 doi:10.20944/preprints201904.0091.v3

Preference Neural Network

Ayman Elgharabawy, Student Member, *IEEE* , Mukesh Parsad, Senior Member, *IEEE*, Chin-Teng Lin, Fellow, *IEEE*

***Abstract*—This paper proposes a preference neural network (*PNN*) to address the problem of indifference preferences orders with new activation function. *PNN* also solves the Multi-label ranking problem, where labels may have indifference preference orders or subgroups are equally ranked. *PNN* follows a multi-layer feedforward architecture with fully connected neurons. Each neuron contains a novel smooth stairstep activation function based on the number of preference orders. *PNN* inputs represent data features and output neurons represent label indexes. The proposed *PNN* is evaluated using new preference mining dataset that contains repeated label values which have not experimented before. *PNN* outperforms five previously proposed methods for strict label ranking in terms of accurate results with high computational efficiency.**

***Index Terms*—Preference learning, Labels ranking, Neural network, Kendall's tau, Preference mining**

I. Introduction

PREFERENCE learning (*PL*) is emerging as an extended paradigm in machine learning by inducing predictive preference models from experimental data [1] [2] [3]. *PL* involves in various research topics such as knowledge discovery and recommender systems [4]. Objects, instances and label ranking are the three main categories of *PL*. However, label ranking (*LR*) is one of the challenging problems that has gained importance in information retrieval of search engine results [5] [6]. Unlike the standard problems of regression and classification, multi-label ranking involves predicting the relation between the orders of strict multiple labels. For a given an instance x from the instance space $\mathbb{x}$, there is a finite label set $\mathscr{L}$= $\{y_1,..,y_n\}$ denoted by y associated with x, where n is the number of labels. *LR* is an extension of multi-class and multi-label classification, where each instance is described by a set of features and assigned labels in a continuous permutation space π=($\lambda_a \succ \lambda_b \succ \lambda_c \succ \lambda_d$), where a, b, c and d are the alphabetical label indexes and λ_a, λ_b, λ_c and λ_d are the ranking values of these labels respectively. Various algorithms have been introduced recently for label ranking [8]; Decomposition, Probabilistic, Similarity and Ensemble methods.

Decomposition methods include pairwise comparison [9] [10], long linear models and constraint classification [11]. The pairwise approach introduced by hüllermeier [21] 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 input x, where i and j are labels indexes. The probabilistic methods are based on probability estimation, i.e., decision trees [12], instance base (Plackett-Luce) [30] and Gaussian mixture model [31]. The similarity methods introduced similarity measures by minimizing the distance instead of maximizing the probability of label values, i.e., Naive Bayes [34] and association rules [13]. The ensemble methods adapt the existing multi-class classifiers to rank multiple labels, i.e., multi-layer perceptron for label ranking (*MLP-LR*) [29] and Rank-net [7]. The instance-based decision tree was introduced by Cheng and Hüllermeier to rank the labels based on predictive probability models of a decision tree [28]. Hüllermeier combined both a decision tree and supervised clustering in two approaches for label ranking by using unsupervised, supervised clustering and creating a model for mapping between instances and multi-labels ranking space [27].

Neural Network (*NN*) for ranking was first introduced as (rank-net) by Burge to solve the problem of object ranking for sorting web documents by search engine [7]. Rank-net uses Gradient descent and probabilistic ranking cost function for each object pairs. Multi-layer perceptron (*MLP*) for label ranking [29] employs *NN* architecture using sigmoid activation function to calculate the error between the actual and expected values for the output labels. However, it does not predict indifference preference relations between labels and unable to represent all labels as output neurons. Zhou and Yangming provided a scalable decision tree structure by implementing a random forest with a parallel computational architecture for extreme label ranking [38]. Claudio Rebelo introduced label random forest (*LRF*) as an ensemble method of ranking [35]. *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 voting of the best learners and scoring the labels for ranking [23]. Song and Huang proposed a framework to solve the vulnerability of multi-label deep learning models [36]. Yan and Wang proposed a long short term memory (*LSTM*) based multi-label ranking model for document classification to identify the relation between labels [37]. 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 [39]. The abovementioned methods and their respective variants have problematic boundaries which can be broadly categorized into two types of drawbacks;

Classification models drawbacks, where ranking model is constructed from the binary classification model of higher dimension label space. Binary classification for ranking is based on minimizing the classification error which is not often equivalent to maximizing the perfor-

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 June 2020 doi:10.20944/preprints201904.0091.v3

mance of the label ranking.

Probability models drawbacks, where ranking model is decomposed from the probabilistic model where individual labels are ranked by probability scoring instead of distance between labels. These drawbacks inspired this research to overcome accuracy and ranking limitations. Thus, *PNN* addresses three problems which currently exists in Multi-label ranking methods.

First, *PNN* uses gradient descent to minimize Kendall's tau error function which helps to provide better ranking results, whereas other methods such as decision trees, probability and clustering, have variations in ranking results for given datasets.

Second, *PNN* proposes stairstep activation function to rank all the labels simultaneously. However, current methods based on *NN* such as *MLP-LR* and *Rank-net* solve only pairwise ranking.

Third, *PNN* ranks the indifference preferences orders, However, the existing methods fail to address this issue.

