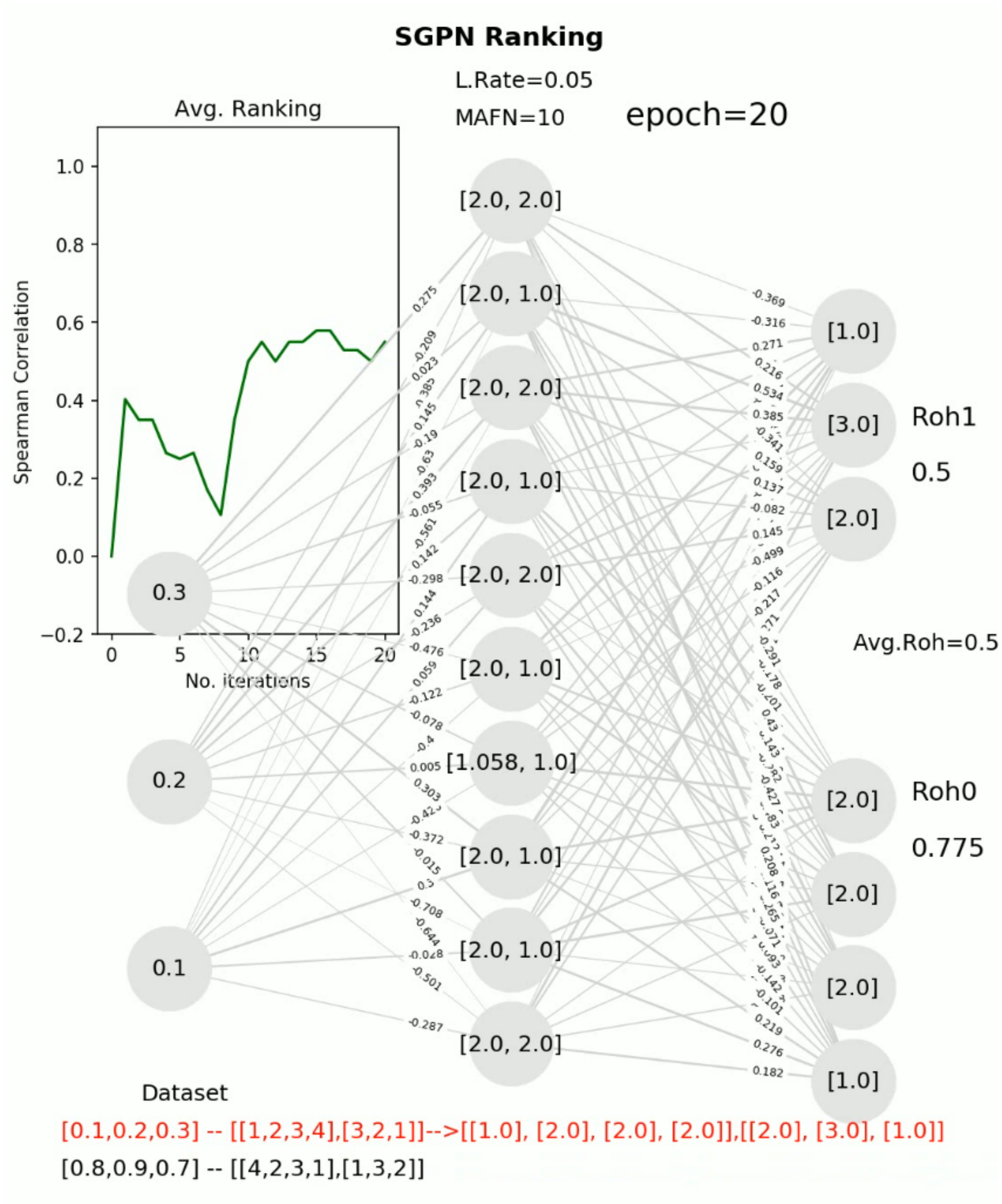


Figure D.5: *SGPNN* at epoch 20

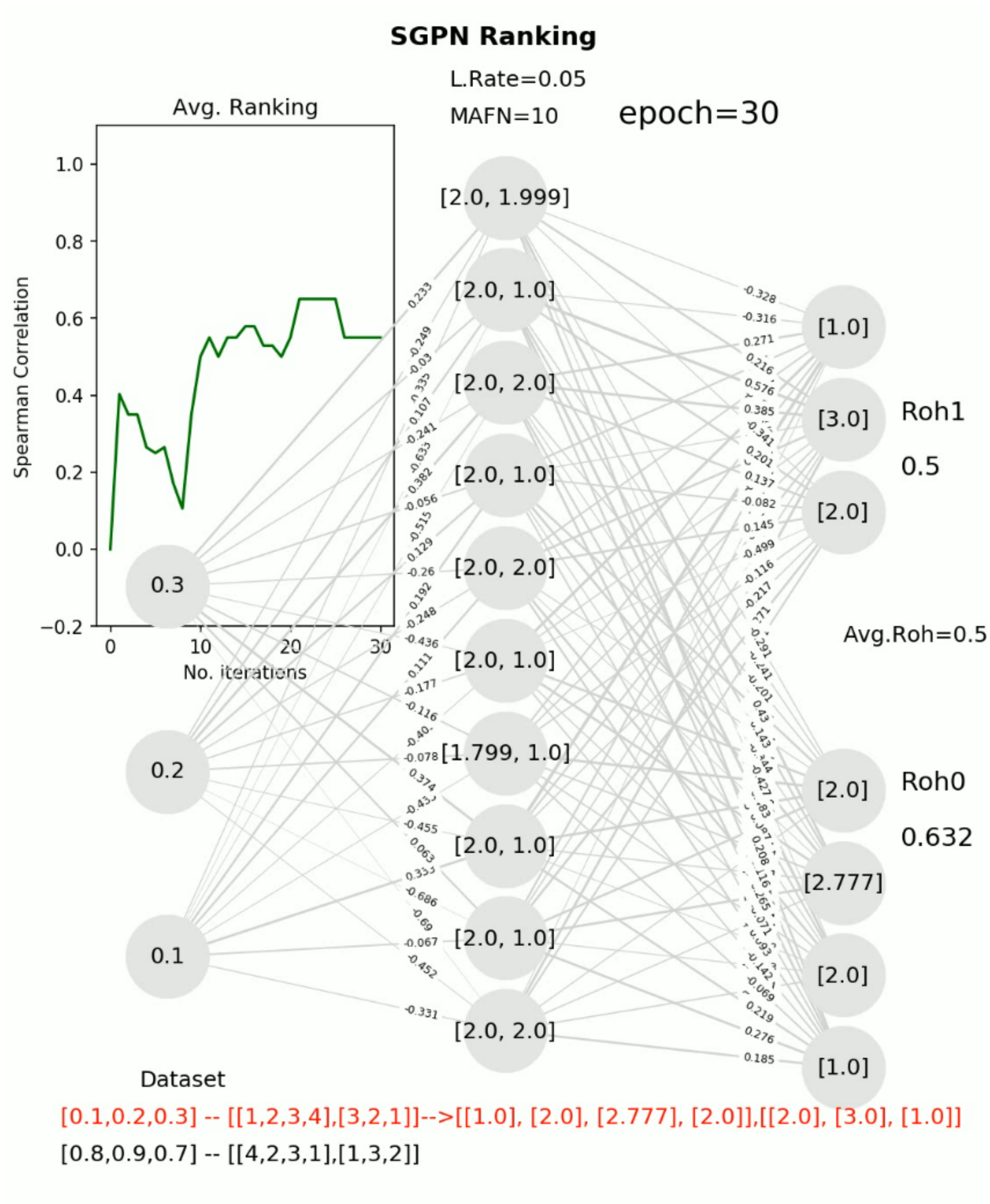
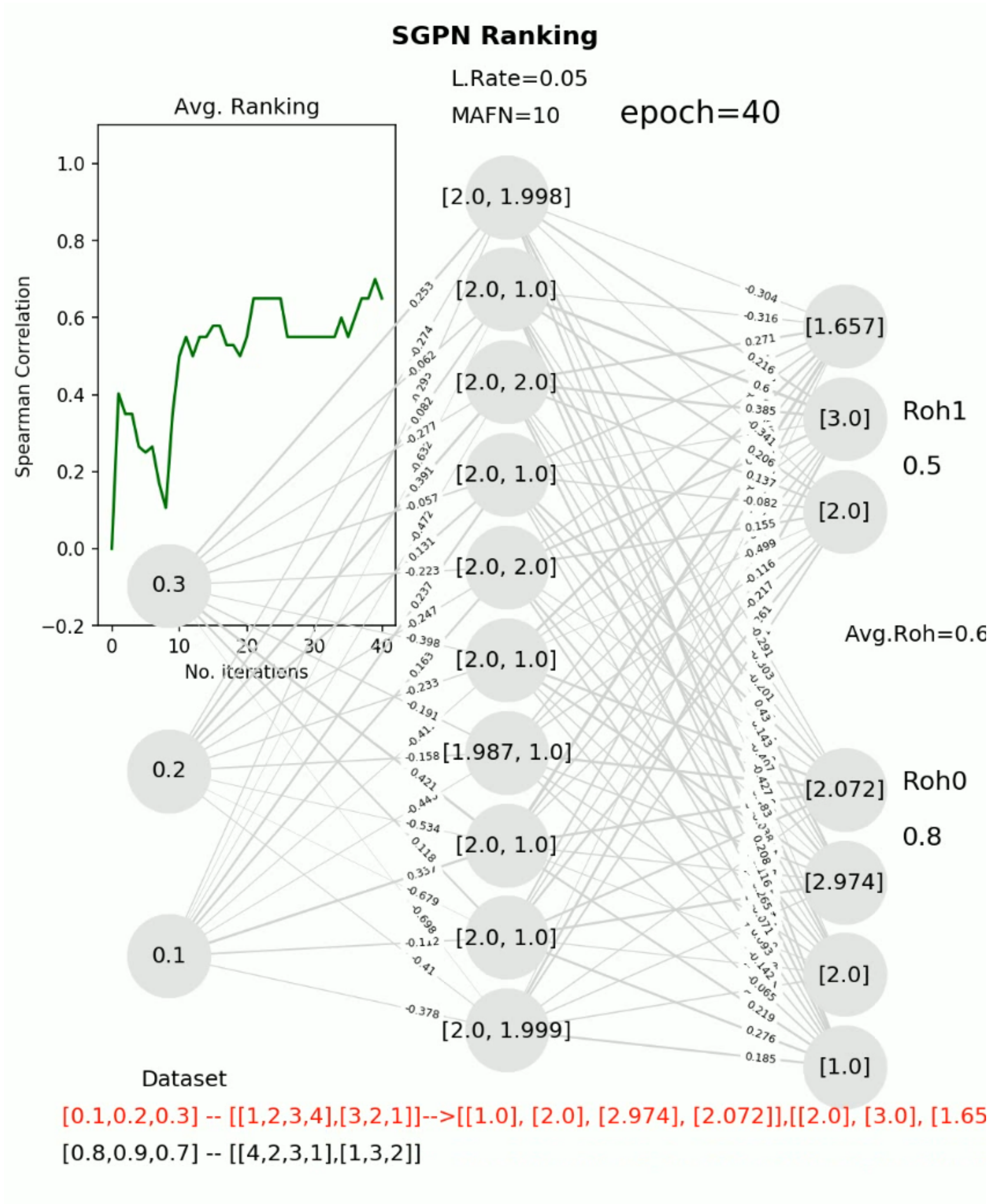


