GaussianMLR: Learning Implicit Class Significance via Calibrated Multi-Label Ranking

V. Bugra Yesilkaynak[*][†]
Technical University of Munich
bugra.yesilkaynak@tum.de

Emine Dari[*]
Istanbul Technical University
dari18@itu.edu.tr

Alican Mertan
University of Vermont
alican.mertan@uvm.edu

Gozde Unal
Istanbul Technical University
gozde.unal@itu.edu.tr

Abstract

Existing multi-label frameworks only exploit the information deduced from the bipartition of the labels into a positive and negative set. Therefore, they do not benefit from the ranking order between positive labels, which is the concept we introduce in this paper. We propose a novel multi-label ranking method: GaussianMLR, which aims to learn implicit class significance values that determine the positive label ranks instead of treating them as of equal importance, by following an approach that unifies ranking and classification tasks associated with multi-label ranking. Due to the scarcity of public datasets, we introduce eight synthetic datasets generated under varying importance factors to provide an enriched and controllable experimental environment for this study. On both real-world and synthetic datasets, we carry out extensive comparisons with relevant baselines and evaluate the performance on both of the two sub-tasks. We show that our method is able to accurately learn a representation of the incorporated positive rank order, which is not only consistent with the ground truth but also proportional to the underlying information. We strengthen our claims empirically by conducting comprehensive experimental studies. Code is available at *https://github.com/MrGranddy/GaussianMLR*.

1 Introduction

Multi-label ranking (MLR) [1, 2, 3] is a supervised learning problem where the objective is to not only classify the labels by their relevancy to the instance, but also to predict a ranking that represents the instance's preferences over chosen relevant labels. This can be considered as a generalization of the two sub-problems: multi-label classification and label ranking. To briefly define, given a set of labels Y, multi-label classification bipartites the set into two and associates the instances with the relevant label set $\mathcal{Y} \subseteq Y$. On the other hand, label ranking aims to map instances to a total order over Y. With the objectives of these sub-problems combined, multi-label ranking can be applied to any scenario where the expected output is a ranked subset of all possible labels.

Although the use cases of multi-label ranking have been defined in the literature, most of the public multi-label datasets has ranking information of labels as being a positive or a negative label only. Ranked datasets specific to MLR, where the label order represents the preference of relevant labels to the instance according to the chosen criteria, are very rare. Therefore, not many studies deal with evaluating the performance of a proposed approach in terms of the ranking order between

---

[*]These authors contributed equally to this work
[†]Contact author

the predicted positive classes. Rather, multi-label ranking has been used as an approach to solve multi-label classification problems, applied to object recognition [4], image classification [5], or used in online algorithms [6], where the ranking of predictions are not taken into consideration as a measure. Furthermore, existing algorithms agree with the restraint that, only the partial information that is deduced from the bipartition of the labels is available.

Given the limitations in the field, the motivation for our work is to study multi-label ranking with a novel approach that exploits the order between positive labels, instead of assuming the positive and negative labels of equal importance in their own set. In the following subsection, we highlight our contributions and state the focus of our study in detail.

1.1 Our Contributions

- In the field of MLR, we establish the paradigm of exploiting the ranking information between positive labels of an instance, where the main objective is to extract significance values of labels that determine their rank, which may not be feasible to obtain numerically during the labeling process.

- We model the MLR problem as a distribution learning problem based on a probabilistic approach that unifies the bi-partition and ranking of the labels in the same space. This way, we incorporate pairs consisting of positive labels into the optimization in addition to positive and negative label pairs, to reveal a preference relation over a varying number of positive labels.

- We introduce ranked image datasets generated under different setups with varying importance factors that determine the ranks, which create a controllable environment to test new approaches while facilitating unambiguous interpretations of their performances.

- We compare our novel framework with different methods related to our approach, and empirically explore and interpret the outcomes, thus setting up a clear set of baselines for the MLR problem.

2 Related Work

In the literature, multi-label ranking has been applied to solve classification problems [4, 5, 7, 8], to learn from incompletely or inconsistently labeled data [3, 9], and has been studied from perspectives of consistency and generalization by [10]. In this section, we review the related works on learning problems associated with multi-label ranking. Due to the misuse of terminology in previous works, we clearly state the differences between concepts and examine the ones in the scope of this paper.

2.1 Label Ranking

Preference learning has been long studied in various contexts, and this paper is concerned with learning label preferences where an instance is associated with a finite set of ordered labels, namely label ranking problem [11, 12]. Label ranking should not be confused with object ranking [13], where the aim is to predict the ranking over a predefined class of objects related to the given user query, such as sorting the responses of search engines according to their relevance to the query [14].

Proposed methods for label ranking include pointwise [15, 16, 17], pairwise [11, 1, 5, 18] and listwise [19, 20] methods. In this paper, we are concerned with pairwise methods. A pairwise method that transforms the problem into a binary classification problem was first introduced in [21], as the Constraint Classification framework (CC). The idea behind CC is to build constraints according to the preference relation between labels, where the relation $\lambda_i > \lambda_j$, denoting that label λ_i precedes λ_j in relevance to the instance, constructs the two constraints $f_i(x) - f_j(x) > 0$ being the positive constraint and $f_j(x) - f_i(x) < 0$ being the negative constraint. Then, both constraints are used as training samples of the single classifier to find the suitable weight vector satisfying the constraints. More efficiently, Ranking by Pairwise Comparison (RPC) [11, 22] transforms the problem into training a binary classifier for each label pair, producing a number of $K(K-1)/2$ models which is half of the number of constraints of CC, where K is the number of classes. The final ranking is determined after obtaining each model's decision for the given pair and calculating the sum of weighted votes. More recently, the Log-Sum-Exp-Pairwise (LSEP) loss function introduced in [5] improves the previous approaches using hinge loss [23, 24] by learning the pairwise comparisons in

a smooth and easier way to optimize. The proposed loss function follows the method of pairing one positive and one negative label from the sets constructed according to the ground truth and wraps BP-MLL (Backpropogation for Multi-Label Learning) [25] in a logarithmic function with a bias term. The objective of LSEP is to enforce the positive labels to be in higher ranks and negative labels to be in lower ranks by optimizing the loss function while training convolutional neural networks [26]. However, LSEP involves an obligatory ordering of the whole set of labels to be followed by a thresholding that determines which labels are to be discarded, while our GaussianMLR introduces a natural label selection and ranking process.

2.2 Label Classification

Classification algorithms can be roughly divided into two categories based on the predicted label count, as multi-class and multi-label classification. Determining the label count of the prediction set is one challenge of multi-label classification, which is not a concern in multi-class classification where instances are associated with a single label only. Although some approaches include setting fixed label counts such as choosing top-k labels or fixed thresholds as a confidence score boundary, forcing a label count by a fixed value is impractical as it ignores the context of the problem at hand. Varying label counts can be realized by setting learnable thresholds or label counts as in [5], or they can be jointly learned in the label ranking step as proposed in the Calibrated Label Ranking method [27], by using virtual labels as split points which are inserted in the label set before the ranking process. Then, labels in higher ranks compared to the virtual label are included in the relevant set. Compared against all previous related work, our GaussianMLR implicitly introduces a zero-point, i.e. an inherent threshold, to perform binary classification of labels into positives and negatives in the same space that we perform ranking, thus combining both tasks in a unified model.

3 Problem and Notation

3.1 Dataset Definiton

We start with a dataset of N examples, $\{(\boldsymbol{x}^{(i)}, R^{(i)})\}_{i=1}^N$ where $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$ is a real-valued sample from the input distribution, and $R^{(i)} = \{(y_j, r_j^{(i)})\}_{j=1}^K$ is the label, where $y_j \in Y$ is a symbolic class representation for an entity that can be semantically present in $\boldsymbol{x}^{(i)}$, $Y = \{y_1, y_2, ..., y_K\}$ is the set of K possible classes, and $r_j^{(i)} \in \mathbb{N}$ is the rank of the associated class y_j for $\boldsymbol{x}^{(i)}$. It is the case that prior work mostly uses standard multi-label classification datasets defined as $\{(\boldsymbol{x}^{(i)}, \mathcal{Y}^{(i)})\}_{i=1}^N$ where $\mathcal{Y}^{(i)} \subseteq Y$ is the set of classes semantically present in $\boldsymbol{x}^{(i)}$ and called the *positive* classes, while the rest of the classes are called *negative* classes. This kind of dataset only provides a ranking information between negative and positive classes where positives have a higher rank than negatives, and can be seen as a special case of the former definition where $R^{(i)} = \{(y_u, 1)|y_u \in \mathcal{Y}^{(i)}\} \cup \{(y_v, 0)|y_v \notin \mathcal{Y}^{(i)}\}$. In our definition, a negative class will always have rank 0. Throughout our work, our findings are under the fair assumptions: **(i)** $\boldsymbol{x}$ are identically and independently distributed (i.i.d.) in all datasets. **(ii)** All label/rank pairs $(y_j, r_j^{(i)})$ are conditionally independent given an input $\boldsymbol{x}^{(i)}$.

