Quasibinary Classifier for Images with Zero and Multiple Labels

Shuai Liao
University of Amsterdam
s.liao@uva.nl

Efstratios Gavves
University of Amsterdam
e.gavves@uva.nl

ChangYong Oh
University of Amsterdam
c.oh@uva.nl

Cees G. M. Snoek
University of Amsterdam
cgmsnoek@uva.nl

***Abstract*—The softmax and binary classifier are commonly preferred for image classification applications. However, as softmax is specifically designed for categorical classification, it assumes each image has just one class label. This limits its applicability for problems where the number of labels does not equal one, most notably zero- and multi-label problems. In these challenging settings, binary classifiers are, in theory, better suited. However, as they ignore the correlation between classes, they are not as accurate and scalable in practice. In this paper, we start from the observation that the only difference between binary and softmax classifiers is their normalization function. Specifically, while the binary classifier self-normalizes its score, the softmax classifier combines the scores from all classes before normalisation. On the basis of this observation we introduce a normalization function that is learnable, constant, and shared between classes and data points. By doing so, we arrive at a new type of binary classifier that we coin quasibinary classifier. We show in a variety of image classification settings, and on several datasets, that quasibinary classifiers are considerably better in classification settings where regular binary and softmax classifiers suffer, including zero-label and multi-label classification. What is more, we show that quasibinary classifiers yield well-calibrated probabilities allowing for direct and reliable comparisons, not only between classes but also between data points.**

***Index Terms*—Binary classifier, softmax classifier, image classification.**

I. Introduction

Learning deep convolutional neural networks to classify an image requires a proper prediction activation function. It maps an internal network representation from feature space to prediction space, where a loss can be more easily defined. Softmax has been the dominant choice, and very successfully so [1]–[4]. In general, the softmax classifier is designed to model the probabilities for *one-vs.-rest* classification problems, where the number of ground truth labels per sample is assumed to be one. As the output predictions are normalized to be summed to one, softmax becomes a natural fit to return the categorical distribution prediction. However, this one-label assumption limits the application of softmax for problems where the number of labels for a sample is not one, most notably zero-label [5], [6] and multi-label [7]–[9] classification problems. Nonetheless, the softmax classifier is sometimes still being considered for zero-label [10], [11] and multi-label [7], [9], [12] classification problems, with the risk of violating the categorical assumption.

Instead of 'abusing' the softmax for out-of-distribution and multi-label problems, one may opt to build a set of binary classifiers for each of the classes, *e.g.* [13]–[18]. However, binary classifiers suffer from modeling class likelihoods independently. As a result, building binary classifiers becomes a suboptimal choice when the number of classes is large or imbalanced [13], [14], [16], [19]. Moreover, given an input image, the confidence scores per classifier are uncalibrated, making them incomparable in practice.

In this paper, we introduce a new definition of binary classifiers, which we coin *quasibinary classifiers*. Similar to regular binary classifiers, quasibinary classifiers compute probabilities for binary outputs. Yet, different from regular binary classifiers, quasibinary classifiers incorporate the information that other labels may exist in the image. We achieve this by having the normalization function of the quasibinary classifiers learnt rather than defined, as with binary and softmax classifiers. Specifically, quasibinary classifiers set the normalization function to be a constant and share it not only between classes but also between data points, allowing for tractability, high efficiency, and most importantly better calibration in computing the probabilities. As a result, quasibinary classifiers can work seamlessly in a variety of classification settings without the need for any specialized adaptation. They work well in regular single-label, multiple-class settings, as well as in multiple-label multiple class settings. They even work well when none of the labels are present in the image, that is out-of-distribution classification [10], [17] where the out-of-distribution samples are visible during training, but without any label attached to them. Importantly, quasibinary classifiers yield calibrated probabilities that can be used to compare predictions more reliably amongst different classes and different images, see Fig. 1.

We make the following contributions. First, we identify restrictions of the widely used binary and softmax classifiers. Second, on the basis of these restrictions, we propose quasibinary classifiers that learn a constant normalization function shared among classes and data points to return well calibrated binary outputs. Third, we show that quasibinary classifiers perform well on a variety of classification settings that are important in realistic application scenarios, including multiple- or even zero-label out-of-distribution image classification.

