CLASSIFICATION AND COMPARISON VIA NEURAL NETWORKS*

İlkay Yıldız†
Dept. of ECE
Northeastern Univ.
yildizi@ece.neu.edu

Peng Tian†
Dept. of ECE
Northeastern Univ.
pengtian@ece.neu.edu

Jennifer Dy
Dept. of ECE
Northeastern Univ.
jdy@ece.neu.edu

Deniz Erdoğmuş
Dept. of ECE
Northeastern Univ.
erdogmus@ece.neu.edu

James Brown
Dept. of Radiology
MGH
jbrown97@mgh.harvard.edu

Jayashree Kalpathy-Cramer
Dept. of Radiology
MGH
kalpathy@nmr.mgh.harvard.edu

Susan Ostmo
Dept. of Ophthalmology
Casey Eye Inst., OHSU
ostmo@ohsu.edu

J. Peter Campbell
Dept. of Ophthalmology
Casey Eye Inst., OHSU
campbelp@ohsu.edu

Michael F. Chiang
Dept. of Ophthalmology
Casey Eye Inst., OHSU
chiangm@ohsu.edu

Stratis Ioannidis
Dept. of ECE
Northeastern Univ.
ioannidis@ece.neu.edu

July 4, 2019

ABSTRACT

We consider learning from *comparison labels* generated as follows: given two samples in a dataset, a labeler produces a label indicating their relative order. Such comparison labels scale quadratically with the dataset size; most importantly, in practice, they often exhibit lower variance compared to class labels. We propose a new neural network architecture based on siamese networks to incorporate both class and comparison labels in the same training pipeline, using Bradley-Terry and Thurstone loss functions. Our architecture leads to a significant improvement in predicting both class and comparison labels, increasing classification AUC by as much as 35% and comparison AUC by as much as 6% on several real-life datasets. We further show that, by incorporating comparisons, training from few samples becomes possible: a deep neural network of 5.9 million parameters trained on 80 images attains a 0.92 AUC when incorporating comparisons.

***K*eywords** neural network · joint learning · comparison · classification · siamese network

1 Introduction

Neural networks have been tremendously successful in tackling supervised learning problems in a variety of domains, including speech recognition [1], natural language processing [2], and image classification [3; 4; 5], to name a few. Unfortunately, in many real-life applications of interest, including medicine [6; 7; 8] and recommender systems [9; 10; 11], class labels are generated by humans and exhibit high variance. This is further exacerbated by the fact that in many domains, such as, e.g., medicine, datasets are often small. Training neural networks over noisy, small datasets is challenging, as high-dimensional parametric models are prone to overfitting in this setting [12; 13].

We propose to overcome this limitation by incorporating *comparisons* in a neural network's training process. In doing so, we exploit the fact that in many domains, beyond producing class labels, labelers can assess the relative order

*Preprint; paper published in the Elsevier Journal of Neural Networks, https://doi.org/10.1016/j.neunet.2019.06.004.
†İlkay Yıldız and Peng Tian are both first authors of this paper.

between two inputs. For example, in a medical diagnosis problem with two classes of images (e.g., diseased and normal), an expert can generate not only diagnostic labels but also order pairs of images w.r.t. disease severity. Similarly, in a recommender system, class labels (e.g., star ratings) are ordered, and labelers can produce relative preferences between any two data samples.

Incorporating comparison labels to the training process has two advantages. First, in the presence of a small dataset, soliciting comparisons in addition to class labels increases the training set size quadratically. Compared to class labels on the same data samples, such labels indeed reveal additional information: comparisons express both inter- and intra-class relationships; the latter are not revealed via class labels alone. Second, in practice, comparisons are often less noisy than (absolute) class labels. Indeed, human labelers disagreeing when generating class judgments often exhibit far less variability when asked to compare pairs of samples instead. This has been extensively documented in a broad array of domains, including medicine [14; 15], movie recommendations [16; 17; 18; 19], travel recommendations [10], music recommendations [11], and web page recommendations [9], to name a few. As a result, incorporating comparisons in training is advantageous even when datasets are large and class labels are abundant; we demonstrate this experimentally in Section 4.1.

We make the following contributions:

- We propose a neural network architecture that is trained on (and can be used to estimate) both class and comparison labels. Our architecture is inspired by siamese networks [20], using Bradley-Terry [21] and Thurstone [22] models as loss functions.
- We extensively evaluate our model w.r.t. several metrics on real-life datasets. We confirm that combining comparisons with class labels via this model consistently improves the classification performance w.r.t. Area Under the ROC Curve (AUC) by $8\% - 35\%$, w.r.t. accuracy by $14\% - 25\%$, w.r.t. F1 score by $12\% - 40\%$, and w.r.t. Area Under the Precision-Recall Curve (PRAUC) by $14\% - 55\%$.
- We validate the benefit of training with comparisons in learning from small datasets. We establish that, by incorporating comparisons, we can train and optimize a neural network with 5,974,577 parameters on a dataset of only 80 samples. In this setting, our combined model attains a classification performance of 0.914 AUC, 0.883 accuracy, 0.64 F1 score, and 0.731 PRAUC, whereas training only on class labels can only achieve 0.835 AUC, 0.705 accuracy, 0.517 F1 score, and 0.177 PRAUC.

The remainder of this paper is organized as follows. Section 2 discusses related work. Section 3 describes our methodology and introduces our model combining class and comparison labels. Section 4 reports our experimental results, evaluated on several real-life datasets. Finally, Section 5 summarizes our contributions.

2 Related Work

Siamese networks were originally proposed to regress similarity between two samples [20]. Many subsequent works using siamese networks thus focus on learning low-dimensional representations to predict similarity, for applications such as multi-task learning [23; 24; 25; 26; 27], drug discovery [28], protein structure prediction [29; 30], person identification [31; 32], and image retrieval [33; 34; 35; 25]. Both the problem we solve and the methodology we employ significantly depart from these works; crucially, the learning objectives used in regressing similarity are not suitable for regressing comparisons. For instance, similarities and respective penalties, such as contrastive loss [33] and triplet loss [34], are sensitive to scaling: inputs further apart are heavily penalized. In contrast, comparisons reflecting relative order and, consequently, penalties for regressing them ought to be scaling-invariant. Such differences lead us to consider altogether different penalties.

A significant body of research in the last decade has focused on comparison and ranking tasks. RankSVM [36] learns a target ranking via a linear Support Vector Machine (SVM), with constraints imposed by all possible pairwise comparisons. Burges et al. [37] estimate pairwise comparisons via a fully connected neural network called RankNet, using the Bradley-Terry generative model [21] to construct a cross-entropy objective function. Following Burges et al. [37], we also model comparison labels as the difference between functions applied to feature pairs. However, we regress *both* comparison *and* class labels, while Burges et al. [37] regress rating scores of compared objects (not comparisons per se). We also depart from cross-entropy and SVM losses used by Burges et al. [37] and Joachims [36], respectively, by learning comparisons via maximum likelihood based on the Bradley-Terry [21] and Thurstone models [38]. As we demonstrate experimentally in Section 4.1, losses induced by these models are more suitable, since these models consider the order between compared input pairs, unlike cross-entropy and SVM losses.

Closer to our work, Chang et al. [39], Dubey et al. [40], and Doughty et al. [41] adopt siamese networks for comparison and ranking tasks. Chang et al. [39] examine the ranking among the burst mode shootings of a scene, using an objective (maximum likelihood based on the Bradley-Terry model) similar to one of the two we consider. Dubey et al. [40] predict the pairwise comparisons of urban appearance images, via a loss function combining cross-entropy loss for

pairwise comparison and hinge loss for ranking. Doughty et al. [41] assess videos with respect to the skill level in their content, by learning similarities and comparisons of video pairs via hinge loss. We differ from these works in jointly learning class labels along with comparison labels within the same pipeline.

Similar to our setting, several works study joint regression and ranking, albeit in shallow learning models. Sculley [42]; Chen et al. [43]; Takamura and Tsujii [44], and Wang et al. [45] learn the rank of two inputs as the difference between their regression outputs, rather than via labeler rankings, and apply this to click prediction and document or image retrieval. Sculley [42] trains a logistic regression model, where the loss function is a weighted combination of the same loss applied to class and ranking predictions. Takamura and Tsujii [44] and Wang et al. [45] employ a similar approach as Sculley [42] and train linear regression models with different loss functions for regression and ranking (mean-square and hinge, respectively). Chen et al. [43] consider joint classification and ranking via a similar approach as RankSVM Joachims [36]. We also minimize a weighted combination of different losses applied to class and comparison labels. However, we model pairwise comparison probabilities via two linear comparison models: Bradley-Terry and Thurstone. Crucially, we also differ by learning both class and comparison labels via a neural network, instead of shallow models such as logistic regression and SVMs.

To the best of our knowledge, Sun et al. [46] is the only previous work incorporating class and comparison labels via a neural network architecture. Nevertheless, they do not incorporate ordered class labels and utilize joint learning only for improving comparison label predictions. Our neural network architecture is more generic and learns class and comparison labels individually or jointly. Finally, we show experimentally in Section 4.2 that our approach outperforms Sun et al. [46] w.r.t. predicting both class and comparison labels on several real-life datasets.

Finally, several works propose methods to reduce overfitting, while training a deep learning model on a small dataset. Mao et al. [47] augment the dataset by generating artificial data samples from a trained SVM classifier. Antoniou et al. [48] and Zhang et al. [49] use generative adversarial networks to produce new images for data augmentation. Hauberg et al. [50] generate synthetic data by using diffeomorphism, which is learned from pairs of data samples within each class. Other works aim to cope with overfitting by means of dimension reduction. Liu et al. [51] propose a method to jointly select features and train a deep learning model. Keshari et al. [52] propose a dictionary based algorithm to learn a single weight for each filter of a Convolutional Neural Network (CNN). Singh and Kingsbury [53] use a dual tree complex wavelet transform (DTCWT) based CNN to learn edge-based invariant representations in the first few layers of the CNN. We differ from these works by not generating artificial data samples but by augmenting the dataset by comparison labels. Moreover, our objective significantly departs from the ones used in these works.

3 Problem Formulation

We consider a dataset containing N items, indexed by $i \in \{1, \dots, N\}$. Every item i has a corresponding d-dimensional feature vector $\boldsymbol{x}_i \in \mathbb{R}^d$. A labeler generates two types of labels for this dataset: absolute labels and comparison labels. Absolute labels characterize the class of an item, while comparison labels show the outcome of a comparison between two items. We denote the absolute label set by D_a, containing tuples of the form (i, y_i), where $y_i \in \{-1, +1\}$ shows which class the item i belongs to. Similarly, we denote the comparison label set by D_c, containing tuples of the form $(i, j, y_{(i,j)})$, where $y_{(i,j)} = +1$ represents that item i is ranked higher than item j and $y_{(i,j)} = -1$, o.w. Neither our model nor our datasets contain any ties: a labeler always selects a comparison label in $\{-1, +1\}$ when presented with two items.

In practice, when class labels are ordered, absolute and comparison labels can be coupled in the following natural fashion: given items (i, j), $y_{(i,j)} = +1$ indicates that i has a higher propensity to receive the absolute label $y_i = +1$, compared to j. As discussed in Section 1, ordered absolute labels naturally occur in applications of interest, such as medicine and recommender systems. For example, in the medical scenario, items $i \in \{1, \dots, N\}$ are images and the absolute label $y_i = +1$ represents the existence of a disease. Given images (i, j), $y_{(i,j)} = +1$ indicates that the labeler deems the presence of the disease to be more severe in i. We describe several additional real-life examples, both medical and non-medical, in Section 4.

Acknowledging the coupling between absolute and comparison labels, our goal is to provide a single probabilistic model that can learn both labels jointly. In particular, learning from both absolute and comparison labels via our model enhances prediction performance on both types. To do so, we propose a combined neural network architecture in the next section.

3.1 Proposed Solution: A Combined Neural Network Architecture

Our neural network architecture is inspired by siamese networks [20], which are extensively used for regressing similarity between two inputs. A siamese network contains two identical *base* networks, followed by a module predicting the similarity between the inputs of the base networks. In this work, we extend the generic application of

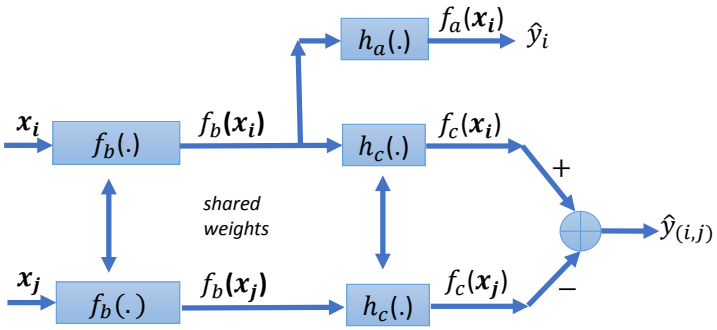


Figure 1: Our Combined Neural Network Architecture. *Base* network f_b has the same structure and parameters for both types of labels. The absolute network receives $\boldsymbol{x}_i$ as input, predicts the corresponding absolute label as $\hat{y}_i$, and consists of f_b and h_a. The comparison network is a siamese network. It receives $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ as inputs, predicts the corresponding comparison label as $\hat{y}_{(i,j)}$, and consists of f_b and h_c.

siamese networks to regress both absolute and comparison labels simultaneously. To do so, we adopt the combined neural network architecture given in Fig. 1.

Our architecture is based on the assumption that absolute and comparison labels depend on the same latent variables. Formally, there exists a function $f_b : \mathbb{R}^d \to \mathbb{R}^p$ representing the coupling between absolute and comparison labels. We interpret $f_b(\boldsymbol{x}_i)$ as a latent vector informative for both absolute and comparison label predictions involving i. Therefore, given items (i, j), the absolute label y_i is regressed from $f_b(\boldsymbol{x}_i)$ and the associated comparison label $y_{(i,j)}$ is regressed from $f_b(\boldsymbol{x}_i)$ and $f_b(\boldsymbol{x}_j)$. We refer to f_b as the *base* network, which has the same structure and parameters for all items $i \in \{1, \ldots, N\}$.