3.2 Problem Definition

We construct the *multi-label ranking* problem, assuming that the two sub-problems, the multi-label classification and label ranking, are independent, i.e., $\mathcal{Y}^{(i)}$ and $\mathcal{B}^{(i)}$ are independent given $\boldsymbol{x}^{(i)}$:

$$P_{\mathcal{Y}, \mathcal{B}}(\mathcal{Y}^{(i)}, \mathcal{B}^{(i)}|\boldsymbol{x}^{(i)}; \theta) = P_{\mathcal{Y}}(\mathcal{Y}^{(i)}|\boldsymbol{x}^{(i)}\theta)P_{\mathcal{B}}(\mathcal{B}^{(i)}|\boldsymbol{x}^{(i)}; \phi). \tag{1}$$

Thus, we formally define the following likelihood optimization problem, given an instance $\boldsymbol{x}^{(i)}$:

$$\max_{\theta, \phi} P_{\mathcal{Y}}(\mathcal{Y}^{(i)}|\boldsymbol{x}^{(i)}; \theta)P_{\mathcal{B}}(\mathcal{B}^{(i)}|\boldsymbol{x}^{(i)}; \phi). \tag{2}$$

Here $\mathcal{Y}^{(i)}$ is the set of *positive* classes for an instance $\boldsymbol{x}^{(i)}$, $P_{\mathcal{Y}}$ is the parameterized family of probability mass functions of possible positive class sets, parameterized by θ, and conditioned by $\boldsymbol{x}^{(i)}$. While $\mathcal{B}^{(i)}$ is a *bucket order*[28], which is a class of partial orders allowing *ties*. A partial order is a reflexive, antisymmetric, and transitive binary relation on a set of items, in our case Y. $\mathcal{B}^{(i)}$ intuitively

partitions labels $y \in Y$ into mutually exclusive bucket of ranks $\langle \mathcal{M}_1^{(i)}, ..., \mathcal{M}_{b^{(i)}}^{(i)} \rangle$ where $b^{(i)}$ is the number of buckets. If two items $y_u, y_v \in \mathcal{M}_k^{(i)}$ are the member of the same bucket then we say they are in tie, meaning they can not be distinguished ordinally and they virtually have the same rank. Formally, for a bucket $\mathcal{M}_k^{(i)}$, $y_u, y_v \in \mathcal{M}_k^{(i)} \iff (y_u, y_v) \notin \mathcal{B}^{(i)} \wedge (y_v, y_u) \notin \mathcal{B}^{(i)}$. Here it should be noted that $\mathcal{M}_k^{(i)} \subseteq Y$ is a set and is only introduced to better visualize the bucket orders, $\mathcal{B}^{(i)}$ is just a relation and is enough to define a bucket order. On the other hand, different buckets for an instance $x^{(i)}$ have total ordinal relationship between them, such that: for any two distinct buckets $\mathcal{M}_k^{(i)}, \mathcal{M}_l^{(i)}$, $y_u \in \mathcal{M}_k^{(i)}, y_v \in \mathcal{M}_l^{(i)}, (y_u, y_v) \in \mathcal{B}^{(i)} \iff k > l$, i.e. $r_u^{(i)} \geq r_v^{(i)} \iff k > l$, where $r_u^{(i)}$ and $r_v^{(i)}$ are corresponding ranks of y_u and y_v. $P_{\mathcal{B}}$ is the parameterized family of probability mass functions of such bucket orders parameterized by ϕ, conditioned by $x^{(i)}$.

The optimization problem in hand can be seen as the joint optimization of two distinct problems, namely: multi-label classification and label ranking. Both of the terms can be divided into practicable sub-problems.

We can simplify the first likelihood $P_{\mathbf{y}}(\mathbf{y}|x^{(i)}; \theta)$ by defining the random variable $\mathbf{y} \in \{0, 1\}^K$ via the random vector:

$$\mathbf{y}_c = \begin{cases} 1 & y_c \in \mathcal{Y} \\ 0 & y_c \notin \mathcal{Y} \end{cases}, \qquad c \in \{1, ..., K\}.$$

Here $\mathcal{Y}$ is any random variable distributed by $P_{\mathcal{Y}}(\mathcal{Y}|x^{(i)}; \theta)$, then we can model the sub-problem with K binary classification models using Bernoulli distribution:

$$\max_{\theta} P_{\mathcal{Y}}(\mathcal{Y}^{(i)}|x^{(i)}; \theta) = \max_{\theta} \prod_{c=1}^{K} P(\mathbf{y}_c = 1|x^{(i)}; \theta)^{\mathbb{I}[y_c \in \mathcal{Y}^{(i)}]} P(\mathbf{y}_c = 0|x^{(i)}; \theta)^{\mathbb{I}[y_c \notin \mathcal{Y}^{(i)}]}, \quad (3)$$

where $\mathbb{I}[.]$ is the indicator function.

Likelihood of a Bucket Order. We can use a random vector $\mathbf{r} \sim P_{\mathbf{r}}(\mathbf{r}|\phi)$ where $\mathbf{r} \in \mathbb{R}^K$ and $|Y| = K$ to model the likelihood of a bucket order, where each element of $\mathbf{r}_i$ corresponds to the significance value of class y_i. Let $\mathbf{r}$ define a weighted directed complete graph $G = (V, E, \omega)$ such that, $V = Y$, $E = Y \times Y$ and $\omega : E \to \mathbb{R}$. We define the edge weights as $\omega(y_u, y_v) = P(\mathbf{r}_u \geq \mathbf{r}_v)$. Let $\mathcal{P}$ be a total partial order defined by $\succ$ such that $\forall y_u, y_v \in Y, (y_u, y_v) \in \mathcal{P}$ or $(y_v, y_u) \in \mathcal{P}$, which describes a unique permutation of the elements. Then we can model the likelihood of $\mathcal{P}$ with independent Bernoulli distributions on the edges, such that for any two nodes $y_u, y_v \in Y$ either $(y_u, y_v) \in \mathcal{P}$ or $(y_v, y_u) \in \mathcal{P}$, let S be the set of unique pairs on Y, such that, if $(u, v) \in S$ then $(v, u) \notin S : P(\mathcal{P}) = \prod_{(y_u, y_v) \in S} P(\mathbf{r}_u \geq \mathbf{r}_v)^{\mathbb{I}[(y_u, y_v) \in \mathcal{P}]} P(\mathbf{r}_u < \mathbf{r}_v)^{\mathbb{I}[(y_u, y_v) \notin \mathcal{P}]}$. $\mathcal{P}$ is defined by $\succ$ relations so we can simplify: $P(\mathcal{P}) = \prod_{(y_u, y_v) \in \mathcal{P}} P(\mathbf{r}_u \geq \mathbf{r}_v)$. A bucket order $\mathcal{B}$ agrees with a unique set of total partial orders $S_{\mathcal{P}}$, formally $\forall \mathcal{P} \in S_{\mathcal{P}}, \forall (y_u, y_v) \in \mathcal{B}, (y_u, y_v) \in \mathcal{P}$. Then we can express $P(\mathcal{B}) = \sum_{\mathcal{P} \in S_{\mathcal{P}}} \prod_{(y_u, y_v) \in \mathcal{P}} P(\mathbf{r}_u \geq \mathbf{r}_v)$. For any pair of tied elements $y_u, y_v \in Y, (y_u, y_v) \notin \mathcal{B} \wedge (y_v, y_u) \notin \mathcal{B}$, there will be $(y_u, y_v) \in \mathcal{P}_1$ and $(y_v, y_u) \in \mathcal{P}_2$ and the rest of the elements are the same, where $\mathcal{P}_1, \mathcal{P}_2 \in S_{\mathcal{P}}$. $P(\mathbf{r}_u \geq \mathbf{r}_v) + P(\mathbf{r}_u < \mathbf{r}_v) = 1$, grouping and simplifying the summed terms in $P(\mathcal{B})$ formula, we have: $P(\mathcal{B}) = \prod_{(y_u, y_v) \in \mathcal{B}} P(\mathbf{r}_u \geq \mathbf{r}_v)$. A visual explanation of the likelihood of a bucket order can be seen in Figure 1.

We parameterize $P_{\mathbf{r}}(\mathbf{r}|x^{(i)}; \phi)$ by ϕ, and re-write $P(\mathcal{B})$ in a likelihood maximization:

$$\max_{\phi} P_{\mathcal{B}}(\mathcal{B}^{(i)}|x^{(i)}; \phi) = \max_{\phi} \prod_{(y_u, y_v) \in \mathcal{B}^{(i)}} P(\mathbf{r}_u \geq \mathbf{r}_v|x^{(i)}; \phi). \tag{4}$$

As a result, the multi-label ranking problem as defined in Equation (2) can be re-written as:

$$\max_{\theta, \phi} \prod_{c=1}^{K} P(\mathbf{y}_c = 1|x^{(i)}; \theta)^{\mathbb{I}[y_c \in \mathcal{Y}^{(i)}]} P(\mathbf{y}_c = 0|x^{(i)}; \theta)^{\mathbb{I}[y_c \notin \mathcal{Y}^{(i)}]} \prod_{(y_u, y_v) \in \mathcal{B}^{(i)}} P(\mathbf{r}_u \geq \mathbf{r}_v|x^{(i)}; \phi). \tag{5}$$

The refined optimization problem in Equation (5) sets up the basis for our proposed unified multi-label ranking method. It establishes a probabilistic foundation for the multi-label ranking problem, which we further develop with a Gaussian probability model next.
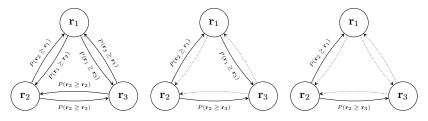
4

Figure 1: Visual explanation of how a real valued random variable, a total partial order and a bucket order is related. Random vector $\mathbf{r}$ defines a complete graph as depicted in (a), a total partial order contains magnitude relations for all pairs of the set that it is defined on. (b) shows how a total partial order can be applied on the random vector $\mathbf{r}$ to calculate its likelihood: $P(\mathbf{r}_2 \geq \mathbf{r}_1)P(\mathbf{r}_1 \geq \mathbf{r}_2)P(\mathbf{r}_2 \geq \mathbf{r}_3)$. Lastly a bucket agrees with a set of total partial orders, since the bucket order depicted in (c) does not specify an ordering between $\mathbf{r}_1$ and $\mathbf{r}_3$, both $\langle \mathbf{r}_2, \mathbf{r}_1, \mathbf{r}_3 \rangle$ and $\langle \mathbf{r}_2, \mathbf{r}_3, \mathbf{r}_1 \rangle$ is valid for (c). Then it is trivial how the elimination process will follow, yielding the likelihood: $P(\mathbf{r}_2 \geq \mathbf{r}_1)P(\mathbf{r}_2 \geq \mathbf{r}_3)$ for the bucket order shown in (c).

4 GaussianMLR

4.1 Core Idea

Multi-label ranking problem can be viewed as both classifying and ranking the classes in a given instance, meaning that we are not only interested in assigning correct classes to an instance, but we also want to measure how relevant these classes are for the given instance.

Significance Values. We start by assuming all $y_j \in \mathcal{Y}^{(i)}$ for a given input $\boldsymbol{x}^{(i)}$ has an underlying significance value $s_j^{(i)} \in \mathbb{R}$, we model these significance values using Gaussian distributions, such that: $s_j^{(i)} \sim \mathcal{N}(\mu_j^{(i)}, \sigma_j^{2(i)})$ where $\mathcal{N}(\cdot, \cdot)$ is a Gaussian distribution, $\mu_j^{(i)}$ is the mean and $\sigma_j^{2(i)}$ is the variance, for a given $\boldsymbol{x}^{(i)}$.