II. Background

Thanks to deep learning, probabilistic learning has become the *de facto* choice for training classifiers [1]–[4]. Probabilistic

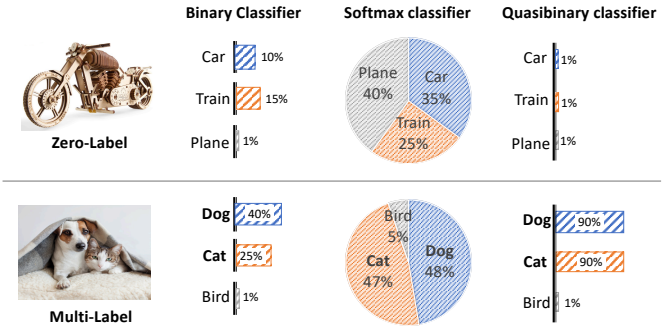


Fig. 1: **Comparison of quasibinary classifiers with binary and softmax classifiers for zero-label and multi-label classification.** While binary classifiers can in principle handle both cases, the confidence scores are not credible. Softmax fails in both cases as it restricts the sum of all the confidence to be equal to one. Our proposed quasibinary classifiers are trained jointly and assign credible confidence scores in both cases.

classifiers, be it binary or multi-class, maximize the (log-) likelihood $p(Y|X)$ on the training data $\{X, Y\} = \{(x^{(i)}, y^{(i)})\}$. Specifically, both binary and multi-class classifiers assume the following decomposition of the total probability

$$p(Y|X) = \prod_i p(y^{(i)}|x^{(i)}) \tag{1}$$

The decomposition in equation (1) is based on the assumption that the class predictions $y^{(i)}$ are conditionally independent given the input data points $x^{(i)}$. The classifiers are then typically the neurons $p_k, k = 1, ..., K$ for K classes in the ultimate layer of the neural network. For probabilistic training of the classifiers the neurons p_k must correspond to valid probabilities, namely, $p_k \in [0, 1]$. In a probabilistic, rather than a frequentist perspective, this probability can also represent the *belief* that one has in the event k being true or not.

A. Ensembles of sigmoid classifiers

For binary classifiers given K classes, the random variables $y_1^{(i)} \in \{0, 1\}, ..., y_K^{(i)} \in \{0, 1\}$ of the i-th sample correspond to the K independent predictions. These K random variables follow a Bernoulli distribution, $y_k^{(i)} \sim \text{Bernoulli}(p_k^{(i)})$, which means p_k can be conveniently modelled by independent sigmoids, that is

$$p_k^{(i)} = p(y_k^{(i)}|x^{(i)}) = \frac{\exp(z_k^{(i)})}{1 + \exp(z_k^{(i)})} \tag{2}$$

where $z_k^{(i)} = h(x^{(i)})$ is the k-th logit computed by the neural network $h(\cdot)$ on the input $x^{(i)}$. The probability output space of the Bernoulli random variable is complemented by $\bar{p}_k$, such that $\bar{p}_k = 1 - p_k$. This is guaranteed by the normalization factor $C_k^{(i)} = 1 + \exp(z_k^{(i)})$. Since binary classifiers work independently for each class given a data sample, it is possible to use the binary classifier in multi-label classification settings. As binary classifiers do not consider the dependency between classes, their performance is usually sub-optimal [13], [14], [16], [19].

We note that while sigmoid activation functions have been conflated with binary and independent classifiers, their function is simply to return binary decisions. Being independent is a modeling choice. This is important to emphasize in the context of the proposed quasibinary classifiers that we introduce later.

B. Softmax classifiers

When having multiple classes in the training set while knowing that each image contains only a single class, *i.e.*, $\#label = 1$, the classifier neurons are usually parameterized by softmax functions. The softmax function models the probability of a categorical distribution parameterized by the means

$$p_k^{(i)} = p(y_k^{(i)}|x^{(i)}; \text{\#label=1}) = \frac{\exp(z_k^{(i)})}{\sum_j \exp(z_j^{(i)})}. \tag{3}$$

We observe that both definitions of the binary classifier and the softmax classifier in equations (2) and (3) have a similar form. The numerator is precisely the same and equal to the exponentiation of the logit $z_k^{(i)}$. We refer to this as the scoring function $s_k^{(i)} = \exp(z_k^{(i)})$. The denominator, however, is different. While in the binary classifier the denominator only depends on the same logit $z_k^{(i)}$, in the softmax classifier the denominator depends on the logits from all classes. This is beneficial for optimization, as using the logits from all classes couples together the otherwise independent scoring functions and trains them jointly.

