Preference Net: Image Recognition using Ranking Reduction to Classification

Preprint · December 2021

DOI: 10.20944/preprints202112.0140.v1

3 authors, including:

Ayman Elgharabawy
University of Technology Sydney
9 PUBLICATIONS **16** CITATIONS
SEE PROFILE

Mukesh Prasad
University of Technology Sydney
138 PUBLICATIONS **2,242** CITATIONS
SEE PROFILE

Preference Net: Image Recognition using Ranking Reduction to Classification

AYMAN ELGHARABAWY, Australian Artificial Intelligence Institute, School of Computer Science, University of Technology Sydney, Australia

MUKESH PRASAD*, Australian Artificial Intelligence Institute, School of Computer Science, University of Technology Sydney, Australia

CHIN TENG LIN, Australian Artificial Intelligence Institute, School of Computer Science, University of Technology Sydney, Australia

Accuracy and computational cost are the main challenges of deep neural networks in image recognition. This paper proposes an efficient ranking reduction to binary classification approach using a new feed-forward network and feature selection based on ranking the image pixels. Preference net (*PN*) is a novel deep ranking learning approach based on Preference Neural Network (*PNN*), which uses new ranking objective function and positive smooth staircase (*PSS*) activation function to accelerate the image pixels' ranking. *PN* has a new type of weighted kernel based on spearman ranking correlation instead of convolution to build the features matrix. The *PN* employs multiple kernels that have different sizes to partial rank image pixels' in order to find the best features sequence. *PN* consists of multiple *PNN*s' have shared output layer. Each ranker kernel has a separate *PNN*. The output results are converted to classification accuracy using the score function. *PN* has promising results comparing to the latest deep learning (*DL*) networks using the weighted average ensemble of each *PN* models for each kernel on CFAR-10 and Mnist-Fashion datasets in terms of accuracy and less computational cost.

Additional Key Words and Phrases: Preference learning; Preference neural network; Deep label ranking; Stairstep; Image Recognition

ACM Reference Format:
Ayman Elgharabawy, Mukesh Prasad, and Chin Teng Lin. 2021. Preference Net: Image Recognition using Ranking Reduction to Classification. 1, 1 (December 2021), 18 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

I 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 [44, 48]. 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 [47], Convolutional Deep Belief Network (*CDBN*) [59], Convolutional Neural Network (*CNN*), Deep Belief Network (*DBN*) [23], Deep Boltzmann Machine (*DBM*) [43], Long short-term Memory Network

Authors' addresses: Ayman Elgharabawy, ayman.elgharabawy@uts.edu.au, Australian Artificial Intelligence Institute, School of Computer Science, University of Technology Sydney, Ultimo, Sydney, NSW, Australia, 2007; Mukesh Prasad, mukesh.prasad@uts.edu.au, Australian Artificial Intelligence Institute, School of Computer Science, University of Technology Sydney, Ultimo, Sydney, NSW, Australia, 2007; Chin Teng Lin, chin-teng.lin@uts.edu.au, Australian Artificial Intelligence Institute, School of Computer Science, University of Technology Sydney, Ultimo, Sydney, NSW, Australia, 2007.

Manuscript submitted to ACM

(*LSTM*) [55, 58], Recurrent Neural Network (*RNN*) [33], and Restricted Boltzmann Machine (*RBM*) [5]) are the main deep learning approaches used in deep learning.

In computer vision, the convolutions architectures is dominant in *DL* by using a variation of CNN-like architectures [21, 32, 34, 49].Some architectures replaced the convolutions entirely [33, 55, 58]. However, these models have succeeded in image classification; they have not yet been scaled effectively on big-size images and use specialized attention patterns. Therefore, in large-scale image recognition, classic *ResNet* like architectures are still state-of-the -art [14, 38]. Therefore, kernel computation and scaling are still fixed and specialized to certain images type.

Label ranking (*LR*) is one of challenging categories of Preference learning (*PL*) that gained importance in information retrieval by search engines [3, 10]. Unlike the common problems of regression and classification, *LR* involves predicting the relationship between multiple label orders. For a given instance x from the instance space x, there is a label $\mathcal{L}$ associated with x, $\mathcal{L} \in \pi$, where $\pi = \{\lambda_1, .., \lambda_n\}$, and 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 set $\pi = \{1, 2, \cdots, n\}$. The label order has the following three features. irreflexive where $\lambda_a \not\succ \lambda_a$,transitive where $(\lambda_a \succ \lambda_b) \wedge (\lambda_b \succ \lambda_c) \implies \lambda_a \succ \lambda_c$ and asymmetric $\lambda_a \succ \lambda_b \implies \lambda_b \not\succ \lambda_a$. Label preference takes one of two forms, strict and non-strict order. The strict label order $(\lambda_a \succ \lambda_b \succ \lambda_c \succ \lambda_d)$ can be represented as $\pi = (1, 2, 3, 4)$ and for non-restricted total order $\pi = (\lambda_a \succ \lambda_b \simeq \lambda_c \succ \lambda_d)$ can be represented as $\pi = (1, 2, 2, 3)$, where a, b, c, and, d are the label indexes and $\lambda_a, \lambda_b, \lambda_c$ and λ_d are the ranking values of these labels.

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 $\pi = (1, 2, 0, 3)$ where 0 represents an incomparable relation since λ_c is not comparable to $(\lambda_a, \lambda_b, \lambda_d)$.