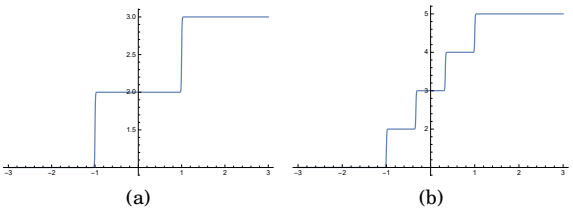
II. *PNN* ARCHITECTURE

A. Ranking activation function

In the proposed preference neural network (*PNN*), each neuron has a generic activation function used to calculate the ranking between labels. The activation function is represented as bounded smooth stairstep function, as shown in Fig. 1. The activation function is a polynomial of multiple $tanh(x)$ functions, therefore, it is considered differentiable and continuous. The function output represents preference value from {*1* to *n*}. The derivative of activation function is mentioned in *PNN* backpropagation section III. The activation function is given in Eq. (1).

$$y = r \cdot \left(q \cdot \sum_{i=0,1,..}^{n-1} \left(\tanh\left(s \cdot x \cdot (k - (i \cdot \frac{t}{n-2})) \right) \right) + \frac{n+1}{2} \right) \quad (1)$$

where k=100, q=0.5, r=-1, s=-100, t=200, and n = number of ranked labels. The activation function has smooth stairsteps shape, where each step represents a rank in Y axis from 1 to n. The input value is normalized between the interval -1 and 1 on the X axis. The Fig. 1(a) and (b) shows the activation functions to rank three and five labels, respectively.



(a) (b)

Fig. 1: Activation functions of (a) n=3 and (b) n=5 where number of ranked labels n equals the number of steps

Eqs.(2)-(3) represent activation functions for n=3 and 5, respectively.

$$y = -0.5\left(\tanh\left(-100(x+1) \right) + \tanh\left(-100(x-1) \right) \right) \quad (2)$$

$$y = -0.5\Big(\tanh\left(-100(x+1) \right) + \tanh\left(-100(x-1) \right) + \tanh\left(-100(x+0.5) \right) + \tanh\left(-100(x-0.5) \right) \Big) \quad (3)$$

Algorithm 1 represents the three main functions of the *PNN* learning process; feedforward, backpropagation, and updating weights. Fig. 2 (a) and (b) represents two different network structures of three and five label ranking, respectively.

Algorithm 1: *PNN* learning flow

Data:

$$D \in \{(x_{i,f}, y_{i,j})\}_{i=1}^{m}, y_{i,j}(y_1, .., y_n) \quad (4)$$

Result: Ranked labels (λ_{y_1},..,λ_{y_n})

1 Activation function $\varphi|_{n=no.of labels}$;
2 Randomly initialize weights $\omega_{i,j} \in \{0, 1.0\}$;
3 **repeat**
4 **forall** $\{x_{i,f}, y_{i,j}\}_{i=1}^{m} \in D$ **do**
5 Feedforward : $a_{(i,j)}$=$\sum a_l.\omega_l$*φ;
6 Backpropagation();
7 Update Weights : $\omega_{i,j,new}$ =$\omega_{i,j,old}$-η . $\delta_{(i,j)}$;
8 **until** τ_{Avg} = *1 or iteration* $\geq 10^6$;

Algorithm 2: *PNN* backpropagation

9 **for** *l = L-1 to 1* **do**
10 **if** *l=L-1* **then**
11 **for** *each i in layer l* **do**
12 E_{rr}=(yt_i-y_i)/m(m-1);
13 **else**
14 **for** *each i in layer l* **do**
15 E_{rri}=$\omega_i.\delta_i$;
16 **for** *each i in layer l* **do**
17 delta δ_i=E_{rr}*φ'

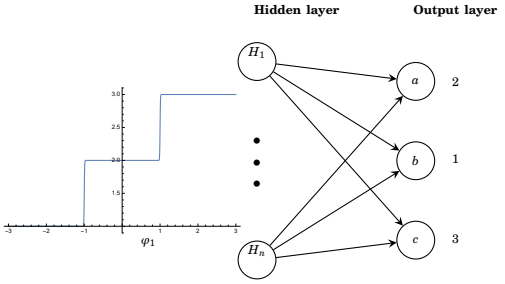
B. Ranking loss function

Two main error functions have been used for label ranking; Kendall's tau [32] and Spearsman [33]. However, both error functions lack continuity and differentiability. Therefore, root mean square (*RMS*) function is used to measure the ranking difference between the *PNN* output value per each iteration and an expected preference value. *RMS* error function is used only for backpropagation, however, Kendall's tau loss function τ given in Eq. (6) is used as convergence stopping criteria. τ_{Avg} is average τ per each label divided by number of instances m, as shown in step 8 of algorithm 1.

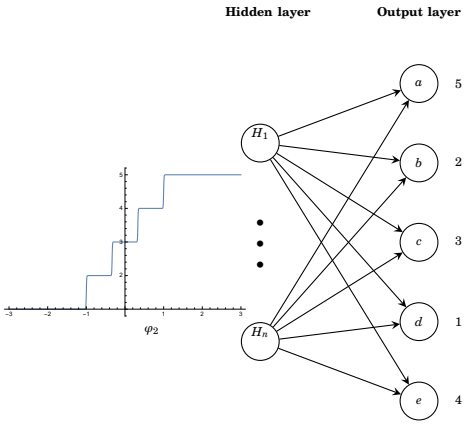
$$E_{rr} = \frac{(y_i - yt_i)^2}{m(m-1)} \quad (5)$$

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

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 June 2020 doi:10.20944/preprints201904.0091.v3



(a)



(b)