Despite the popularity and accuracy of softmax over the binary classifier, however, the softmax classifier also suffers from a limitation: the number of labels is not always known at test time. Most notable for out-of-distribution data (#label=0) and multiple-label data (#label=n). In both cases, the probability outputs are meaningless.

In this work, we are interested in defining a classifier that inherits the flexibility of binary classifiers, while leveraging the prior knowledge of the number of labels per sample that is given for free for optimization. To this end, we present *quasibinary classifiers*. Quasibinary classifiers seamlessly handle any-label (even zero-label) classifications like a binary classifier, but they can be trained in a coupled way like softmax. They also return much more calibrated probabilities as per the definitions recently introduced by Guo *et al.* [20].

III. Quasibinary Classifiers

A. Definition

As with the ensembles of binary classifiers, we want binary decisions since in the image there can be multiple classes present. That is, we have again random variables $y_1^{(i)} \in \{0, 1\}, ..., y_K^{(i)} \in \{0, 1\}$ mapped to the binary output space. Different from the ensembles of binary classifiers, we do not assume that these binary classifiers are independent to each other. That is, we want the likelihood terms of the random variables to be both binary and correlated.

In defining our model, we also start from Bernoulli random variables. However, in the mean parameters of the Bernoulli distributions we introduce a shared constant C that is not a random variable and is shared across all likelihood terms across classes (like softmax) and -importantly- all images. Namely, our Bernoulli variables are modelled as $y_k^{(i)} \sim Bernoulli(q_k)$, where

$$q_k^{(i)} = q(y_k^{(i)}|x^{(i)}, X_{\setminus(i)}) = \frac{\exp(z_k^{(i)})}{C(x^{(i)}, X_{\setminus(i)})} = \frac{\exp(z_k^{(i)})}{C(X)}. \quad (4)$$

Note that, as intended, the normalization constant is shared between samples and does not depend on specific classes. Therefore, the predictions modelled by the Bernoulli random variables are still binary but not independent. As such, quasibinary classifiers can seamlessly work in multiple-label or zero-label setting, where an image may contain several objects from different classes or none at all. In the case of multiple labels present in an image, each of the Bernoulli variables shall return their confidence in the class being present or not, albeit these confidences are not independent to each other. And in the case the image contains no relevant objects, the Bernoulli variables shall all return low confidence without being forced to either select wrongly one of the classes as being present or to assign the same likelihood to all wrong classes.

For the quasibinary classifiers we use the same scoring function as the binary and softmax classifiers, that is an exponential $\exp(\cdot)$, which ensures positivity. Importantly, note that now the normalization function depends on all the training data, $C(X)$. Namely, we have a normalization function that is *shared* across all classes and training data points, as it depends on the logits of all the training data points. This is crucial, since *(i)* a shared normalization function across classes is able to couple individual binary classifiers together, thus jointly optimizing them while taking into account the knowledge of how many labels are present. Moreover, *(ii)* having a shared normalization function across training data points allows the quasibinary scoring functions to "communicate" with each other. Thus, the classifiers learn to predict confidence scores that are comparable across classes and different data points, thus returning better calibrated probabilities.

As with ensembles of binary classifiers, the probability output space is complemented by $\bar{p}_k = 1 - p_k$, thus having the total sum $\bar{p}_k + p_k = 1$, which is a requirement for a valid probability space. Furthermore, for a valid probability space we need to make sure that $0 < p_k < 1$, which is possible by a careful choice of an activation function for the classifier neurons. In the end, the joint likelihood is equal to

$$p(Y|X) = \prod_i p(y^{(i)}|X) = \prod_i p(y^{(i)}|x^{(i)}, X_{\setminus(i)}), \quad (5)$$

where the conditioning variables $x^{(i)}, X_{\setminus(i)})$ together make up for the X variable in the normalization constant $C(X)$.