Various *LR* methods have been introduced in recent years [60], such as decomposition-based methods, statistical methods, similarity, and ensemble-based methods. decomposition methods include pairwise comparison [17, 18], log-linear models and constraint classification [20]. The pairwise approach introduced by Hüllermeier [11] divides the *LR* 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 [19], instance-based methods (Plackett-Luce) [7] and Gaussian mixture model [40] based approaches. For example, Mihajlo uses Gaussian mixture models to learn soft pairwise label preferences [40].

Using ranking to minimize classification loss was introduced by kotlowski [29] 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 [2] confirmed that it is hard to reach faultless ranking of all preference labels. Also, Balcan, Ailon, Abdulrahman, and Mamman [1, 6] proposed different robust approaches to reduce the ranking for better classification. The artificial neural network (*ANN*) for ranking was first introduced as (RankNet) by Burge to solve the problem of object ranking for sorting web documents by a search engine [8]. Rank net uses gradient descent and probabilistic ranking cost function for each object pair. The multilayer perceptron for label ranking (*MLP-LR*) [42] employs a network 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 output neuron by subtracting the actual - predicted value and using Kendall error as a global approach. Neither direction uses a ranking objective function in backpropagation (*BP*) nor learning steps.

A multi-valued activation function has been proposed by Aizenberg [4] using convex shape to support multi-values and complex numbers neural network. In addition, Moraga [41] introduced a similar function to design networks

for realizing any multivalued function; however, Moraga used exponential function derivative did not give promising results in the *PNN* implementation using the ranking objective function in *FF* and backpropagation (*BP*) steps.

Preference neural network (*PNN*) was introduced by Elgharabawy [13] 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 [13]. The deep neural network (*DNN*) is introduced for object ranking to solve document retrieval problems [50]. RankNet [8], RankBoost [16], and Lambda MART [52], and deep pairwise label ranking models [26], are convolution neural Network (*CNN*) approaches for the vector representation of the query and document-based. *CNN* is used for image retrieval [37] and label classification [25].

The *CNN* mentioned above, and their variants have some issues that can be broadly summarized into:

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) Slowing in computational performance due to *CNN* several layers.
3) Challenging 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 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 (Activation functions, Objective function, and network structure).

II *PN* COMPONENTS

II.I Positive Smooth Staircase (*PSS*)

PSS is the positive part of the *SS* function that was introduced in *SGPNN* [13]. *PSS* is a non-linear and monotonic activation function, positive smooth staircase (*PSS*) is represented as a bounded smooth staircase function starts from x=0 to ∞. Thus, it is not geometrically symmetrical around the y-axis as shown in Fig. 1. *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. The activation function is given in Eq. I. *PSS* is scaled by increasing the step width w

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

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

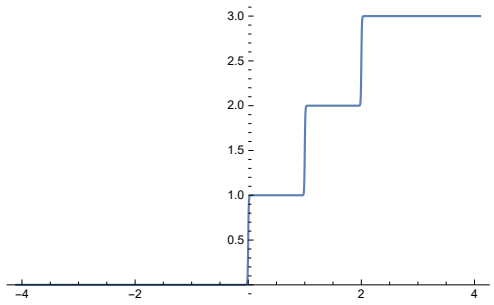


Fig. I. *PSS* activation function where $n = 3$ and step width $w = 1$

II.II Ranking Loss Function

Two main error functions have been used for label ranking; kendall τ [28] and *spearman* ρ [45]. 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 $\tau = 0.3$.

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

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

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

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

II.III Preference Neuron

Preference Neuron are a multi-valued neurons uses a *PSS*, that is the positive part of the *SS* activation function introduced by subgroup preference neural network [13] as an activation function. *PSS* 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 *PSS* 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. I.

II.IV Preference Neural Network

The *PNN* is fully connected to multiple-valued neurons and a single-hidden layer proposed by ELgharabawy [12, 13]. The input layer represents the number of features per data instance. The hidden neurons are equal to or greater than

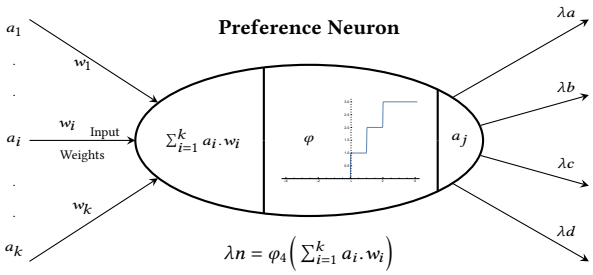


Fig. II. The structure of preference neuron where $\varphi_{n=4}$.

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. III.

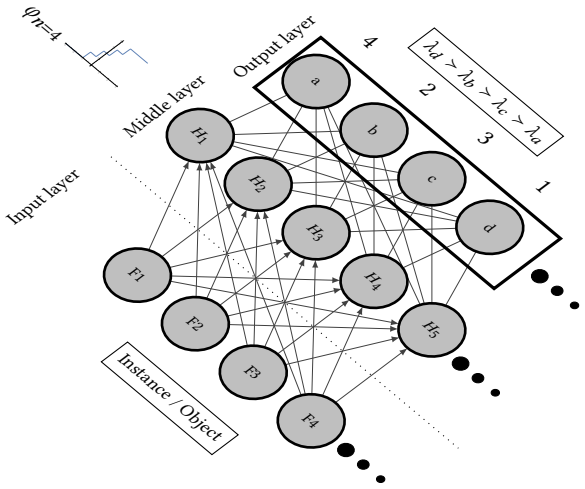


Fig. III. Architecture of Preference Neural Network (*PNN*)

Section III describes the data preprocessing steps, feature selections.