Figure D.6: *SGPNN* at epoch 30



[0.1,0.2,0.3] -- [[1,2,3,4],[3,2,1]]-->[[1.0], [2.0], [2.974], [2.072]],[[2.0], [3.0], [1.65

[0.8,0.9,0.7] -- [[4,2,3,1],[1,3,2]]

Figure D.7: *SGPNN* at epoch 40

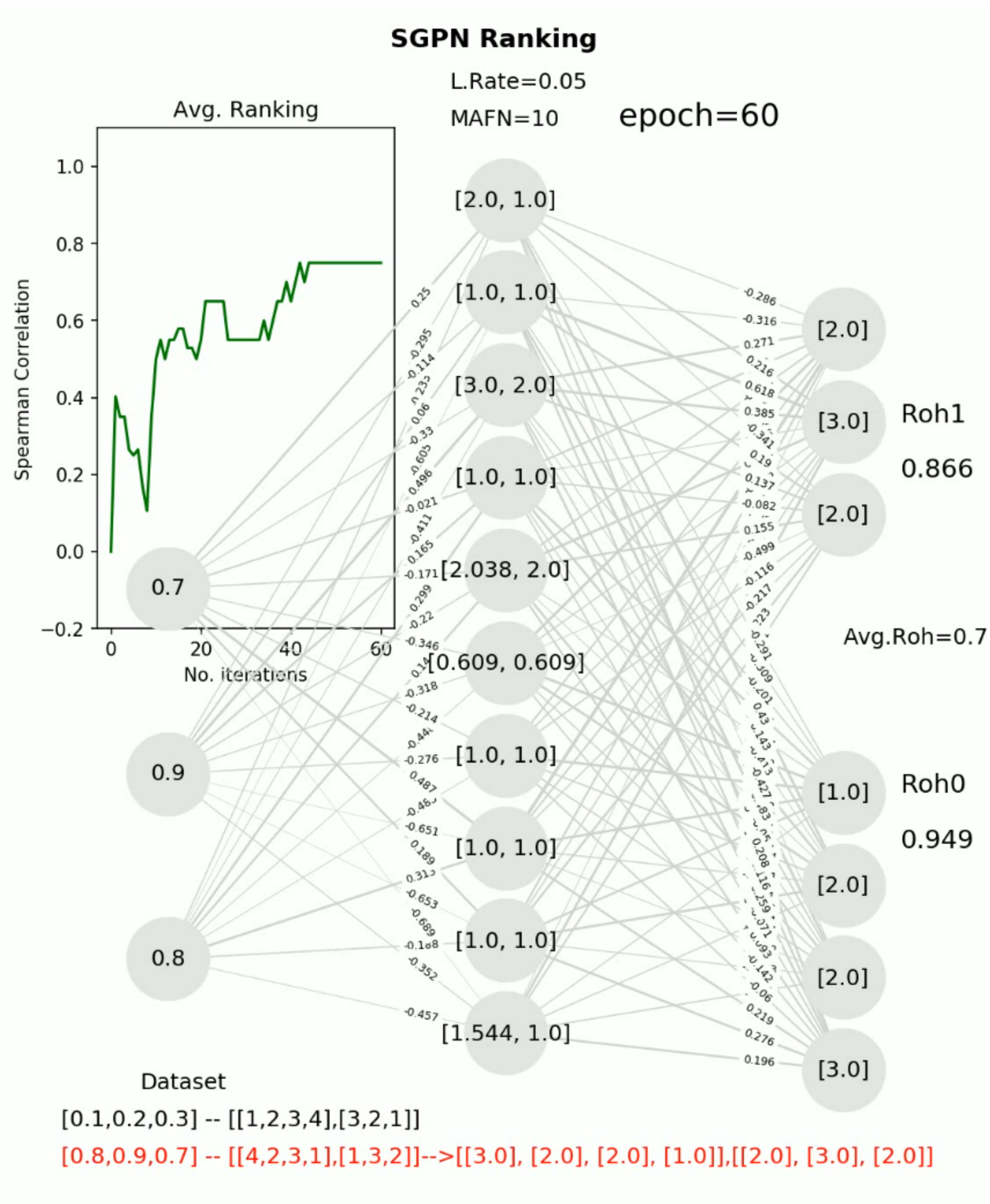


Figure D.8: *SGPNN* at epoch 60

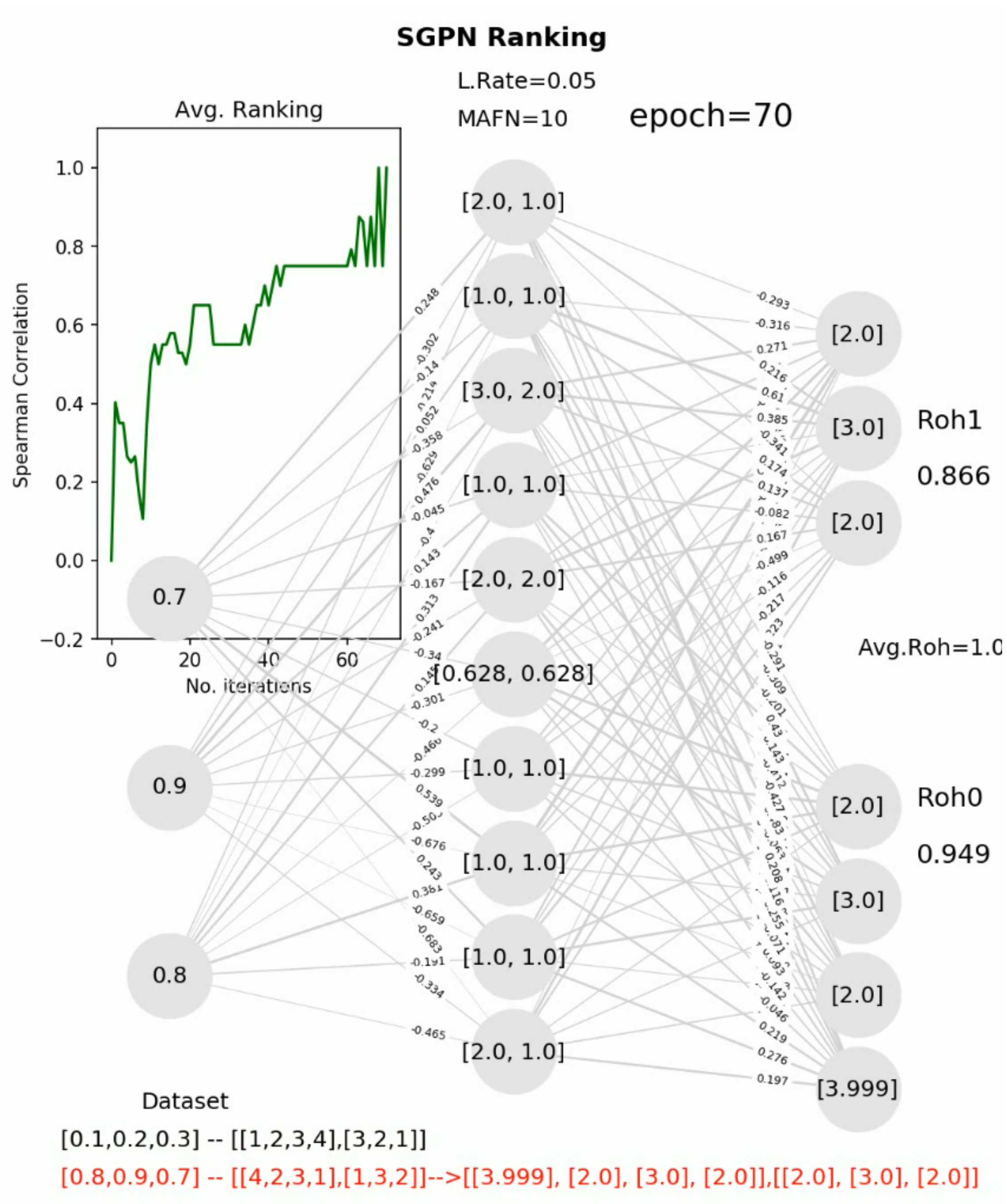


Figure D.9: *SGPNN* at epoch 70

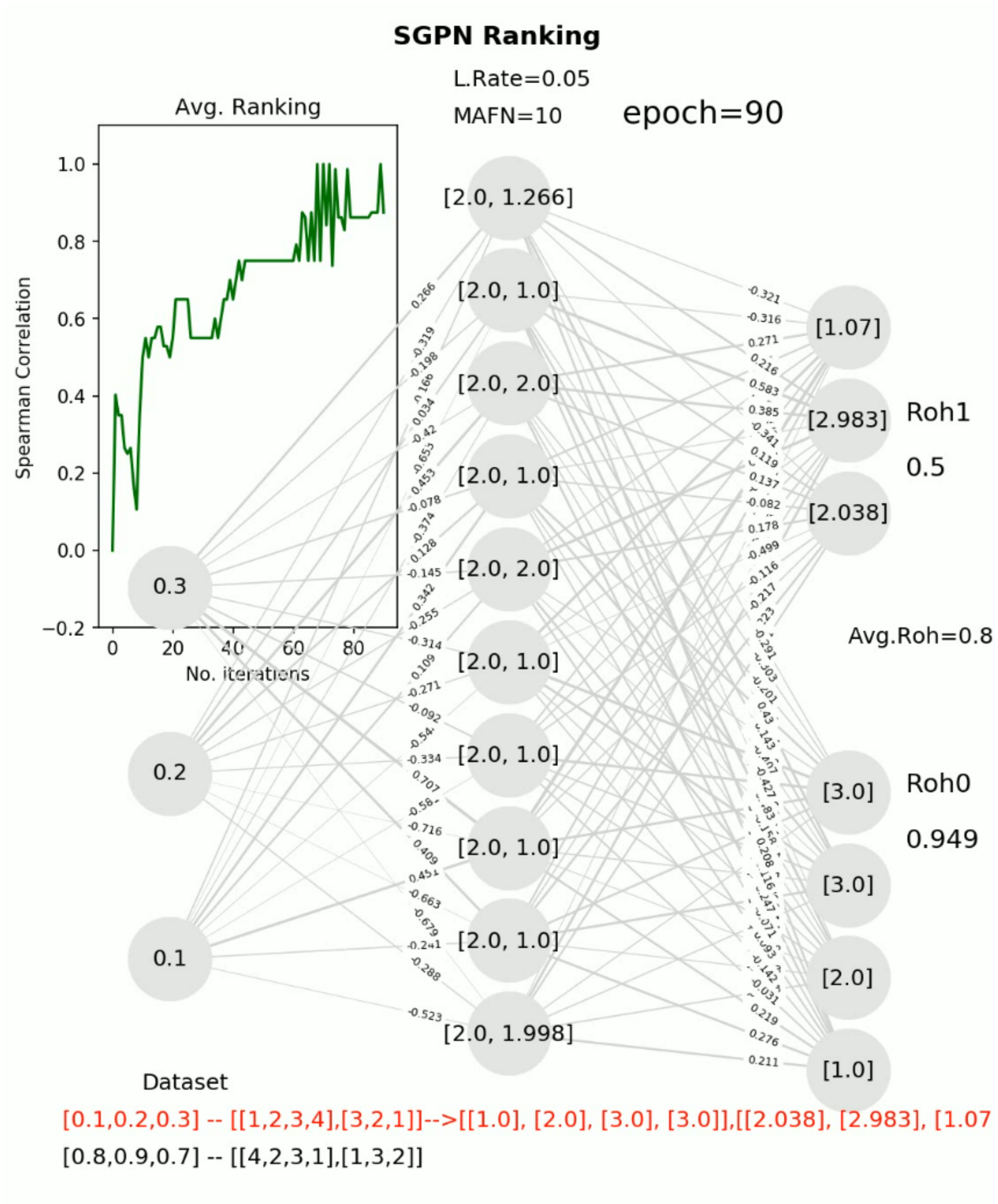


Figure D.10: *SGPNN* at epoch 90