4.2 Methodology

Our goal is to obtain a function f such that it produces significance values matching with the ranking information in the ground truth. Let $f : \mathbb{R}^d \rightarrow \mathbb{R}^{2K}$ be a trainable function parameterized by ζ, where the output of the function is the predicted Gaussian distribution parameters $\hat{\mu}^{(i)} \in \mathbb{R}^K$ and $\hat{\sigma}^{2(i)} \in \mathbb{R}^K$ for the input $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$ such that $f(\boldsymbol{x}^{(i)}; \zeta) = [\hat{\mu}^{(i)} \ \hat{\sigma}^{2(i)}]$ and the predicted significance values $\hat{s}_j^{(i)} \sim \mathcal{N}(\hat{\mu}_j^{(i)}, \hat{\sigma}^2{}_j^{(i)})$.

We adapt the optimization problem in Equation (5) as follows: instead of using a separate binary variable for the classification task, we model the significance values such that for a class y_c, $\hat{s}_c^{(i)} \geq 0$ indicates a predicted positive, and $\hat{s}_c^{(i)} < 0$ indicates a predicted negative. The reformulated optimization problem reads:

$$\max_{\zeta} \prod_{c=1}^{K} P(\hat{s}_c^{(i)} \geq 0)^{\mathbb{I}[y_c \in \mathcal{Y}^{(i)}]} P(\hat{s}_c^{(i)} < 0)^{\mathbb{I}[y_c \notin \mathcal{Y}^{(i)}]} \prod_{(y_u, y_v) \in \mathcal{B}^{(i)}} P(\hat{s}_u^{(i)} \geq \hat{s}_v^{(i)}). \quad (6)$$

Letting $\hat{d}_{(u,v)}^{(i)} = \hat{s}_u^{(i)} - \hat{s}_v^{(i)}$, and using the above-mentioned definition: $\hat{d}_{(u,v)}^{(i)} \sim \mathcal{N}(\hat{\mu}_u^{(i)} - \hat{\mu}_v^{(i)}, \hat{\sigma}_u^{2(i)} + \hat{\sigma}_v^{2(i)})$, we can re-write Equation 6 as follows:

$$\max_{\zeta} \prod_{c=1}^{K} P(\hat{s}_c^{(i)} \geq 0)^{\mathbb{I}[y_c \in \mathcal{Y}^{(i)}]} P(\hat{s}_c^{(i)} < 0)^{\mathbb{I}[y_c \notin \mathcal{Y}^{(i)}]} \prod_{(y_u, y_v) \in \mathcal{B}^{(i)}} P(\hat{d}_{(u,v)}^{(i)} \geq 0). \quad (7)$$

For a Gaussian random variable $z \sim \mathcal{N}(\mu, \sigma^2)$, $P(z > 0) = (1/2)[1 - \mathrm{erf}(-\mu/(\sigma\sqrt{2}))]$ where $\mathrm{erf}(\cdot)$ is the Gaussian error function. Let $Q(\mu, \sigma) = (1/2)[1 - \mathrm{erf}(-\mu/(\sigma\sqrt{2}))]$, then we can

re-write Equation 7 as:

$$\max_{\zeta} \prod_{c=1}^{K} Q(\hat{\mu}_c^{(i)}, \hat{\sigma}_c^{(i)})^{\beta_c^{(i)}} (1 - Q(\hat{\mu}_c^{(i)}, \hat{\sigma}_c^{(i)}))^{(1-\beta_c^{(i)})} \prod_{(y_u, y_v) \in \mathcal{B}^{(i)}} Q(\hat{\mu}_{(u,v)}^{(i)}, \hat{\sigma}_{(u,v)}^{(i)}). \tag{8}$$

Here $\beta_c^{(i)} = \mathbb{I}[y_c \in \mathcal{Y}^{(i)}]$, $\hat{\mu}_{(u,v)}^{(i)} = \hat{\mu}_u^{(i)} - \hat{\mu}_v^{(i)}$ and $\hat{\sigma}_{(u,v)}^{(i)} = \sqrt{\hat{\sigma}_u^{2(i)} - \hat{\sigma}_v^{2(i)}}$. Applying negative log likelihood, we define our loss function in two parts as follows:

$L_c(\hat{\mu}^{(i)}, \hat{\sigma}^{(i)}, \mathcal{Y}^{(i)}) = \sum_{c=1}^{K} -\beta_c^{(i)} \log\left(Q(\hat{\mu}_c^{(i)}, \hat{\sigma}_c^{(i)})\right) - (1 - \beta_c^{(i)}) \log\left(1 - Q(\hat{\mu}_c^{(i)}, \hat{\sigma}_c^{(i)})\right),$

$L_r(\hat{\mu}^{(i)}, \hat{\sigma}^{(i)}, \mathcal{B}^{(i)}) = \sum_{(y_u, y_v) \in \mathcal{B}^{(i)}} -\log\left(Q(\hat{\mu}_{(u,v)}^{(i)}, \hat{\sigma}_{(u,v)}^{(i)})\right).$

Summing up, the objective function of the GaussianMLR is given by:

$$\min_{\zeta} \frac{1}{N} \sum_{i=1}^{N} L_c(\hat{\mu}^{(i)}, \hat{\sigma}^{(i)}, \mathcal{Y}^{(i)}) + L_r(\hat{\mu}^{(i)}, \hat{\sigma}^{(i)}, \mathcal{B}^{(i)}), \tag{9}$$

Training and inference. GaussianMLR provides a differentiable loss function (9), thus in the context of our work we choose to use our objective to train neural networks using stochastic gradient descent. After training a network with any dataset $\mathcal{D}$, at inference we use $\hat{\mu}^{(i)}$ as the predicted ranking score for a given $\boldsymbol{x}^{(i)}$. Further details on the implementation can be found in Appendix B.

Learning Implicit Class Significance. For a large enough dataset, we claim that our predictions will be proportional to the real underlying significance values. This claim is theoretically supported in Appendix D and we further support our claim with empirical studies in Section 5.

5 Experiments

5.1 Datasets

We conduct our experiments on three distinct datasets: natural scene images database[3], architectural VDP dataset[29] and Ranked MNIST datasets, which we introduce Section 5.2. These datasets have the common trait that they not only bipartite the labels into negatives and positives, but also provide how relevant each of these positive classes are to the instances. All datasets we use follow the notation provided in Section 3.1, further details can be found in Appendix A. In the paper, we only provide Ranked MNIST Gray experiments, for the Ranked MNIST Color, the experiments can be found in Appendix F.

5.2 Ranked MNIST

Ranked MNIST is a family of datasets with two main branches named as Ranked MNIST Gray and Ranked MNIST Color, where the first is in grayscale and the latter has varying random hue and saturation values for each digit. These datasets are generated by placing unique digits from the MNIST dataset [30] on a 224x224 canvas, where the number of digits in a single image vary from 1 up to 10. For each branch, we have two different importance factors that change: scale and brightness. According to these factors, we rank each positive digit such that for scale: the larger digits have greater ranks and for brightness: the brighter digits have greater ranks. For both the Ranked MNIST Gray-S/B (scale/brightness) and Color-S/B datasets we have four different setups: changing scale, changing brightness, changing both and training on scales, changing both and training on brightness. Further explanation and examples can be seen in Appendix E.

5.3 Baselines

To evaluate our GaussianMLR (GMLR) method, we selected two pairwise baseline methods, namely: CRPC[3], which is the calibrated[31] version of RPC, and LSEP[5]. To our knowledge, GMLR is the first multi-label ranking method which utilizes the positive class ranks, thus we aim to provide fairness in competition of the baseline algorithms with our method. To that end, we introduce CRPC-Strong

and LSEP-Strong, where we develop the existing baselines into *Strong* versions that can process the positive class ranks by adding all (y_u, y_v) pairs, where y_u and y_v are positive classes and $y_u \succ y_v$. Similarly, we call the methods which do not use the positive class relations as *Weak* versions.

Table 1: Quantitative results on Ranked MNIST Gray datasets. Ranked MNIST S and B stands for changing the scale or brightness of the digits, while (Mix) means both of the features are changing, but the ground truth indicates only one of the features. Bold-marked results show the best scores in Strong(S) baselines, and underlined scores show the best scores in Weak(W) baselines.

Method	Ranked MNIST Gray-S						Ranked MNIST Gray-B						Ranked MNIST Gray-S (Mix)						Ranked MNIST Gray-B (Mix)					
	$\tau_b\uparrow$	$S\rho\uparrow$	$\gamma\uparrow$	HL↓	M-1↓	F1↑	$\tau_b\uparrow$	$S\rho\uparrow$	$\gamma\uparrow$	HL↓	M-1↓	F1↑	$\tau_b\uparrow$	$S\rho\uparrow$	$\gamma\uparrow$	HL↓	M-1↓	F1↑	$\tau_b\uparrow$	$S\rho\uparrow$	$\gamma\uparrow$	HL↓	M-1↓	F1↑
CRPC(W)	49.26	59.92	59.80	17.13	0.20	86.45	51.36	61.45	59.27	12.85	0.44	89.44	51.73	61.71	58.83	13.29	0.43	89.12	52.35	62.50	59.29	12.89	0.47	89.42
LSEP(W)	61.38	70.67	61.49	0.49	0.10	99.56	59.45	68.67	59.77	0.96	0.14	99.12	60.45	69.47	60.68	0.97	0.14	99.11	60.04	69.20	60.31	0.99	0.11	99.10
GMLR(W)	62.52	71.86	62.62	0.51	0.10	99.54	60.18	69.36	60.54	0.97	0.10	99.12	60.56	69.74	60.80	0.92	0.15	99.16	60.00	69.23	60.24	0.93	0.13	99.15
CRPC(S)	64.09	75.56	75.20	18.69	0.18	85.37	61.71	73.62	74.70	24.15	0.37	81.73	62.58	74.03	73.80	18.89	0.32	85.17	63.32	74.85	74.44	18.58	0.34	85.37
LSEP(S)	93.99	97.35	94.50	1.38	0.23	98.75	93.62	97.01	94.46	1.95	0.24	98.21	91.71	95.89	92.54	2.15	0.27	98.04	93.03	96.81	93.64	1.80	0.23	98.36
GMLR(S)	94.23	97.41	94.43	0.58	0.20	99.47	93.38	96.65	94.49	2.04	0.33	98.15	90.99	95.05	91.75	1.45	0.79	98.67	92.94	96.46	93.73	1.52	0.44	98.62

5.4 Quantitative Results

We provide quantitative results on both the Ranked MNIST and real datasets using two set of metrics: ranking and classification. For ranking we use Kendall's Tau-b (τ_b), Spearman's Rho (S_ρ) and Goodman and Kruskal's Gamma (γ), for classification we use Hamming Loss (HL), Max-1 (M-1) loss and F1 score. Max-1 loss yields the percentage of instances such that

Table 2: Quantitative results on Natural Scene Images Database (NSID) and Architectural VDP Dataset (AVDP). The annotations of scores are the same with Table 1.