III *PN* STRUCTURE AND PROCESSING

The paper approach converts the multi-class classification problem into a multi-label ranking problem in two steps.

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 $s :\longrightarrow \mathbb{R}$. For output labels to choose the max value $f_s = Max(\lambda)$ for the binary classification that has $c : \mathcal{X} \longrightarrow \{\pm 1\}$. *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 ρ =1 correlated to class categorical output (1, 0, 0, 0, 0, 0, 0, 0, 0, 0).

III.I Image Prepossessing

III.I.I 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.

III.I.II Pixels' Ranking. The image pixels are ranked by flatten the image and rank the pixels values from 0-255 to 1-156 as illustrated in Fig. IV 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. IV (c).

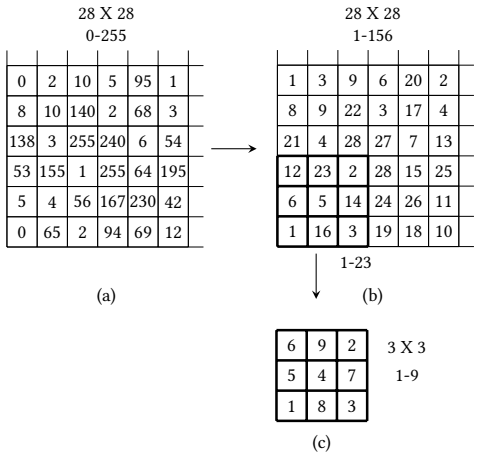


Fig. IV. Image pixel ranking for each flattened window.

III.I.III 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. VIII. 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).

The two approaches, Pixel ranking and Averaging, has been tested on two sample images of remote sensing and faces images to detect the similarity, and it shows high ranking correlations using different window size as shown in Fig VI. It detects the high correlation by starting from the large window size = image size. It reduces the size and scan until it reaches the highest correlation.

III.II Feature Extraction

This paper 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.

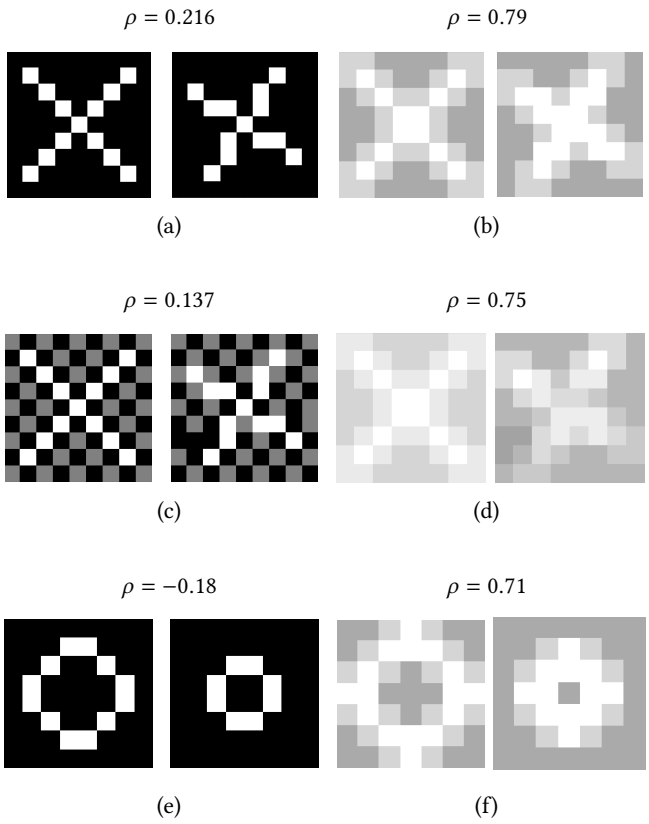


Fig. V. Sample of moving objects in (a) and (b) without and with averaging by window 2x2. The ranking of two flattened images are $\rho = 0.216$ and 0.79 in (a) and (b), respectively. Sample of moving noisy object in (c) and (d) without and with image averaging by a window of 2x2. The ranking of two 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.

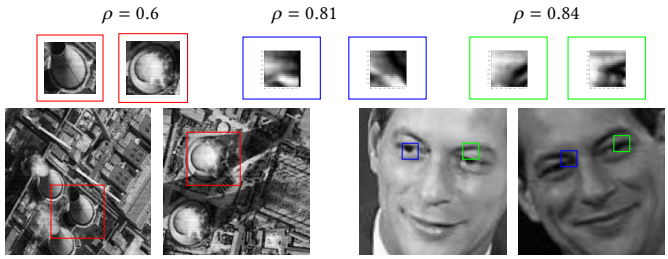


Fig. VI. Detecting the similarity in remote sensing and face recognition by ranking the image pixels after averaging the pixels using a 2x2 window.

III.II.I 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. IV shows the window size 3X3 range from $\lambda_{k_1} = 23$ to $\lambda_{k_2} = 9$.

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

III.II.II 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. III

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

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.

III.II.III 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*.