Fig. 2: *PNN* structures, (a) n=3 and (b) n=5

C. Neural Network structure

PNN is fully connected multi-layer *NN*. 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, $H_n \geq \mathscr{L}$, in order to reach error convergence after a finite number of iterations. The output layer represents the labels indexes as neurons, where the labels are displayed in alphabetical order as shown in Fig. 4. In order to map sub-grouping to the label ranking, *PNN* assigns the same ranking values to more than one label. Thus, *NN* structure solves the problem of multiple subgroup ranking, i.e., π=λ_d>λ_b>(λ_c,λ_a), where λ_d=1, λ_b=2 and ($\lambda_c \simeq \lambda_a$) = 3 as shown in Fig. 4. *PNN* calculates preference values of the output labels based on bounded smooth stairstep activation function. Each neuron uses activation function in feedforward propagation to calculate preference number from 1 to n, where n is the number of label classes. During backpropagation, both stairstep and error function are differentiated as shown in Eqs. (10)-(13) and (14)-(15), respectively. The process of feedforward and backpropagation are iterated until the average of kendall's tao coefficient of all data equals to one (τ_{Avg}=1) or number of iterations reaches (10^6) as shown in Algorithm 1.

$$\tau = \frac{\sum_{ij} Sgn(y_i - y_j).Sgn(yt_i - yt_j)}{m(m-1)} \tag{6}$$

where τ, Sgn,y_i, yt_i, i, j and m are Kendall's tau coefficient, sign function, output ranking value, expected ranking value, labels indexes and number of instances.

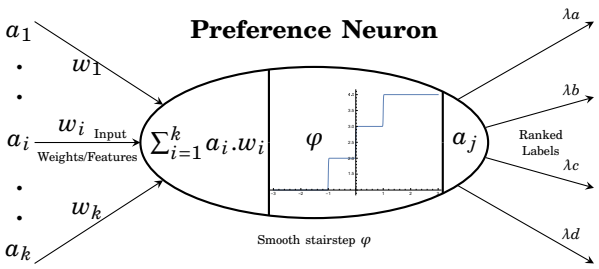


Fig. 3: Preference neuron where n=4.

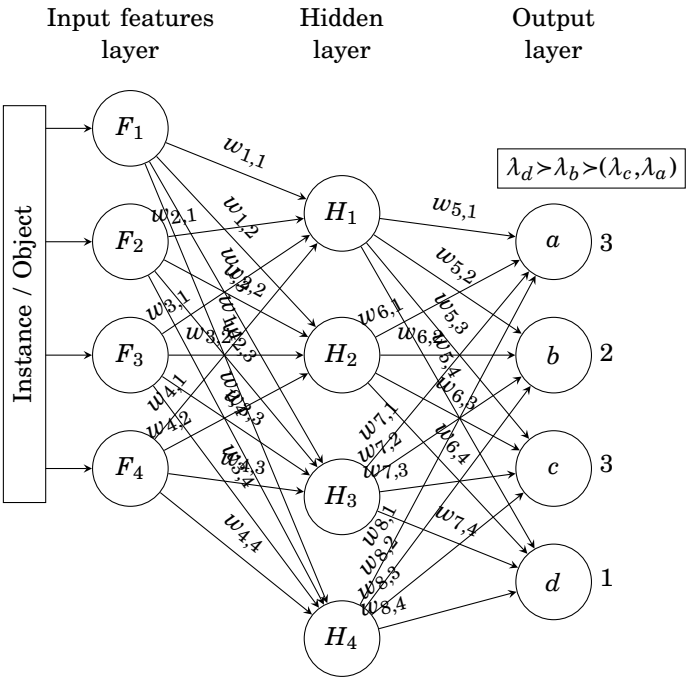


Fig. 4: *PNN* architecture where n = 4, H_n=4, f_n=4

III. *PNN* BACKPROPAGATION

A. Backpropagation of the last layer

In this step, stairstep activation and error function are differentiated for every hidden neuron as given in Eq. (7).

$$\frac{\partial \tau}{\partial \omega_{5,1}} = \frac{\partial \tau}{\partial y_1} \cdot \frac{\partial y_1}{\partial y_{a1}} \cdot \frac{\partial y_{a1}}{\partial \omega_{5,1}} \tag{7}$$

where y_{a1} is the neuron 1 output before activation function. The differentiation of the four labels ranking activation function is represented in Eqs. (10)-(13).

$$\frac{\partial y_1}{\partial y_{a1}} = r \cdot q \cdot \sum_{i=0}^{3} (1 - tanh(s \cdot y_{a1} \cdot (k - i \cdot t/2))^2) \tag{8}$$

$$\frac{\partial \tau}{\partial w_{5,1}} = \frac{\partial \tau}{\partial y_1} \cdot \frac{\partial y_{a1}}{\partial w_{5,1}} \cdot r \cdot q \cdot \sum_{i=0}^{3} (1 - tanh(s \cdot y_{a1} \cdot (k - i \cdot t/2))^2) \tag{9}$$

$$\frac{\partial \tau}{\partial w_{5,1}} = \frac{\partial \tau}{\partial y_1} \cdot H_1 \cdot r \cdot q \cdot \sum_{i=0}^{3} (1 - tanh(s \cdot y_{a1} \cdot (k - i \cdot t/2))^2) \tag{10}$$

Similarly,

$$\frac{\partial \tau}{\partial w_{5,2}} = \frac{\partial t}{\partial y_1} \cdot H_2 \cdot r \cdot q \cdot \sum_{i=0}^{3} (1 - tanh(s \cdot y_{a1} \cdot (k - i \cdot t/2))^2) \tag{11}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 June 2020 doi:10.20944/preprints201904.0091.v3

$$\frac{\partial \tau}{\partial w_{5,3}} = \frac{\partial t}{\partial y_1} \cdot H_3 \cdot r \cdot q \cdot \sum_{i=0}^{3} (1 - tanh(s \cdot y_{a1} \cdot (k - i \cdot t/2))^2) \quad (12)$$

$$\frac{\partial \tau}{\partial w_{5,4}} = \frac{\partial t}{\partial y_1} \cdot H_4 \cdot r \cdot q \cdot \sum_{i=0}^{3} (1 - tanh(s \cdot y_{a1} \cdot (k - i \cdot t/2))^2) \quad (13)$$