Choosing normalization function $C(X)$**.** Clearly, the choice of the normalization function is critical. There exist several requirements.

First, we want our quasibinary classifiers to be jointly optimized while exploiting the free prior knowledge of the number of labels per training sample, *i.e.* $\#\text{label} = n$ where $n \in [0, 1, \cdots, K]$. We can show that this prior knowledge is equivalent to the following constraint in the predicted probability q_k,

$$\sum_k q_k = \#\text{label}. \quad (6)$$

We provide the proof in the supplementary material. Note that the reason for the sum of all q_k being possibly greater than 1 is that the presence (or not) of the various labels y_k in a training sample is not mutually exclusive. As a special case, consider $\#$label=1, equation (6) is exactly what the softmax classifier enforces. A second requirement regarding the normalizing function is computational tractability, as depending on all data, $x^{(i)}, X_{\setminus(i)}$, is potentially prohibitive.

B. Algorithm

In the previous section we explained that the normalization function of quasibinary classifiers is a function shared across classes and across data points. As this normalization function is shared across all data points and classes, it needs to be a constant function. This constant normalization function must then be learned throughout the training so as to satisfy several constraints. First, by the end of the training the probability estimates must lie within the $[0, 1]$ range. Second, the normalization function must be trained to adhere to the constraint of equation (6). That is, we want our quasibinary classifiers to satisfy

$$\arg\max \sum_{i,k} y_k^{(i)} \log(q_k^{(i)}) \quad (7)$$

$$\text{s.t. } q_k^{(i)} \in [0, 1] \quad \forall i, k \quad (8)$$

$$\sum_k q_k^{(i)} = \#\text{labels}^{(i)} \quad (9)$$

Similar to stochastic gradient descent, we rely on stochastic mini-batches B instead of the whole training set.

Training. Rather than enforcing the constraint in equation (6) for all samples, we relax the constraint to batch level for computational tractability. Namely, given that our normalizing function is a constant we have that

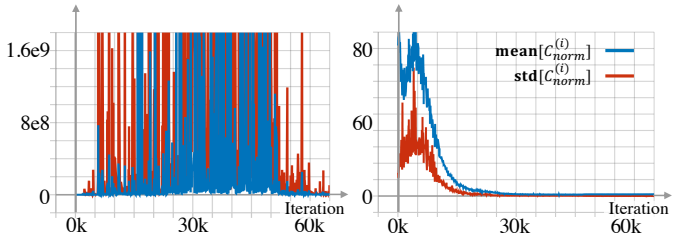
$$\sum_i^B \sum_k^K \exp(z_k^{(i)})/C = \sum_i \#\text{label}^{(i)}$$

$$\Rightarrow \quad C = \frac{1}{N} \sum_{i=1}^B \sum_{k=1}^K \exp(z_k^{(i)}) \quad (10)$$

where N is the total number of labels a batch of data has. Following [21] we resort to Lagrange relaxation to derive the final learning objective and optimize for the model parameters θ

$$\mathcal{L}(\theta) = \frac{1}{B} \sum_{i=1}^B \sum_{k=1}^K y_k^{(i)} \cdot \log q_k^{(i)} - \max(\log q_k^{(i)}, 0), \quad (11)$$

$$\text{where } \log q_k^{(i)} = z_k^{(i)} - \log C \quad (12)$$



(a) Softmax (b) Quasibinary

Fig. 2: **The normalization function $C(X)$ converges over time to a constant value on CIFAR100.** We observe that the mean and variance of the softmax normalization are up to 8x larger than for the quasibinary normalization. The variance is extreme due to alternating spikes (from 1.6e9 to 1e2). The fluctuations continue with no convergence. That is expected, as for softmax the normalization needs not to converge for accurate classification. However, a fluctuating normalizing constant means that scores between different images are hardly comparable. Overall, softmax behaves completely opposite to the quasibinary.

The summands correspond to the losses by the q_k likelihood terms. The second term makes sure that q_k yields by the end of the training a valid probability (in log-space the maximum probability is 0).