Method	NSID						AVDP					
	$\tau_b\uparrow$	$S\rho\uparrow$	$\gamma\uparrow$	HL↓	M-1↓	F1↑	$\tau_b\uparrow$	$S\rho\uparrow$	$\gamma\uparrow$	HL↓	M-1↓	F1↑
CRPC(W)	57.89	64.62	70.65	20.36	9.60	68.25	37.57	40.24	41.70	24.28	33.81	50.75
LSEP(W)	71.85	75.80	79.13	10.74	6.03	79.92	39.79	41.93	44.54	19.23	30.29	54.66
GMLR(W)	73.41	77.77	80.81	11.16	5.58	79.94	40.29	42.69	41.12	20.10	31.75	54.31
CRPC(S)	59.54	66.02	72.15	19.22	9.82	69.61	39.69	41.95	44.74	22.17	34.43	52.57
LSEP(S)	72.57	76.57	80.58	10.54	4.69	80.12	40.54	42.60	42.55	18.89	31.41	55.86
GMLR(S)	75.44	78.66	82.87	11.06	6.03	80.08	41.27	43.34	45.27	20.08	29.22	52.54

the label with the maximum predicted score is not in the ground truth positive set. The details for the metrics are described in Appendix C. Table 1 shows the results for each method trained on each Ranked MNIST Gray dataset. Here, GMLR slightly outperforms LSEP, and CRPC performs the worst amongst the three on all metrics. Quantitative results of our experiments on real datasets are given in Table 2. GMLR visibly outperforms the baselines for the ranking metrics, and produces comparable results to LSEP for the classification. It should be noted that both of the real datasets comprise inherent noise due to the labeling process being subjective.

5.5 Adjusting Significance Effects Experiment

To analyze the learned rank scores of GMLR and baseline methods, we first conduct an experiment where we gradually adjusted the selected effects. We have two setups in the experiment: changing scale and changing brightness. For each setup, we generate a set of sequences $\mathcal{D}_a = \{\mathcal{S}_1, ..., \mathcal{S}_{50}\}$, where each sequence consists of gradually changing images, i.e. $\mathcal{S}_i = \langle x_i^{(1)}, ..., x_i^{(50)} \rangle$. Each sequence $\mathcal{S}_i$ consists of three random MNIST digits, let us name them y_i^{low}, y_i^{middle} and y_i^{high}, the starting image of the sequence $x_i^{(1)}$ has the corresponding significance values s^{low}, s^{middle} and s^{high}. Iterating over the images of any sequence $\mathcal{S}_i$, the significance value for y_i^{low} linearly changes from s^{low} to s^{high}, for y_i^{high} changes from s^{high} to s^{low}, and y_i^{middle} remains constant. In the top row of Figure 2 you can see how the images change for each setup. For each method and setup, we obtain the rank scores of the images in $\mathcal{D}_a$ using the network trained with the corresponding method on the corresponding dataset Ranked MNIST Gray-S/B. The average value of each position over all sequences $\mathcal{S}_i$ are calculated for y_i^{low}, y_i^{middle} and y_i^{high}, and we show how the predicted rank scores change in Figure 2 for both strong and weak baselines, and GMLR.

5.6 Calibration Experiment

The calibration experiments are conducted to see how the scores for each baseline are distributed for different instances of the same significance values. We start by generating an image set $\mathcal{D}_C = \{x^{(1)}, ..., x^{(50)}\}$, where each $x^{(i)}$ consists of MNIST digits and is associated with a label set $\mathcal{Y}^{(i)} \subseteq Y$ where $|\mathcal{Y}^{(i)}| = 4$. Each positive class $y_j \in \mathcal{Y}^{(i)}$ in an image $x^{(i)}$ has a one-to-one mapping to $\{1.0, 1.5, 2.0, 2.5\}$ which defines their significance value, in this case the scale value. For each
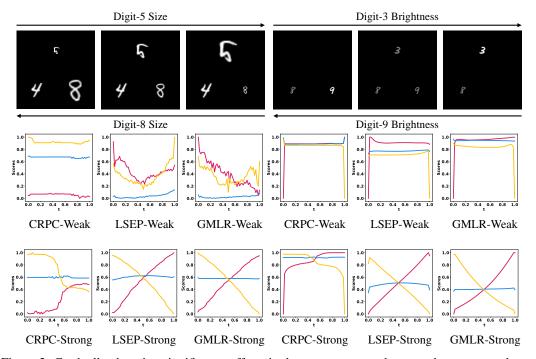
Figure 2: Gradually changing significance effects in the sequences are shown at the top row, where the importance factor is the size of digits in top-left and brightness of digits in top-right. Lines demonstrate changes in scores of $\langle$**1st**, **2nd**, **3rd**$\rangle$ digits, which are in the order of $\langle 5, 4, 8 \rangle$ in top-left and $\langle 3, 8, 9 \rangle$ in top-right. As GaussianMLR (GMLR) produces concurrently adjusting significance scores, it compares favorably over the baseline methods: CRPC and LSEP.

Figure 3: Annotated samples of 4-digit Ranked MNIST Gray-S dataset are shown in the first two columns from the left, where the scale factors of each digit is demonstrated by the bounding boxes for **Scale=1.0**, **Scale=1.5**, **Scale=2.0**, **Scale=2.5**. Resulting plots for each method when the set of scores for each significance value (scale) is fit to a Gaussian distribution are given in the other three columns. GMLR captures proportional scores to significance values of each digit.

method in our experimental setup, we train a network on Ranked MNIST Gray-S, then we feed each image $x^{(i)}$ to the network to produce score vectors $\hat{s}^{(i)}$. From each score vector $\hat{s}^{(i)}$ we select the scores associated with each value in $\{1.0, 1.5, 2.0, 2.5\}$ then create a set of scores for each underlying significance value $\mathcal{S}_{1.0}, \mathcal{S}_{1.5}, \mathcal{S}_{2.0}, \mathcal{S}_{2.5}$. For each set we fit a Gaussian distribution to the values, the resulting plots for each method are given in Figure 3 with a visual expressing the experiment. Both LSEP and GMLR extract an inherent calibration of significance scores, whereas in terms of their magnitude and variance, the order of distributions for each digit is best reflected by GMLR. Another observation from our experiments is that, which is also exemplified in Figure 3, GMLR produces

significance scores with larger variance for larger objects. This is due to naturally increased data variations owing to larger object extent. This positively distinct impact is not observed with other methods that entail intrinsic noise surpassing this effect, which is further explained in Appendix I.

5.7 Extracted Significance Value Experiment

In this experiment, instead of creating gradually changing images to test if an MLR method produces consistent predictions with the underlying process, we do the opposite. Assuming the underlying process exists, we visualize which images in our test set would generate consistent scores with the process. We order the images according to the predicted significance value by our trained model for each class. Then, we select 10 images as checkpoints among the set of 400 test images, by choosing equidistant images in the interval. These image sequences sorted as in Figure 4 demonstrate that as the predicted significance value for a class increases, the dominance of that class also increases. These results suggest that GaussianMLR extracts a proportional score to the underlying significance value for a class.
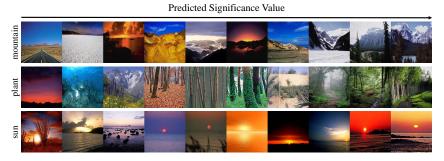

Figure 4: Three sequences of images sampled from the test set of Natural Scene Images database, sorted in the order of predicted significance values for each class by GMLR-Strong. The ordering of images demonstrates that GaussianMLR extracts class significance values that determine their rank proportionally to the dominant class in the image.

6 Conclusion

While previously studied weak multi-label ranking methods learn almost no useful information about the underlying significance values of the positive classes, the strong multi-label ranking paradigm of GMLR yields remarkably calibrated significance values. GMLR compares favorably to the competing baselines as demonstrated by the experiments, where the concurrent gradual changes in the scores for the changing effects in images as well as the constant scores for the static effects indicate that the underlying appearance and geometric characteristics pertinent to ranking are learned by GMLR.

Broad Impact. GaussianMLR encourages new ideas by providing a fresh perspective into the field of MLR. It introduces a set of datasets (Ranked MNISTs), which construct a controllable experimental environment for the new MLR paradigm. Not only GMLR shows the potential on learning more than a ranking between labels but also its output converges to a distribution that is proportional to the underlying significance value process of data characteristics. For a dataset where the labels of any instance are weighted by an unknown factor which controls their relevancy to the instance, GMLR provides a way to extract these unknown factors by only using the ordering between the labels. These findings we believe have potential in numerous applications where ordering of factors are of value for instance in generation, design, or in decision making where alternative choices are typically pairwise ranked.

Limitations and Ethical Concerns. In the absence of strong MLR paradigm, it is an open question whether the MLR can reach or surpass the results provided by GMLR. The true potential of Strong MLR paradigm can be further appreciated with availability of more public synthetic and real-life datasets. GMLR does not impact the explainability and fairness of the decisions made by the underlying design choices, such as the architecture of the used neural nets in the pipeline. GMLR calls for further experimental and theoretical studies in learning calibrated significance values.

A   Datasets

Natural Scene Images Database. Natural scene images database[3] consists of 2000 images of natural scenes, for example: cloud, desert, mountain etc. The dataset has multiple labels per image, to convert them into single label, we applied mean rank ordering[32] as a ranking aggregation method described in their paper. We took a random split of images into sets with size 1600 and 400 to create our train and test sets accordingly. The dataset is public and can be found in http://ldl.herokuapp.com/download.