The differentiation of error function of label 1 is given in Eq. (14)

$$\frac{\partial \tau}{\partial y_1} = \frac{2(y_1 - y_{t1})}{n(n-1)} \quad (14)$$

Similarly, the differentiation of error function for all labels is given in Eqs. (15)-(17).

$$\frac{\partial \tau}{\partial y_2} = \frac{2(y_2 - y_{t2})}{n(n-1)} \quad (15)$$

$$\frac{\partial \tau}{\partial y_3} = \frac{2(y_3 - y_{t3})}{n(n-1)} \quad (16)$$

$$\frac{\partial \tau}{\partial y_4} = \frac{2(y_4 - y_{t4})}{n(n-1)} \quad (17)$$

$$\omega_{5,1new} = \omega_{5,1old} - \eta . \frac{\partial \tau}{\partial w_{5,1}} \quad (18)$$

where η, $\omega_{5,1}$ and a are the learning rate, weight of H_1 and output node, respectively as shown in Fig. 4.

$$\omega_{5,1new} = \omega_{5,1old} - \eta . \frac{\partial \tau}{\partial y_1} \cdot \frac{\partial y_1}{\partial y_{a1}} \cdot H_1 \quad (19)$$

Similarly,

$$\omega_{5,2new} = \omega_{5,2old} - \eta . \frac{\partial \tau}{\partial y_1} \cdot \frac{\partial y_1}{\partial y_{a1}} \cdot H_2 \quad (20)$$

$$\omega_{5,3new} = \omega_{5,3old} - \eta . \frac{\partial \tau}{\partial y_1} \cdot \frac{\partial y_1}{\partial y_{a1}} \cdot H_3 \quad (21)$$

$$\omega_{5,4new} = \omega_{5,4old} - \eta . \frac{\partial \tau}{\partial y_1} \cdot \frac{\partial y_1}{\partial y_{a1}} \cdot H_4 \quad (22)$$

B. Backpropagation for hidden and input layers

This section shows the calculation of the weights of input neurons x_1 to x_4 using Eqs. (23)-(36).

$$\frac{\partial \tau_{total}}{\partial w_{1,1}} = \frac{\partial \tau}{\partial H_1} \cdot \frac{\partial H_1}{\partial H_{a1}} \cdot \frac{\partial H_{a1}}{\partial w_{1,1}} \quad (23)$$

where H_a is the hidden neuron output before activation function.

$$\frac{\partial \tau_{total}}{\partial w_{1,1}} = \frac{\partial \tau_{total}}{\partial H_1} \cdot \frac{\partial H_1}{\partial H_{a1}} \cdot \frac{\partial H_{a1}}{\partial \omega_{1,1}} \quad (24)$$

$$\frac{\partial \tau_{total}}{\partial w_{1,1}} = (\frac{\partial \tau_1}{\partial H_1} + \frac{\partial \tau_2}{\partial H_2} + \frac{\partial \tau_3}{\partial H_3} + \frac{\partial \tau_4}{\partial H_4}) \cdot \frac{\partial H_1}{\partial H_{a1}} \cdot \frac{\partial H_{a1}}{\partial \omega_{1,1}} \quad (25)$$

$$\frac{\partial \tau_1}{\partial H_1} = \frac{\partial \tau_1}{\partial y_{a1}} \cdot \frac{\partial y_{a1}}{\partial H_1} \quad (26)$$

$$\frac{\partial \tau_1}{\partial y_1} = \frac{\partial \tau_1}{\partial y_1} \cdot \frac{\partial y_1}{\partial y_{a1}} \quad (27)$$

$$\frac{\partial \tau_1}{\partial H_1} = \frac{\partial \tau_1}{\partial y_1} \cdot \frac{\partial y_1}{\partial y_{a1}} \cdot \frac{\partial y_{a1}}{\partial H_1} \quad (28)$$

Similarly,

$$\frac{\partial \tau_2}{\partial H_2} = \frac{\partial \tau_2}{\partial y_2} \cdot \frac{\partial y_2}{\partial y_{a2}} \cdot \frac{\partial y_{a2}}{\partial H_2} \quad (29)$$

$$\frac{\partial \tau_3}{\partial H_{3out}} = \frac{\partial \tau_3}{\partial y_3} \cdot \frac{\partial y_3}{\partial y_{a3}} \cdot \frac{\partial y_{a3}}{\partial H_3} \quad (30)$$

$$\frac{\partial \tau_4}{\partial H_4} = \frac{\partial \tau_4}{\partial y_4} \cdot \frac{\partial y_4}{\partial y_{a4}} \cdot \frac{\partial y_{a4}}{\partial H_4} \quad (31)$$

$$\omega_{1,1new} = \omega_{1,1old} - \eta . \frac{\partial \tau_{total}}{\partial \omega_{1,1}} \quad (32)$$

Similarly,

$$\omega_{1,2new} = \omega_{1,2old} - \eta . \frac{\partial \tau_{total}}{\partial \omega_{1,2}} \quad (33)$$

$$\omega_{1,3new} = \omega_{1,3old} - \eta . \frac{\partial \tau_{total}}{\partial \omega_{1,3}} \quad (34)$$

$$\omega_{1,4new} = \omega_{1,4old} - \eta . \frac{\partial \tau_{total}}{\partial \omega_{1,4}} \quad (35)$$

$$\omega_{1,1new} = \omega_{1,1old} - \eta . \left(\frac{\partial \tau_1}{\partial H_1} + \frac{\partial \tau_2}{\partial H_2} + \frac{\partial \tau_3}{\partial H_3} + \frac{\partial \tau_4}{\partial H_4} \right) \cdot r \cdot q \cdot \sum_{i=0}^{3} (1 - tanh(s \cdot x_1 \cdot (k - i \cdot t/2))^2) \cdot H_{a1} \quad (36)$$