Given f_b, our combined neural network comprises two sub-networks: the absolute network f_a and the comparison network f_c. The absolute network $f_a : \mathbb{R}^d \to \mathbb{R}$ receives $\boldsymbol{x}_i$ as input and outputs $\hat{y}_i$, the absolute label prediction. Formally, for all $(i, y_i) \in D_a$:

$$\hat{y}_i = f_a(\boldsymbol{x}_i) = (h_a \circ f_b)(\boldsymbol{x}_i), \tag{1}$$

where $\circ$ denotes function composition, $f_b : \mathbb{R}^d \to \mathbb{R}^p$ is the base network, and $h_a : \mathbb{R}^p \to \mathbb{R}$ is an additional network linking latent features to absolute labels. The comparison network $f_c : \mathbb{R}^d \to \mathbb{R}$ is trained via a so-called siamese architecture. Given two inputs $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, the corresponding comparison label prediction $\hat{y}_{(i,j)}$ is given by:

$$\hat{y}_{(i,j)} = f_c(\boldsymbol{x}_i) - f_c(\boldsymbol{x}_j) = (h_c \circ f_b)(\boldsymbol{x}_i) - (h_c \circ f_b)(\boldsymbol{x}_j), \tag{2}$$

where $f_b : \mathbb{R}^d \to \mathbb{R}^p$ is again the base network and $h_c : \mathbb{R}^p \to \mathbb{R}$ is an additional network linking latent features to a one-dimensional score $f_c = h_c \circ f_b$. We denote the parametrization of neural network functions f_b, h_a, and h_c w.r.t. their weights as $f_b(\cdot; \boldsymbol{W}_b)$, $h_a(\cdot; \boldsymbol{W}_a)$, and $h_c(\cdot; \boldsymbol{W}_c)$, respectively. Note that, as a result, $f_a(\cdot) = f_a(\cdot; \boldsymbol{W}_a, \boldsymbol{W}_b)$ and $f_c(\cdot) = f_c(\cdot; \boldsymbol{W}_c, \boldsymbol{W}_b)$. Given *both* D_a and D_c, we learn $f_b(\cdot; \boldsymbol{W}_b)$, $h_a(\cdot; \boldsymbol{W}_a)$, and $h_c(\cdot; \boldsymbol{W}_c)$ via a minimization of the form:

$$\begin{aligned} \min_{\boldsymbol{W}_a, \boldsymbol{W}_b, \boldsymbol{W}_c} \quad & \alpha \sum_{(i,y_i) \in D_a} L_a\big(\boldsymbol{x}_i, y_i, f_a(\boldsymbol{x}_i; \boldsymbol{W}_a, \boldsymbol{W}_b)\big) \\ & + (1-\alpha) \sum_{(i,j,y_{(i,j)}) \in D_c} L_c\Big(\boldsymbol{x}_i, \boldsymbol{x}_j, y_{(i,j)}, \big(f_c(\boldsymbol{x}_i; \boldsymbol{W}_c, \boldsymbol{W}_b) - f_c(\boldsymbol{x}_j; \boldsymbol{W}_c, \boldsymbol{W}_b)\big)\Big). \end{aligned} \tag{3}$$

We train our combined neural network via stochastic gradient descent over Eq. (3). Here, L_a is a loss function for the absolute network and L_c is a loss function for the comparison network; we instantiate both below, in Section 3.2. The balance parameter $\alpha \in [0, 1]$ establishes a trade-off between the absolute label set D_a and the comparison label set D_c. When $\alpha = 0$, our model is restricted to the comparison/siamese network setting. When $\alpha = 1$, our model is restricted to the absolute/classical neural network setting.

This architecture is generic and flexible. In particular, the special case where $h_a = h_c$ naturally captures the relationship between binary absolute and comparison labels described in the beginning of Section 3. Indeed, under this design, the event $y_{i,j} = +1$ becomes more likely when i has a higher propensity to receive the absolute label $y_i = +1$, compared to j. Nevertheless, we opt for the more generic scenario in which h_a and h_c may differ; joint prediction of absolute and comparison labels is thus more flexible. Finally, our design naturally extends to multi-class absolute labels, multiple

labelers, and rankings. We present and apply an extension to multi-class absolute labels collected from multiple labelers in our experiments (c.f. ROP in Section 4). We discuss how to generalize to multi-class absolute labels in A, and how to generalize to rankings via the so-called Plackett-Luce model [54] in B.

3.2 Loss Functions

We train f_a using classical loss functions L_a, such as cross-entropy, hinge loss, and mean-square loss. To train f_c via L_c, we use two linear comparison models: Bradley-Terry and Thurstone. In contrast to cross-entropy and hinge losses, linear comparison models consider the order between compared items. The main advantage of these models is the capability of learning a score for each item, even though the labelers provide only binary comparative information, i.e., $y_{(i,j)} \in \{-1, +1\}$ [22]. This score is then used to predict not only the absolute label of the item, but also the rank of the item among all items in the dataset.

Formally, linear models for pairwise comparisons assert that every item i in a labeled comparison dataset D_c is parametrized by a (non-random) score s_i [21; 55]. Then, all labelling events in D_c are independent of each other, and their marginal probabilities are:

$$P(y_{(i,j)} = +1) = F(s_i - s_j), \qquad \forall (i, j, y_{(i,j)}), \tag{4}$$

where F is a cumulative distribution function (c.d.f.). When F is the standard normal c.d.f., Eq. (4) becomes the Thurstone model, yet when F is the logistic c.d.f., Eq. (4) becomes the Bradley-Terry model [22]. We extend the linear comparison model in Eq. (4) by introducing a generic representation of the scores s_i. In particular, we assume that $s_i = f_c(\boldsymbol{x}_i)\,, , \forall i \in \{1, \ldots, N\}$, where f_c is the neural network described in Eq. (2). Then, given $\boldsymbol{x}_i, \boldsymbol{x}_j$, and $y_{(i,j)}$ under the Bradley-Terry model, the (negative log-likelihood) comparison loss becomes:

$$L_c(y_{(i,j)}, \hat{y}_{(i,j)}) = \log(1 + e^{-y_{(i,j)}\hat{y}_{(i,j)}}), \tag{5}$$

where $\hat{y}_{(i,j)} = f_c(\boldsymbol{x}_i) - f_c(\boldsymbol{x}_j)$. Similarly, under the Thurstone model, the (negative log-likelihood) comparison loss becomes:

$$L_c(y_{(i,j)}, \hat{y}_{(i,j)}) = -\log\left(\int_{-\infty}^{\hat{y}_{(i,j)} y_{(i,j)}} \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}} dx\right). \tag{6}$$

4 Experiments

Datasets. We evaluate our model on four real-life datasets, GIFGIF Happiness, GIFGIF Pleasure, ROP, and FAC, summarized in Table 1a.

Dataset	Absolute Label			Comparison Label		
	N_{tr}	N_{val}	N_{test}	N_{tr}	N_{val}	N_{test}
GIFGIF Happiness	823	274	274	8830	390	390
GIFGIF Pleasure	675	225	225	5530	215	215
ROP	60	20	5561	10650	1140	1140
FAC	2112	704	704	4549	224	224

(a) Datasets

Optimized on Validation Set	Values
L_a	{C.E., H.}
L_c	{B.T., T., C.E., H.}
λ	$[2 \times 10^{-4}, 2 \times 10^{-2}]$
L.R.	$[10^{-4}, 10^{-2}]$

(b) Loss Functions and Hyperparameters

Table 1: **(a)** We evaluate our combined network on four real-life datasets: GIFGIF Happiness, GIFGIF Pleasure, ROP, and FAC (c.f. Section 4 for details). For each dataset, we allocate 60% of the total of N images for training, 20% for validation, and 20% for test. Here, N_{tr}, N_{val}, and N_{test} denote training, validation, and test set sizes, respectively. For ROP, we augment the test set by adding the 5561 images in set *H*, which contains only absolute labels and no comparisons. **(b)** For each α, we tune the following on the validation set: absolute loss function L_a, comparison loss function L_c, regularization parameter λ, and learning rate (L.R.). We consider cross-entropy (C.E), hinge (H.), Bradley-Terry (B.T., given in Eq. (5)), and Thurstone (T., given in Eq. (6)) as loss functions.

GIFGIF is an MIT Media Lab project aiming to understand the emotional content of animated GIF images [56]. Labelers are provided with two images and are asked to identify which image better represents a given emotion. A labeler can choose either image, or use a third option: neither. We discard the outcomes that resulted in neither. This dataset does not contain absolute labels. We manually label $N = 1371$ GIF images within the category *happiness*, labeling image i with the absolute label $y_i = +1$ if regarded as happy and $y_i = -1$ if sad. There are 9617 pairwise comparisons among these 1371 images, where $y_{(i,j)} = +1$ if image i is regarded as happier than j, and $y_{(i,j)} = -1$,

o.w. We repeat the same procedure for the emotion category *pleasure* and manually label $N = 1125$ GIF images within this category with $y_i = +1$ if the subject of the image i is pleased and $y_i = -1$ o.w. There are 5667 pairwise comparisons among these 1125 images.

Retinopathy of Prematurity (ROP) is a retinal disease occuring in premature infants; it is a leading cause of childhood blindness [57]. The ROP dataset [58] contains two sets of images: a small set of $N = 100$ retinal images, which we denote by *S*, and a larger set of $N = 5561$ images that we denote by *H*. Images in both sets receive Reference Standard Diagnostic (RSD) labels [59]: To create an RSD label for a retinal image, a committee of three experts labels the image as 'Plus', 'Pre-plus', or 'Normal' based on the existence of ROP, indicating severe, intermediate level, and no ROP, respectively. These ordered categorical labels constitute our absolute labels. In addition, five experts independently label all 4950 pairwise comparisons of the images in set *S* only. Note that some pairs are labeled more than once by different experts. For each pair (i, j), the comparison label is $y_{(i,j)} = +1$ if image i's ROP severity is higher. Note that only images in *S* contain both absolute and comparison labels. We use *S* for training and reserve *H* for testing purposes.

The Filter Aesthetic Comparison (FAC) dataset [46] contains 1280 unfiltered images pertaining to 8 different categories. Twenty-two different image filters are applied to each image and resulting images are labelled by Amazon Mechanical Turk users. Filtered image pairs are labelled such that the comparison label $y_{(i,j)} = +1$ if filtered image i has better quality than filtered image j, and $y_{(i,j)} = -1$ o.w. Each filtered image appears in three comparison pairs. At the same time, absolute labels are generated by the creators of FAC dataset as follows. For each pair (i, j) such that $y_{(i,j)} = +1$, i and j receive scores $+1$ and -1, respectively. Hence, each filtered image receives a score in $[-3, +3]$. Subsequently, image i receiving a $+3$ (-3) score is assigned the label $y_i = +1$ ($y_i = -1$); images that do not receive $+3$ or -3 score are discarded. Thus, the absolute label $y_i = +1$ if the filtered image i has high quality and $y_i = -1$, o.w. We choose one of the categories with $N = 3520$ filtered images as our dataset, since comparison labels only exist for filtered image pairs within the same category. All in all, there are 3520 binary absolute labels and 4964 pairwise comparison labels in this dataset.

Network Architectures. We use GoogLeNet [5] without the last two fully connected layers as our base network f_b. We resize all images, so that $\boldsymbol{x}_i \in \mathbb{R}^{3\times 224\times 224}$. The corresponding base network output $f_b(\boldsymbol{x}_i)$ has $p = 1024$ dimensions $\forall i \in \{1, 2, \cdots, N\}$. To leverage the well-known transfer learning properties of neural networks trained on images [60], we initialize the layers from GoogLeNet with weights pre-trained on the ImageNet dataset [61]. We choose h_a and h_c as single fully connected layers with sigmoid activations. We add L2 regularizers with a regularization parameter λ to all layers and train our combined neural network end-to-end with stochastic gradient descent.[3] We instantiate our loss functions and hyperparameters as shown in Table 1b. Specifically for ROP, we first pre-process retinal images with a pre-trained U-Net [62] to convert the colorful images into black and white masks for retinal vessels [58]. Recall that the special case where $h_a = h_c$, combining cross-entropy as L_a and Bradley-Terry loss as L_c, naturally captures the relationship between absolute and comparison labels (c.f. Section 3), at least in binary classification. In our experiments, we implement both $h_a \neq h_c$ and $h_a = h_c$ (c.f. Section 4.2).

Experiment Setup. We allocate 60% of the total of N images in the GIFGIF Happiness, GIFGIF Pleasure, ROP *S*, and FAC datasets for training, 20% for validation, and 20% for test. Thus, images in the training set are not paired with images in the validation or test sets for the generation of comparison labels. For ROP, we augment the test set by adding the 5561 images in set *H* and predict the existence of ROP during test time, i.e. $\hat{y}_i = -1$ if image i is predicted to be 'Normal' and $\hat{y}_i = +1$, o.w. Recall that *H* contains only absolute labels and no comparisons (c.f. Table 1a).

Given $\alpha \in [0, 1]$, for each combination of the loss functions L_a, L_c, and hyperparameters λ, L.R. shown in Table 1b, we train our architecture over the training set. Then, we evaluate the resulting models on the validation set to determine the optimal loss functions and hyperparameters. We predict the absolute and comparison labels in the test set using the corresponding optimal models. When $\alpha \in (0, 1)$, functions f_b, h_a, and h_c are trained (and tested) on both datasets D_a and D_c. Note that when $\alpha = 0.0$, by Eq. (3), we only learn f_b and h_c, as dataset D_a is not used in training; in this case, we set $h_a = h_c$ to predict the absolute labels on the validation and test sets. Similarly, when $\alpha = 1.0$, we only learn f_b and h_a, and set $h_c = h_a$ to predict comparisons.

Competing Methods. We implement four alternative methods which incorporate absolute and comparison labels: the method by Sun et al. [46], logistic regression, a support vector machine (SVM), and an ensemble method combining the predictions of logistic regression and SVM. We use the norm-comparison loss function L_c by Sun et al. [46], introduced in detail in C, combined with the best performing L_a for each dataset (c.f. Table 1b). In both logistic and SVM methods, we first use GoogLeNet pre-trained on the ImageNet dataset [61] to map input images to features. Then, we train a model via logistic regression and an SVM, respectively, on these features. Training procedures for these methods are reported in more detail in Appendix D. Finally, we implement a soft voting ensemble method combining logistic

[3] Our code is publicly available at `https://github.com/neu-spiral/ClassificationAndComparisonViaNeuralNetworks`.

regression and SVM [63]. More precisely, we train another logistic classifier on the pairs of logistic regression and SVM predictions.

Metrics. To evaluate the performance of our model on validation and test sets, we use Area Under the ROC Curve (AUC), accuracy (Acc.), F1-score (F1), and Area Under the Precision-Recall Curve (PRAUC) on both absolute labels D_a and comparison labels D_c. We report these metrics along with the 95% confidence interval, which we compute as $1.96 \times \sigma_A$, where σ_A^2 is the variance. Variance for AUC and PRAUC are computed by:

$$\sigma_A^2 = \frac{1}{mn}\left(A(1-A) + (m-1)(P_x - A^2) + (n-1)(P_y - A^2)\right), \tag{7}$$

where A is the AUC or PRAUC, $P_x = A/(2-A)$, $P_y = 2A^2/(1+A)$, and m, n are the number of positive and negative samples, respectively [64]. Variance for accuracy and F1-score are computed by:

$$\sigma_A^2 = A(1-A)/(m+n), \tag{8}$$

for A being the accuracy or F1-score. Finally, when comparing the prediction performances of different methods, we assess the statistical significance of relative improvements. We do so via the p-value of one-sided Welch T-test for unequal variances [65], which is computed by:

$$p_{A_l, A_s} = \Phi\Big((A_l - A_s)/\sqrt{\sigma_{A_l}^2 + \sigma_{A_s}^2}\Big), \tag{9}$$

where Φ is the standard normal cumulative distribution function (c.d.f.), and A_l and A_s are the metric values with larger and smaller magnitudes, respectively.

4.1 Prediction Performance

Tables 2a and 2b show the prediction performance of our model trained on GIFGIF Happiness, GIFGIF Pleasure, ROP, and FAC datasets. For each α, we optimize the hyperparameters on the validation set following the procedure explained in Experiment Setup. For AUC, Acc., F1, and PRAUC on both D_a and D_c, we evaluate all eight metrics on the test set, for $\alpha = 0.0$ (comparison labels only), $\alpha = 1.0$ (absolute labels only), and best performing $\alpha \in [0, 1]$. Optimal hyperparameters are reported in Appendix E (c.f. Tables 4 to 11).