Architectural VDP Dataset. Architectural Visual Design Principles (VDP)[29] dataset consists of 3654 train and 407 test images where the labels are: asymmetric, color, crystallographic, flowing, isolation, progressive, regular, shape, symmetric. Images are associated with a maximum of 3 positive classes, and each positive class is ranked by their dominance over the other classes in representing the image. The authors of the paper can not provide the dataset publicly, and we obtained the dataset by asking from the authors. Some of the publishable images and their scores are provided in Figure 8.

The datasets have no harmful or offensive content in them. In our work we only provide publishable material, further details of the datasets are held by the respective authors.

B   Implementation Details

For all experiments we use a neural network with ResNet18[33] feature extractor with feature size 512 and an additional 512×512 fully connected layer.

For **CRPC** we add a fully connected layer with output size of $K(K-1)/2$, where K is the number of classes including the virtual label. The virtual label is used to determine *positive* and *negative* predictions such that if predicted score is higher than virtual label's score it is *positive* and vice versa. Here each output value corresponds to the logits for the relation between a unique pair of items. Then for each logit sigmoid function is applied and used for a binary classification, where for a pair (y_u, y_v) if $y_u \succ y_v$ then the ground truth is *positive*, else it is *negative*.

$$L_{CRPC(weak)} = \sum_{i=1}^{N} \sum_{(y_u,y_v)\in\mathcal{S}} -\log\sigma(f_{(u,v)}(\boldsymbol{x}^{(i)}))\beta_u - \log\Big(1 - \sigma(f_{(u,v)}(\boldsymbol{x}^{(i)}))\Big)\beta_v,$$

where $\beta_u = \mathbb{I}[y_u \in \mathcal{Y}^{(i)} \wedge (y_v \notin \mathcal{Y}^{(i)} \vee y_v = v_0)]$, $\beta_v = \mathbb{I}[(y_u \notin \mathcal{Y}^{(i)} \vee y_u = v_0) \wedge y_v \in \mathcal{Y}^{(i)}]$, v_0 is the virtual label, $\mathcal{Y}^{(i)}$ is the set of positive classes for the instance $\boldsymbol{x}^{(i)}$, $\sigma(\cdot)$ is the sigmoid function, $\mathcal{S}$ is the set of unique pairs on the class set Y, where each unique pair corresponds to one output value $f_{(u,v)}$. Further details can be found in [27].
For the strong version of CRPC we change the loss function into:

$$L_{CRPC(strong)} = \sum_{i=1}^{N} \sum_{(y_u,y_v)\in\mathcal{B}^{(i)}} -\log\sigma(f_{(u,v)}(\boldsymbol{x}^{(i)})),$$

where $\mathcal{B}^{(i)}$ is the ground truth bucket order for instance $\boldsymbol{x}^{(i)}$.

For **LSEP** we add two parallel fully connected layers with output size K on top of the feature extractor for scores and thresholds. LSEP consists of two stages, first the score layer is trained with a ranking loss:

$$L_{LSEP/R(weak)} = \sum_{i=1}^{N} \log\left(1 + \sum_{(y_u,y_v)\in\mathcal{B}'^{(i)}} \exp\Big(f_v(\boldsymbol{x}^{(i)}) - f_u(\boldsymbol{x}^{(i)})\Big)\right),$$

where $\mathcal{B}'^{(i)}$ only consists of *positive* and *negative* pairs, such that $(y_u, y_v) \in \mathcal{B}'^{(i)} \iff y_u \in \mathcal{Y}^{(i)} \wedge y_v \notin \mathcal{Y}^{(i)}$. The strong version can be modeled similarly by replacing $\mathcal{B}'^{(i)}$ with the real bucket order for $\boldsymbol{x}^{(i)}$, $\mathcal{B}^{(i)}$:

$$L_{LSEP/R(strong)} = \sum_{i=1}^{N} \log\left(1 + \sum_{(y_u, y_v) \in \mathcal{B}^{(i)}} \exp\left(f_v(\boldsymbol{x}^{(i)}) - f_u(\boldsymbol{x}^{(i)})\right)\right).$$

After training the score head, to be able the determine if a class is *positive* or *negative* all layers of the network except the threshold layer is frozen, then the network is optimized for classification:

$$L_{LSEP/C} = -\sum_{i=1}^{N} \sum_{j=1}^{K} \mathbb{I}[y_j \in \mathcal{Y}^{(i)}] \log\left(\delta_j(\boldsymbol{x}^{(i)})\right) + \mathbb{I}[y_j \notin \mathcal{Y}^{(i)}] \log\left(1 - \delta_j(\boldsymbol{x}^{(i)})\right),$$

where $\delta_k(\boldsymbol{x}^{(i)}) = \sigma(f_k(\boldsymbol{x}^{(i)}) - g_k(\boldsymbol{x}^{(i)}))$, $f(\cdot)$ is the score head and $g(\cdot)$ is the threshold head. Further details for LSEP can be found in [5].

For **GaussianMLR** we add a fully connected layer with output size $2K$ for mean and log variance. The first K values of the output vector are used as predicted mean $\hat{\mu}$ and the second K values of the output vector are used as predicted logarithm variance $\log(\hat{\sigma}^2)$ as commonly practiced in variational autoencoder frameworks. Our proposed loss function consists of two parts, to balance the magnitude of the two terms we use additional weights:

$$\min_{\zeta} \frac{1}{N} \sum_{i=1}^{N} \lambda_1^{(i)} L_c(\hat{\mu}^{(i)}, \hat{\sigma}^{(i)}, \mathcal{Y}^{(i)}) + \lambda_2^{(i)} L_r(\hat{\mu}^{(i)}, \hat{\sigma}^{(i)}, \mathcal{B}^{(i)}),$$

where $\lambda_1^{(i)} = 1/K$ and $\lambda_2^{(i)} = 1/|\mathcal{B}^{(i)}|$, K is the number of classes and $\mathcal{B}^{(i)}$ is the bucket order for the instance $\boldsymbol{x}^{(i)}$.

All networks are trained until convergence with Adam optimizer of learning rate $1.e - 4$ and weight decay $1.e - 5$. For real datasets we use the frozen ResNet18 feature extractor pretrained on ImageNet[34] and train for the rest of the layers. For Ranked MNIST the learning rate is decayed by 0.9 each epoch and for real datasets every 5 epochs. For real datasets we also use RandAugment[35] and for RankedMNIST we apply no augmentation.

C   Metrics

To evaluate our method quantitatively, we use both ranking and classification metrics. For ranking: Kendall's Tau-b (τ_b), Spearman's rank correlation coefficient ($S\rho$), Goodman and Kruskal's gamma (γ), and for classification: Hamming Loss (HL), Max-1 error (M-1), and F1 Score are used. For a predicted and a ground truth ranking, the following are the notations used in formulations:

N_c : the number of concordant pairs,
N_d : the number of discordant pairs,
$N_0 = K(K-1)/2$: the total number of pairs,
N_1 : total number of tied pairs in the prediction,
N_2 : total number of tied pairs in the labels,
r_u : the rank of y_u in the ground truth,
$\hat{r}_u$: the predicted rank of y_u,
$\mathbf{y}$: the binary classification vector of the ground truth, where 1 indicates a positive class,
$\hat{\mathbf{y}}$: the binary classification vector of the prediction,
TP, FP, FN : True Positive, False Positive and False Negative terms.

Formulations for each measure can be found in Table 3. All metrics provided in the tables of quantitative results are the average of the individual metrics for the test sets.

Table 3: Metrics used for quantitative evaluation.

Algorithm	Measure	Formulation
Ranking	↑ Kendall's Tau-b (τ_b)	$\frac{(N_c - N_d)}{\sqrt{(N_0 - N_1)(N_0 - N_2)}}$
	↑ Spearman's ρ ($S\rho$)	$1 - \frac{6 \sum d_i^2}{K(K^2 - 1)}$ (where $d_i = \hat{r}_i - r_i$)
	↑ Goodman and Kruskal's gamma (γ)	$\frac{N_c - N_d}{N_c + N_d}$
Classification	↓ Hamming Loss (HL)	$\sum_{i=1}^{K} \mathbb{I}[\hat{\mathbf{y}}_i \neq \mathbf{y_i}]/K$
	↓ Max-1 Error (M-1)	$\mathbb{I}[\mathbf{y}_{\arg\max_i r_i} \neq 1]$
	↑ F1 Score	TP / (TP + 0.5(FP + FN))

D Learning Implicit Class Significance

$$\min_{\zeta} \frac{1}{N} \sum_{i=1}^{N} L_c(\hat{\mu}^{(i)}, \hat{\sigma}^{(i)}, \mathcal{Y}^{(i)}) + L_r(\hat{\mu}^{(i)}, \hat{\sigma}^{(i)}, \mathcal{B}^{(i)}) \tag{10}$$

Our dataset can be seen as a sample of a continuous data distribution $(\boldsymbol{x}, \mathcal{Y}, \mathcal{B}) \sim \mathcal{D}$, with our objective function given in Equation 10, we optimize the Monte Carlo estimation for the following intractable objective:

$$\min_{\zeta} \mathbb{E}_{(\boldsymbol{x}, \mathcal{Y}, \mathcal{B}) \sim \mathcal{D}} \left[L_c(\hat{\mu}, \hat{\sigma}, \mathcal{Y}) + L_r(\hat{\mu}, \hat{\sigma}, \mathcal{B}) \right] \tag{11}$$

where $f_\mu(\boldsymbol{x}; \zeta) = \hat{\mu}$ and $f_\sigma(\boldsymbol{x}; \zeta) = \hat{\sigma}$. For any input $\boldsymbol{x}^{(i)}$ and for any two classes y_u and y_v, let $s_u^{(i)}$ and $s_v^{(i)}$ be the underlying significance values of the classes, such that $s_u^{(i)} > s_v^{(i)} \iff y_u \succ y_v$ given $\boldsymbol{x}^{(i)}$. Function $f(\boldsymbol{x}; \zeta)$ is limited by ζ in terms of capacity, assuming ζ has finite dimensionality. Let us define $p = P(\hat{s}_u^{(i)} \geq \hat{s}_v^{(i)})$ where $\hat{s}_u^{(i)} \sim \mathcal{N}(\hat{\mu}_u^{(i)}, \hat{\sigma}_u^{2(i)})$ and $\hat{s}_v^{(i)} \sim \mathcal{N}(\hat{\mu}_v^{(i)}, \hat{\sigma}_v^{2(i)})$.