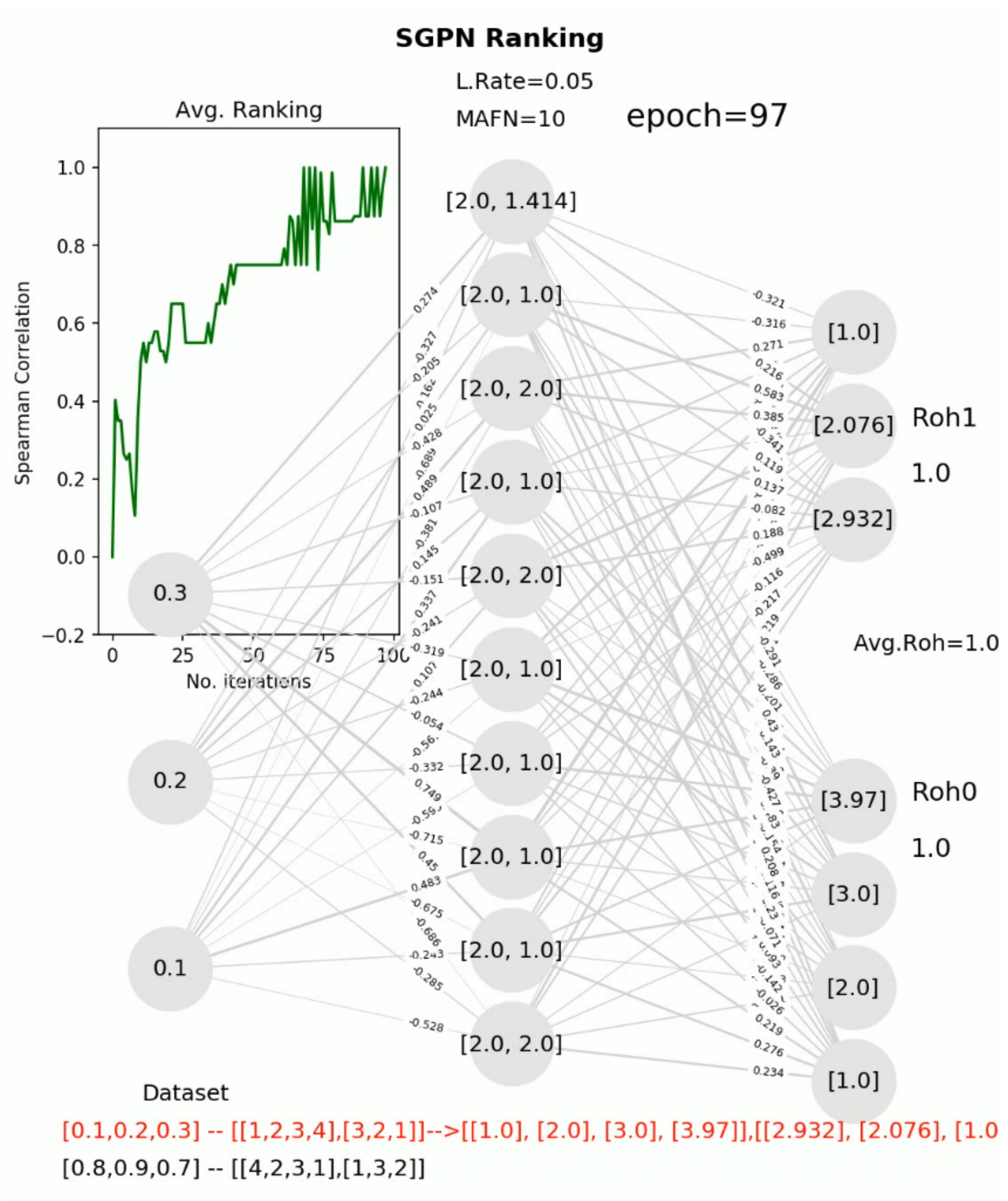
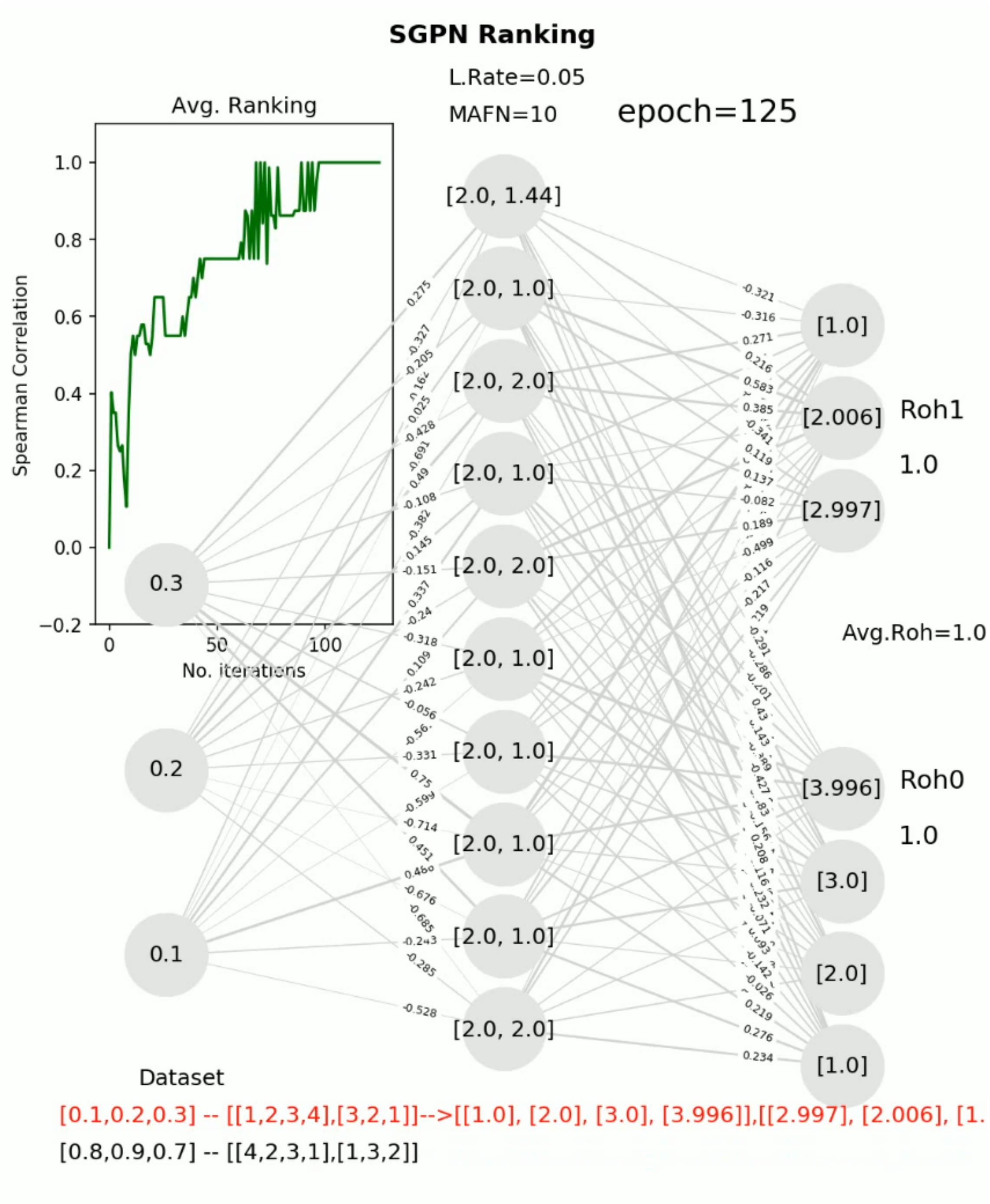


Figure D.11: *SGPNN* at epoch 97



[0.1,0.2,0.3] -- [[1,2,3,4],[3,2,1]]-->[[1.0], [2.0], [3.0], [3.996]],[[2.997], [2.006], [1.