Tables 2a and 2b indicate that, on all datasets and metrics, comparisons significantly enhance the predictions of both label types. Compared to training on absolute labels alone ($\alpha = 1.0$), combining absolute and comparison labels ($\alpha \in (0, 1)$) or training only on comparison labels ($\alpha = 0$) considerably improves absolute label prediction, by $8\% - 35\%$ AUC, $14\% - 25\%$ Acc., $12\% - 40\%$ F1, and $14\% - 55\%$ PRAUC across all datasets. For all of these improvements over training with $\alpha = 1.0$, we compute the p-value of Welch T-test. In all cases, p-values are above 0.99. Interestingly, in several cases indicated on Table 2a by *, comparisons lead to better predictions of absolute labels *even when the latter are ignored during training.* This indicates the relative advantage of incorporating comparison labels into training. Adding absolute labels to comparisons also improves comparison prediction performance, albeit modestly (by at most 6% AUC/0.923 p-value, 5% Acc./0.877 p-value, 3% F1/0.793 p-value, and 6% PRAUC/0.913 p-value).

Our performance over the ROP dataset is especially striking. Recall that its training set contains only 80 images and its test set contains 5561 images, unseen during training. Despite this extreme imbalance, by incorporating comparisons into training, *we successfully train and optimize a neural network involving 5,974,577 parameters, on 80 images* (60 for training, 20 for validation)! Specifically, our combined model attains an absolute label prediction performance of 0.914 AUC, 0.883 Acc., 0.64 F1, and 0.731 PRAUC. Training only on absolute labels, i.e. $\alpha = 1.0$, can only achieve 0.835 AUC, 0.705 Acc., 0.517 F1, and 0.177 PRAUC (c.f. ROP rows in Table 2a). This further attests to the fact that comparisons are more informative than absolute labels and that our methodology is able to exploit this information even in the presence of a small dataset.

Our architecture is robust in the selection of a metric to optimize over the validation set. In E (c.f. Tables 4 to 11), we show that optimizing hyperparameters w.r.t. a metric (e.g., AUC) leads to good performance w.r.t. other metrics (e.g., Acc., and F1). Furthermore, for all metrics and datasets, comparison loss functions (L_c) induced by Bradley-Terry and Thurstone models (given in Eq. (5) and Eq. (6)) outperform cross-entropy and hinge losses. This further motivates the use of these linear comparison models for learning relative orders, compared to standard loss functions such as cross-entropy and hinge.

Learning with Fewer Samples. The striking performance of our model on ROP motivates us to study settings in which we learn from small datasets. To that end, we repeat our experiments, training our model on a subset of the training set containing both absolute and comparison labels, selected uniformly at random. The effect of the number of training images on AUC, Acc., and F1 for different values of α can be seen in Fig. 2, for GIFGIF Happiness, GIFGIF Pleasure, and FAC datasets. We illustrate that, despite the small number of images, learning from comparisons consistently

Dataset	α	Performance Metrics on Test Set of Absolute Labels			
		AUC on D_a	Acc. on D_a	F1 on D_a	PRAUC on D_a
Happiness	0.0	0.915 ± 0.041	0.85 ± 0.042	0.779 ± 0.049	0.824 ± 0.057
	1.0	0.561 ± 0.072	0.664 ± 0.055	0.39 ± 0.057	0.401 ± 0.07
	Best(α)	$\mathbf{0.915 \pm 0.041(0.0)^*}$	$\mathbf{0.85 \pm 0.042(0.0)^*}$	$\mathbf{0.791 \pm 0.048(0.75)}$	$\mathbf{0.824 \pm 0.057(0.0)^*}$
Pleasure	0.0	0.887 ± 0.045	0.805 ± 0.052	0.819 ± 0.051	0.873 ± 0.047
	1.0	0.579 ± 0.075	0.56 ± 0.065	0.667 ± 0.062	0.591 ± 0.075
	Best(α)	$\mathbf{0.887 \pm 0.045(0.0)^*}$	$\mathbf{0.814 \pm 0.051(0.25)}$	$\mathbf{0.823 \pm 0.05(0.5)}$	$\mathbf{0.88 \pm 0.046(0.25)}$
ROP	0.0	0.909 ± 0.013	0.824 ± 0.011	0.624 ± 0.013	0.731 ± 0.019
	1.0	0.835 ± 0.017	0.705 ± 0.012	0.517 ± 0.014	0.177 ± 0.012
	Best(α)	$\mathbf{0.914 \pm 0.013(0.75)}$	$\mathbf{0.883 \pm 0.009(0.75)}$	$\mathbf{0.64 \pm 0.013(0.5)}$	$\mathbf{0.731 \pm 0.019(0.0)^*}$
FAC	0.0	0.87 ± 0.028	0.82 ± 0.029	0.809 ± 0.03	0.82 ± 0.033
	1.0	0.746 ± 0.037	0.677 ± 0.035	0.684 ± 0.035	0.678 ± 0.041
	Best(α)	$\mathbf{0.87 \pm 0.028(0.0)^*}$	$\mathbf{0.82 \pm 0.029(0.0)^*}$	$\mathbf{0.809 \pm 0.03(0.0)^*}$	$\mathbf{0.82 \pm 0.033(0.0)^*}$

(a) Absolute label predictions

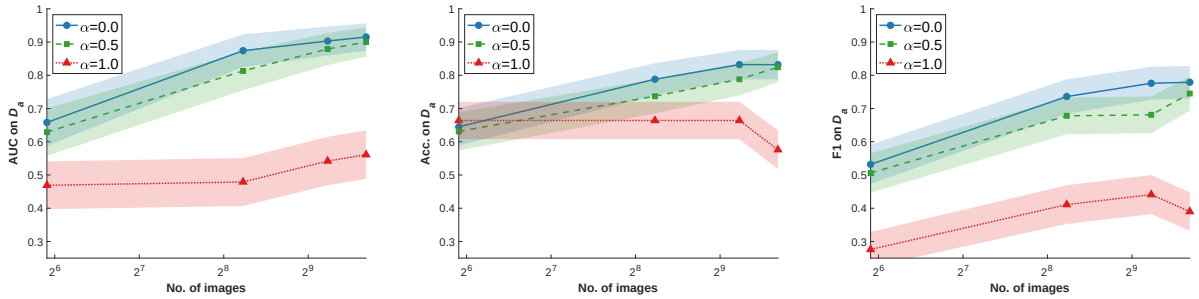
Dataset	α	Performance Metrics on Test Set of Comparison Labels			
		AUC on D_c	Acc. on D_c	F1 on D_c	PRAUC on D_c
Happiness	0.0	0.898 ± 0.031	0.828 ± 0.036	0.819 ± 0.037	0.906 ± 0.031
	1.0	0.419 ± 0.055	0.42 ± 0.047	0.397 ± 0.047	0.448 ± 0.056
	Best(α)	$\mathbf{0.902 \pm 0.03(0.5)}$	$\mathbf{0.83 \pm 0.036(0.25)}$	$\mathbf{0.819 \pm 0.037(0.0)}$	$\mathbf{0.906 \pm 0.031(0.0)}$
Pleasure	0.0	0.836 ± 0.052	0.729 ± 0.058	0.742 ± 0.057	0.849 ± 0.05
	1.0	0.531 ± 0.076	0.492 ± 0.065	0.662 ± 0.062	0.618 ± 0.073
	Best(α)	$\mathbf{0.84 \pm 0.052(0.25)}$	$\mathbf{0.755 \pm 0.056(0.25)}$	$\mathbf{0.775 \pm 0.055(0.25)}$	$\mathbf{0.855 \pm 0.049(0.25)}$
ROP	0.0	0.955 ± 0.013	0.882 ± 0.019	0.885 ± 0.019	0.963 ± 0.012
	1.0	0.941 ± 0.015	0.859 ± 0.021	0.865 ± 0.02	0.942 ± 0.015
	Best(α)	$\mathbf{0.955 \pm 0.013(0.0)}$	$\mathbf{0.883 \pm 0.019(0.25)}$	$\mathbf{0.886 \pm 0.019(0.25)}$	$\mathbf{0.963 \pm 0.012(0.0)}$
FAC	0.0	0.751 ± 0.065	0.71 ± 0.06	0.706 ± 0.06	0.735 ± 0.066
	1.0	0.809 ± 0.058	0.733 ± 0.058	0.72 ± 0.059	0.797 ± 0.06
	Best(α)	$\mathbf{0.814 \pm 0.057(0.5)}$	$\mathbf{0.759 \pm 0.057(0.75)}$	$\mathbf{0.736 \pm 0.058(0.5)}$	$\mathbf{0.797 \pm 0.06(1.0)}$

(b) Comparison label predictions

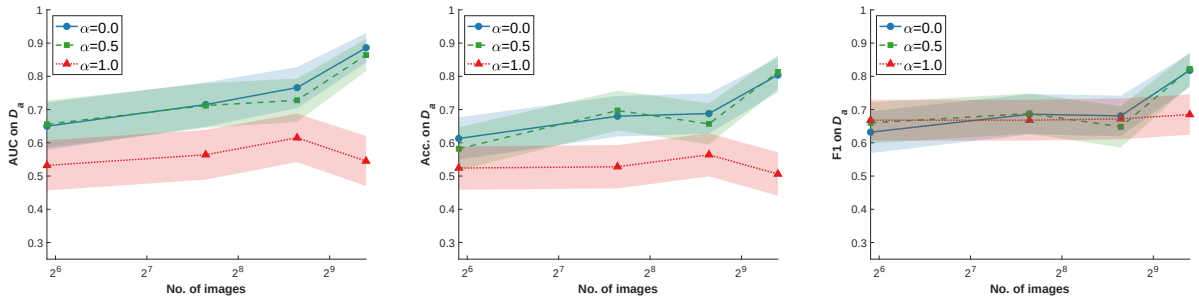
Table 2: Predictions on the GIFGIF Happiness, GIFGIF Pleasure, ROP, and FAC datasets. For each α, we optimize the hyperparameters on the validation set following the procedure explained in Experiment Setup. For AUC, Acc., F1, and PRAUC on both D_a and D_c, we evaluate all eight metrics on the test set, for $\alpha = 0.0$ (comparison labels only), $\alpha = 1.0$ (absolute labels only), and best performing $\alpha \in [0, 1]$. Optimal hyperparameters are reported E (c.f. Tables 4 to 11). We use $h_a \neq h_c$ in these experiments.

improves absolute label predictions. In fact, e.g., in Fig. 2a, using comparisons ($\alpha \in \{0.0, 0.5\}$), we can reduce the training set size from 823 to 300 with a performance loss of only 4% on AUC, Acc., and F1. In contrast, using only absolute labels in training ($\alpha = 1.0$) leads to a significant performance drop, by $12\% - 39\%$ AUC, $13\% - 30\%$ Acc., and $14\% - 39\%$ F1. These results confirm that, in the presence of a small dataset with few samples, soliciting comparisons along with training via our combined architecture significantly boosts prediction performance.

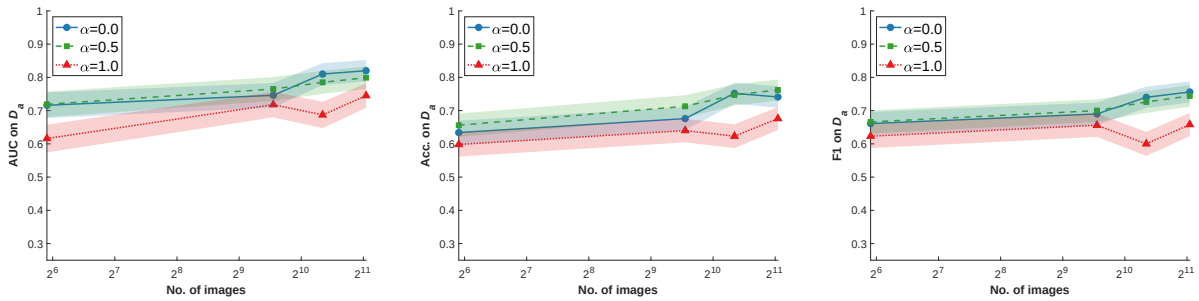
Comparison Labels Exhibit Less Noise Than Absolute Labels. In the previous experiment, given the number of samples, the number of comparisons is significantly larger than the number of absolute labels. This raises the question of how training performs when the number of comparisons is equal to the number of absolute labels. To answer this question, we repeat the previous experiment by subsampling labels rather than images and train our model on the same number of absolute or comparison labels. In Fig. 3, we show AUC, Acc., and F1 as functions of the number of training labels for GIFGIF Happiness, GIFGIF Pleasure, and FAC datasets. We demonstrate that, for the same number of training labels, learning from comparisons ($\alpha = 0.0$) instead of absolute labels ($\alpha = 1.0$) boosts absolute label predictions by $11\% - 30\%$ AUC, $6\% - 24\%$ Acc., and $14\% - 17\%$ F1. This further corroborates our claim that comparisons are indeed less noisy and more informative than absolute labels. We note that this has been observed in other settings [14; 15; 16].



(a) AUC, accuracy, F1 on D_a vs. N_{tr} for GIFGIF Happiness



(b) AUC, accuracy, F1 on D_a vs. N_{tr} for GIFGIF Pleasure



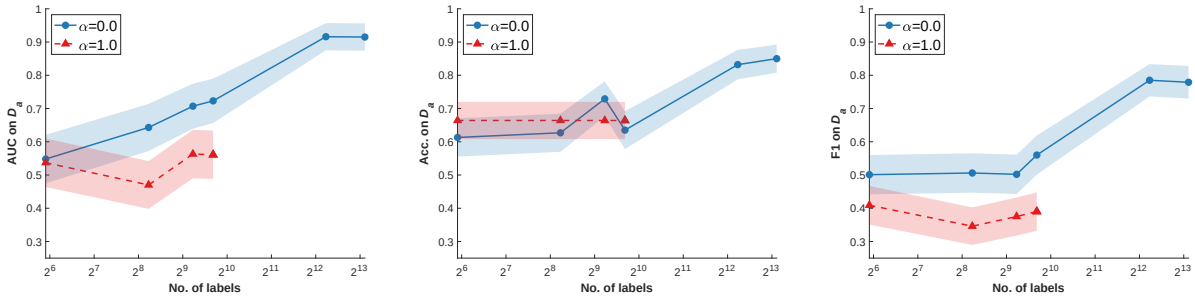
(c) AUC, accuracy, F1 on D_a vs. N_{tr} for FAC

Figure 2: Prediction performance on absolute labels (D_a) vs. number of training images (N_{tr}) for the (a) GIFGIF Happiness, (b) GIFGIF Pleasure, and (c) FAC datasets. We train our model on the absolute labels D_a and comparison labels D_c pertaining to N_{tr} images randomly sampled from the training set. For each $N_{tr} \in \{60, ...\}$ and $\alpha \in \{0.0, 0.5, 1.0\}$, we optimize the hyperparameters on the validation set following the procedure explained in Experiment Setup. We evaluate AUC, Acc., and F1 on the test set w.r.t. D_a.