For the optimal parameter ζ^*, the difference between significance values $\epsilon = s_u^{(i)} - s_v^{(i)}$, and the maximal difference Δ, we expect as $\lim_{\epsilon \to \Delta} p = 1$, and $\lim_{\epsilon \to 0} p = 0.5$. This is the case since higher ϵ indicates higher perceivable difference in the instance $\boldsymbol{x}^{(i)}$, thus the relation between $s_u^{(i)}$ and $s_v^{(i)}$ for the instance $\boldsymbol{x}^{(i)}$ will be fairly obvious if ϵ is higher. Since $p \propto \hat{\mu}_u^{(i)} - \hat{\mu}_v^{(i)}$, for $\epsilon = s_u^{(i)} - s_v^{(i)}$, $\epsilon \propto \hat{\mu}_u^{(i)} - \hat{\mu}_v^{(i)}$ for optimal ζ^*. Thus we can say the difference between our predicted means $\hat{\epsilon} = \hat{\mu}_u^{(i)} - \hat{\mu}_v^{(i)}$ is proportional to the real difference of underlying significance values $\epsilon = s_u^{(i)} - s_v^{(i)}$ for an instance $\boldsymbol{x}^{(i)}$. Given the same soft constraint is applied to all pairs on our predicted vector, for a large enough dataset we claim that our predictions will be proportional to the real significance values, which we further supported with empirical studies in the paper Section 5.5, 5.6 and 5.7.

E Ranked MNIST

Ranked MNIST is a family of datasets with two main branches named as Ranked MNIST Gray and Ranked MNIST Color, where the first is in grayscale and the latter has varying random hue and saturation values for each digit. These datasets are generated by placing unique digits from the MNIST dataset [30] on a 224x224 canvas, where the number of digits in a single image vary from 1 up to 10. Here we would like the further explain how we generate the datasets and how each of the 8 different datasets are defined.

For all the datasets we randomly pick the number of digits in the image from a discrete uniform distribution of numbers from 1 to 10, then we randomly place the digits on a 224x224 RGB canvas. For colored datasets we assign a random hue and saturation to color each digit.

Besides the coloring of the digits we have 3 different dataset setups to create the images: 1. Changing Scale, 2. Changing Brightness, 3. Changing Both. Yet we have 4 different datasets since for the scenario where we change both of the importance factors, the labels can be either for the changing scale or for the changing brightness, thus for each coloring we have 4 different in total 8 datasets, namely: Ranked MNIST Gray-S, Ranked MNIST Gray-B, Ranked MNIST Gray-S (Mix), Ranked MNIST Gray-B (Mix), Ranked MNIST Color-S, Ranked MNIST Color-B, Ranked MNIST Color-S (Mix), Ranked MNIST Color-B (Mix), where the letter in the end S and B stands for Scale and Brightness respectively.

For brightness change we use HSV color space and sample brightness values from uniform distribution $\mathcal{U}(0, 1)$ and for the scale change we sample coefficients from $\mathcal{U}(1, 3)$ then resize the digits such that the height and width of the digits are multiplied with the scale coefficients.

Each dataset consists of 60000 train, 10000 validation and 10000 test images. The test digits are sampled from the original MNIST test digits and the rest are sampled from the train digits.

The code to create the datasets will be provided in the supplementary material, and sample images can be seen in Figure 14.

F Ranked MNIST Color Experiments

The Adjusting Significance Effects Experiment explained in Section 5.5 of the main paper is performed on the Ranked MNIST Color dataset, for all four different setups. Corresponding results on Ranked MNIST Color-S and Color-B are given in Figure 5,and for the Mix datasets Color-S (Mix) and Color-B (Mix), results are given in Figure 6. Quantitative results are provided in Table 4. It can be observed that developing the baselines into "*stronger*" versions that enhance the models' ability to adjust to the changing importance factors when compared to the "*weaker*" versions which are the original versions that do not benefit from positive class relations, in terms of the predicted scores. When compared with the Strong versions, our method GMLR performs better in predicting consistent scores both for the increasing or decreasing gradual changes and for the digits with unchanged importance factors, while LSEP produces inconstant scores for the latter. Another interpretation is that the predictions of GMLR are rather independent of the correlation between the digits and it evaluates each digit separately, while LSEP bases the predictions on the relation between digits, causing the convex score line for the 2nd digit in the experiments. These results are in compliance with the experiments of the same setup on Ranked MNIST Gray on the main paper.

The additional Adjusting Significance Effects Experiment on Ranked MNIST Gray (Mix) that we could not provide in the main paper due to space limitations can also be seen in Figure 7.
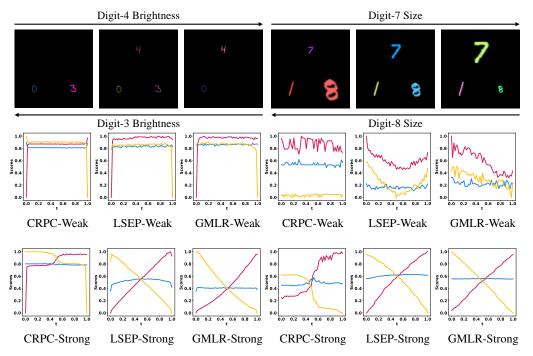
Figure 5: Gradually changing significance effects in the sequences are shown at the top row, where the importance factor is the brightness of digits in top-left, Ranked MNIST Color-B, and size of digits in top-right, Ranked MNIST Color-S. Lines demonstrate changes in scores of ⟨**1st**, **2nd**, **3rd**⟩ digits, which are in the order of ⟨4, 0, 3⟩ in top-left and ⟨7, 1, 8⟩ in top-right.
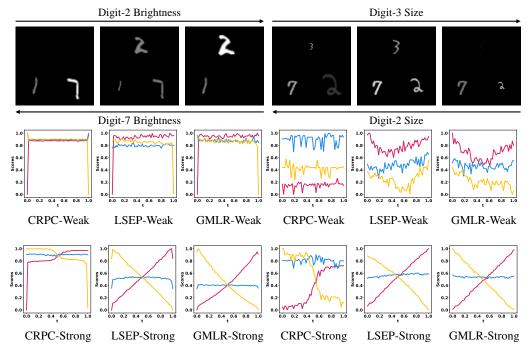

Figure 7: Gradually changing significance effects in the sequences are shown at the top row, where the importance factor is the brightness of digits in top-left, Ranked MNIST Gray-B (Mix), and size of digits in top-right, Ranked MNIST Gray-S (Mix). Size of the digits for Ranked MNIST Gray-B (Mix) and brightness of the digits for Ranked MNIST Gray-S (Mix) change randomly as explained in Section E. Lines demonstrate changes in scores of ⟨**1st**, **2nd**, **3rd**⟩ digits, which are in the order of ⟨2, 1, 7⟩ in top-left and ⟨3, 7, 2⟩ in top-right.
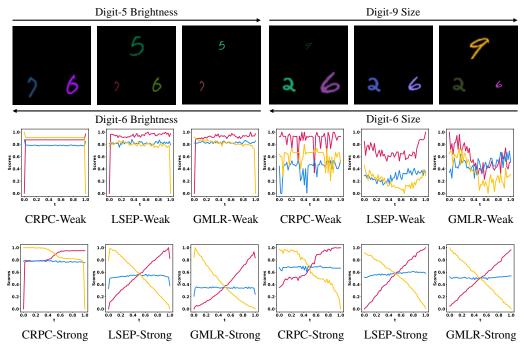
Figure 6: Gradually changing significance effects in the sequences are shown at the top row, where the importance factor is the brightness of digits in top-left, Ranked MNIST Color-B (Mix), and size of digits in top-right, Ranked MNIST Color-S (Mix). Size of the digits for Ranked MNIST Color-B (Mix) and brightness of the digits for Ranked MNIST Color-S (Mix) change randomly as explained in Section E. Lines demonstrate changes in scores of ⟨1st, 2nd, 3rd⟩ digits, which are in the order of ⟨5, 7, 6⟩ in top-left and ⟨9, 2, 6⟩ in top-right.

Table 4: Quantitative results on Ranked MNIST Color datasets. Ranked MNIST S and B stands for changing the scale or brightness of the digits, while (Mix) means both of the features are changing, but the ground truth indicates only one of the features. Bold-marked results show the best scores in Strong(S) baselines, and underlined scores show the best scores in Weak(W) baselines.

Method	Ranked MNIST Color-S						Ranked MNIST Color-B						Ranked MNIST Color-S (Mix)						Ranked MNIST Color-B (Mix)					
	$\tau_b\uparrow$	$S\rho\uparrow$	$\gamma\uparrow$	HL↓	M-1↓	F1↑	$\tau_b\uparrow$	$S\rho\uparrow$	$\gamma\uparrow$	HL↓	M-1↓	F1↑	$\tau_b\uparrow$	$S\rho\uparrow$	$\gamma\uparrow$	HL↓	M-1↓	F1↑	$\tau_b\uparrow$	$S\rho\uparrow$	$\gamma\uparrow$	HL↓	M-1↓	F1↑
CRPC (W)	49.80	60.40	60.11	17.07	0.32	86.45	36.06	62.67	59.29	11.69	0.41	90.28	52.07	62.40	60.15	14.77	0.35	87.93	50.76	61.36	60.23	15.99	0.38	87.05
LSEP (W)	61.85	71.06	61.98	0.51	0.10	99.53	59.70	69.03	60.03	1.07	0.17	99.02	61.58	70.75	61.85	1.01	0.12	99.07	61.12	70.44	61.39	1.00	0.16	99.08
GMLR (W)	63.60	73.05	63.77	0.49	0.08	99.55	60.0	69.25	60.33	1.04	0.19	99.04	62.43	71.60	62.73	0.94	0.11	99.13	61.68	71.10	61.98	1.06	0.11	99.03
CRPC (S)	63.91	75.42	74.88	19.06	0.23	85.08	62.62	74.41	74.78	21.02	0.42	83.69	61.56	73.41	74.13	22.63	0.32	82.53	64.77	76.07	75.0	16.33	0.29	86.78
LSEP (S)	93.76	97.32	94.34	1.42	0.09	98.70	92.91	96.58	93.72	2.02	0.31	98.14	91.74	95.93	92.61	2.29	0.20	97.90	92.48	96.51	93.24	2.07	0.21	98.09
GMLR (S)	94.18	97.52	94.44	0.66	0.07	99.40	89.43	93.44	92.69	3.15	0.88	97.15	92.18	96.27	92.67	1.27	0.15	98.83	88.80	92.77	90.41	2.40	3.03	97.74