[0.8,0.9,0.7] -- [[4,2,3,1],[1,3,2]]

Figure D.12: *SGPNN* at epoch 125

Bibliography

Abdulrahman, S. M., Brazdil, P., Zainon, W. M. N. W. & Adamu, A. (2019), Simplifying the algorithm selection using reduction of rankings of classification algorithms, *in* 'Proceedings of the 2019 8th International Conference on Software and Computer Applications', ICSCA '19, Association for Computing Machinery, New York, NY, USA, p. 140148.
URL: *https://doi.org/10.1145/3316615.3316674*

Adomavicius, G. & Tuzhilin., A. (2005), 'Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions', pp. 734–749.

Aiguzhinov, A., Soares, C. & Serra, A. P. (2010), A similarity-based adaptation of naive bayes for label ranking: Application to the metalearning problem of algorithm recommendation, *in* 'Discovery Science'.

Ailon, N. & Mohri, M. (2007), 'An efficient reduction of ranking to classification', *arXiv* .

Aiolli, F. (2005), 'A preference model for structured supervised learning tasks', pp. 557–560.

Aizenberg, I., Aizenberg, N. & Vandewalle, J. P. (2000), *Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications*, Kluwer Academic Publishers, Norwell, MA, USA.

Alphonse, A., Shankar, K., Rakkini, M. J. J., Ananthakrishnan, S., Athisayamani, S., Singh, A. R. & Gobi, R. (2021), 'A multi-scale and rotation-invariant phase pattern (mripp) and a stack of restricted boltzmann machine (rbm) with preprocessing for facial expression classification', *J. Ambient Intell. Humaniz. Comput.* **12**, 3447–3463.

Balcan, M.-F., Bansal, N., Beygelzimer, A., Coppersmith, D., Langford, J. & Sorkin, G. B. (2008), 'Robust reductions from ranking to classification', **72**(12), 139153.
URL: *https://doi.org/10.1007/s10994-008-5058-6*

Belfodil, A., Belfodil, A., Bendimerad, A., Lamarre, P., Robardet, C., Kaytoue, M. & Plantevit, M. (2019), Fssd - a fast and efficient algorithm for subgroup set discovery, *in* '2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)', pp. 91–99.

Bergelson, V. (2008), 'Part v: Theorems and problems - 09. ergodic theorems.', pp. 3–691.

Bologna, G. (2000), 'Rule extraction from a multilayer perceptron with staircase activation functions'.

Brafman, R. & Domshlak, C. (2009), 'Preference handling - an introductory tutorial', pp. 58–86.

Brinker, K. & Hüllermeier, E. (2007), Label ranking in case-based reasoning, *in* 'International Conference on Case-Based Reasoning', Springer, pp. 77–91.

Brinker, K. & Hüllermeier, E. (2019), A reduction of label ranking to multiclass classification, *in* 'ECML/PKDD'.

Carranza-Alarcon, Y.-C., Messoudi, S. & Destercke, S. (2020), Cautious label-wise ranking with constraint satisfaction, *in* M.-J. Lesot, S. Vieira,

M. Z. Reformat, J. P. Carvalho, A. Wilbik, B. Bouchon-Meunier & R. R. Yager, eds, ‘Information Processing and Management of Uncertainty in Knowledge-Based Systems’, Springer International Publishing, Cham, pp. 96–111.

Chankong, V. & Haimes, Y. Y. (2008), *Multiobjective Decision Making: Theory and Methodology*, Courier Dover Publications.

Chen, M., Challita, U., Saad, W., Yin, C. & Debbah, M. (2019), ‘Artificial neural networks-based machine learning for wireless networks: A tutorial’, *IEEE Communications Surveys Tutorials* **21**(4), 3039–3071.

Cheng, W., Hühn, J. & Hüllermeier, E. (2009), Decision tree and instance-based learning for label ranking, *in* ‘Proceedings of the 26th Annual International Conference on Machine Learning’, ICML 09, ACM, p. 161168.

Cheng, W. & Hüllermeier, E. (2008), ‘Instance-based label ranking using the mallows model’, pp. 143–157.

Chiclana, F., Herrera-Viedma, E. & Alonso, S. (2009), ‘A note on two methods for estimating missing pairwise preference values’, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **39**(6), 1628–1633.

Chollet, F. (2017), ‘Xception: Deep learning with depthwise separable convolutions’, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* pp. 1800–1807.

Chouhan, N., Khan, A. & Khan, H.-U.-R. (2019), ‘Network anomaly detection using channel boosted and residual learning based deep convolutional neural network’, *Appl. Soft Comput.* **83**.

Chris Burges, T. S. (2005), ‘Learning to rank using gradient descent’, pp. 58–86.

Chu, Y., Zhao, X., Zou, Y., Xu, W., Han, J. & Zhao, Y. (2018), ‘A decoding scheme for incomplete motor imagery eeg with deep belief network’, *Frontiers in Neuroscience* **12**, 680.
URL: *https://www.frontiersin.org/article/10.3389/fnins.2018.00680*

Ciresan, D. C., Meier, U. & Schmidhuber, J. (2012), ‘Multi-column deep neural networks for image classification’, *2012 IEEE Conference on Computer Vision and Pattern Recognition* pp. 3642–3649.

Cláudio, R. (2018), ‘algae dataset’.
URL: *http://dx.doi.org/10.17632/3mv94c8jpc.2*

Crammer, K. & Singer, Y. (2002), ‘Pranking with ranking’, pp. 641–647.

de Sá, C. R., Soares, C., Knobbe, A. & Cortez, P. (2017), ‘Label ranking forests’, *Expert Syst. J. Knowl. Eng.* **34**.

de S*á*, C. R. & Duivesteijn, W. (2018), ‘Discovering a taste for the unusual: exceptional models for preference mining’, pp. 1775–1807.

de Sá, C. R. (2018), ‘Label ranking datasets (sushi)’, Mendeley Data. v2.

E. Hüllermeier, J. Furnkranz, W. (2008), ‘Label ranking by learning pairwise preferences’, pp. 1897–1916.

Elgharabawy, A. (2020*a*), ‘German 2007, 2009 dataset as subgroup’. Available at `https://drive.google.com//file//d//1acsVJIuR-cgW_uA2dfUnteuL3CGnUFSk//view?usp=sharing`.

Elgharabawy, A. (2020*b*), ‘motions dataset as subgroup’.

Available at `https://drive.google.com//file//d//1kmsIxUyjNz4nC6paPuwAsJMSGqz0q1FH//view?usp=sharing`.

Elgharabawy, A. (2020*c*), ‘Preference neural network convergence performance’.
Available at `https://drive.google.com/drive/folders/1yxuqYoQ3Kiuch-2sLeVe2ocMj12QVsRM?usp=sharing`.

Elgharabawy, A. (2020*d*), ‘Resturant food and service survey dataset as subgroup’.
Available at `https://drive.google.com/file/d/1bDALTELymFJLwd5hznVDFK7KH_TgTBj2/view?usp=sharing`.

Elgharabawy, A. (2020*e*), ‘Sushi dataset as subgroup’.
Available at `https://drive.google.com/file/d/1cXbSgONItGXsNDj5YN_qEac1cAsowqhD/view?usp=sharing`.

Elgharabawy, A. (2021*a*), ‘Initial ranker neural network convergence’, Video file.
Available at `https://drive.google.com/file/d/12Fm6Yj8rjKLw6LFpzJJZag26U8--UvAY/view?usp=sharing`.