IV. Proof of Convergence

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

$$a_j = \varphi(\omega^T . a_i) \quad (37)$$

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

$$\{a_1, t_1\}, \{a_2, t_2\}, .., \{a_j, t_q\} \quad (38)$$

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.

$$a_j = \omega^T . a_i = \omega^T . z \quad (39)$$

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

$$\omega_{new} = \omega_{old} + E_{err} . z \quad (40)$$

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).

$$\omega(k) = \omega(k-1) + z'(k-1) \quad (41)$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 June 2020 doi:10.20944/preprints201904.0091.v3

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)

$$z_1, z_2, z_3, .., z_Q. \tag{42}$$

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

$$\omega_s^T z_1 > \delta > 0 \tag{43}$$

$$\omega_s^T z_2 > \delta > 1 \tag{44}$$

$$\omega_s^T z_3 > \delta > 2 \tag{45}$$

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)

$$\omega(k) = z'(0) + z'(1) + .. + z'(k-1) \tag{46}$$

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

$$\omega_s^T . x(k) = \omega_s^T . z'(0) + \omega_s^T . z'(1) + .. + \omega_s^T . z'(k-1) \tag{47}$$

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

$$\omega_s^T . z'(i) > \delta \tag{48}$$

Therefore,

$$\omega_s^T . \omega(k) > k\delta \tag{49}$$

From the Cauchy-Schwarts inequality [40]

$$(\omega_s^T . \omega(k))^2 \leq \| \omega_s \|^2 \| \omega(k) \|^2 \tag{50}$$

where

$$\| \omega \|^2 = \omega^T \omega \tag{51}$$

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

$$\| \omega(k) \|^2 \geq \frac{(\omega_s^T \omega(k))^2}{\| \omega_s \|^2} > \frac{(k\delta)^2}{\| \omega_s \|^2} \tag{52}$$

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

$$\| \omega(k) \|^2 = \omega^T(k) . \omega(k) \tag{53}$$

$$\| \omega(k) \|^2 = [\omega(k-1) + z'(k-1)]^T [\omega(k-1) + z'(k-1)] \tag{54}$$

$$\| \omega(k) \|^2 = \omega^T(k-1)\omega(k-1) + 2\omega^T(k-1)z'(k-1) + z'^T(k-1)z'(k-1) \tag{55}$$

Eq. (52) can be simplified as

$$\| \omega(k) \|^2 \leq \| \omega(k-1) \|^2 + \| z'(k-1) \|^2 \tag{56}$$

Eq. (53) can be repeated for $\| \omega(k-1) \|^2$, $\| \omega(k-2) \|^2$, to obtain

$$\| \omega(k) \|^2 \leq \| z'(0) \|^2 + \| z'(1) \|^2 + ... + \| z'(k-1) \|^2 \tag{57}$$

If $\Pi = max\{\| z'(i) \|\}$, this upper bound can be simplified to Eq. (58).

TABLE I: Three benchmark datasets for label ranking; preference mining [25], semi-synthetic (*SS*) [28] and real-world datasets

type	dataset	category	# inst.	# attr.	# labels
Mining data	algae	chemical stat.	317	11	4
	german.2005	user pref.	413	29	5
	german.2009	user pref.	413	32	5
	sushi	user pref.	5000	10	10
	top7movies	user pref.	602	7	7
Real data	cold	biology	2,465	23	4
	diau	biology	2,465	24	6
	dtt	biology	2,465	24	4
	heat	biology	2,465	24	6
	spo	biology	2,465	24	11
Semi-Synthesis data	authorship	A	841	70	4
	bodyfat	B	252	7	7
	calhousing	B	20,640	6	5
	cpu-small	B	8192	3	4
	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	3	4
	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

$$\| \omega(k) \|^2 \leq K\Pi \tag{58}$$

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

V. Experimental Results

A. Datasets

PNN is experimented using three different types of benchmark datasets to evaluate the multi-label ranking performance. The first type of dataset focuses on exceptions preference mining [24], 'algae' dataset is one the first type that highlights indifferences preferences problem, where labels have repeated preference value [25]. German elections 2005, 2009 and modified sushi are considered new and restricted preference datasets. The second type is real-world data related to biological science [21]. The third type of dataset is semi-synthetic (*SS*) taken from the *KEBI* Data Repository at the Philipps University of Marburg [28]. All datasets do not have ranking ground truth and all labels have a continuous permutation space of relations between labels. All data are normalized between -1 and 1. Table I summarizes the main characteristics of the datasets.

B. Results

PNN is evaluated by restricted and non-restricted label ranking datasets. The results are derived in terms of Kendall's tau coefficient with ten-fold cross validation.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 June 2020 doi:10.20944/preprints201904.0091.v3

TABLE II: Preference mining ranking performance in terms of Kendall's tau coefficient and learning step and number of hidden neurons

Preference mining exceptions Data			
dataset	**kendall's tau**	**l.step**	**# h.neurons**
algae	0.651	0.0001	250
german2005	0.93	0.0001	17
german2009	0.85	0.0001	17
sushi	0.78	0.0003	300
top7 movies	0.692	0.004	16

1) Preference Mining results: Ranking performance of new preference mining dataset is represented in table II. To enhance the ranking performance of the repeated label values of algea dataset, total 250 hidden neurons are used. However, restricted labels ranking datasets of the same type, i.e, (german elections and sushi) did not require high number of hidden neurons and took less computation cost.