III.III *PN* 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. 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 VII. *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 [35]. Three kernels with sizes (6, 7, 8) are used for *CFAR-10* [30].

Table I represents a comparison between *CNN* and *PN* in terms of components. Table I shows a brief comparison between *CNN* and *PN*.

Table I. 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

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

IV ALGORITHMS

IV.I 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 has 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 II

IV.II Baseline Algorithm

Algorithm 1 represents the three functions of the network learning process; feed-forward (*FF*), *BP*, and updating weights. Algorithm 2 represents the learning flow of *PN*. Algorithm 3 represents the simplified BP function in two steps.

Algorithm 1: *PNN* learning flow

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

IV.III Complexity Analysis

IV.III.I 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 of number of instances t. The time complexity in Eq. IV

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

- *BB* starts with calculating the error of output layer $E_{ot} = \rho'_o \; Delta_o = E_{ot} \cdot SS'$ and $Delta_m = E_{mt} \cdot SS'$ then UW

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

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

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

IV.III.II Input Neurons.

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

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

where K_W is kernel width and K_H is kernel height.

Algorithm 2: *PN* Learning flow

Converting image to greyscale
Flattening image
Image pixel ranking
2D Image
Pixel averaging by a 2X2 window
Flattening image
Select one/more kernel sizes.
Random init. Kernel $K\omega_{x,y} \in \{-0.05, 0.05\}$
Random init. *PNN* $\omega_{i,j} \in \{-0.05, 0.05\}$
repeat
    2D Image
    Scanned window pixel ranking Img_w
    Compute $\rho(Img_w, Kw)$ feature map
    Max. Pooling.
    Flattening image
    PNN FF()
    PNN BP()
    PNN UW()
    Max. Pooling BP()
    Ranker kernel BP and UW()
until $\rho_{Avg} = 1$ *or* $\#iterations \geq 10^6$;

Algorithm 3: *PNN* BP

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

Table II. The number of window Correlated of two images in Fig. VIII (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

V 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.