Elgharabawy, A. (2021*b*), ‘Preference neural network convergence’, Video file.
Available at `https://drive.google.com/file/d/1j5DUczsrNqI9aP6RyiDJuN2wxq2goIbP/view?usp=sharing`.

Elgharabawy, A. (2021*c*), ‘Subgroup preference networks log file’, Video file.
Available at `https://drive.google.com/file/d/1U42c-xysVxZK4YYvCEIQ5LIaTwLpanrK/view?usp=sharing`.

Elgharabawy, A. (2021*d*), 'Subgroup preference neural network convergence'.
Available at `https://drive.google.com/file/d/1e1QjRhmlor3_QUAFrtw30jim0crEm1iG/view?usp=sharing`.

Elgharabawy, A. (2022), 'Preference neural network source code', Python Code, Mathematica code.
URL: *https://github.com/ayman-elgharabawy/PNN*

Elgharabawy, A., Prasad, M. & Lin, C.-T. (2021*a*), 'Preference neural network'.

Elgharabawy, A., Prasad, M. & Lin, C.-T. (2021*b*), 'Subgroup preference neural network', *Sensors* **21**(18).
URL: *https://www.mdpi.com/1424-8220/21/18/6104*

Emmert-Streib, F., Yang, Z., Feng, H., Tripathi, S. & Dehmer, M. (2020), 'An introductory review of deep learning for prediction models with big data', *Frontiers in Artificial Intelligence* **3**, 4.
URL: *https://www.frontiersin.org/article/10.3389/frai.2020.00004*

Foret, P., Kleiner, A., Mobahi, H. & Neyshabur, B. (2021), 'Sharpness-aware minimization for efficiently improving generalization', *ArXiv* **abs/2010.01412**.

Freund, Y., Iyer, R., Schapire, R. E. & Singer, Y. (2003), 'An efficient boosting algorithm for combining preferences', *J. Mach. Learn. Res.* **4**(null), 933969.

Frnkranz, J. & Hllermeier, E. (2010), *Preference Learning*, 1st edn, Springer-Verlag, Berlin, Heidelberg.

Fukushima, K. (1988), 'Neocognitron: A hierarchical neural network capable of visual pattern recognition', *Neural Networks* **1**, 119–130.

Fukushima, K., Miyake, S. & Ito, T. (1983), 'Neocognitron: A neural network model for a mechanism of visual pattern recognition', *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-13**, 826–834.

Furnkranz, J. & Hüllermeier, E. (2003), 'Pairwise preference learning and ranking in machine learning', pp. 145–156.

Fürnkranz, J. & Hüllermeier, E. (2010), 'Preference learning'.

Furnkranz, J. & Hüllermeier, E. (2011), 'Decision tree modeling for ranking data', pp. 83–106.

Gamberger, D. & Lavrac, N. (2002), 'Expert-guided subgroup discovery: Methodology and application', *J. Artif. Intell. Res.* **17**, 501–527.

Grbovic, M., Djuric, N., Guo, S. & Vucetic, S. (2013), 'Supervised clustering of label ranking data using label preference information', *Machine Learning* **93**(2-3), 191–225.

Guo, B., Hou, C., Shan, J. & Yi, D. (2018), 'Low rank multi-label classification with missing labels', *2018 24th International Conference on Pattern Recognition (ICPR)* pp. 417–422.

Gupta, A. K., Seal, A., Prasad, M. & Khanna, P. (2020), 'Salient object detection techniques in computer visiona survey', *Entropy* **22**.

Han, D., Kim, J. & Kim, J. (2017), Deep pyramidal residual networks, *in* '2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)', pp. 6307–6315.

Har-Peled, S., Roth, D. & Zimak, D. (2002), 'Constraint classification: A new approach to multiclass classification'.

He, K., Zhang, X., Ren, S. & Sun, J. (2016*a*), Deep residual learning for image recognition, *in* '2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)', pp. 770–778.

He, K., Zhang, X., Ren, S. & Sun, J. (2016*b*), 'Deep residual learning for image recognition', *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* pp. 770–778.

Henzgen, S. & Hüllermeier, E. (2014), Mining rank data, *in* 'International Conference on Discovery Science', Lecture Notes in Computer Science, Springer, pp. 123–134.

Hinton, G. E. (2009), 'Deep belief networks', *Scholarpedia* **4**, 5947.

Holland, S., Ester, M. & Kießling, W. (2003), Preference mining: A novel approach on mining user preferences for personalized applications, *in* N. Lavrač, D. Gamberger, L. Todorovski & H. Blockeel, eds, 'Knowledge Discovery in Databases: PKDD 2003', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 204–216.

Hu, J., Shen, L., Albanie, S., Sun, G. & Wu, E. (2020), 'Squeeze-and-excitation networks', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **42**, 2011–2023.

Hu, Y., Wen, G., Luo, M., Dai, D. & Ma, J. (2018), 'Competitive inner-imaging squeeze and excitation for residual network', *ArXiv* **abs/1807.08920**.

Huang, G., Liu, S., Maaten, L. V. D. & Weinberger, K. Q. (2018), 'Condensenet: An efficient densenet using learned group convolutions', *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* pp. 2752–2761.

Huang, G., Liu, Z., Van Der Maaten, L. & Weinberger, K. Q. (2017), Densely connected convolutional networks, *in* '2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)', pp. 2261–2269.

Inc., W. (n.d.), 'Mathematica, Version 12.3.1'.
Champaign, IL, 2021.
URL: *https://www.wolfram.com/mathematica*

Ince, N. F., Goksu, F., Tewfik, A. H. & Arica, S. (2009), 'Adapting subject specific motor imagery eeg patterns in spacetimefrequency for a brain computer interface', *Biomedical Signal Processing and Control* **4**(3), 236–246.
New Trends in Voice Pathology Detection and Classification.
URL: *https://www.sciencedirect.com/science/article/pii/S1746809409000214*

Ince, N., Tewfik, A. & Arica, S. (2005), Classification of movement eeg with local discriminant bases, *in* 'Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.', Vol. 5, pp. v/413–v/416 Vol. 5.

Ji, Z., Cui, B., Li, H., Jiang, Y.-G., Xiang, T., Hospedales, T. & Fu, Y. (2020), 'Deep ranking for image zero-shot multi-label classification', *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society* .
Date created - 2020-05-15; Date revised - 2020-12-20; Last updated - 2021-01-29.
URL: *http://ezproxy.lib.uts.edu.au/login?url=https://www-proquest-com.ezproxy.lib.uts.edu.au/scholarly-journals/deep-ranking-image-zero-shot-multi-label/docview/2403037643/se-2?accountid=17095*

Jian, Y., Xiao, J., Cao, Y., Khan, A. & Zhu, J. (2019), Deep pairwise ranking with multi-label information for cross-modal retrieval, *in* '2019

IEEE International Conference on Multimedia and Expo (ICME)', pp. 1810–1815.

Jiang, F., Kong, B., Li, J., Dashtipour, K. & Gogate, M. (2020), 'Robust visual saliency optimization based on bidirectional markov chains', *Cognitive Computation* pp. 1–12.

Jorge, P. J. A. & J. P. da, C. (2011), 'Mining association rules for label ranking', pp. 432–443.

Jung, Y. H. & Tewari, A. (2018), 'Online boosting algorithms for multilabel ranking'.

Kabir, H. M. D., Abdar, M., Jalali, S. M. J., Khosravi, A., Atiya, A. F., Nahavandi, S. & Srinivasan, D. (2020), 'Spinalnet: Deep neural network with gradual input', *ArXiv* **abs/2007.03347**.

Kamishima, T. (2003), Nantonac collaborative filtering: Recommendation based on order responses, *in* 'Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', KDD 03, ACM, p. 583588.

Kavsek, B., Lavrac, N. & Jovanoski, V. (2003), Apriori-sd: Adapting association rule learning to subgroup discovery, *in* 'IDA'.

Kendall, M. (1948), 'Rank correlation methods'.