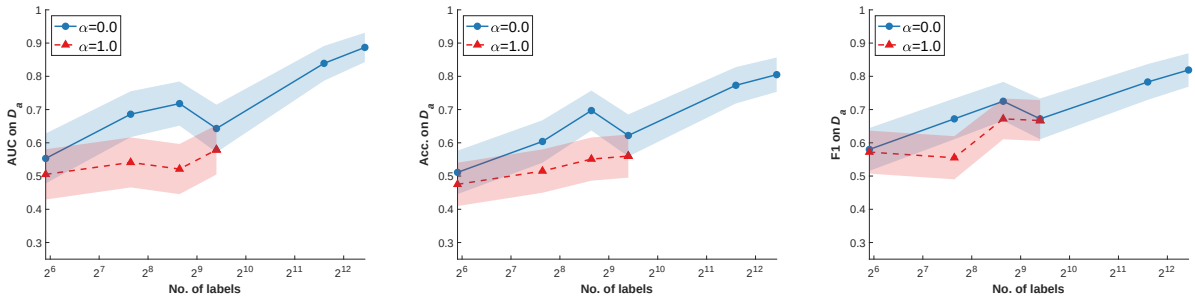
Impact of α. Fig. 4 shows the prediction AUC of our model vs. α on GIFGIF Happiness, GIFGIF Pleasure, ROP, and FAC datasets. For each α, we optimize the hyperparameters on the validation set following the procedure explained in Section 4.1 and evaluate AUC on the test set w.r.t. both D_a and D_c. On all datasets, comparisons significantly boost the prediction performance on both D_a and D_c. Our model achieves the peak prediction AUCs on both labels for learning from comparisons ($\alpha \in [0.0, 1.0)$) and the lowest prediction AUCs for learning from absolute labels alone ($\alpha = 1.0$), except for the FAC dataset. The FAC dataset differs from other datasets, since absolute labels are generated from comparisons instead of being independently produced by labelers (c.f. Datasets in Section 4).

4.2 Comparison with Competing Methods

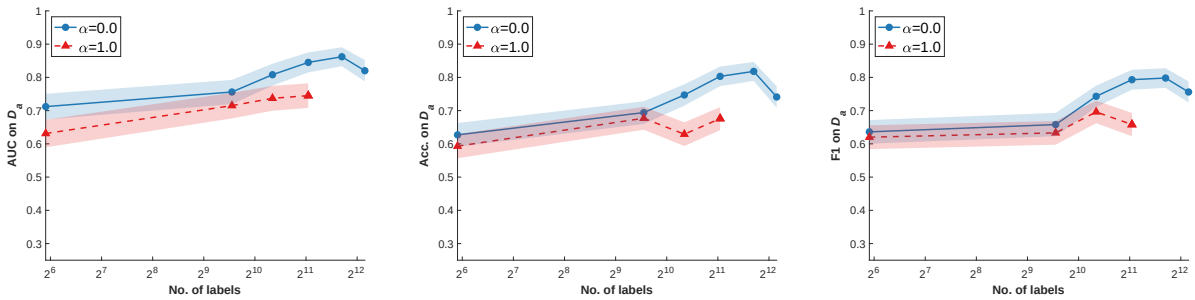
In Tables 3a and 3b, we compare our combined neural network (CNN) with the Sun et al. [46] method (Norm), logistic regression (Log.), a support vector machine (SVM), and a soft voting ensemble method (Ensemble) linearly combining the predictions of logistic regression and SVM. We evaluate our CNN model both for $h_a \neq h_c$, and the special case where $h_a = h_c$ $\big($CNN $(h_a = h_c)\big)$, combining cross-entropy as L_a and Bradley-Terry loss as L_c. For each α, we optimize the hyperparameters on the validation set following the procedure explained in Experiment Setup. For the best performing $\alpha \in [0, 1]$, we evaluate AUC, Acc., and F1 on the test set w.r.t. both D_a and D_c. We demonstrate



(a) AUC, accuracy, F1 on D_a vs. N_{lb} for GIFGIF Happiness

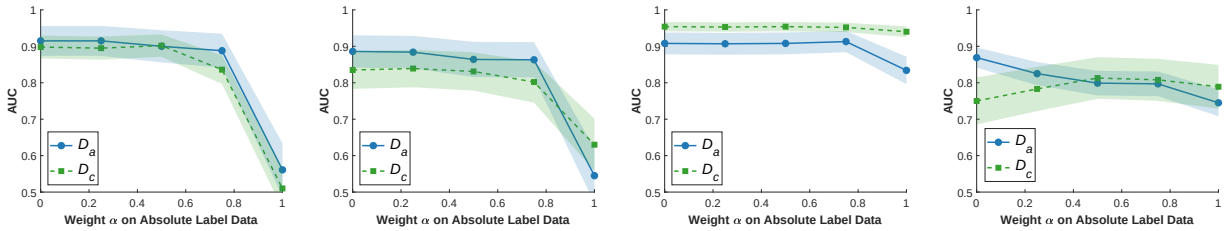


(b) AUC, accuracy, F1 on D_a vs. N_{lb} for GIFGIF Pleasure



(c) AUC, accuracy, F1 on D_a vs. N_{lb} for FAC

Figure 3: Prediction performance on absolute labels (D_a) vs. number of labels (N_{lb}) in the training set for the (a) GIFGIF Happiness, (b) GIFGIF Pleasure, and (c) FAC datasets. We train our model on N_{lb} absolute labels (D_a) and N_{lb} comparison labels (D_c) randomly sampled from the training set. For each $N_{lb} \in \{60, ...\}$ and $\alpha \in \{0.0, 1.0\}$, we optimize the hyperparameters on the validation set following the procedure explained in Experiment Setup. We evaluate AUC, Acc., and F1 on the test set w.r.t. D_a.



(a) AUC on Happiness (b) AUC on Pleasure (c) AUC on ROP (d) AUC on FAC

Figure 4: Prediction performance vs. α on GIFGIF Happiness, GIFGIF Pleasure, ROP, and FAC datasets. For each $\alpha \in \{0.0, 0.25, 0.5, 0.75, 1.0\}$, we optimize the hyperparameters on the validation set following the procedure explained in Section 4.1. Finally, we evaluate the optimal models on the test set of both D_a and D_c w.r.t. AUC.

that learning through the norm comparison loss (Norm) instead of maximum likelihood based on Bradley-Terry or Thurstone models (given in Eq. (5) and (6)) is not suitable for our architecture. Moreover, our method outperforms logistic, SVM, and ensemble methods, particularly in absolute label prediction, by $2\% - 13\%$ AUC, $5\% - 13\%$ Acc.,

Dataset	Method	Performance Metrics on Test Set of Absolute Labels		
		AUC on D_a (α)	Acc. on D_a (α)	F1 on D_a (α)
Pleasure	**CNN**	$\mathbf{0.887 \pm 0.045(0.0)}$	$0.814 \pm 0.051(0.25)$	$0.823 \pm 0.05(0.5)$
	CNN($h_a = h_c$)	0.882 ± 0.046	$\mathbf{0.823 \pm 0.05}$	$\mathbf{0.827 \pm 0.05}$
	Log.	$0.846 \pm 0.051(0.25)$	$0.773 \pm 0.054(0.75)$	$0.775 \pm 0.054(0.75)$
	SVM	$0.862 \pm 0.048(0.75)$	$0.768 \pm 0.055(0.0)$	$0.796 \pm 0.052(0.0)$
	Ensemble	$0.852 \pm 0.051(0.25)$	$0.787 \pm 0.054(0.5)$	$0.794 \pm 0.053(0.5)$
	Norm	$0.5 \pm 0.07(0.5)$	$0.5 \pm 0.065(0.5)$	$0.67 \pm 0.061(0.5)$
ROP	**CNN**	$0.914 \pm 0.013(0.75)$	$\mathbf{0.883 \pm 0.009(0.75)}$	$\mathbf{0.64 \pm 0.013(0.5)}$
	CNN($h_a = h_c$)	$\mathbf{0.929 \pm 0.012}$	0.876 ± 0.009	0.64 ± 0.013
	Log.	$0.788 \pm 0.017(0.0)$	$0.823 \pm 0.01(0.0)$	$0.484 \pm 0.013(1.0)$
	SVM	$0.793 \pm 0.017(0.5)$	$0.822 \pm 0.01(0.0)$	$0.497 \pm 0.013(0.5)$
	Ensemble	$0.79 \pm 0.018(0.0)$	$0.823 \pm 0.011(0.0)$	$0.478 \pm 0.014(1.0)$
	Norm	$0.63 \pm 0.04(0.0)$	$0.82 \pm 0.01(0.5)$	$0.3 \pm 0.012(0.0)$
FAC	**CNN**	$\mathbf{0.87 \pm 0.028(0.0)}$	$\mathbf{0.82 \pm 0.029(0.0)}$	$0.809 \pm 0.03(0.0)$
	CNN($h_a = h_c$)	0.839 ± 0.031	0.817 ± 0.029	$\mathbf{0.818 \pm 0.029}$
	Log.	$0.746 \pm 0.036(0.5)$	$0.681 \pm 0.034(0.5)$	$0.655 \pm 0.035(0.0)$
	SVM	$0.741 \pm 0.037(0.5)$	$0.684 \pm 0.034(0.25)$	$0.648 \pm 0.035(0.5)$
	Ensemble	$0.746 \pm 0.037(0.5)$	$0.692 \pm 0.035(0.5)$	$0.658 \pm 0.036(0.5)$
	Norm	$0.52 \pm 0.04(0.5)$	$0.55 \pm 0.036(1.0)$	$0.61 \pm 0.036(0.5)$

(a) Absolute label predictions of competing methods.

Dataset	Method	Performance Metrics on Test Set of Comparison Labels		
		AUC on D_c (α)	Acc. on D_c (α)	F1 on D_c (α)
Pleasure	**CNN**	$0.84 \pm 0.052(0.25)$	$0.755 \pm 0.056(0.25)$	$0.775 \pm 0.055(0.25)$
	CNN($h_a = h_c$)	$\mathbf{0.866 \pm 0.048}$	$\mathbf{0.764 \pm 0.056}$	$\mathbf{0.796 \pm 0.053}$
	Log.	$0.805 \pm 0.056(0.0)$	$0.741 \pm 0.056(0.0)$	$0.763 \pm 0.055(0.0)$
	SVM	$0.803 \pm 0.056(0.75)$	$0.736 \pm 0.057(0.0)$	$0.754 \pm 0.055(0.0)$
	Ensemble	$0.813 \pm 0.056(0.25)$	$0.75 \pm 0.057(0.25)$	$0.772 \pm 0.055(0.25)$
	Norm	$0.5 \pm 0.07(0.5)$	$0.46 \pm 0.064(0.5)$	$0 \pm 0(0.5)$
ROP	**CNN**	$\mathbf{0.955 \pm 0.013(0.0)}$	$\mathbf{0.883 \pm 0.019(0.25)}$	$\mathbf{0.886 \pm 0.019(0.25)}$
	CNN($h_a = h_c$)	0.953 ± 0.013	0.874 ± 0.02	0.88 ± 0.019
	Log.	$0.842 \pm 0.023(0.5)$	$0.762 \pm 0.024(0.25)$	$0.757 \pm 0.024(0.75)$
	SVM	$0.829 \pm 0.024(0.0)$	$0.732 \pm 0.025(0.75)$	$0.743 \pm 0.025(0.5)$
	Ensemble	$0.837 \pm 0.024(0.5)$	$0.758 \pm 0.025(0.5)$	$0.758 \pm 0.025(0.5)$
	Norm	$0.65 \pm 0.03(1.0)$	$0.47 \pm 0.028(0.5)$	$0.5 \pm 0.028(1.0)$
FAC	**CNN**	$0.814 \pm 0.057(0.5)$	$\mathbf{0.759 \pm 0.057(0.75)}$	$0.736 \pm 0.058(0.5)$
	CNN($h_a = h_c$)	0.816 ± 0.057	0.755 ± 0.057	0.749 ± 0.057
	Log.	$0.82 \pm 0.055(0.0)$	$0.75 \pm 0.056(0.25)$	$0.75 \pm 0.056(0.25)$
	SVM	$0.821 \pm 0.055(0.25)$	$0.75 \pm 0.056(0.25)$	$0.75 \pm 0.056(0.25)$
	Ensemble	$\mathbf{0.822 \pm 0.056(0.25)}$	$0.755 \pm 0.057(0.25)$	$\mathbf{0.756 \pm 0.057(0.25)}$
	Norm	$0.52 \pm 0.07(0.5)$	$0.54 \pm 0.065(0.5)$	$0.37 \pm 0.063(0.5)$

(b) Comparison label predictions of competing methods.

Table 3: As alternatives to our Combined Neural Network (CNN) method, we implement [46]'s method (Norm), logistic regression (Log.), SVM, and a soft voting ensemble method (Ensemble) linearly combining the predictions of logistic regression and SVM using the objectives explained in C and D. We further evaluate our CNN method for the special case where $h_a = h_c$ $\big($CNN ($h_a = h_c$)$\big)$, combining cross-entropy as L_a and Bradley-Terry loss as L_c. For each method and each α, we optimize the hyperparameters on the validation set following the procedure explained in Experiment Setup. Finally, for the best performing $\alpha \in [0, 1]$ on the test set, we evaluate AUC, Acc., and F1 on GIFGIF Pleasure, ROP, and FAC datasets w.r.t. both D_a and D_c.

and $3\% - 15\%$ F1 across all datasets. These improvements correspond to $0.772 - 1.0$ p-value for AUC, $0.91 - 1.0$ p-value for Acc., and $0.8 - 1.0$ p-value for F1 under Welch T-test. Our method also outperforms other methods on comparison label prediction, specifically on ROP, by 11% AUC, 12% Acc., and 13% F1. For all of these improvements, p-values are 1.0.

Note that the performance of our architecture is robust to the design choice between single task learning with $h_a = h_c$ vs. generic joint learning with $h_a \neq h_c$. More precisely, our model balances the variance introduced by allowing h_a and h_c to differ by improving the bias.

5 Conclusion

In this paper, we tackle the problem of limited and noisy data that emerge in real-life applications. To do so, we propose a neural network architecture that can collectively learn from both absolute and comparison labels. We extensively evaluate our model on several real-life datasets and metrics, demonstrating that learning from both labels immensely improves predictive power on both label types. By incorporating comparisons into training, we successfully train an architecture containing 5.9 million parameters with only 80 images. All in all, we observe the benefit of learning from comparison labels.

Given the quadratic nature of pairwise comparisons, designing active learning algorithms that identify which comparisons to solicit from labelers is an interesting open problem. For example, generalizing active learning algorithms, as the ones proposed by Guo et al. [66] for shallow models of comparisons to the neural network model we consider here, is a promising direction.

Acknowledgments

Our work is supported by NIH (R01EY019474), NSF (SCH-1622542 at MGH; SCH-1622536 at Northeastern; SCH-1622679 at OHSU), and by unrestricted departmental funding from Research to Prevent Blindness (OHSU).

References

[1] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

[2] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[4] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[5] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. In *Computer Vision and Pattern Recognition, 2015.*, 2015.

[6] James D Reynolds, Velma Dobson, Graham E Quinn, Alistair R Fielder, Earl A Palmer, Richard A Saunders, Robert J Hardy, Dale L Phelps, John D Baker, Michael T Trese, et al. Evidence-based screening criteria for retinopathy of prematurity: natural history data from the CRYO-ROP and LIGHT-ROP studies. *Archives of Ophthalmology*, 120(11):1470–1476, 2002.

[7] Michael F Chiang, Lei Jiang, Rony Gelman, Yunling E Du, and John T Flynn. Interexpert agreement of plus disease diagnosis in retinopathy of prematurity. *Archives of Ophthalmology*, 125(7):875–880, 2007.

[8] David K Wallace, Graham E Quinn, Sharon F Freedman, and Michael F Chiang. Agreement among pediatric ophthalmologists in diagnosing plus and pre-plus disease in retinopathy of prematurity. *Journal of American Association for Pediatric Ophthalmology and Strabismus*, 12(4):352–356, 2008.