2) Restricted preferences results: Table III summarizes *PNN* ranking performance of strict label ranking datasets by learning rate and the total number of hidden neurons. The results are compared with the four methods for label ranking; supervised clustering [27], supervised decision tree [28], multi-layer perceptron label ranking [29] and label ranking tree forest (*LRT*) [35]. The comparison selects only the best approach for each method.

C. PNN Performance

During the experiment, it was found that ranking performance increases with an increase in the number of hidden neurons in hidden layers. All the results are held using 1 hidden layer with a various number of hidden neurons from (50 to 300). Kendall's tau error converges and reaches close to 1 after 50,000 iterations as shown in fig 6.

Few datasets have low labels classes separability, i.e., (wisconsin, sushi). To enhance the ranking performance, the number of hidden neurons are increased to 300.

D. Missing labels evaluation

PNN is evaluated by removing a random number of labels per each instance. *PNN* marked the missing label as -1; *PNN* neglects error calculation during backpropagation, $\delta = 0$. Thus, the missing label weights remain constants per each learning iteration. Missing label approach is applied to the dataset by 20% and 60% of the training data.

It is noticed that ranking performance decreases when the number of the missing labels increases. This evaluation is performed on iris dataset as shown in Fig. 7.

Table IV compares *PNN* with the similar approaches used for multi-label ranking. These approaches are; Decision trees [27], *MLP-LR* [29] and label ranking trees forest *LRT* [35]. In this comparison, we choose the method that have the best results for each approach. The biological real world dataset was experimented using supervised clustering (*SC*) [27], Table V represents the comparison

TABLE III: *PNN* ranking performance in terms of Kendall's tau coefficient and learning step and number of hidden neurons

type	**dataset**	**kendall's tau**	**l.step**	**# h.neurons**
Real data	cold	0.78	0.004	10
	diau	0.49	0.001	12
	dtt	0.98	0.002	10
	heat	0.976	0.004	10
	spo	0.972	0.0007	12
Semi-Synthesis data	authorship	0.92	0.004	30
	bodyfat	0.31	0.004	14
	calhousing	0.356	0.009	14
	cpu-small	0.58	0.009	50
	elevators	0.82	0.05	20
	fried	0.997	0.02	100
	glass	0.99	0.006	50
	housing	0.85	0.004	50
	iris	0.998	0.004	15
	pendigits	0.97	0.004	22
	segment	0.963	0.001	19
	stock	0.92	0.004	12
	vehicle	0.9	0.004	200
	vowel	0.94	0.004	22
	wine	0.977	0.01	50
	wisconsin	0.72	0.0007	200

TABLE IV: *PNN* performance comparison in terms of Kendall's tau coefficient

Multi Label Ranking Methods					
dataset	**S.Clust.**	***DT***	***MLP-LR***	***LRT***	***PNN***
authorship	0.854	0.936(IBLR)	0.889(LA)	0.882	0.92
bodyfat	0.09	0.281(CC)	0.075(CA)	0.117	0.31
calhousing	0.28	0.351(IBLR)	0.130(SSGA)	0.324	0.356
cpu-small	0.274	0.50(IBLR)	0.357(CA)	0.447	0.58
elevators	0.332	0.768(CC)	0.687(LA)	0.760	0.82
fried	0.176	0.99(CC)	0.660(CA)	0.890	0.997
glass	0.766	0.883(LRT)	0.818(LA)	0.883	0.99
housing	0.246	0.797(LRT)	0.574(CA)	0.797	0.85
iris	0.814	0.966(IBLR)	0.911(LA)	0.947	0.998
pendigits	0.422	0.944(IBLR)	0.752(CA)	0.935	0.97
segment	0.572	0.959(IBLR)	0.842(CA)	0.949	0.963
stock	0.566	0.927(IBLR)	0.745(CA)	0.895	0.92
vehicle	0.738	0.862(IBLR)	0.801(LA)	0.827	0.9
vowel	0.49	0.90(IBLR)	0.545(CA)	0.794	0.94
wine	0.898	0.949(IBLR)	0.931(LA)	0.882	0.977
wisconsin	0.09	0.629(CC)	0.235(CA)	0.343	0.72
Average	0.475	0.79	0.621	0.730	0.825

between *PNN* and supervised clustering on biological real world data in terms of $Loss_{LR}$ as given in Eq. (59).

$$\tau = 1 - 2.Loss_{LR} \tag{59}$$

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

E. Discussion and Future Work

It can be noticed from table IV that *PNN*outperforms on *SS* datasets with τ_{Avg}=0.825, whereas other methods such as, supervised clustering, decision tree, *MLP-ranker* and *LRT*, have results τ_{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 biological real world dataset as shown

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 June 2020 doi:10.20944/preprints201904.0091.v3