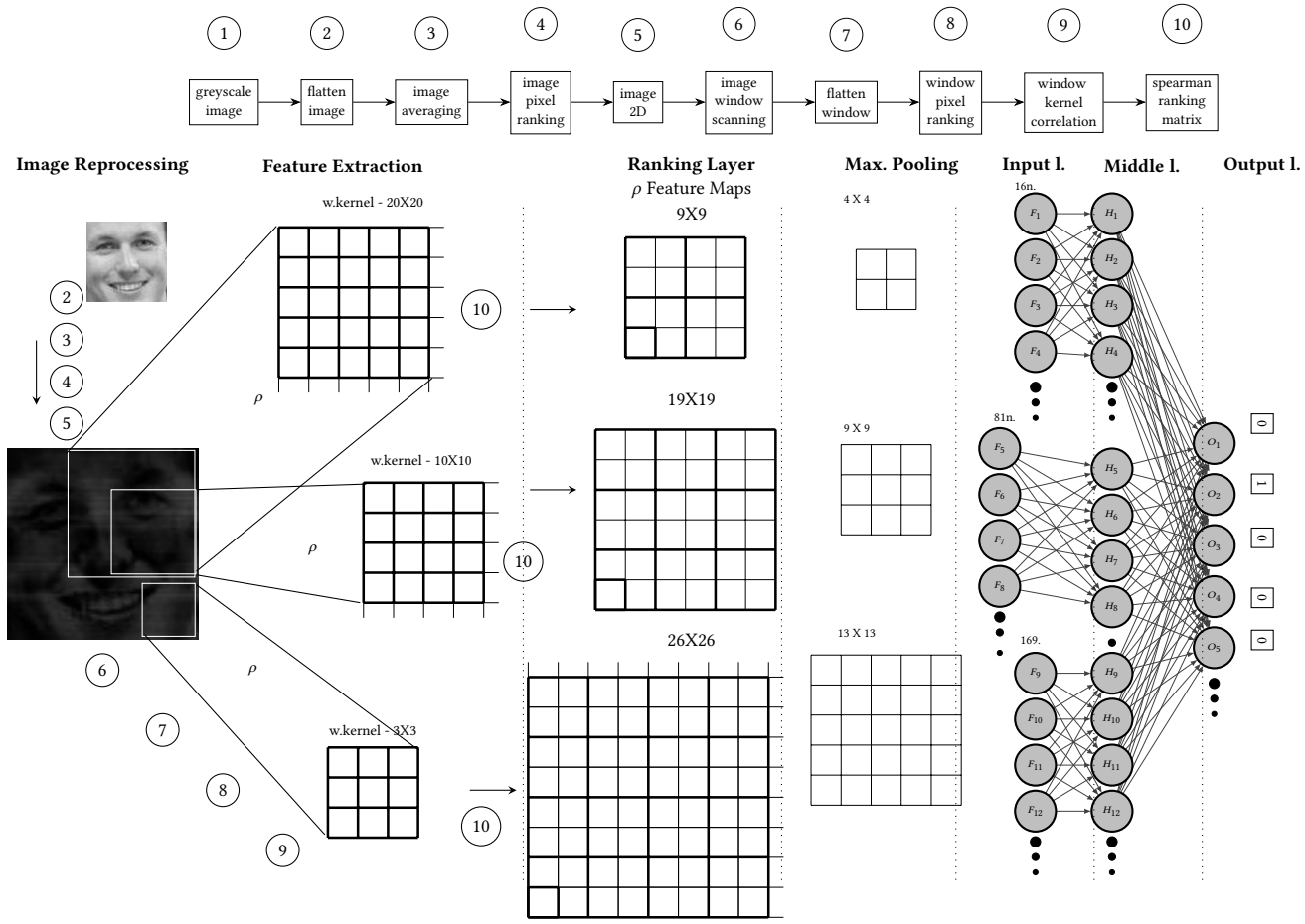


Fig. VII. The *PN* structure has three kernels and three *PNN*s where $\varphi_{n=2}$, $f_{1in} = 16, f_{2in} = 81, f_{3in} = 169$ and $\lambda_{out} = 15$, per $\langle x_1, \pi_1 \rangle, \pi \in \{\lambda_1, \lambda_2, \lambda_3 \cdots, \lambda_{15}\}$.

V..I 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. IX.

V..II 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.

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

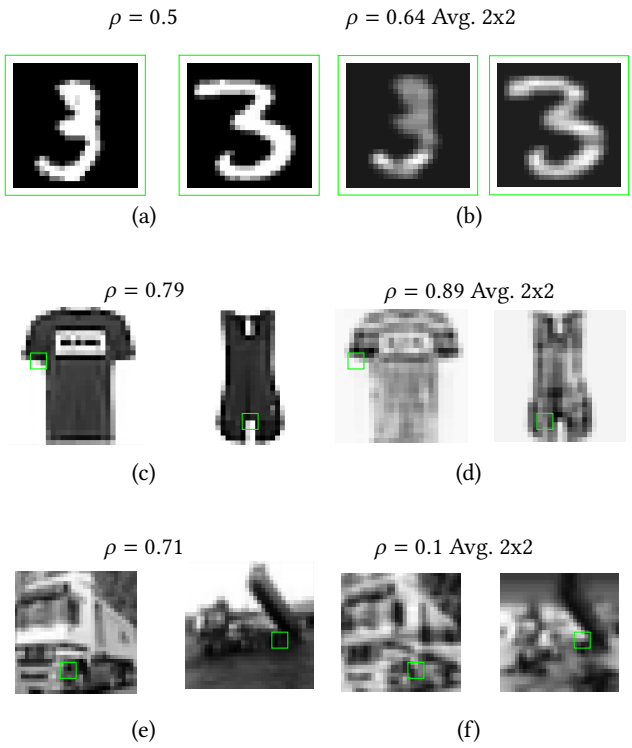


Fig. VIII. Image averaging using image width window size applied in Mnist , Mnist-Fashion and Cifar10 datasets in (a),(c), and (e) and the images after averaging using 2X2 in (b), (d), (f) in order to choose the best kernel windows by searching for the best correlation for all window sizes

VI EXPERIMENTS

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

VI.I Data sets

PN is evaluated using *Mnist* [36], Fashion-Mnist [53], and *CIFAR-10* [30] data sets.

VI.I.I Mnist. It consists of hand-written digits and is the most commonly used dataset within the deep learning community. The dataset is trivial to learn and simple to reach good performance [36]. It is included in the experiment for algorithm completeness of benchmarks.

VI.I.II Mnist-Fashion. is a rather new dataset with different classes of clothing and is a drop-in replacement for MNIST [54]. It is harder but has the same size, input dimension, and number of classes as Mnist.

VI.I.III CIFAR-10. The CIFAR-10 dataset contains 60,000 32x32 color images in 10 different classes. [31]. The 10 different classes represent airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 6,000 images of each class.

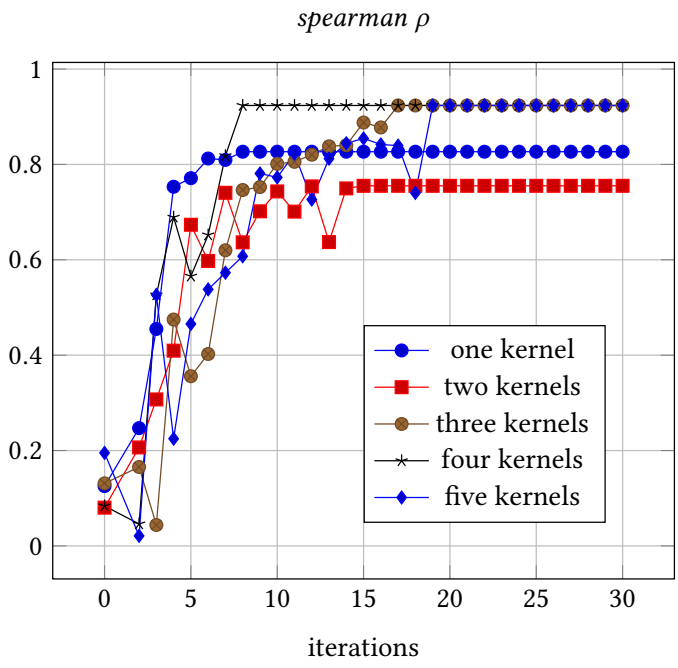


Fig. IX. Comparison between the number of ranker kernels used in PN; one kernel (15X15) , two kernels (15X15 and 20X20), three kernels (15X15, 20X20 and 25X25),four kernels (10X10, 15X15, 20X20 and 25X25), five kernels (5X5, 10X10, 15X15, 20X20 and 25X25) for training 10 images of Mnist dataset where 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 [2].