[9] Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. In *Advances in Neural Information Processing Systems*, pages 41–48, 2004.

[10] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining interesting locations and travel sequences from gps trajectories. In *Proceedings of the 18th International Conference on World Wide Web*, pages 791–800. ACM, 2009.

[11] Yehuda Koren and Joe Sill. Ordrec: an ordinal model for predicting personalized item rating distributions. In *Proceedings of the fifth ACM Conference on Recommender Systems*, pages 117–124. ACM, 2011.

[12] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[13] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[14] Neil Stewart, Gordon DA Brown, and Nick Chater. Absolute identification by relative judgment. *Psychological Review*, 112(4):881, 2005.

[15] Jayashree Kalpathy-Cramer, J Peter Campbell, Deniz Erdoğmuş, Peng Tian, Dharanish Kedarisetti, Chace Moleta, James D Reynolds, Kelly Hutcheson, Michael J Shapiro, Michael X Repka, et al. Plus disease in retinopathy of prematurity: Improving diagnosis by ranking disease severity and using quantitative image analysis. *Ophthalmology*, 123(11):2345–2351, 2016.

[16] Armelle Brun, Ahmad Hamad, Olivier Buffet, and Anne Boyer. Towards preference relations in recommender systems. In *Preference Learning (PL 2010) ECML/PKDD 2010 Workshop*, 2010.

[17] Maunendra Sankar Desarkar, Sudeshna Sarkar, and Pabitra Mitra. Aggregating preference graphs for collaborative rating prediction. In *Proceedings of the fourth ACM Conference on Recommender Systems*, pages 21–28. ACM, 2010.

[18] Maunendra Sankar Desarkar, Roopam Saxena, and Sudeshna Sarkar. Preference relation based matrix factorization for recommender systems. In *International conference on user modeling, adaptation, and personalization*, pages 63–75. Springer, 2012.

[19] Shaowu Liu, Truyen Tran, Gang Li, and Yuan Jiang. Ordinal random fields for recommender systems. In *ACML 2014: Proceedings of the Sixth Asian Conference on Machine Learning*, pages 283–298. JMLR Workshop and Conference Proceedings, 2014.

[20] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In *Advances in Neural Information Processing Systems*, pages 737–744, 1994.

[21] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[22] Manuela Cattelan. Models for paired comparison data: A review with emphasis on dependent data. *Statistical Science*, pages 412–433, 2012.

[23] Rongkai Xia, Yan Pan, Hanjiang Lai, Cong Liu, and Shuicheng Yan. Supervised hashing for image retrieval via image representation learning. In *AAAI*, page 2, 2014.

[24] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.

[25] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. In *European Conference on Computer Vision*, pages 241–257. Springer, 2016.

[26] Edgar Simo-Serra and Hiroshi Ishikawa. Fashion style in 128 floats: joint ranking and classification using weak data for feature extraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 298–307, 2016.

[27] Chen Shen, Zhongming Jin, Yiru Zhao, Zhihang Fu, Rongxin Jiang, Yaowu Chen, and Xian-Sheng Hua. Deep siamese network with multi-level similarity perception for person re-identification. In *Proceedings of the 2017 ACM on Multimedia Conference*, pages 1942–1950. ACM, 2017.

[28] Natalie Stephenson, Emily Shane, Jessica Chase, Jason Rowland, David Ries, Nicola Justice, Jie Zhang, Leong Chan, and Renzhi Cao. Survey of machine learning techniques in drug discovery. *Current drug metabolism*, 2018.

[29] Jie Hou, Tianqi Wu, Renzhi Cao, and Jianlin Cheng. Protein tertiary structure modeling driven by deep learning and contact distance prediction in casp13. *bioRxiv*, page 552422, 2019.

[30] Spencer Moritz, Jonas Pfab, Tianqi Wu, Jie Hou, Jianlin Cheng, Renzhi Cao, Liguo Wang, and Dong Si. Cascaded-cnn: Deep learning to predict protein backbone structure from high-resolution cryo-em density maps. *BioRxiv*, page 572990, 2019.

[31] Shi-Zhe Chen, Chun-Chao Guo, and Jian-Huang Lai. Deep ranking for person re-identification via joint representation learning. *IEEE Transactions on Image Processing*, 25(5):2353–2367, 2016.

[32] Haoran Wu, Zhiyong Xu, Jianlin Zhang, Wei Yan, and Xiao Ma. Face recognition based on convolution siamese networks. In *Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), 2017 10th International Congress on*, pages 1–5. IEEE, 2017.

[33] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1735–1742. IEEE, 2006.

[34] Mohammad Norouzi, David J Fleet, and Ruslan R Salakhutdinov. Hamming distance metric learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1061–1069, 2012.

[35] Jiang Wang, Thomas Leung, Chuck Rosenberg, Jinbin Wang, James Philbin, Bo Chen, Ying Wu, et al. Learning fine-grained image similarity with deep ranking. *arXiv preprint arXiv:1404.4661*, 2014.

[36] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142. ACM, 2002.

[37] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the International Conference on Machine Learning*, pages 89–96. ACM, 2005.

[38] Louis L Thurstone. A law of comparative judgment. *Psychological review*, 34(4):273, 1927.

[39] Huiwen Chang, Fisher Yu, Jue Wang, Douglas Ashley, and Adam Finkelstein. Automatic triage for a photo series. *ACM Transactions on Graphics (TOG)*, 35(4):148, 2016.

[40] Abhimanyu Dubey, Nikhil Naik, Devi Parikh, Ramesh Raskar, and César A Hidalgo. Deep learning the city: Quantifying urban perception at a global scale. In *European Conference on Computer Vision*, pages 196–212. Springer, 2016.

[41] Hazel Doughty, Dima Damen, and Walterio Mayol-Cuevas. Who's better? who's best? pairwise deep ranking for skill determination. *arXiv:1703.09913*, 2017.

[42] David Sculley. Combined regression and ranking. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 979–988. ACM, 2010.

[43] Lin Chen, Peng Zhang, and Baoxin Li. Fusing pointwise and pairwise labels for supporting user-adaptive image retrieval. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 67–74. ACM, 2015.

[44] Hiroya Takamura and Jun'ichi Tsujii. Estimating numerical attributes by bringing together fragmentary clues. In *HLT-NAACL*, pages 1305–1310, 2015.

[45] Yilin Wang, Suhang Wang, Jiliang Tang, Huan Liu, and Baoxin Li. PPP: Joint pointwise and pairwise image label prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6005–6013, 2016.

[46] Wei-Tse Sun, Ting-Hsuan Chao, Yin-Hsi Kuo, and Winston H Hsu. Photo filter recommendation by category-aware aesthetic learning. *IEEE Transactions on Multimedia*, 19(8):1870–1880, 2017.

[47] Rongfu Mao, Haichao Zhu, Linke Zhang, and Aizhi Chen. A new method to assist small data set neural network learning. In *Intelligent Systems Design and Applications, 2006. ISDA'06. Sixth International Conference on*, volume 1, pages 17–22. IEEE, 2006.

[48] Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.

[49] Xiaofeng Zhang, Zhangyang Wang, Dong Liu, and Qing Ling. Dada: Deep adversarial data augmentation for extremely low data regime classification. *arXiv preprint arXiv:1809.00981*, 2018.

[50] Søren Hauberg, Oren Freifeld, Anders Boesen Lindbo Larsen, John Fisher, and Lars Hansen. Dreaming more data: Class-dependent distributions over diffeomorphisms for learned data augmentation. In *Artificial Intelligence and Statistics*, pages 342–350, 2016.

[51] Bo Liu, Ying Wei, Yu Zhang, and Qiang Yang. Deep neural networks for high dimension, low sample size data. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2287–2293, 2017.

[52] Rohit Keshari, Mayank Vatsa, Richa Singh, and Afzel Noore. Learning structure and strength of cnn filters for small sample size training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9349–9358, 2018.

[53] Amarjot Singh and Nick Kingsbury. Efficient convolutional network learning using parametric log based dual-tree wavelet scatternet. In *Computer Vision Workshop (ICCVW), 2017 IEEE International Conference on*, pages 1140–1147. IEEE, 2017.

[54] R Duncan Luce. *Individual choice behavior: A theoretical analysis*. Courier Corporation, 2012.

[55] Peter B Imrey. Bradley–Terry model. *Encyclopedia of Biostatistics*, 1998.

[56] MIT Media Lab. GIFGIF. `http://gifgif.media.mit.edu`. Accessed: 2018-12-01.

[57] Glen A Gole, Anna L Ells, Ximena Katz, G Holmstrom, Alistair R Fielder, A Capone Jr, John T Flynn, William G Good, Jonathan M Holmes, JA McNamara, et al. The international classification of retinopathy of prematurity revisited. *JAMA Ophthalmology*, 123(7):991–999, 2005.

[58] James M Brown, J Peter Campbell, Andrew Beers, Ken Chang, Susan Ostmo, RV Paul Chan, Jennifer Dy, Deniz Erdogmus, Stratis Ioannidis, and Jayashree Kalpathy-Cramer. Automated diagnosis of plus disease in retinopathy of prematurity using deep convolutional neural networks. *JAMA Ophthalmology*, 2018.

[59] Michael C Ryan, Susan Ostmo, Karyn Jonas, Audina Berrocal, Kimberly Drenser, Jason Horowitz, Thomas C Lee, Charles Simmons, Maria-Ana Martinez-Castellanos, RV Paul Chan, et al. Development and evaluation of reference standards for image-based telemedicine diagnosis and clinical research studies in ophthalmology. In *AMIA Annual Symposium Proceedings*, volume 2014, page 1902. American Medical Informatics Association, 2014.

[60] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[61] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[62] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015.

[63] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[64] James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36, 1982.

[65] Shlomo S Sawilowsky. Fermat, schubert, einstein, and behrens-fisher: The probable difference between two means when $\sigma_1^2 \neq \sigma_2^2$. *Journal of Modern Applied Statistical Methods*, 1(2):55, 2002.

[66] Yuan Guo, Peng Tian, Jayashree Kalpathy-Cramer, Susan Ostmo, J Peter Campbell, Michael F Chiang, Deniz Erdogmus, Jennifer G Dy, and Stratis Ioannidis. Experimental design under the bradley-terry model. In *IJCAI*, pages 2198–2204, 2018.

[67] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[68] Xin Lu, Zhe Lin, Hailin Jin, Jianchao Yang, and James Z Wang. Rapid: Rating pictorial aesthetics using deep learning. In *Proceedings of the 22nd ACM International Conference on Multimedia*, pages 457–466. ACM, 2014.

A Multi-Class Absolute Labels

Our design naturally extends to multi-class absolute labels. In this setting, we consider one-hot encoded vectors $y_i \in [0,1]^K \ \forall i$ as absolute labels, where K is the number of classes. When item i belongs to class $k \in \{1, ..., K\}$, only the i^{th} element of y_i is 1, and the rest of its elements are 0. Recall that when class labels are ordered, absolute and comparison labels are coupled (c.f. Section 3). This implies for multi-class absolute labels, that given items (i, j) with absolute labels (k_i, k_j), $y_{(i,j)} = +1$ indicates that $k_i > k_j$ is more likely than $k_j > k_i$.

When trained on multi-class absolute labels, the absolute network takes the form $f_a : \mathbb{R}^d \to [0,1]^K$. Thus, the additional network linking the base network output to absolute labels becomes $h_a : \mathbb{R}^p \to [0,1]^K$ (c.f.Fig. 1). In our experiments with multi-class absolute labels (c.f. ROP in Section 4), we choose h_a as a single fully connected layer with softmax activation. We train f_a using classical loss functions L_a, such as multi-class cross-entropy and multi-class hinge loss. Further details on multi-class classification can be found in [67].

B Plackett-Luce Model

Rankings frequently occur in real-life applications, including, i.e. medicine and recommender systems. In top-1 ranking settings, given a subset of N alternative items, a labeler chooses the item they prefer the most. Formally, a labeler produces a top-1 ranking label of the form (w, A), where $A \subseteq \{1, ..., N\}$ is a set of alternative items and $w \in A$ is the chosen item. Given a top-1 ranking dataset, the total order of N items can be learned through the Plackett-Luce (PL) model [54]. PL model, being the generalization of the Bradley-Terry (BT) model to rankings, asserts that every item i in the ranking dataset is parametrized by a score s_i. Then, all labelling events are independent of each other, where the probability of choosing item w over the alternatives in set A is:

$$P(w \,|A) = \frac{s_w}{\sum_{i \in A} s_i}. \tag{10}$$

Another common setting is when labelers provide a total ranking over the alternatives, instead of a single choice. More precisely, a labeler produces a total ranking label of the form $\big(\sigma_A, A\big)$, where σ_A is a permutation of the alternative items in A. In fact, a total ranking is equivalent to a sequence of $|A| - 1$ independent choices, where $|A|$ is the size of set A: a labeler starts by choosing the first item $\sigma_A(1)$ out of A, then chooses the second item $\sigma_A(2)$ out of $A \setminus \{\sigma_A(1)\}$, then chooses the third item $\sigma_A(3)$ out of $A \setminus \{\sigma_A(1), \sigma_A(2)\}$, and so on. Hence, under the PL model, probability of a total ranking over the alternatives in set A becomes:

$$P(\sigma_A \,|A) = \prod_{r=1}^{|A|-1} \frac{s_{\sigma_A(r)}}{\sum_{p=r}^{|A|} s_{\sigma_A(p)}}. \tag{11}$$

Given $\{s_1, \cdots, s_N\}$, all ranking events are fully determined by Eq. (11).

Recall that our architecture introduces a generic representation of the score s_i through linear comparison models: we assume that $s_i = f_c(\boldsymbol{x}_i) \ \forall i \in \{1, \ldots, N\}$, where f_c is the comparison network. Hence, our formulation naturally generalizes to learning from rankings under PL model. To implement this generalization, given a total ranking $\big(\sigma_A, A\big)$ with the corresponding feature vectors $\{\boldsymbol{x}_i\}_{i \in A}$, we can train our architecture by adopting the (negative log-likelihood) loss function:

$$\min_{\boldsymbol{W}_b, \boldsymbol{W}_c} \quad \sum_{r=1}^{|A|-1} \left(\log \Big(\sum_{p=r}^{|A|} f_c(\boldsymbol{x}_{\sigma_A(p)}; \boldsymbol{W}_c, \boldsymbol{W}_b) \Big) - \log \Big(f_c(\boldsymbol{x}_{\sigma_A(r)}; \boldsymbol{W}_c, \boldsymbol{W}_b) \Big) \right), \tag{12}$$

instead of L_c in our combined loss in Eq. (3).

C Sun et.al. 's Method

[46] leverage a siamese network to learn comparisons, where the base network architecture is either AlexNet [3] or RAPID net [68] with a fully-connected output layer. The base network $f : \mathbb{R}^d \to \mathbb{R}^{128}$ receives an input image and generates the corresponding 128-dimensional feature vector. To regress the comparison labels from these features, [46] employ a norm comparison loss function:

$$L_c(\boldsymbol{x}_i, \boldsymbol{x}_j, y_{(i,j)}) = - \sum_{(i,j,y_{(i,j)}) \in D_c} y_{(i,j)} \big(\|f(\boldsymbol{x}_i)\|_2^2 - \|f(\boldsymbol{x}_j)\|_2^2 \big). \tag{13}$$

To evaluate the performance of their architecture, for each image i, they use $\|f(\boldsymbol{x}_i)\|_2^2$ as the quality score.