G   Bar Graphs

Bar graphs of the predictions of strong versions of CRPC, LSEP and GMLR are given for the four datasets: AVDP Dataset on Figure 8, NSID on Figure 9, Ranked MNIST Gray-S on Figure 10 and Ranked MNIST Color-S on Figure 11. The graphs visualize the working principles of each baseline, where there is an additional virtual label for CRPC, learnable thresholds for each class for LSEP and the zero-point as the inherent threshold for GMLR, to perform binary classification where the positive predicted classes are denoted by green bars and negatives are in red, and the thresholding method is given in purple. The ranks of positive predicted classes are determined by the sorted scores and written below each graph with the ≻ operator denoting precedence.
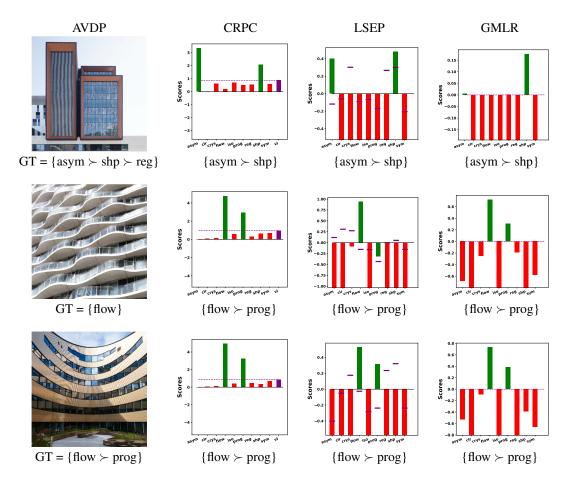
Figure 8: Samples from the test set of the AVDP dataset are given in the first column from left, with the corresponding ground truth labels denoted as GT. Each bar plot represents the predicted scores of baselines CRPC, LSEP and our method GMLR, in their respective columns.
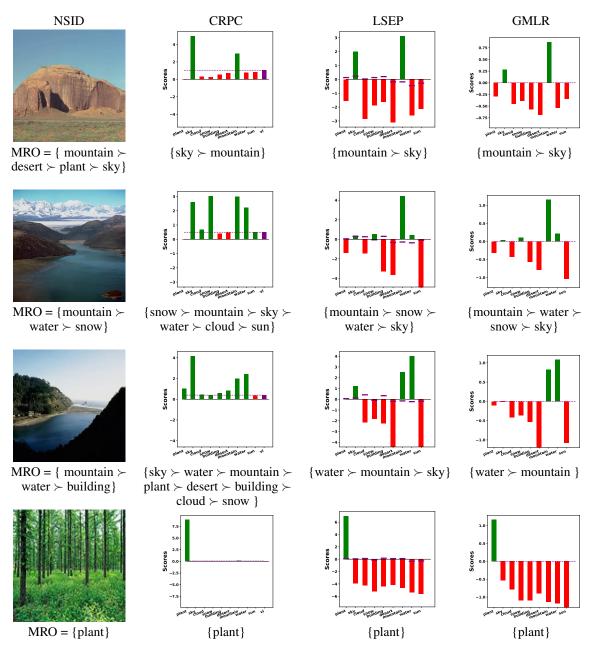
Figure 9: Samples from the test set of the NSID are given in the first column from left, with the corresponding aggregated rankings from ten rankers by the mean rank ordering method denoted as MRO. Each bar plot represents the predicted scores of baselines CRPC, LSEP and our method GMLR, in their respective columns.
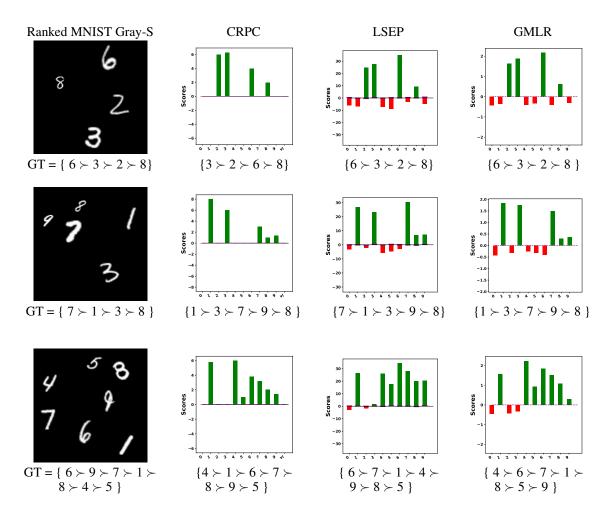
Figure 10: Samples from the test set of the Ranked MNIST Gray dataset are given in the first column from left, with the corresponding ground truth labels denoted as GT. Each bar plot represents the predicted scores of baselines CRPC, LSEP and our method GMLR, in their respective columns.
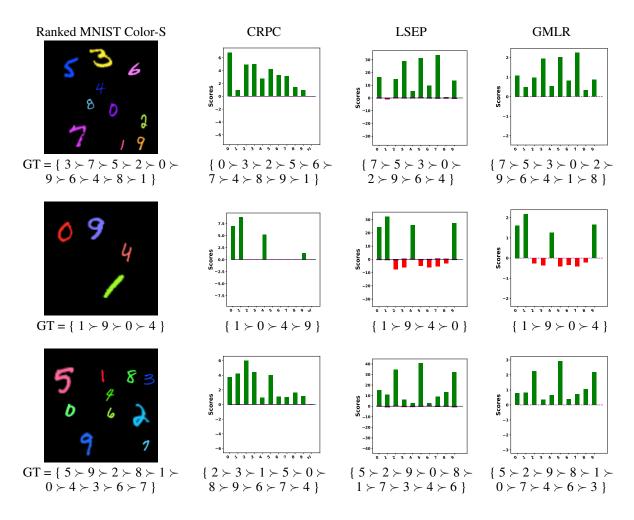
Figure 11: Samples from the test set of the Ranked MNIST Color dataset are given in the first column from left, with the corresponding ground truth labels denoted as GT. Each bar plot represents the predicted scores of baselines CRPC, LSEP and our method GMLR, in their respective columns.

H   Compute Resources

All the trainings, evaluations and experiments are done with an Intel i9-9900K CPU, 64 GB of RAM and two GPUs, NVIDIA TITAN RTX and NVIDIA RTX 2080-ti on an Ubuntu 18.04 machine.

I   Error Bars

The error bar table for real datasets is given in Table 5. It can be seen that Strong methods are consistently better compared to the weak methods. It should be noted that both of the real datasets consists of noisy and subjective labels which can affect the classification scores, while pairwise ranking explains the performance more accurately. Here GMLR manages to outperform the rest of the methods on most of the ranking metrics and yields comparable scores for classification, due to the noisy nature of the datasets we can say the difference between GMLR and LSEP for the classification does not strongly indicate that one is better while the other is not.

We ran each of the training setup given in Table 5 for 5 times with random seeds, the scores on the table are the mean and standard deviation of the metrics for each random run. We are not providing error bars for Ranked MNIST datasets, due to resource limitations, since they are considerably larger compared to the real datasets.

Table 5: Error bar for real datasets NSID and AVDP. Mean scores and standard deviations of each baseline after 5 runs are reported. The annotations of scores are the same with Table 4.

Method	NSID						AVDP					
	$\tau_b \uparrow$	$S\rho \uparrow$	$\gamma \uparrow$	HL$\downarrow$	M-1$\downarrow$	F1$\uparrow$	$\tau_b \uparrow$	$S\rho \uparrow$	$\gamma \uparrow$	HL$\downarrow$	M-1$\downarrow$	F1$\uparrow$
CRPC(W)	57.68 ± 1.0	64.39 ± 1.0	70.43 ± 0.6	20.73 ± 0.9	10.67 ± 0.6	67.97 ± 1.0	38.71 ± 1.0	41.29 ± 1.0	42.83 ± 1.9	23.86 ± 0.4	33.43 ± 0.8	51.8 ± 0.8
LSEP (W)	72.8 ± 0.6	77.0 ± 0.6	80.3 ± 0.5	10.51 ± 0.2	5.31 ± 0.6	80.35 ± 0.3	39.21 ± 2.8	41.3 ± 3.0	41.93 ± 2.7	19.21 ± 0.2	30.66 ± 1.4	54.94 ± 1.2
GMLR (W)	73.49 ± 0.9	77.84 ± 1.0	80.96 ± 1.0	10.64 ± 0.3	5.67 ± 0.5	80.63 ± 0.6	41.13 ± 0.4	43.41 ± 0.4	43.48 ± 1.8	20.03 ± 0.2	30.21 ± 1.0	54.31 ± 0.5
CRPC(S)	59.34 ± 0.5	65.87 ± 0.4	72.47 ± 0.5	19.36 ± 0.3	9.51 ± 0.4	69.44 ± 0.4	39.91 ± 0.6	42.43 ± 0.6	44.71 ± 1.2	23.11 ± 0.4	33.59 ± 0.7	52.66 ± 0.3
LSEP (S)	71.95 ± 1.8	75.94 ± 1.9	79.25 ± 1.8	10.51 ± 0.2	5.49 ± 1.1	80.17 ± 0.4	40.95 ± 1.7	43.02 ± 1.8	44.98 ± 1.5	18.94 ± 0.2	28.76 ± 1.4	55.42 ± 0.7
GMLR (S)	75.54 ± 0.4	78.86 ± 0.4	82.89 ± 0.6	10.87 ± 0.3	6.21 ± 0.5	80.21 ± 0.6	41.68 ± 1.6	43.84 ± 1.7	44.56 ± 2.8	19.88 ± 0.4	29.48 ± 1.2	54.12 ± 1.5

J Variance Experiment

The variance experiments are conducted to observe whether our method distinguishes the small changes in importance factors and is able to assign ranks accordingly. The results of this experiment ran on a similar dataset to Ranked MNIST Gray-S which is provided in Table 6 but instead of sampling the scale from $\mathcal{U}(1,3)$ we sampled it from $\mathcal{U}(1,1.5)$. Superior to other baselines, GMLR-Strong captures the small differences in importance factors for both classification and ranking task. Sample images created for the small variance experiment can be seen in Figure 12.