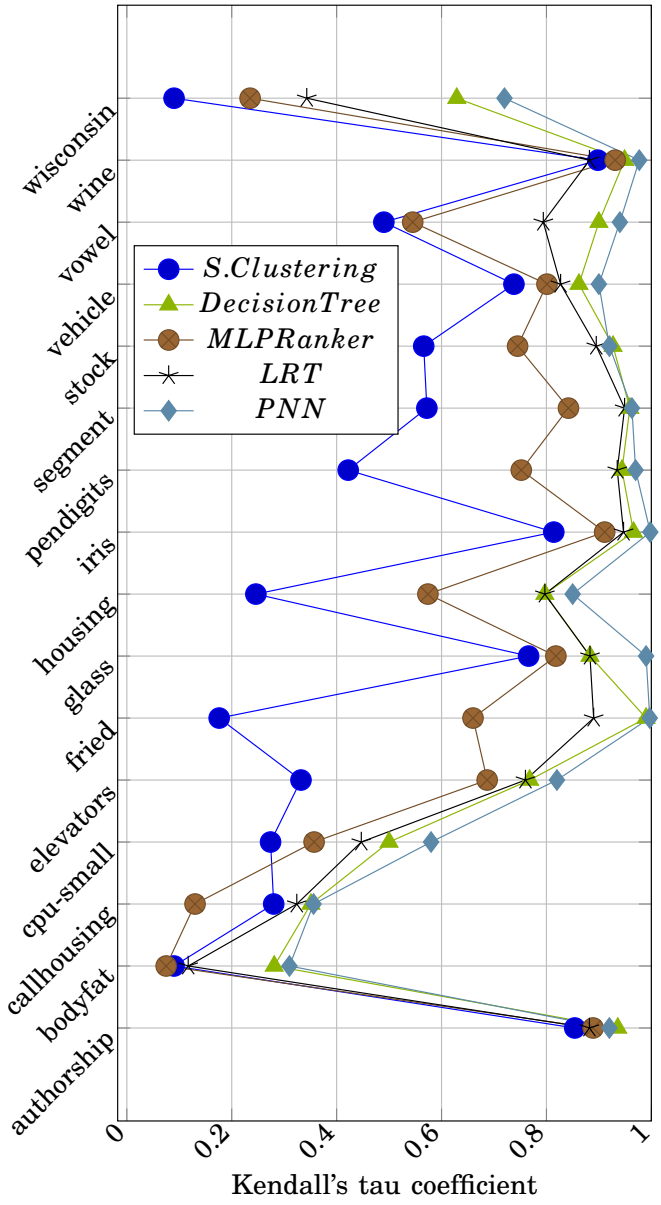


Fig. 5: Ranking performance comparison of *PNN* with other approaches.

in table V. *PNN* increases the number of iterations to enhance the ranking performance of missing labels of iris dataset as shown in Fig. 7.

The superiority of *PNN* for ranking is encoding the Multi-label preference relation to numeric values and rank the output labels simultaneously. *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.

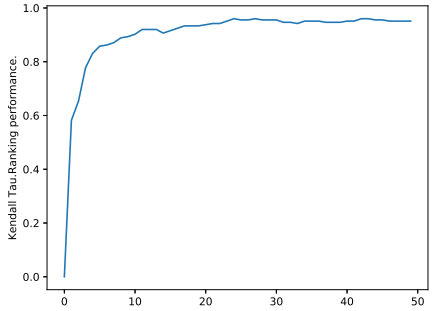


Fig. 6: Ranking convergence of iris dataset over 50,000 iterations

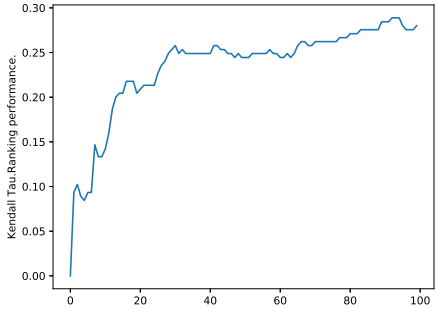


Fig. 7: Ranking convergence of iris dataset has 60% missing labels over 100,000 iterations

VI. Conclusion

This paper proposed a novel method to rank a complete Multi-label space. The preference neural network uses a smooth stairsteps single activation function to rank the labels. The novelty of this neural network is indexing all output labels as output neurons and proposing new activation function for ranking. The neuron output structure can be mapped to numeric ranking value; thus, *PNN* solves sub grouping ranking problems by assigning the rank value to more than one output index.

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

Biological real world data		
dataset	**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

 doi:10.20944/preprints201904.0091.v3

Acknowledgement

This work was supported in part by the Australian Research Council (ARC) under discovery grant DP180100670 and DP180100656.