To compare the performance of our method with [46]'s, we adapt our experimental setup as follows: we transfer the features extracted by our base network, i.e. GoogLeNet [5], to a fully-connected layer with 128-d output. We train the resulting architecture with our combined loss function (c.f. Eq. (3)), where L_a is the optimal absolute loss function for each dataset (c.f. Tables 4 to 11), and L_c is the norm comparison loss. For each $\alpha \in \{0.0, 0.5, 1.0\}$, we optimize the learning rate ranging from 10^{-6} to 10^{-2} and λ ranging from 2×10^{-4} to 2×10^{-2} on the validation set. For this method, we also normalize the input images to aid convergence. Finally, we evaluate the test set predictions of the optimal models obtained through this setup, using $s_i = \|f(\boldsymbol{x}_i)\|_2^2$ as the quality score for each image i.

D Logistic Regression and SVMs

For both logistic and SVM methods, we employ linear comparison models (c.f. Sec. 3.2) by assuming that the score of each item i, i.e. s_i, is parametrized by a common parameter vector $\boldsymbol{\beta} \in \Re^p$ and a common bias $b \in \Re$, such that $s_i = \boldsymbol{\beta}^T f_b(\boldsymbol{x}_i) + b$. In other words, we assign $h_c(\cdot) = h_a(\cdot) = \boldsymbol{\beta}^T \cdot + b$ in Eq.(1) and Eq.(2). Then, absolute and comparison label predictions become $\hat{y}_i = \boldsymbol{\beta}^T f_b(\boldsymbol{x}_i) + b$ and $\hat{y}_{(i,j)} = \boldsymbol{\beta}^T \big(f_b(\boldsymbol{x}_i) - f_b(\boldsymbol{x}_j)\big)$, respectively. In the exposition below, as well as in our experiments in Sec. 4.2, f_b is fixed, i.e. not trained, and given by GoogLeNet with weights pre-trained on the ImageNet dataset [61].

Under logistic method, maximum aposteriori (MAP) estimates of $\boldsymbol{\beta}$ and b from absolute and comparison labels correspond to minimizing the negative log-likelihood functions $L_a(y_i, \hat{y}_i) = \log(1+e^{-y_i \hat{y}_i})$ and $L_c(y_{(i,j)}, \hat{y}_{(i,j)}) = \log(1+e^{-y_{(i,j)} \hat{y}_{(i,j)}})$, respectively. Motivated by the combined loss function Eq.(3), we estimate $\boldsymbol{\beta}$ and b by solving:

$$\begin{aligned} \min_{\boldsymbol{\beta}, b} \quad & \alpha \sum_{(i,y_i) \in D_a} L_a\big(\boldsymbol{x}_i, y_i, \boldsymbol{\beta}^T f_b(\boldsymbol{x}_i) + b\big) \\ & + (1-\alpha) \sum_{(i,j,y_{(i,j)}) \in D_c} L_c\Big(\boldsymbol{x}_i, \boldsymbol{x}_j, y_{(i,j)}, \boldsymbol{\beta}^T \big(f_b(\boldsymbol{x}_i) - f_b(\boldsymbol{x}_j)\big)\Big) + \lambda \|\boldsymbol{\beta}\|_2, \end{aligned} \tag{14}$$

where λ is the regularization parameter. Similarly, under SVM method, we estimate $\boldsymbol{\beta}$ and b by solving the convex program:

$$\begin{aligned} \min_{\boldsymbol{\beta}, b} \quad & \alpha \sum_{(i,y_i) \in D_a} \epsilon_i + (1-\alpha) \sum_{(i,j,y_{(i,j)}) \in D_c} \epsilon_{(i,j)} + \lambda \|\boldsymbol{\beta}\|_2 \\ \text{subject to:} \quad & y_i\big(\boldsymbol{\beta}^T f_b(\boldsymbol{x}_i) + b\big) \geq 1 - \epsilon_i, \\ & \epsilon_i \geq 0 \qquad \forall (i, y_i) \in D_a, \\ & y_{(i,j)}\Big(\boldsymbol{\beta}^T \big(f_b(\boldsymbol{x}_i) - f_b(\boldsymbol{x}_j)\big)\Big) \geq 1 - \epsilon_{(i,j)}, \\ & \epsilon_{(i,j)} \geq 0 \quad \forall (i, j, y_{(i,j)}) \in D_c. \end{aligned} \tag{15}$$

We optimize the regularization parameter λ (ranging from 10^{-6} to 10^4) on the validation set of each dataset.

E Robustness to Metric Selection

In this section, we illustrate that our model is robust in the selection of a metric to optimize over the validation set. We employ the experimental setup explained in Section 4.1 to train and evaluate our model on these datasets. Tables 4 to 11 report the test set performance of the models optimized on the validation set for each metric, over all eight metrics. These four tables correspond to GIFGIF Happiness, GIFGIF Pleasure, ROP, and FAC datasets, respectively. Note that the diagonals of each table correspond to the rows of Tables 2a and 2b. We conclude that optimizing hyperparameters w.r.t. a metric, e.g., AUC, still results in high performance w.r.t. other metrics, e.g., Acc., and F1.

	α	Hyperparameters Optimized on Validation Set				Performance Metrics on Test Set			
		L_a	L_c	λ	L.R.	AUC on D_a	Acc. on D_a	F1 on D_a	PRAUC on D_a
AUC on D_a	0.0	C.E.	T.	0.002	0.001	0.915 ± 0.041	0.832 ± 0.044	0.783 ± 0.048	0.824 ± 0.057
	1.0	C.E.	B.T.	0.002	0.001	0.561 ± 0.072	0.638 ± 0.056	0.287 ± 0.053	0.401 ± 0.07
	Best (0.0)	C.E.	T.	0.002	0.001	$\mathbf{0.915 \pm 0.041}$	0.832 ± 0.044	0.783 ± 0.048	0.824 ± 0.057
Acc. on D_a	0.0	C.E.	B.T.	0.002	0.001	0.883 ± 0.047	0.85 ± 0.042	0.789 ± 0.048	0.762 ± 0.064
	1.0	C.E.	B.T.	0.02	0.001	0.504 ± 0.072	0.664 ± 0.055	0.0 ± 0.0	0.336 ± 0.066
	Best (0.0)	C.E.	B.T.	0.002	0.001	0.883 ± 0.047	$\mathbf{0.85 \pm 0.042}$	0.789 ± 0.048	0.762 ± 0.064
F1 on D_a	0.0	C.E.	T.	0.0002	0.01	0.925 ± 0.038	0.824 ± 0.045	0.779 ± 0.049	0.848 ± 0.054
	1.0	C.E.	B.T.	0.0002	0.001	0.519 ± 0.072	0.613 ± 0.057	0.39 ± 0.057	0.383 ± 0.069
	Best (0.75)	C.E.	T.	0.0002	0.01	0.888 ± 0.046	0.85 ± 0.042	$\mathbf{0.791 \pm 0.048}$	0.764 ± 0.064
PRAUC on D_a	0.0	C.E.	T.	0.002	0.001	0.916 ± 0.042	0.833 ± 0.045	0.784 ± 0.049	0.824 ± 0.057
	1.0	C.E.	B.T.	0.002	0.001	0.562 ± 0.073	0.639 ± 0.057	0.288 ± 0.054	0.401 ± 0.07
	Best (0.0)	C.E.	T.	0.002	0.001	0.916 ± 0.042	0.833 ± 0.045	0.784 ± 0.049	$\mathbf{0.824 \pm 0.057}$
AUC on D_c	0.0	C.E.	T.	0.0002	0.001	0.917 ± 0.04	0.835 ± 0.043	0.782 ± 0.048	0.803 ± 0.06
	1.0	C.E.	B.T.	0.0002	0.001	0.487 ± 0.072	0.565 ± 0.058	0.32 ± 0.055	0.383 ± 0.069
	Best (0.5)	C.E.	T.	0.002	0.001	0.9 ± 0.044	0.828 ± 0.044	0.77 ± 0.049	0.792 ± 0.061
Acc. on D_c	0.0	C.E.	T.	0.002	0.001	0.915 ± 0.041	0.832 ± 0.044	0.783 ± 0.048	0.824 ± 0.057
	1.0	C.E.	B.T.	0.0002	0.001	0.487 ± 0.072	0.565 ± 0.058	0.32 ± 0.055	0.401 ± 0.07
	Best (0.25)	C.E.	T.	0.0002	0.01	0.915 ± 0.041	0.843 ± 0.043	0.786 ± 0.048	0.815 ± 0.058
F1 on D_c	0.0	C.E.	T.	0.002	0.001	0.915 ± 0.041	0.832 ± 0.044	0.783 ± 0.048	0.824 ± 0.057
	1.0	C.E.	B.T.	0.0002	0.001	0.487 ± 0.072	0.565 ± 0.058	0.32 ± 0.055	0.383 ± 0.069
	Best (0.0)	C.E.	T.	0.002	0.001	0.915 ± 0.041	0.832 ± 0.044	0.783 ± 0.048	0.824 ± 0.057
PRAUC on D_c	0.0	C.E.	T.	0.0002	0.001	0.918 ± 0.041	0.836 ± 0.044	0.783 ± 0.049	0.803 ± 0.06
	1.0	H.	B.T.	0.0002	0.01	0.466 ± 0.072	0.665 ± 0.056	0.0 ± 0.0	0.308 ± 0.063
	Best (0.0)	C.E.	T.	0.0002	0.001	0.918 ± 0.041	0.836 ± 0.044	0.783 ± 0.049	0.803 ± 0.06

Table 4: Absolute label prediction performance on the GIFGIF Happiness dataset. For each α, we find the optimal absolute loss function (L_a), comparison loss function (L_c), regularization parameter λ, and learning rate (L.R.). We consider cross-entropy (C.E), hinge (H.), Bradley-Terry (B.T.), and Thurstone (T.) as loss functions. We repeat this optimization for AUC, accuracy (Acc.), F1 score, and PRAUC metrics on the absolute (D_a) and the comparison labels (D_c) of the validation set. Accordingly, each row triplet corresponds to the metric on which we optimize the hyperparameters. We then report all eight metrics on the test set, for training with $\alpha = 0.0$ (comparison labels only), $\alpha = 1.0$ (absolute labels only), and best performing $\alpha \in [0, 1]$.

	α	Hyperparameters Optimized on Validation Set				Performance Metrics on Test Set			
		L_a	L_c	λ	L.R.	AUC on D_c	Acc. on D_c	F1 on D_c	PRAUC on D_c
AUC on D_a	0.0	C.E.	T.	0.002	0.001	0.909 ± 0.029	0.828 ± 0.036	0.819 ± 0.037	0.917 ± 0.029
	1.0	C.E.	B.T.	0.002	0.001	0.461 ± 0.055	0.476 ± 0.048	0.443 ± 0.048	0.491 ± 0.057
	Best (0.0)	C.E.	T.	0.002	0.001	0.909 ± 0.029	0.828 ± 0.036	0.819 ± 0.037	0.917 ± 0.029
Acc. on D_a	0.0	C.E.	B.T.	0.002	0.001	0.896 ± 0.031	0.805 ± 0.038	0.797 ± 0.039	0.895 ± 0.032
	1.0	C.E.	B.T.	0.02	0.001	0.508 ± 0.056	0.496 ± 0.048	0.22 ± 0.04	0.499 ± 0.057
	Best (0.0)	C.E.	B.T.	0.002	0.001	0.896 ± 0.031	0.805 ± 0.038	0.797 ± 0.039	0.895 ± 0.032
F1 on D_a	0.0	C.E.	T.	0.0002	0.01	0.913 ± 0.028	0.832 ± 0.036	0.83 ± 0.036	0.912 ± 0.03
	1.0	C.E.	B.T.	0.0002	0.001	0.51 ± 0.056	0.493 ± 0.048	0.477 ± 0.048	0.516 ± 0.057
	Best (0.75)	C.E.	T.	0.0002	0.01	0.892 ± 0.032	0.81 ± 0.038	0.807 ± 0.038	0.898 ± 0.032
PRAUC on D_a	0.0	C.E.	T.	0.002	0.001	0.91 ± 0.03	0.829 ± 0.037	0.82 ± 0.038	0.917 ± 0.029
	1.0	C.E.	B.T.	0.002	0.001	0.462 ± 0.056	0.477 ± 0.049	0.444 ± 0.049	0.491 ± 0.057
	Best (0.0)	C.E.	T.	0.002	0.001	0.91 ± 0.03	0.829 ± 0.037	0.82 ± 0.038	0.917 ± 0.029
AUC on D_c	0.0	C.E.	T.	0.0002	0.001	0.898 ± 0.031	0.813 ± 0.037	0.808 ± 0.038	0.906 ± 0.031
	1.0	C.E.	B.T.	0.0002	0.001	0.419 ± 0.055	0.42 ± 0.047	0.397 ± 0.047	0.516 ± 0.057
	Best (0.5)	C.E.	T.	0.002	0.001	$\mathbf{0.902 \pm 0.03}$	0.813 ± 0.037	0.81 ± 0.038	0.893 ± 0.033
Acc. on D_c	0.0	C.E.	T.	0.002	0.001	0.909 ± 0.029	0.828 ± 0.036	0.819 ± 0.037	0.917 ± 0.029
	1.0	C.E.	B.T.	0.0002	0.001	0.419 ± 0.055	0.42 ± 0.047	0.397 ± 0.047	0.491 ± 0.057
	Best (0.25)	C.E.	T.	0.0002	0.01	0.911 ± 0.029	$\mathbf{0.83 \pm 0.036}$	0.828 ± 0.036	0.904 ± 0.031
F1 on D_c	0.0	C.E.	T.	0.002	0.001	0.909 ± 0.029	0.828 ± 0.036	0.819 ± 0.037	0.917 ± 0.029
	1.0	C.E.	B.T.	0.0002	0.001	0.419 ± 0.055	0.42 ± 0.047	0.397 ± 0.047	0.516 ± 0.057
	Best (0.0)	C.E.	T.	0.002	0.001	0.909 ± 0.029	0.828 ± 0.036	$\mathbf{0.819 \pm 0.037}$	0.917 ± 0.029
PRAUC on D_c	0.0	C.E.	T.	0.0002	0.001	0.899 ± 0.032	0.814 ± 0.038	0.809 ± 0.039	0.906 ± 0.031
	1.0	H.	B.T.	0.0002	0.01	0.445 ± 0.056	0.465 ± 0.049	0.442 ± 0.049	0.448 ± 0.056
	Best (0.0)	C.E.	T.	0.0002	0.001	0.899 ± 0.032	0.814 ± 0.038	0.809 ± 0.039	$\mathbf{0.906 \pm 0.031}$

Table 5: Comparison label prediction performance on the GIFGIF Happiness dataset. For each α, we find the optimal absolute loss function (L_a), comparison loss function (L_c), regularization parameter λ, and learning rate (L.R.). We consider cross-entropy (C.E), hinge (H.), Bradley-Terry (B.T.), and Thurstone (T.) as loss functions. We repeat this optimization for AUC, accuracy (Acc.), F1 score, and PRAUC metrics on the absolute (D_a) and the comparison labels (D_c) of the validation set. Accordingly, each row triplet corresponds to the metric on which we optimize the hyperparameters. We then report all eight metrics on the test set, for training with $\alpha = 0.0$ (comparison labels only), $\alpha = 1.0$ (absolute labels only), and best performing $\alpha \in [0, 1]$.