Table 6: Quantitative results of Variance Experiment on a variation of Ranked MNIST Gray-S where there are small changes in importance factors. The annotations of scores are the same with Table 4.

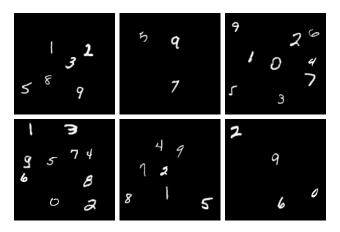| Method	Ranked MNIST Small Change					
	$\tau_b \uparrow$	$S\rho \uparrow$	$\gamma \uparrow$	HL$\downarrow$	M-1$\downarrow$	F1$\uparrow$
CRPC (W)	50.57	60.94	59.81	14.74	0.19	88.05
LSEP (W)	61.50	70.67	61.61	_0.39_	_0.10_	_99.64_
GMLR (W)	_62.18_	_71.31_	_62.32_	0.47	_0.12_	99.57
CRPC (S)	62.46	74.30	74.77	21.27	**0.16**	83.59
LSEP (S)	92.10	96.50	92.61	1.32	0.25	98.79
GMLR (S)	**92.65**	**96.77**	**92.86**	**0.52**	**0.15**	**99.52**

Figure 12: Sample images used in the small variance experiment.

We also conducted a similar experiment to the experiment given in Figure 5, Figure 6 and Figure 7, but instead of providing scores we provide variance as predicted by GaussianMLR. We provide the results in Figure 13 for Ranked MNIST Gray-S, Ranked MNIST Color-S, Ranked MNIST Gray-B, Ranked MNIST Color-B, and Ranked MNIST Small Change. As can be seen from the Figure 13, even for the small change experiment the variance is learned in such a way that the digit with higher significance value always has higher variance. From this finding we hypothesize that this is due to a digit of high importance having a higher $\hat{\mu}$ and a digit of lower importance having a lower $\hat{\mu}$, the same percentage of error will make a higher impact on the loss for the more important digit and this error is being compensated by increasing the variance.
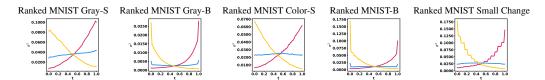
Figure 13: The change of variance for some of the experiments, conducted in Figure 5, Figure 6, Figure 7 and also the Small Change experiment.
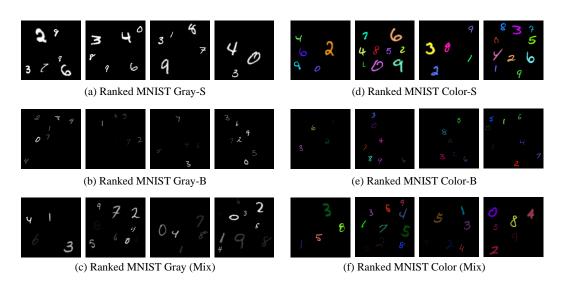

(a) Ranked MNIST Gray-S

(d) Ranked MNIST Color-S

(b) Ranked MNIST Gray-B

(e) Ranked MNIST Color-B

(c) Ranked MNIST Gray (Mix)

(f) Ranked MNIST Color (Mix)

Figure 14: Sample images from the Ranked MNIST family.

K Extracted Significance Value Benchmark

In order to benchmark against other baselines, we provide the results of the Extracted Significance Value experiment in Section 5.7 of the main paper for all baselines on NSID. For the three classes we provided for GMLR on the main paper, Figure 15 compares the results for the mountain class, Figure 16 compares for the plant class and Figure 17 compares for the sun class.

Predicted Significance Value

(a) CLR- Weak

(b) LSEP- Weak

(c) GaussianMLR- Weak

Predicted Significance Value

(d) CLR- Strong

(e) LSEP- Strong

(f) GaussianMLR- Strong

Figure 15: Mountain class benchmark for the extracted significance values.

Predicted Significance Value

(a) CLR- Weak

(b) LSEP- Weak

(c) GaussianMLR- Weak

Predicted Significance Value

(d) CLR- Strong

(e) LSEP- Strong

(f) GaussianMLR- Strong

Figure 16: Plant class benchmark for the extracted significance values.

Predicted Significance Value

(a) CLR- Weak

(b) LSEP- Weak

(c) GaussianMLR- Weak

Predicted Significance Value

(d) CLR- Strong

(e) LSEP- Strong

(f) GaussianMLR- Strong

Figure 17: Sun class benchmark for the extracted significance values.

References

[1] Yangming Zhou, Yangguang Liu, Jiangang Yang, Xiaoqi He, and Liangliang Liu. A taxonomy of label ranking algorithms. *J. Comput.*, 9:557–565, 2014.

[2] Lihi Naamani Dery. Multi-label ranking: Mining multi-label and label ranking data. *ArXiv*, abs/2101.00583, 2021.

[3] Xin Geng, Renyi Zheng, Jiaqi Lv, and Yu Zhang. Multilabel ranking with inconsistent rankers. In *PAMI*, pages 1–1, 2021.

[4] Serhat Selcuk Bucak, Pavan Kumar Mallapragada, Rong Jin, and Anil K. Jain. Efficient multi-label ranking for multi-class learning: Application to object recognition. *ICCV*, pages 2098–2105, 2009.

[5] Y. Li, Yale Song, and Jiebo Luo. Improving pairwise ranking for multi-label image classification. In *CVPR*, pages 1837–1845, 2017.

[6] Young Hun Jung and Ambuj Tewari. Online boosting algorithms for multi-label ranking. In *AISTATS*, volume 84, pages 279–287, 2018.

[7] Raed Alazaidah, Farzana Kabir Ahmad, and Mohamad Farhan Mohamad Mohsin. Multi label ranking based on positive pairwise correlations among labels. *International Arab Journal of Information Technology*, 17, 2019.

[8] Krzysztof Dembczynski, Wojciech Kotłowski, and Eyke Hüllermeier. Consistent multilabel ranking through univariate loss minimization. In *ICML*, page 1347–1354, 2012.

[9] Atsushi Kanehira and Tatsuya Harada. Multi-label ranking from positive and unlabeled data. In *CVPR*, pages 5138–5146, 2016.

[10] Guoqiang Wu, Chongxuan Li, Kun Xu, and Jun Zhu. Rethinking and reweighting the univariate losses for multi-label ranking: Consistency and generalization. In *NeurIPS*, 2021.

[11] Eyke Hüllermeier, Johannes Fürnkranz, Weiwei Cheng, and Klaus Brinker. Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172(16-17):1897–1916, 2008.

[12] Shankar Vembu and Thomas Gärtner. Label ranking algorithms: A survey. In *Preference Learning*, 2010.

[13] William W Cohen, Robert E Schapire, and Yoram Singer. Learning to order things. In *NeurIPS*, volume 10, 1997.

[14] Toshihiro Kamishima, Hideto Kazawa, and Shotaro Akaho. A survey and empirical comparison of object ranking methods. In *Preference Learning*, pages 181–201, 2011.

[15] Minmin Chen, Alice Zheng, and Kilian Weinberger. Fast image tagging. In *ICML*, pages 1274–1282, 2013.

[16] Börkur Sigurbjörnsson and Roelof van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceedings of the 17th International Conference on World Wide Web*, WWW, page 327–336, 2008.

[17] George Toderici, Hrishikesh B. Aradhye, Marius Pasca, Luciano Sbaiz, and Jay Yagnik. Finding meaning on youtube: Tag recommendation and category discovery. *CVPR*, pages 3447–3454, 2010.

[18] Alican Mertan, Yusuf Huseyin Sahin, Damien Jade Duff, and Gozde Unal. A new distributional ranking loss with uncertainty: Illustrated in relative depth estimation. In *3DV*, pages 1079–1088, 2020.

[19] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: From pairwise approach to listwise approach. In *ICML*, pages 129–136, 01 2007.

[20] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. Listwise approach to learning to rank: Theory and algorithm. In *ICML*, page 1192–1199, 2008.

[21] Sariel Har-Peled, Dan Roth, and Dav Zimak. Constraint classification for multiclass classification and ranking. In *NeurIPS*, 2002.

[22] Johannes Fürnkranz and Eyke Hüllermeier. Preference learning and ranking by pairwise comparison. In *Preference Learning*, 2010.

[23] Yunchao Gong, Yangqing Jia, Thomas Leung, Alexander Toshev, and Sergey Ioffe. Deep convolutional ranking for multilabel image annotation. In *CoRR*, 2014.

[24] Jason Weston, Samy Bengio, and Nicolas Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, 2011.

[25] Min-Ling Zhang and Zhi-Hua Zhou. Multilabel neural networks with applications to functional genomics and text categorization. In *IEEE Trans. Knowl. Data Eng.*, pages 1338–1351, 2006.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, volume 25, 2012.

[27] Klaus Brinker, Johannes Fürnkranz, and Eyke Hüllermeier. A unified model for multilabel classification and ranking. In *ECAI*, page 489–493, 2006.

[28] Ronald Fagin, Ravi Kumar, Mohammad Mahdian, D. Sivakumar, and Erik Vee. Comparing and aggregating rankings with ties. In *SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, 2004.

[29] Gözdenur Demir, Aslı Çekmiş, Vahit Buğra Yeşilkaynak, and Gozde Unal. Detecting visual design principles in art and architecture through deep convolutional neural networks. In *Automation in Construction*, 2021.

[30] Li Deng. The mnist database of handwritten digit images for machine learning research. In *IEEE Signal Process*, pages 141–142, 2012.

[31] Johannes Fürnkranz, Eyke Hüllermeier, Eneldo Loza Mencía, and Klaus Brinker. Multilabel classification via calibrated label ranking. *Machine Learning*, 73:133–153, 2008.

[32] Klaus Brinker and Eyke Hüllermeier. Case-based multilabel ranking. In *IJCAI*, 2007.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[34] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009.

[35] Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. Randaugment: Practical automated data augmentation with a reduced search space. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *NeurIPS*, volume 33, pages 18613–18624. Curran Associates, Inc., 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 6.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] All relevant data will be provided in the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] All details are provided in the Appendix B.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Appendix H.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] All details are provided in the Appendix G.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

   (a) If your work uses existing assets, did you cite the creators? [Yes]

   (b) Did you mention the license of the assets? [Yes] See Appendix A.

   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] The code to reproduce the Ranked MNIST datasets is provided in supplemental material

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See Appendix A.

   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] See Appendix A.

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]