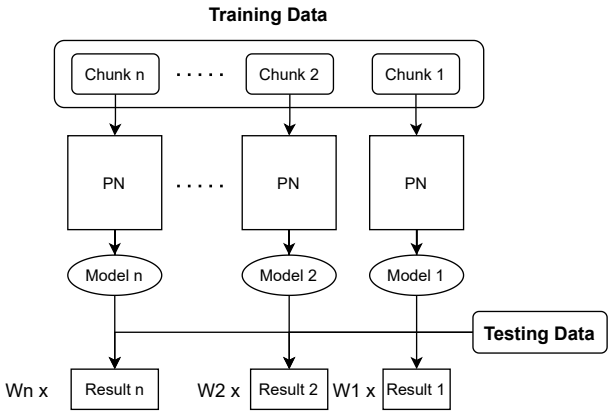
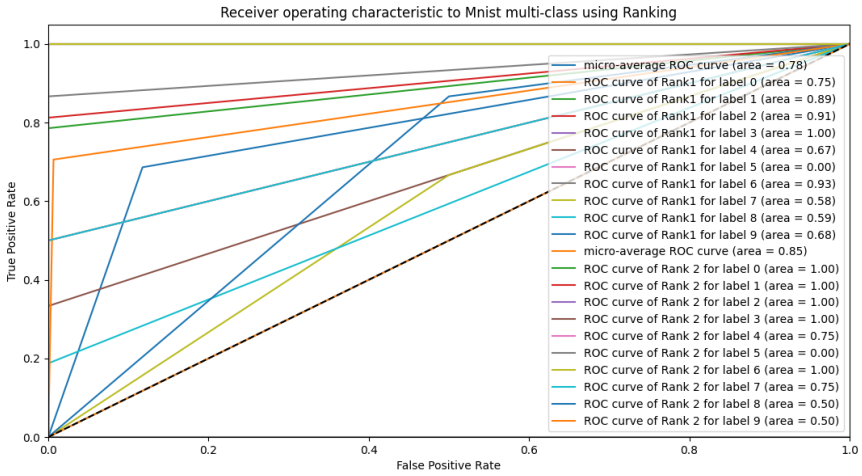
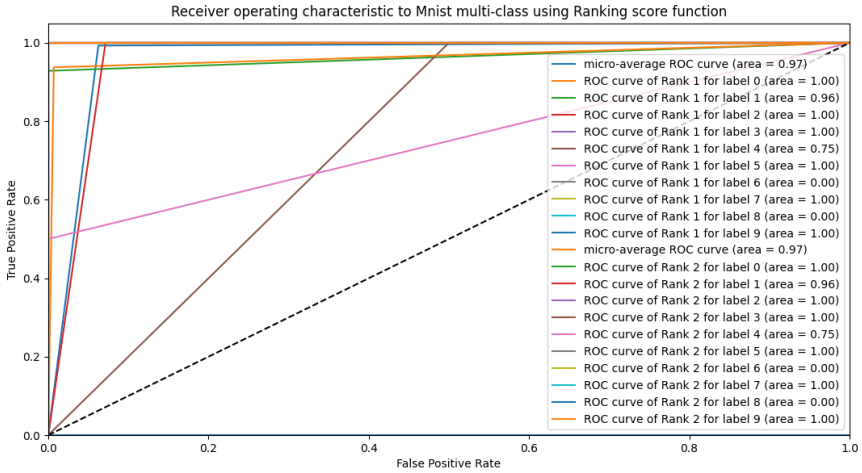


Fig. X. weighted ensemble average models flow of the image classification using *PN*s parallel processing for each chunk of training data.



(a)



(b)

Fig. XI. *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 $f_{s1} = notMax(\lambda)$

VI.II Results

VI.II.I 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. X. According

to the validation result, the weight is determined, then testing data 20,000 are executed on fifty models, the average weight is taken of the 50 models to determine the final result.

VI.II.II Output Ranking Results. In terms of ranking, the output of *PN* almost 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, which exceeds the state-of-the-art in Mnist, Mnist-fashion, and CIFAR-10 as shown in table III.

VI.II.III Image Classification Results. *PN* has 3 kernel sizes is tested on the *CFAR-10* [30], Fashion-MNIST data set [53] and Mnist as shown in Table III. Table IV shows the results compared to other convolutions networks.

Table III. *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

Table IV. Comparison of classification on CIFAR-10 [30] and Fashion-Mnist data set [53] Data sets using different convolution models

DS	Model	Baseline
MNIST	CapsNet [39]	97.22
	MCDNN [9]	99.26
	SpinalNet [27]	98.73
	PrefNet	97.4
Fashion-MNIST	DARTS [46]	0.96
	SAM [15]	0.951
	Wide-ResNet [56]	0.955
	PrefNet	0.9316
CIFAR-10	ResNet [22]	92.22
	WRN [57]	96.26
	Dense [24]	93.73
	PrefNet	90.83

VI.III Computational Platform

PNN and *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.

VI.IV Discussion and Future Work

This paper confirms that it hardly detects a complete ranking set of data where $\rho = 1$. However, using initial ranking to solve classification is a promising step in terms of computational cost and reach 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 IV that *PN* is performing better than *CapsNet* [39] 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.

VII CONCLUSION

This paper 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 [51].