	α	Hyperparameters Optimized on Validation Set				Performance Metrics on Test Set			
		L_a	L_c	λ	L.R.	AUC on D_a	Acc. on D_a	F1 on D_a	PRAUC on D_a
AUC on D_a	0.0	C.E.	T.	0.002	0.001	0.887 ± 0.045	0.805 ± 0.052	0.819 ± 0.051	0.873 ± 0.047
	1.0	C.E.	B.T.	0.0002	0.001	0.579 ± 0.075	0.56 ± 0.065	0.557 ± 0.065	0.591 ± 0.075
	Best (0.0)	C.E.	T.	0.002	0.001	$\mathbf{0.887 \pm 0.045}$	0.805 ± 0.052	0.819 ± 0.051	0.873 ± 0.047
Acc. on D_a	0.0	C.E.	T.	0.002	0.001	0.887 ± 0.045	0.805 ± 0.052	0.819 ± 0.051	0.873 ± 0.047
	1.0	C.E.	B.T.	0.0002	0.001	0.579 ± 0.075	0.56 ± 0.065	0.557 ± 0.065	0.591 ± 0.075
	Best (0.25)	H.	T.	0.0002	0.01	0.865 ± 0.049	$\mathbf{0.814 \pm 0.051}$	0.813 ± 0.052	0.852 ± 0.051
F1 on D_a	0.0	C.E.	T.	0.002	0.001	0.887 ± 0.045	0.805 ± 0.052	0.819 ± 0.051	0.873 ± 0.047
	1.0	C.E.	B.T.	0.002	0.001	0.616 ± 0.074	0.516 ± 0.066	0.667 ± 0.062	0.633 ± 0.073
	Best (0.5)	H.	T.	0.0002	0.001	0.853 ± 0.051	0.814 ± 0.051	$\mathbf{0.823 \pm 0.05}$	0.827 ± 0.055
PRAUC on D_a	0.0	C.E.	T.	0.002	0.001	0.887 ± 0.045	0.805 ± 0.052	0.819 ± 0.051	0.873 ± 0.047
	1.0	C.E.	B.T.	0.0002	0.001	0.579 ± 0.075	0.56 ± 0.065	0.557 ± 0.065	0.591 ± 0.075
	Best (0.25)	H.	T.	0.0002	0.001	0.9 ± 0.042	0.832 ± 0.049	0.826 ± 0.05	$\mathbf{0.88 \pm 0.046}$
AUC on D_c	0.0	C.E.	T.	0.0002	0.01	0.886 ± 0.045	0.809 ± 0.052	0.806 ± 0.052	0.874 ± 0.047
	1.0	H.	B.T.	0.0002	0.01	0.595 ± 0.074	0.507 ± 0.066	0.673 ± 0.062	0.599 ± 0.074
	Best (0.25)	H.	T.	0.0002	0.01	0.865 ± 0.049	0.814 ± 0.051	0.813 ± 0.052	0.852 ± 0.051
Acc. on D_c	0.0	C.E.	T.	0.002	0.001	0.887 ± 0.045	0.805 ± 0.052	0.819 ± 0.051	0.873 ± 0.047
	1.0	H.	B.T.	0.0002	0.01	0.595 ± 0.074	0.507 ± 0.066	0.673 ± 0.062	0.599 ± 0.074
	Best (0.25)	C.E.	T.	0.0002	0.01	0.865 ± 0.049	0.814 ± 0.051	0.813 ± 0.052	0.852 ± 0.051
F1 on D_c	0.0	C.E.	T.	0.002	0.001	0.887 ± 0.045	0.805 ± 0.052	0.819 ± 0.051	0.873 ± 0.047
	1.0	H.	B.T.	0.002	0.001	0.615 ± 0.074	0.56 ± 0.065	0.686 ± 0.061	0.599 ± 0.074
	Best (0.25)	C.E.	T.	0.0002	0.01	0.865 ± 0.049	0.814 ± 0.051	0.813 ± 0.052	0.852 ± 0.051
PRAUC on D_c	0.0	C.E.	T.	0.0002	0.01	0.886 ± 0.045	0.809 ± 0.052	0.806 ± 0.052	0.874 ± 0.047
	1.0	H.	B.T.	0.002	0.001	0.615 ± 0.074	0.56 ± 0.065	0.686 ± 0.061	0.599 ± 0.074
	Best (0.25)	H.	T.	0.0002	0.01	0.865 ± 0.049	0.814 ± 0.051	0.813 ± 0.052	0.852 ± 0.051

Table 6: Absolute label prediction performance on the GIFGIF Pleasure dataset. For each α, we find the optimal absolute loss function (L_a), comparison loss function (L_c), regularization parameter λ, and learning rate (L.R.). We consider cross-entropy (C.E), hinge (H.), Bradley-Terry (B.T.), and Thurstone (T.) as loss functions. We repeat this optimization for AUC, accuracy (Acc.), F1 score, and PRAUC metrics on the absolute (D_a) and the comparison labels (D_c) of the validation set. Accordingly, each row triplet corresponds to the metric on which we optimize the hyperparameters. We then report all eight metrics on the test set, for training with $\alpha = 0.0$ (comparison labels only), $\alpha = 1.0$ (absolute labels only), and best performing $\alpha \in [0, 1]$.

	α	Hyperparameters Optimized on Validation Set				Performance Metrics on Test Set			
		L_a	L_c	λ	L.R.	AUC on D_c	Acc. on D_c	F1 on D_c	PRAUC on D_c
AUC on D_a	0.0	C.E.	T.	0.002	0.001	0.835 ± 0.053	0.729 ± 0.058	0.742 ± 0.057	0.856 ± 0.049
	1.0	C.E.	B.T.	0.0002	0.001	0.53 ± 0.076	0.509 ± 0.065	0.538 ± 0.065	0.601 ± 0.074
	Best (0.0)	C.E.	T.	0.002	0.001	0.835 ± 0.053	0.729 ± 0.058	0.742 ± 0.057	0.856 ± 0.049
Acc. on D_a	0.0	C.E.	T.	0.002	0.001	0.835 ± 0.053	0.729 ± 0.058	0.742 ± 0.057	0.856 ± 0.049
	1.0	C.E.	B.T.	0.0002	0.001	0.53 ± 0.076	0.509 ± 0.065	0.538 ± 0.065	0.601 ± 0.074
	Best (0.25)	H.	T.	0.0002	0.01	0.84 ± 0.052	0.755 ± 0.056	0.775 ± 0.055	0.855 ± 0.049
F1 on D_a	0.0	C.E.	T.	0.002	0.001	0.835 ± 0.053	0.729 ± 0.058	0.742 ± 0.057	0.856 ± 0.049
	1.0	C.E.	B.T.	0.002	0.001	0.597 ± 0.074	0.593 ± 0.064	0.627 ± 0.063	0.613 ± 0.073
	Best (0.5)	H.	T.	0.0002	0.001	0.832 ± 0.053	0.733 ± 0.058	0.754 ± 0.056	0.836 ± 0.052
PRAUC on D_a	0.0	C.E.	T.	0.002	0.001	0.835 ± 0.053	0.729 ± 0.058	0.742 ± 0.057	0.856 ± 0.049
	1.0	C.E.	B.T.	0.0002	0.001	0.53 ± 0.076	0.509 ± 0.065	0.538 ± 0.065	0.601 ± 0.074
	Best (0.25)	H.	T.	0.0002	0.001	0.828 ± 0.053	0.702 ± 0.06	0.715 ± 0.059	0.843 ± 0.051
AUC on D_c	0.0	C.E.	T.	0.0002	0.01	0.836 ± 0.052	0.742 ± 0.057	0.764 ± 0.056	0.849 ± 0.05
	1.0	H.	B.T.	0.0002	0.01	0.531 ± 0.076	0.492 ± 0.065	0.463 ± 0.065	0.618 ± 0.073
	Best (0.25)	H.	T.	0.0002	0.01	$\mathbf{0.84 \pm 0.052}$	0.755 ± 0.056	0.775 ± 0.055	0.855 ± 0.049
Acc. on D_c	0.0	C.E.	T.	0.002	0.001	0.835 ± 0.053	0.729 ± 0.058	0.742 ± 0.057	0.856 ± 0.049
	1.0	H.	B.T.	0.0002	0.01	0.531 ± 0.076	0.492 ± 0.065	0.463 ± 0.065	0.618 ± 0.073
	Best (0.25)	C.E.	T.	0.0002	0.01	0.84 ± 0.052	$\mathbf{0.755 \pm 0.056}$	0.775 ± 0.055	0.855 ± 0.049
F1 on D_c	0.0	C.E.	T.	0.002	0.001	0.835 ± 0.053	0.729 ± 0.058	0.742 ± 0.057	0.856 ± 0.049
	1.0	H.	B.T.	0.002	0.001	0.631 ± 0.072	0.615 ± 0.064	0.662 ± 0.062	0.618 ± 0.073
	Best (0.25)	C.E.	T.	0.0002	0.01	0.84 ± 0.052	0.755 ± 0.056	$\mathbf{0.775 \pm 0.055}$	0.855 ± 0.049
PRAUC on D_c	0.0	C.E.	T.	0.0002	0.01	0.836 ± 0.052	0.742 ± 0.057	0.764 ± 0.056	0.849 ± 0.05
	1.0	H.	B.T.	0.002	0.001	0.631 ± 0.072	0.615 ± 0.064	0.662 ± 0.062	0.618 ± 0.073
	Best (0.25)	H.	T.	0.0002	0.01	0.84 ± 0.052	0.755 ± 0.056	0.775 ± 0.055	$\mathbf{0.855 \pm 0.049}$

Table 7: Comparison label prediction performance on the GIFGIF Pleasure dataset. For each α, we find the optimal absolute loss function (L_a), comparison loss function (L_c), regularization parameter λ, and learning rate (L.R.). We consider cross-entropy (C.E), hinge (H.), Bradley-Terry (B.T.), and Thurstone (T.) as loss functions. We repeat this optimization for AUC, accuracy (Acc.), F1 score, and PRAUC metrics on the absolute (D_a) and the comparison labels (D_c) of the validation set. Accordingly, each row triplet corresponds to the metric on which we optimize the hyperparameters. We then report all eight metrics on the test set, for training with $\alpha = 0.0$ (comparison labels only), $\alpha = 1.0$ (absolute labels only), and best performing $\alpha \in [0, 1]$.

	α	Hyperparameters Optimized on Validation Set				Performance Metrics on Test Set			
		L_a	L_c	λ	L.R.	AUC on D_a	Acc. on D_a	F1 on D_a	PRAUC on D_a
AUC on D_a	0.0	C.E.	B.T.	0.02	0.001	0.909 ± 0.013	0.824 ± 0.011	0.624 ± 0.013	0.731 ± 0.019
	1.0	C.E.	B.T.	0.002	0.01	0.835 ± 0.017	0.824 ± 0.011	0.0 ± 0.0	0.528 ± 0.021
	Best (0.75)	C.E.	B.T.	0.02	0.001	$\mathbf{0.914 \pm 0.013}$	0.883 ± 0.009	0.622 ± 0.013	0.736 ± 0.019
Acc. on D_a	0.0	C.E.	B.T.	0.02	0.001	0.909 ± 0.013	0.824 ± 0.011	0.624 ± 0.013	0.731 ± 0.019
	1.0	C.E.	B.T.	0.0002	0.01	0.87 ± 0.015	0.705 ± 0.012	0.517 ± 0.014	0.549 ± 0.021
	Best (0.75)	C.E.	B.T.	0.02	0.001	0.914 ± 0.013	$\mathbf{0.883 \pm 0.009}$	0.622 ± 0.013	0.736 ± 0.019
F1 on D_a	0.0	C.E.	B.T.	0.02	0.001	0.909 ± 0.013	0.824 ± 0.011	0.624 ± 0.013	0.731 ± 0.019
	1.0	C.E.	B.T.	0.0002	0.01	0.87 ± 0.015	0.705 ± 0.012	0.517 ± 0.014	0.549 ± 0.021
	Best (0.5)	C.E.	B.T.	0.02	0.001	0.909 ± 0.013	0.87 ± 0.009	$\mathbf{0.64 \pm 0.013}$	0.731 ± 0.019
PRAUC on D_a	0.0	C.E.	B.T.	0.02	0.001	0.913 ± 0.013	0.822 ± 0.011	0.625 ± 0.013	0.731 ± 0.019
	1.0	C.E.	B.T.	0.02	0.01	0.5 ± 0.02	0.177 ± 0.011	0.301 ± 0.013	0.177 ± 0.012
	Best (0.0)	C.E.	B.T.	0.02	0.001	0.913 ± 0.013	0.822 ± 0.011	0.625 ± 0.013	$\mathbf{0.731 \pm 0.019}$
AUC on D_c	0.0	C.E.	B.T.	0.02	0.001	0.909 ± 0.013	0.824 ± 0.011	0.624 ± 0.013	0.731 ± 0.019
	1.0	C.E.	B.T.	0.002	0.01	0.835 ± 0.017	0.824 ± 0.011	0.0 ± 0.0	0.528 ± 0.021
	Best (0.0)	C.E.	B.T.	0.02	0.001	0.909 ± 0.013	0.824 ± 0.011	0.624 ± 0.013	0.731 ± 0.019
Acc. on D_c	0.0	C.E.	B.T.	0.02	0.001	0.909 ± 0.013	0.824 ± 0.011	0.624 ± 0.013	0.731 ± 0.019
	1.0	C.E.	B.T.	0.002	0.01	0.835 ± 0.017	0.824 ± 0.011	0.0 ± 0.0	0.528 ± 0.021
	Best (0.25)	C.E.	B.T.	0.02	0.001	0.908 ± 0.013	0.882 ± 0.009	0.616 ± 0.013	0.731 ± 0.019
F1 on D_c	0.0	C.E.	B.T.	0.02	0.001	0.909 ± 0.013	0.824 ± 0.011	0.624 ± 0.013	0.731 ± 0.019
	1.0	C.E.	B.T.	0.002	0.01	0.835 ± 0.017	0.824 ± 0.011	0.0 ± 0.0	0.528 ± 0.021
	Best (0.25)	C.E.	B.T.	0.02	0.001	0.908 ± 0.013	0.882 ± 0.009	0.616 ± 0.013	0.731 ± 0.019
PRAUC on D_c	0.0	C.E.	B.T.	0.02	0.001	0.913 ± 0.013	0.822 ± 0.011	0.625 ± 0.013	0.731 ± 0.019
	1.0	C.E.	B.T.	0.002	0.01	0.835 ± 0.017	0.177 ± 0.011	0.301 ± 0.013	0.528 ± 0.021
	Best (0.0)	C.E.	B.T.	0.02	0.001	0.913 ± 0.013	0.822 ± 0.011	0.625 ± 0.013	0.731 ± 0.019

Table 8: Absolute label prediction performance on the ROP dataset. For each α, we find the optimal absolute loss function (L_a), comparison loss function (L_c), regularization parameter λ, and learning rate (L.R.). We consider cross-entropy (C.E), hinge (H.), Bradley-Terry (B.T.), and Thurstone (T.) as loss functions. We repeat this optimization for AUC, accuracy (Acc.), F1 score, and PRAUC metrics on the absolute (D_a) and the comparison labels (D_c) of the validation set. Accordingly, each row triplet corresponds to the metric on which we optimize the hyperparameters. We then report all eight metrics on the test set, for training with $\alpha = 0.0$ (comparison labels only), $\alpha = 1.0$ (absolute labels only), and best performing $\alpha \in [0, 1]$.