References

[1] J. Fürnkranz and E. Hüllermeier, "Preference Learning: An Introduction," in Preference Learning, J. Fürnkranz and E. Hüllermeier, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 1-17.
[2] R. Brafman and C. Domshlak. "Preference handling - an introductory tutorial". AI Magazine, 30(1): 58-86, 2009.
[3] G. Adomavicius and A. Tuzhilin. "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions." IEEE Transactions on Knowledge and Data Engineering, 17(6):734-749, 2005.
[4] M. Montaner, B. Lopez and J.L. De La Rosa, "A Taxonomy of Recommender Agents on the Internet." Artif. Intell. Rev. 19(4), 285-330 (2003)
[5] F. Aiolli,"A preference model for structured supervised learning tasks", in Proceedings of the IEEE International Conference on Data Mining (ICDM) (2005), pp. 557-560
[6] K. Crammer and Y. Singer, "Pranking with ranking", in Advances in Neural Information Processing Systems (NIPS) (2002), pp. 641-647
[7] C. Burges et al., "Learning to rank using gradient descent," presented at the Proceedings of the 22nd international conference on Machine learning, Bonn, Germany, 2005.
[8] Y. Zhou, Y. Liu, J. Yang, X. He, and L. Liu, "A Taxonomy of Label Ranking Algorithms," JCP, vol. 9, pp. 557-565, 2014.
[9] J. Furnkranz and E. Hüllermeier, "Pairwise Preference Learning and Ranking," in Machine Learning: ECML 2003, Berlin, Heidelberg, 2003, pp. 145-156: Springer Berlin Heidelberg.
[10] J. Frnkranz and E. Hüllermeier, "Preference Learning," Springer-Verlag, 2010, p. 454.
[11] S. Har-Peled, D. Roth, and D. Zimak, "Constraint classification for multiclass classification and ranking," presented at the Proceedings of the 15th International Conference on Neural Information Processing Systems, 2002.
[12] P. L. H. Yu, W. M. Wan, and P. H. Lee, "Decision Tree Modeling for Ranking Data," in Preference Learning, J. Furnkranz and E. Hüllermeier, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 83-106.
[13] C. R. de Sá, C. Soares, A. M. Jorge, P. J. Azevedo, and J. P. da Costa, "Mining association rules for label ranking" in PAKDD (2), 2011, pp. 432-443.
[14] A. G. e. Ivakhnenko, V. G. v. Lapa, S. United, and S. Joint Publications Research, Cybernetic predicting devices. New York: CCM Information Corp., 1965.
[15] A. G. e. Ivakhnenko, V. G. Lapa, and R. N. McDonough, "Cybernetics and forecasting techniques." New York: American Elsevier, 1967.
[16] A. G. Ivakhnenko, "The Group Method of Data Handling-A Rival of the Method of Stochastic Approximation," Soviet Automatic Control, Vol. 1, No. 3, 1968, pp. 43-55.
[17] R. Isermann, S. Ernst, and O. Nelles,"Identification with Dynamic Neural Networks - Architectures, Comparisons, Applications," IFAC Proceedings Volumes, vol. 30, no. 11, pp. 947-972, 1997/07/01/ 1997.
[18] J. Schmidhuber, "Deep learning in neural networks: An overview," Neural Networks, vol. 61, pp. 85-117, 2015/01/01/ 2015.
[19] X. Wang, T. Huang, and X. Liu, "Handwritten Character Recognition Based on BP Neural Network," in 2009 Third International Conference on Genetic and Evolutionary Computing, 2009, pp. 520-524.
[20] G. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme Learning Machine for Regression and Multiclass Classification," IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 42, no. 2, pp. 513-529, 2012.
[21] E. Hüllermeier, J. Furnkranz, W. Cheng, K. Brinker, "Label ranking by learning pairwise preferences". Artif. Intell. 178, 1897-1916 2008.
[22] O. Dekel, D.Manning, Y. Singer, "Log-linear models for label ranking", in Advances in Neural Information Processing Systems, vol. 16 2003.
[23] Y. H. Jung and A. Tewari, "Online Boosting Algorithms for Multi-label Ranking," ed. Ithaca: Cornell University Library, arXiv.org, 2018.
[24] C. R. de Sá, W. Duivesteijn, P. Azevedo, A. M. Jorge, C. Soares, and A. Knobbe, "Discovering a taste for the unusual: exceptional models for preference mining," Machine Learning, vol. 107, no. 11, pp. 1775-1807, 2018/11/01 2018.
[25] http://dx.doi.org/10.17632/3mv94c8jpc.2
[26] S. Har-Peled, D. Roth, D. Zimak, Constraint classification: A new approach to multiclass classification, in Proceedings of the Thirteenth International Conference on Algorithmic Learning, Theory 2002.
[27] M. Grbovic, N. Djuric, S. Guo, and S. Vucetic, "Supervised clustering of label ranking data using label preference information," Machine Learning, vol. 93, no. 2, pp. 191-225, 2013/11/01 2013.
[28] W. Cheng et al., "Decision tree and instance-based learning for label ranking," presented at the Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, Quebec, Canada, 2009.
[29] G. Ribeiro, W. Duivesteijn, C. Soares, and A. Knobbe,"Multilayer perceptron for label ranking," presented at the Proceedings of the 22nd international conference on Artificial Neural Networks and Machine Learning - Volume Part II, Lausanne, Switzerland, 2012.
[30] W. Cheng and E. Hüllermeier, "Instance-based label ranking using the mallows model," in ECCBR Workshops, 2008, pp. 143-157.
[31] M. Grbovic, N. Djuric, and S. Vucetic,"Learning from pairwise preference data using Gaussian mixture model," Preference Learning: Problems and Applications in AI, p. 33, 2012.
[32] M. G. Kendall, "Rank Correlation Methods." London, England: Griffin, 1970.
[33] C. Spearman, "The proof and measurement of association between two things,"The American Journal of Psychology, vol. 15, no. 1, pp. 72-101, 1904.
[34] A. Aiguzhinov, C. Soares, and A. P. Serra, "A Similarity-Based Adaptation of Naive Bayes for Label Ranking: Application to the Metalearning Problem of Algorithm Recommendation,"in Discovery Science, Berlin, Heidelberg, 2010, pp. 16-26: Springer Berlin Heidelberg.
[35] C. R. de Sá, C. Soares, A. Knobbe, and P. Cortez, "Label Ranking Forests," Expert Systems, vol. 34, no. 1, 2017.
[36] Q. Song, H. Jin, X. Huang, and X. Hu, "Multi-Label Adversarial Perturbations," ed. Ithaca: Cornell University Library, arXiv.org, 2019.
[37] Y. Yan, Y. Wang, G. Wen-Chao, Z. Bo-Wen, C. Yang, and Y. Xu-Cheng, "LSTM 2: Multi-Label Ranking for Document Classification," (in English), Neural Processing Letters, vol. 47, no. 1, pp. 117-138, 2018 2018-03-03 2018.
[38] Y. Zhou and G. Qiu, "Random forest for label ranking," (in English), Expert Systems with Applications, vol. 112, p. 99, 2018 Dec 01 2018-10-05 2018.
[39] B. Guo, C. Hou, J. Shan, and D. Yi, "Low Rank Multi-Label Classification with Missing Labels," in 2018 24th International Conference on Pattern Recognition (ICPR), 2018, pp. 417-422.
[40] V. Bergelson, "Part V: Theorems and Problems - 09. Ergodic Theorems." Princeton: Princeton University Press, 2008, pp. 3-691.