Klösgen, W. (1996), Explora: A multipattern and multistrategy discovery assistant, *in* 'Advances in Knowledge Discovery and Data Mining'.

Klösgen, W. & Zytkow, J. M. (2002), *Handbook of Data Mining and Knowledge Discovery*, Oxford University Press.

Korba, A., Garcia, A. & d'Alché Buc, F. (2018), A structured prediction approach for label ranking, *in* 'NeurIPS'.

Kotlowski, W., Dembczynski, K. & Hüllermeier, E. (2011), Bipartite ranking through minimization of univariate loss, *in* 'ICML'.

Krizhevsky, A. (2009*a*), Learning multiple layers of features from tiny images, Technical report.

Krizhevsky, A. (2009*b*), Learning multiple layers of features from tiny images, Technical report.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012*a*), 'Imagenet classification with deep convolutional neural networks', *Communications of the ACM* **60**, 84 – 90.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012*b*), Imagenet classification with deep convolutional neural networks, *in* 'Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1', NIPS'12, Curran Associates Inc., Red Hook, NY, USA, p. 10971105.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012*c*), Imagenet classification with deep convolutional neural networks, *in* 'Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1', NIPS'12, Curran Associates Inc., Red Hook, NY, USA, p. 10971105.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012*d*), Imagenet classification with deep convolutional neural networks, *in* F. Pereira, C. J. C. Burges, L. Bottou & K. Q. Weinberger, eds, 'Advances in Neural Information Processing Systems', Vol. 25, Curran Associates, Inc.

Kuen, J., Kong, X., Wang, G. & Tan, Y.-P. (2017), 'Delugenets: Deep networks with efficient and flexible cross-layer information inflows', *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)* pp. 958–966.

Kumar, J., Raja, K., Pooja, R. S. & Charan, V. S. (2020), 'Generation of lyrics using recurrent neural network (rnn)', *International Journal of Research* **7**, 654–659.

Kwok, T.-Y. & Yeung, D.-Y. (1997), 'Constructive algorithms for structure learning in feedforward neural networks for regression problems', *IEEE Transactions on Neural Networks* **8**(3), 630–645.

Larsson, G., Maire, M. & Shakhnarovich, G. (2017), 'Fractalnet: Ultra-deep neural networks without residuals', *ArXiv* **abs/1605.07648**.

Lawhern, V. J., Solon, A. J., Waytowich, N. R., Gordon, S. M., Hung, C. P. & Lance, B. J. (2018), 'EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces', *Journal of Neural Engineering* **15**(5), 056013.
URL: *https://doi.org/10.1088/1741-2552/aace8c*

Lecun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998), 'Gradient-based learning applied to document recognition', *Proceedings of the IEEE* **86**(11), 2278–2324.

LeCun, Y. & Cortes, C. (2010), 'MNIST handwritten digit database'.
URL: *http://yann.lecun.com/exdb/mnist/*

LeCun, Y., Cortes, C. & Burges, C. (2010), 'Mnist handwritten digit database', *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* **2**.

Lei, H., Lei, T. & Yue-nian, T. (2021), 'Sports image detection based on particle swarm optimization algorithm', *Microprocess. Microsystems* **80**, 103345.

Li, J., Ng Wing, W. Y., Xing, T., Kwong, S. & Wang, H. (2020), 'Weighted multi-deep ranking supervised hashing for efficient image retrieval', *International Journal of Machine Learning and Cybernetics* **11**(4), 883–897.
Copyright - 2019Ï' Springer-Verlag GmbH Germany, part of Springer Nature 2019; Last updated - 2020-03-08.
URL: *http://ezproxy.lib.uts.edu.au/login?url=https://www-proquest-com.ezproxy.lib.uts.edu.au/scholarly-journals/weighted-multi-deep-ranking-supervised-hashing/docview/2373640699/se-2?accountid=17095*

Li, M.-a., Zhu, W., Liu, H.-n. & Yang, J.-f. (2017), 'Adaptive feature extraction of motor imagery eeg with optimal wavelet packets and se-isomap', *Applied Sciences* **7**(4).
URL: *https://www.mdpi.com/2076-3417/7/4/390*

Liu, D., Baskett, W., Beversdorf, D. & Shyu, C. R. (2020), 'Exploratory data mining for subgroup cohort discoveries and prioritization', *IEEE Journal of Biomedical and Health Informatics* **24**(5), 1456–1468.

Liu, Y., Liu, S., Wang, Y., Lombardi, F. & Han, J. (2021), 'A survey of stochastic computing neural networks for machine learning applications', *IEEE Transactions on Neural Networks and Learning Systems* **32**(7), 2809–2824.

Lucas, T., Vimieiro, R. & Ludermir, T. (2018), Ssdp+: A diverse and more informative subgroup discovery approach for high dimensional data, *in* '2018 IEEE Congress on Evolutionary Computation (CEC)', pp. 1–8.

Mazzia, V., Salvetti, F. & Chiaberge, M. (2021), 'Efficient-capsnet: capsule network with self-attention routing', *Scientific Reports* **11**.

Mihajlo, G., Nemanja, D. & Slobodan, V. (2014), ‘Learning from pairwise preference data using gaussian mixture model’.

Mirchandani, G. & Cao, W. (1989), ‘On hidden nodes for neural nets’, *IEEE Transactions on Circuits and Systems* **36**(5), 661–664.

Montaner, M. & López, B. (2003), ‘A taxonomy of recommender agents on the internet.’, pp. 285–330.

Moraga, C. & Heider, R. (1999), "New lamps for old!" (generalized multiple-valued neurons), *in* ‘Proceedings 1999 29th IEEE International Symposium on Multiple-Valued Logic (Cat. No. 99CB36329)’, IEEE, pp. 36–41.

Paeedeh, N. & Ghiasi-Shirazi, K. (2021), ‘Improving the backpropagation algorithm with consequentialism weight updates over mini-batches’, *Neurocomputing* **461**, 86–98.
URL: *https://www.sciencedirect.com/science/article/pii/S0925231221010444*

Pandeya, Y. R., Bhattarai, B. & Lee, J. (2021), ‘Deep-learning-based multimodal emotion classification for music videos’, *Sensors* **21**(14).
URL: *https://www.mdpi.com/1424-8220/21/14/4927*

Polat, K. & Güne, S. (2007), ‘Classification of epileptiform eeg using a hybrid system based on decision tree classifier and fast fourier transform’, *Applied Mathematics and Computation* **187**(2), 1017–1026.
URL: *https://www.sciencedirect.com/science/article/pii/S0096300306012380*

Rebelo, C. (2018*a*), ‘Label ranking datasets (german2005)’, Mendeley Data.
v2.

Rebelo, C. (2018*b*), ‘Label ranking datasets (german2009)’, Mendeley Data.

v2.

Ribeiro, G., Duivesteijn, W., Soares, C. & Knobbe, A. (2012), Multilayer perceptron for label ranking, *in* 'Proceedings of the 22nd International Conference on Artificial Neural Networks and Machine Learning - Volume Part II', ICANN12, Springer, p. 2532.

Ridnik, T., Sharir, G., Ben-Cohen, A., Ben-Baruch, E. & Noy, A. (2021), Ml-decoder: Scalable and versatile classification head.

Roy, A. G., Navab, N. & Wachinger, C. (2018), 'Concurrent spatial and channel squeeze & excitation in fully convolutional networks', *ArXiv* **abs/1803.02579**.

Rueping, S. (2009), Ranking interesting subgroups, *in* 'Proceedings of the 26th Annual International Conference on Machine Learning', ICML '09, Association for Computing Machinery, New York, NY, USA, p. 913920.
URL: *https://doi.org/10.1145/1553374.1553491*

Salakhutdinov, R. & Hinton, G. E. (2009), Deep boltzmann machines, *in* 'AISTATS'.