VIII ACKNOWLEDGMENT

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.

REFERENCES

[1] Salisu Mamman Abdulrahman, Pavel Brazdil, Wan Mohd Nazmee Wan Zainon, and Alhassan Adamu. 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* (Penang, Malaysia) *(ICSCA '19)*. Association for Computing Machinery, New York, NY, USA, 140–148. https://doi.org/10.1145/3316615.3316674

[2] Nir Ailon and Mehryar Mohri. 2007. An efficient reduction of ranking to classification. *arXiv* (2007).

[3] F. Aiolli. 2005. A preference model for structured supervised learning tasks. (2005), 557–560.

[4] I. Aizenberg, N. Aizenberg, and Joos P. Vandewalle. 2000. *Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications.* Kluwer Academic Publishers, Norwell, MA, USA.

[5] A. Alphonse, K. Shankar, M. J. J. Rakkini, S. Ananthakrishnan, S. Athisayamani, A. R. Singh, and R. Gobi. 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 (2021), 3447–3463.

[6] Maria-Florina Balcan, Nikhil Bansal, Alina Beygelzimer, Don Coppersmith, John Langford, and Gregory B. Sorkin. 2008. Robust Reductions from Ranking to Classification. 72, 1–2 (Aug. 2008), 139–153. https://doi.org/10.1007/s10994-008-5058-6

[7] W. Cheng and E. Hüllermeier. 2008. Instance-based label ranking using the mallows model. (2008), 143–157.

[8] Tale Shaked Chris Burges. 2005. Learning to rank using gradient descent. (2005), 58–86.

[9] Dan C. Ciresan, Ueli Meier, and Jürgen Schmidhuber. 2012. Multi-column deep neural networks for image classification. *2012 IEEE Conference on Computer Vision and Pattern Recognition* (2012), 3642–3649.

[10] K. Crammer and Y. Singer. 2002. Pranking with ranking. (2002), 641–647.

[11] J. Furnkranz E. Hüllermeier. 2008. Label ranking by learning pairwise preferences. (2008), 1897–1916.

[12] Ayman Elgharabawy, Mukesh Parsad, and Chin-Teng Lin. 2020. Preference neural network. (2020). https://doi.org/10.20944/preprints201904.0091.v3

[13] Ayman Elgharabawy, Mukesh Prasad, and Chin-Teng Lin. 2021. Subgroup Preference Neural Network. *Sensors* 21, 18 (2021). https://doi.org/10.3390/s21186104

[14] Muhammad Khalid Farooq and Abdul Hafeez. 2020. COVID-ResNet: A Deep Learning Framework for Screening of COVID19 from Radiographs. *ArXiv* abs/2003.14395 (2020).

[15] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. 2021. Sharpness-Aware Minimization for Efficiently Improving Generalization. *ArXiv* abs/2010.01412 (2021).

[16] Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. 2003. An Efficient Boosting Algorithm for Combining Preferences. *J. Mach. Learn. Res.* 4, null (Dec. 2003), 933–969.

[17] J. Furnkranz and E. Hüllermeier. 2003. Pairwise Preference Learning and Ranking in Machine Learning. (2003), 145–156.

[18] J. Fürnkranz and E. Hüllermeier. 2010. Preference Learning. (2010).

[19] J. Furnkranz and E. Hüllermeier. 2011. Decision Tree Modeling for Ranking Data. (2011), 83–106.

[20] Sariel Har-Peled, D. Roth, and Dav Zimak. 2002. Constraint Classification: A New Approach to Multiclass Classification. (2002).

[21] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 770–778.

[22] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778. https://doi.org/10.1109/CVPR.2016.90

[23] Geoffrey E. Hinton. 2009. Deep belief networks. *Scholarpedia* 4 (2009), 5947.

[24] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. 2017. Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2261–2269. https://doi.org/10.1109/CVPR.2017.243

[25] Zhong Ji, Biying Cui, Huihui Li, Yu-Gang Jiang, Tao Xiang, Timothy Hospedales, and Yanwei Fu. 2020. Deep Ranking for Image Zero-Shot Multi-Label Classification. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society* (May 14 2020). 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 Date created - 2020-05-15; Date revised - 2020-12-20; Last updated - 2021-01-29.

[26] Y. Jian, J. Xiao, Y. Cao, A. Khan, and J. Zhu. 2019. Deep Pairwise Ranking with Multi-label Information for Cross-Modal Retrieval. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*. 1810–1815. https://doi.org/10.1109/ICME.2019.00311

[27] H. M. D. Kabir, Moloud Abdar, Seyed Mohammad Jafar Jalali, Abbas Khosravi, Amir F. Atiya, Saeid Nahavandi, and Dipti Srinivasan. 2020. SpinalNet: Deep Neural Network with Gradual Input. *ArXiv* abs/2007.03347 (2020).

[28] Maurice Kendall. 1948. Rank correlation methods. (1948).

[29] Wojciech Kotlowski, Krzysztof Dembczynski, and Eyke Hüllermeier. 2011. Bipartite Ranking through Minimization of Univariate Loss. In *ICML*.

[30] Alex Krizhevsky. 2009. *Learning multiple layers of features from tiny images.* Technical Report.

[31] Alex Krizhevsky. 2009. *Learning multiple layers of features from tiny images.* Technical Report.