	α	Hyperparameters Optimized on Validation Set				Performance Metrics on Test Set			
		L_a	L_c	λ	L.R.	AUC on D_c	Acc. on D_c	F1 on D_c	PRAUC on D_c
AUC on D_a	0.0	C.E.	B.T.	0.02	0.001	0.955 ± 0.013	0.882 ± 0.019	0.885 ± 0.019	0.963 ± 0.012
	1.0	C.E.	B.T.	0.002	0.01	0.941 ± 0.015	0.859 ± 0.021	0.865 ± 0.02	0.942 ± 0.015
	Best (0.75)	C.E.	B.T.	0.02	0.001	0.953 ± 0.013	0.869 ± 0.02	0.872 ± 0.02	0.957 ± 0.013
Acc. on D_a	0.0	C.E.	B.T.	0.02	0.001	0.955 ± 0.013	0.882 ± 0.019	0.885 ± 0.019	0.963 ± 0.012
	1.0	C.E.	B.T.	0.0002	0.01	0.89 ± 0.02	0.785 ± 0.024	0.785 ± 0.024	0.761 ± 0.028
	Best (0.75)	C.E.	B.T.	0.02	0.001	0.953 ± 0.013	0.869 ± 0.02	0.872 ± 0.02	0.957 ± 0.013
F1 on D_a	0.0	C.E.	B.T.	0.02	0.001	0.955 ± 0.013	0.882 ± 0.019	0.885 ± 0.019	0.963 ± 0.012
	1.0	C.E.	B.T.	0.0002	0.01	0.89 ± 0.02	0.785 ± 0.024	0.785 ± 0.024	0.761 ± 0.028
	Best (0.5)	C.E.	B.T.	0.02	0.001	0.955 ± 0.013	0.872 ± 0.02	0.875 ± 0.02	0.963 ± 0.012
PRAUC on D_a	0.0	C.E.	B.T.	0.02	0.001	0.959 ± 0.013	0.886 ± 0.019	0.89 ± 0.019	0.963 ± 0.012
	1.0	C.E.	B.T.	0.02	0.01	0.5 ± 0.034	0.475 ± 0.029	0.0 ± 0.0	0.526 ± 0.034
	Best (0.0)	C.E.	B.T.	0.02	0.001	0.959 ± 0.013	0.886 ± 0.019	0.89 ± 0.019	0.963 ± 0.012
AUC on D_c	0.0	C.E.	B.T.	0.02	0.001	0.955 ± 0.013	0.882 ± 0.019	0.885 ± 0.019	0.963 ± 0.012
	1.0	C.E.	B.T.	0.002	0.01	0.941 ± 0.015	0.859 ± 0.021	0.865 ± 0.02	0.942 ± 0.015
	Best (0.0)	C.E.	B.T.	0.02	0.001	$\mathbf{0.955 \pm 0.013}$	0.882 ± 0.019	0.885 ± 0.019	0.963 ± 0.012
Acc. on D_c	0.0	C.E.	B.T.	0.02	0.001	0.955 ± 0.013	0.882 ± 0.019	0.885 ± 0.019	0.963 ± 0.012
	1.0	C.E.	B.T.	0.002	0.01	0.941 ± 0.015	0.859 ± 0.021	0.865 ± 0.02	0.942 ± 0.015
	Best (0.25)	C.E.	B.T.	0.02	0.001	0.954 ± 0.013	$\mathbf{0.883 \pm 0.019}$	0.886 ± 0.019	0.963 ± 0.012
F1 on D_c	0.0	C.E.	B.T.	0.02	0.001	0.955 ± 0.013	0.882 ± 0.019	0.885 ± 0.019	0.963 ± 0.012
	1.0	C.E.	B.T.	0.002	0.01	0.941 ± 0.015	0.859 ± 0.021	0.865 ± 0.02	0.942 ± 0.015
	Best (0.25)	C.E.	B.T.	0.02	0.001	0.954 ± 0.013	0.883 ± 0.019	$\mathbf{0.886 \pm 0.019}$	0.963 ± 0.012
PRAUC on D_c	0.0	C.E.	B.T.	0.02	0.001	0.959 ± 0.013	0.886 ± 0.019	0.89 ± 0.019	0.963 ± 0.012
	1.0	C.E.	B.T.	0.002	0.01	0.941 ± 0.015	0.859 ± 0.021	0.865 ± 0.02	0.942 ± 0.015
	Best (0.0)	C.E.	B.T.	0.02	0.001	0.959 ± 0.013	0.886 ± 0.019	0.89 ± 0.019	$\mathbf{0.963 \pm 0.012}$

Table 9: Comparison label prediction performance on the ROP dataset. For each α, we find the optimal absolute loss function (L_a), comparison loss function (L_c), regularization parameter λ, and learning rate (L.R.). We consider cross-entropy (C.E), hinge (H.), Bradley-Terry (B.T.), and Thurstone (T.) as loss functions. We repeat this optimization for AUC, accuracy (Acc.), F1 score, and PRAUC metrics on the absolute (D_a) and the comparison labels (D_c) of the validation set. Accordingly, each row triplet corresponds to the metric on which we optimize the hyperparameters. We then report all eight metrics on the test set, for training with $\alpha = 0.0$ (comparison labels only), $\alpha = 1.0$ (absolute labels only), and best performing $\alpha \in [0, 1]$.

	α	Hyperparameters Optimized on Validation Set				Performance Metrics on Test Set			
		L_a	L_c	λ	L.R.	AUC on D_a	Acc. on D_a	F1 on D_a	PRAUC on D_a
AUC on D_a	0.0	C.E.	T.	0.0002	0.01	0.87 ± 0.028	0.82 ± 0.029	0.809 ± 0.03	0.82 ± 0.033
	1.0	H.	B.T.	0.002	0.01	0.746 ± 0.037	0.677 ± 0.035	0.659 ± 0.036	0.678 ± 0.041
	Best (0.0)	C.E.	T.	0.0002	0.01	$\mathbf{0.87 \pm 0.028}$	0.82 ± 0.029	0.809 ± 0.03	0.82 ± 0.033
Acc. on D_a	0.0	C.E.	T.	0.0002	0.01	0.87 ± 0.028	0.82 ± 0.029	0.809 ± 0.03	0.82 ± 0.033
	1.0	H.	B.T.	0.002	0.01	0.746 ± 0.037	0.677 ± 0.035	0.659 ± 0.036	0.678 ± 0.041
	Best (0.0)	C.E.	T.	0.0002	0.01	0.87 ± 0.028	$\mathbf{0.82 \pm 0.029}$	0.809 ± 0.03	0.82 ± 0.033
F1 on D_a	0.0	C.E.	T.	0.0002	0.01	0.87 ± 0.028	0.82 ± 0.029	0.809 ± 0.03	0.82 ± 0.033
	1.0	C.E.	B.T.	0.02	0.001	0.741 ± 0.038	0.618 ± 0.036	0.684 ± 0.035	0.66 ± 0.041
	Best (0.0)	C.E.	T.	0.0002	0.01	0.87 ± 0.028	0.82 ± 0.029	$\mathbf{0.809 \pm 0.03}$	0.82 ± 0.033
PRAUC on D_a	0.0	C.E.	T.	0.0002	0.01	0.87 ± 0.028	0.82 ± 0.029	0.809 ± 0.03	0.82 ± 0.033
	1.0	H.	B.T.	0.002	0.01	0.746 ± 0.037	0.677 ± 0.035	0.659 ± 0.036	0.678 ± 0.041
	Best (0.0)	C.E.	T.	0.0002	0.01	0.87 ± 0.028	0.82 ± 0.029	0.809 ± 0.03	$\mathbf{0.82 \pm 0.033}$
AUC on D_c	0.0	C.E.	B.T.	0.002	0.01	0.783 ± 0.035	0.733 ± 0.033	0.733 ± 0.033	0.698 ± 0.04
	1.0	C.E.	B.T.	0.02	0.001	0.721 ± 0.039	0.554 ± 0.037	0.025 ± 0.012	0.638 ± 0.042
	Best (0.5)	C.E.	H.	0.02	0.001	0.764 ± 0.036	0.706 ± 0.034	0.679 ± 0.035	0.644 ± 0.042
Acc. on D_c	0.0	C.E.	B.T.	0.002	0.01	0.783 ± 0.035	0.733 ± 0.033	0.733 ± 0.033	0.698 ± 0.04
	1.0	C.E.	B.T.	0.02	0.001	0.721 ± 0.039	0.554 ± 0.037	0.025 ± 0.012	0.638 ± 0.042
	Best (0.75)	C.E.	H.	0.02	0.001	0.758 ± 0.037	0.682 ± 0.035	0.633 ± 0.036	0.68 ± 0.04
F1 on D_c	0.0	C.E.	B.T.	0.002	0.01	0.783 ± 0.035	0.733 ± 0.033	0.733 ± 0.033	0.698 ± 0.04
	1.0	C.E.	B.T.	0.02	0.001	0.721 ± 0.039	0.554 ± 0.037	0.025 ± 0.012	0.638 ± 0.042
	Best (0.5)	H.	B.T.	0.002	0.01	0.784 ± 0.035	0.718 ± 0.034	0.733 ± 0.033	0.674 ± 0.041
PRAUC on D_c	0.0	C.E.	B.T.	0.002	0.01	0.783 ± 0.035	0.733 ± 0.033	0.733 ± 0.033	0.698 ± 0.04
	1.0	C.E.	B.T.	0.02	0.001	0.741 ± 0.038	0.618 ± 0.036	0.684 ± 0.035	0.66 ± 0.041
	Best (1.0)	C.E.	B.T.	0.02	0.001	0.741 ± 0.038	0.618 ± 0.036	0.684 ± 0.035	0.66 ± 0.041

Table 10: Absolute label prediction performance on the FAC dataset. For each α, we find the optimal absolute loss function (L_a), comparison loss function (L_c), regularization parameter λ, and learning rate (L.R.). We consider cross-entropy (C.E), hinge (H.), Bradley-Terry (B.T.), and Thurstone (T.) as loss functions. We repeat this optimization for AUC, accuracy (Acc.), F1 score, and PRAUC metrics on the absolute (D_a) and the comparison labels (D_c) of the validation set. Accordingly, each row triplet corresponds to the metric on which we optimize the hyperparameters. We then report all eight metrics on the test set, for training with $\alpha = 0.0$ (comparison labels only), $\alpha = 1.0$ (absolute labels only), and best performing $\alpha \in [0, 1]$.

	α	Hyperparameters Optimized on Validation Set				Performance Metrics on Test Set			
		L_a	L_c	λ	L.R.	AUC on D_c	Acc. on D_c	F1 on D_c	PRAUC on D_c
AUC on D_a	0.0	C.E.	T.	0.0002	0.01	0.72 ± 0.068	0.706 ± 0.06	0.716 ± 0.06	0.66 ± 0.072
	1.0	H.	B.T.	0.002	0.01	0.792 ± 0.06	0.75 ± 0.057	0.746 ± 0.058	0.757 ± 0.064
	Best (0.0)	C.E.	T.	0.0002	0.01	0.72 ± 0.068	0.706 ± 0.06	0.716 ± 0.06	0.66 ± 0.072
Acc. on D_a	0.0	C.E.	T.	0.0002	0.01	0.72 ± 0.068	0.706 ± 0.06	0.716 ± 0.06	0.66 ± 0.072
	1.0	H.	B.T.	0.002	0.01	0.792 ± 0.06	0.75 ± 0.057	0.746 ± 0.058	0.757 ± 0.064
	Best (0.0)	C.E.	T.	0.0002	0.01	0.72 ± 0.068	0.706 ± 0.06	0.716 ± 0.06	0.66 ± 0.072
F1 on D_a	0.0	C.E.	T.	0.0002	0.01	0.72 ± 0.068	0.706 ± 0.06	0.716 ± 0.06	0.66 ± 0.072
	1.0	C.E.	B.T.	0.02	0.001	0.821 ± 0.057	0.75 ± 0.057	0.736 ± 0.058	0.797 ± 0.06
	Best (0.0)	C.E.	T.	0.0002	0.01	0.72 ± 0.068	0.706 ± 0.06	0.716 ± 0.06	0.66 ± 0.072
PRAUC on D_a	0.0	C.E.	T.	0.0002	0.01	0.72 ± 0.068	0.706 ± 0.06	0.716 ± 0.06	0.66 ± 0.072
	1.0	H.	B.T.	0.002	0.01	0.792 ± 0.06	0.75 ± 0.057	0.746 ± 0.058	0.757 ± 0.064
	Best (0.0)	C.E.	T.	0.0002	0.01	0.72 ± 0.068	0.706 ± 0.06	0.716 ± 0.06	0.66 ± 0.072
AUC on D_c	0.0	C.E.	B.T.	0.002	0.01	0.751 ± 0.065	0.71 ± 0.06	0.706 ± 0.06	0.735 ± 0.066
	1.0	C.E.	B.T.	0.02	0.001	0.809 ± 0.058	0.733 ± 0.058	0.72 ± 0.059	0.734 ± 0.066
	Best (0.5)	C.E.	H.	0.02	0.001	$\mathbf{0.814 \pm 0.057}$	0.755 ± 0.057	0.74 ± 0.058	0.749 ± 0.065
Acc. on D_c	0.0	C.E.	B.T.	0.002	0.01	0.751 ± 0.065	0.71 ± 0.06	0.706 ± 0.06	0.735 ± 0.066
	1.0	C.E.	B.T.	0.02	0.001	0.809 ± 0.058	0.733 ± 0.058	0.72 ± 0.059	0.734 ± 0.066
	Best (0.75)	C.E.	H.	0.02	0.001	0.817 ± 0.057	$\mathbf{0.759 \pm 0.057}$	0.743 ± 0.058	0.774 ± 0.062
F1 on D_c	0.0	C.E.	B.T.	0.002	0.01	0.751 ± 0.065	0.71 ± 0.06	0.706 ± 0.06	0.735 ± 0.066
	1.0	C.E.	B.T.	0.02	0.001	0.809 ± 0.058	0.733 ± 0.058	0.72 ± 0.059	0.734 ± 0.066
	Best (0.5)	H.	B.T.	0.002	0.01	0.785 ± 0.061	0.737 ± 0.058	$\mathbf{0.736 \pm 0.058}$	0.76 ± 0.064
PRAUC on D_c	0.0	C.E.	B.T.	0.002	0.01	0.751 ± 0.065	0.71 ± 0.06	0.706 ± 0.06	0.735 ± 0.066
	1.0	C.E.	B.T.	0.02	0.001	0.821 ± 0.057	0.75 ± 0.057	0.736 ± 0.058	0.797 ± 0.06
	Best (1.0)	C.E.	B.T.	0.02	0.001	0.821 ± 0.057	0.75 ± 0.057	0.736 ± 0.058	$\mathbf{0.797 \pm 0.06}$

Table 11: Comparison label prediction performance on the FAC dataset. For each α, we find the optimal absolute loss function (L_a), comparison loss function (L_c), regularization parameter λ, and learning rate (L.R.). We consider cross-entropy (C.E), hinge (H.), Bradley-Terry (B.T.), and Thurstone (T.) as loss functions. We repeat this optimization for AUC, accuracy (Acc.), F1 score, and PRAUC metrics on the absolute (D_a) and the comparison labels (D_c) of the validation set. Accordingly, each row triplet corresponds to the metric on which we optimize the hyperparameters. We then report all eight metrics on the test set, for training with $\alpha = 0.0$ (comparison labels only), $\alpha = 1.0$ (absolute labels only), and best performing $\alpha \in [0, 1]$.