Scherer, D., Müller, A. C. & Behnke, S. (2010), Evaluation of pooling operations in convolutional architectures for object recognition, *in* 'ICANN'.

Schmidhuber, J. (2009), 'Deep learning in neural networks: An overview', **61**, 85–117.

Skarding, J., Gabrys, B. & Musial, K. (2021), 'Foundations and modeling of dynamic networks using dynamic graph neural networks: A survey', *IEEE Access* **9**, 79143–79168.

Song, Q., Jin, H., Huang, X. & Hu, X. (2018), 'Multi-label adversarial perturbations', *2018 IEEE International Conference on Data Mining (ICDM)* pp. 1242–1247.

Sornam, M., Muthusubash, K. & Vanitha, V. (2017), 'A survey on image classification and activity recognition using deep convolutional neural network architecture', *2017 Ninth International Conference on Advanced Computing (ICoAC)* pp. 121–126.

Spearman, C. (1961), 'The proof and measurement of association between two things'.

Srivastava, R., Greff, K. & Schmidhuber, J. (2015), 'Highway networks', *ArXiv* **abs/1505.00387**.

Szegedy, C., Ioffe, S., Vanhoucke, V. & Alemi, A. A. (2017), 'Inception-v4, inception-resnet and the impact of residual connections on learning', p. 42784284.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. & Rabinovich, A. (2015), Going deeper with convolutions, *in* '2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)', pp. 1–9.

Tan, M. & Le, Q. V. (2021), 'Efficientnetv2: Smaller models and faster training', *ArXiv* **abs/2104.00298**.

Tangermann, M., Müller, K.-R., Aertsen, A., Birbaumer, N., Braun, C., Brunner, C., Leeb, R., Mehring, C., Miller, K., Mueller-Putz, G., Nolte, G., Pfurtscheller, G., Preissl, H., Schalk, G., Schlögl, A., Vidaurre, C., Waldert, S. & Blankertz, B. (2012), 'Review of the bci competition iv', *Frontiers in Neuroscience* **6**, 55.
URL: *https://www.frontiersin.org/article/10.3389/fnins.2012.00055*

Tangermann, M., Müller, K.-R., Aertsen, A., Birbaumer, N., Braun, C., Brunner, C., Leeb, R., Mehring, C., Miller, K., Mueller-Putz, G., Nolte, G., Pfurtscheller, G., Preissl, H., Schalk, G., Schlögl, A., Vidaurre, C., Waldert, S. & Blankertz, B. (2012), 'Review of the bci competition iv', *Frontiers in Neuroscience* **6**, 55.
URL: *https://www.frontiersin.org/article/10.3389/fnins.2012.00055*

Tanveer, M., Khan, M. U. K. & Kyung, C. M. (2021), 'Fine-tuning darts for image classification', *2020 25th International Conference on Pattern Recognition (ICPR)* pp. 4789–4796.

Trohidis, K. & Tsoumakas, G. (2011), 'Multi-label classification of music by emotion', *EURASIP Journal on Audio, Speech, and Music Processing* .

Vahdat, A. & Kautz, J. (2020), 'Nvae: A deep hierarchical variational autoencoder', *ArXiv* **abs/2007.03898**.

Vargas-Govea, B., González-Serna, G. & Ponce-Medellın, R. (2011), 'Effects of relevant contextual features in the performance of a restaurant recommender system', *ACM RecSys* **11**(592), 56.

Vedantam, V. K. (2021), The survey: Advances in natural language processing using deep learning*.

Vembu, S. & Gärtner, T. (2010), *Label Ranking Algorithms: A Survey*, Springer, pp. 45–64.

Wang, F., Jiang, M., Qian, C., Yang, S., Li, C., Zhang, H., Wang, X. & Tang, X. (2017), 'Residual attention network for image classification', *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* pp. 6450–6458.

Wang, L., Luo, Z., Li, J.-R. & Chen, C. (2020), 'Target-oriented transformation networks for document retrieval', *Proceedings of the 2020 9th International Conference on Computing and Pattern Recognition* .

Woo, S., Park, J., Lee, J.-Y. & Kweon, I.-S. (2018), Cbam: Convolutional block attention module, *in* 'ECCV'.

Wrobel, S. (1997), An algorithm for multi-relational discovery of subgroups, *in* 'PKDD'.

Wu, H., Xiao, B., Codella, N. C. F., Liu, M., Dai, X., Yuan, L. & Zhang, L. (2021), 'Cvt: Introducing convolutions to vision transformers', *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* pp. 22–31.

Wu, Q., Burges, C. J., Svore, K. M. & Gao, J. (2010), 'Adapting boosting for information retrieval measures', **13**(3), 254270.
URL: *https://doi.org/10.1007/s10791-009-9112-1*

Xiao, H., Rasul, K. & Vollgraf, R. (2017*a*), 'Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms'.
cite arxiv:1708.07747Comment: Dataset is freely available at https://github.com/zalandoresearch/fashion-mnist Benchmark is available at http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/.
URL: *http://arxiv.org/abs/1708.07747*

Xiao, H., Rasul, K. & Vollgraf, R. (2017*b*), 'Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms', *CoRR* **abs/1708.07747**.
URL: *http://arxiv.org/abs/1708.07747*

Xiao, X. & Fang, Y. (2021), ‘Motor imagery eeg signal recognition using deep convolution neural network’, *Frontiers in Neuroscience* **15**, 312.
URL: *https://www.frontiersin.org/article/10.3389/fnins.2021.655599*

Xie, X., Li, L., Lian, S., Chen, S. & Luo, Z. (2020), ‘Seru: A cascaded seresnext unet for kidney and tumor segmentation’, *Concurrency and Computation: Practice and Experience* **32**.

Yan, Y., Wang, Y., Gao, W., Zhang, B.-W., Yang, C. & Yin, X.-C. (2017), ‘Lstm 2: Multi-label ranking for document classification’, *Neural Processing Letters* **47**, 117–138.

Zagoruyko, S. & Komodakis, N. (2016*a*), ‘Wide residual networks’, *ArXiv* **abs/1605.07146**.

Zagoruyko, S. & Komodakis, N. (2016*b*), ‘Wide residual networks’, *ArXiv* **abs/1605.07146**.

Zagoruyko, S. & Komodakis, N. (2016*c*), ‘Wide residual networks’, *ArXiv* **abs/1605.07146**.

Zeiler, M. D. & Fergus, R. (2014), Visualizing and understanding convolutional networks, *in* D. Fleet, T. Pajdla, B. Schiele & T. Tuytelaars, eds, ‘Computer Vision – ECCV 2014’, Springer International Publishing, Cham, pp. 818–833.

Zell, A. (1994), Simulation neuronaler netze.

Zhang, R. (2022), ‘Sports action recognition based on particle swarm optimization neural networks’, *Wireless Communications and Mobile Computing* .

Zhang, S., Choromaska, A. & LeCun, Y. (2015), Deep learning with elastic averaging sgd: 3rd international conference on learning representations, iclr 2015, *in* 'ICLR 2015'.

Zhang, X., Li, Z., Loy, C. C. & Lin, D. (2017), 'Polynet: A pursuit of structural diversity in very deep networks', *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* pp. 3900–3908.

Zhao, Z., Chen, W., Wu, X., Chen, P. C. Y. & Liu, J. (2017), 'Lstm network: a deep learning approach for short-term traffic forecast', *Iet Intelligent Transport Systems* **11**, 68–75.

Zhong, Y. & Fang, J. (2020), 'Text detection in multi-feature fusion natural scenes based on convolution deep belief network'.

Zhou, Y., Liu, Y., Yang, J., He, X. & Liu, L. (2014), 'A taxonomy of label ranking algorithms', *JCP* **9**, 557–565.

Zhou, Y. & Qiu, G. (2018), 'Random forest for label ranking', *Expert Syst. Appl.* **112**, 99–109.