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.), Vol. 25. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[33] Jayendra Kumar, K. Raja, R. S. Pooja, and V. S. Charan. 2020. GENERATION OF LYRICS USING Recurrent Neural Network (RNN). *International Journal of Research* 7 (2020), 654–659.

[34] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324. https://doi.org/10.1109/5.726791

[35] Yann LeCun and Corinna Cortes. 2010. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/. (2010). http://yann.lecun.com/exdb/mnist/

[36] Yann LeCun, Corinna Cortes, and CJ Burges. 2010. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* 2 (2010).

[37] Jiayong Li, W. Y. Ng Wing, Tian Xing, Sam Kwong, and Hui Wang. 2020. Weighted multi-deep ranking supervised hashing for efficient image retrieval. *International Journal of Machine Learning and Cybernetics* 11, 4 (2020), 883–897. 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 Copyright - 2019© Springer-Verlag GmbH Germany, part of Springer Nature 2019; Last updated - 2020-03-08.

[38] Arpana Mahajan and Sanjay Chaudhary. 2019. Categorical Image Classification Based On Representational Deep Network (RESNET). *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)* (2019), 327–330.

[39] Vittorio Mazzia, Francesco Salvetti, and Marcello Chiaberge. 2021. Efficient-CapsNet: capsule network with self-attention routing. *Scientific Reports* 11 (2021).

[40] Grbovic Mihajlo, Djuric Nemanja, and Vucetic Slobodan. 2014. Learning from Pairwise Preference Data using Gaussian Mixture Model. (2014).

[41] Claudio Moraga and Ralph Heider. 1999. "New lamps for old!" (Generalized Multiple-valued Neurons). In *Proceedings 1999 29th IEEE International Symposium on Multiple-Valued Logic (Cat. No. 99CB36329)*. IEEE, 36–41.

[42] Geraldina Ribeiro, Wouter Duivesteijn, Carlos Soares, and Arno Knobbe. 2012. Multilayer Perceptron for Label Ranking. In *Proceedings of the 22nd International Conference on Artificial Neural Networks and Machine Learning - Volume Part II (ICANN'12)*. Springer, 25–32.

[43] R. Salakhutdinov and Geoffrey E. Hinton. 2009. Deep Boltzmann Machines. In *AISTATS*.

[44] M. Sornam, Kavitha Muthusubash, and V. Vanitha. 2017. A Survey on Image Classification and Activity Recognition using Deep Convolutional Neural Network Architecture. *2017 Ninth International Conference on Advanced Computing (ICoAC)* (2017), 121–126.

[45] Charles Spearman. 1961. The proof and measurement of association between two things. (1961).

[46] Muhammad Tanveer, Muhammad Umar Karim Khan, and C. M. Kyung. 2021. Fine-Tuning DARTS for Image Classification. *2020 25th International Conference on Pattern Recognition (ICPR)* (2021), 4789–4796.

[47] Arash Vahdat and J. Kautz. 2020. NVAE: A Deep Hierarchical Variational Autoencoder. *ArXiv* abs/2007.03898 (2020).

[48] Vamsi Krishna Vedantam. 2021. The Survey: Advances in Natural Language Processing using Deep Learning*.

[49] Fei Wang, Mengqing Jiang, Chen Qian, S. Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, and Xiaoou Tang. 2017. Residual Attention Network for Image Classification. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), 6450–6458.

[50] Le Wang, Ze Luo, Jing-Rebecca Li, and Can Chen. 2020. Target-Oriented Transformation Networks for Document Retrieval. *Proceedings of the 2020 9th International Conference on Computing and Pattern Recognition* (2020).

[51] Mathematica Wolfram Research, Inc. [n. d.]. Wolfram. https://www.wolfram.com/mathematica/

[52] Qiang Wu, Christopher J. Burges, Krysta M. Svore, and Jianfeng Gao. 2010. Adapting Boosting for Information Retrieval Measures. 13, 3 (June 2010), 254–270. https://doi.org/10.1007/s10791-009-9112-1

[53] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. http://arxiv.org/abs/1708.07747 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/.

[54] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *CoRR* abs/1708.07747 (2017). arXiv:1708.07747 http://arxiv.org/abs/1708.07747

[55] Yan Yan, Ying Wang, Wenchao Gao, Bo-Wen Zhang, Chun Yang, and Xu-Cheng Yin. 2017. LSTM 2: Multi-Label Ranking for Document Classification. *Neural Processing Letters* 47 (2017), 117–138.

[56] Sergey Zagoruyko and Nikos Komodakis. 2016. Wide Residual Networks. *ArXiv* abs/1605.07146 (2016).

[57] Sergey Zagoruyko and Nikos Komodakis. 2016. Wide Residual Networks. *ArXiv* abs/1605.07146 (2016).

[58] Z. Zhao, Weihai Chen, Xingming Wu, Peter C. Y. Chen, and Jingmeng Liu. 2017. LSTM network: a deep learning approach for short-term traffic forecast. *Iet Intelligent Transport Systems* 11 (2017), 68–75.

[59] Y. Zhong and Jian'an Fang. 2020. Text Detection in Multi-feature Fusion Natural Scenes Based on Convolution Deep Belief Network.

[60] Yangming Zhou, Yangguang Liu, Jiangang Yang, Xiaoqi He, and Liangliang Liu. 2014. A Taxonomy of Label Ranking Algorithms. *JCP* 9 (2014), 557–565.