Note that because of the Lagrange relaxation, it is *not theoretically guaranteed* that q_k will always be within the $(0, 1)$ range and thus yield a probability. However, we find the optimizer is able to find good enough solution to satisfy the constraint. In practice, we only observed a negligible amount ($\sim 0.1\%$) of violations *i.e.* ($q_k > 1$) on test data, which are simply clipped to [0,1] to makes sure q_k are proper probabilities. Similar optimizations for learning to compute probabilities were also previously proposed in [22] with success. Note also that during training time as the model gets updated per iteration, C varies till convergence.

Training in batches. It is important to note that we define the normalization function in equation (10) as a batch-level implementation of equation (6). This means that the number of labels changes per image. Also, just like other multi-label classification settings, the total number of classes K is fixed both at training and test. As the training is in batches, with imbalanced datasets it is important to account for the class frequencies in the batch constitution. We simply follow the spirit of SGD to randomly select a mini-batch of samples [8], [12], [18].

Testing. After training we substitute the normalization function $C(X)$ in equation (4) with the moving average of $C^{(\text{mavg})}$ as a constant to make prediction at test time. Thus, unlike softmax classifiers that assume #label=1 on test data to compute normalization factor, we do not need to make any such assumptions.

Moving average of normalization constant. Since it is inefficient to compute at test time the normalization factor C based on training data, we track a moving average of $C^{(t)}$ over training iterations,

$$C^{(\text{mavg})} = C^{(\text{mavg})} + \alpha \cdot C^{(t)}, \tag{13}$$

where $C^{(t)}$ is the constant normalization function at iteration t, see Fig 2. This is similar to batch-normalization. Note that just like in batch normalization, the α smooths out the fluctuations and helps the learnt constant to converge to a single value consistent for all images, as shown in Fig. 2. Further, we empirically find that the training is not negatively affected by the moving average computation in terms of accuracy. We find that the algorithm is rather robust in the choice of α with the differences in the variance of $C^{(\text{mavg})}$ up to 1e-2. We find that setting $\alpha = 0.1$ is good enough.

IV. Related work

A. Zero-label problems

In general, zero-label data refers to irrelevant samples that belong to neither class of the in-distribution training data. Recent deep models achieve good accuracy on in-distribution data, but are known to be over-confident on out-of-distribution data [10], [11]. Based on statistics of the softmax prediction, Hendrycks and Gimpel [10] introduce a baseline detector to differentiate misclassified samples from out-of-distribution samples. They point out the softmax probabilities have a poor direct correspondence to model confidence, but do not provide a solution. Liang *et al*. [5] find that using temperature scaling and adding small perturbations to the input data increases the statistical significance of the softmax output for the in- and out-of-distribution samples. This leads to an improvement in detection performance.

Passively detecting zero-label out-of-distribution data based on a trained model is not enough. Rather, one may opt to build a model that is aware of out-of-distribution data during training. To do so, Lee *et al*. [6] add two additional loss terms upon the standard cross-entropy loss to train a softmax classifier. The first loss models out-of-distribution data by enforcing the softmax to output a uniform distribution prediction. The second loss guides a generative adversarial network to generate the most effective out-of-distribution samples for training. In contrast, Hein *et al*. [11] introduce noise data as out-of-distribution samples during training. With the same intention as Lee *et al*. [6] to encourage an uniform distribution prediction for out-of-distribution data, Hein *et al*. [11] propose a loss function that suppresses the largest predicted confidence from softmax.

In this paper, we continue the line of work on explicitly modeling out-of-distribution data. However, instead of introducing new losses for the softmax classifier, we introduce the quasibinary classifier which is able to handle out-of-distribution data by design.

B. Multi-label problems

In most real life classification problems, one sample may be associated with multiple labels at the same time, rather than just one. For example, an image from a social network can have multiple user tags and a medical image may show

none or multiple symptoms of a certain diseases. Multi-label classification considers problems where the number of labels for a sample is unknown. The traditional strategy is to reduce the multi-label classification problem to multiple binary classification problems [15]–[18]. When each class is being modeled independently, an ensemble of binary classifiers is able to make multi-label predictions. However, as binary classifiers are trained independently, their confidence scores lack calibration. As a result, binary classifiers have difficulty when extended to multi-label problems with a large number of classes.

Although softmax is exclusively designed for one-vs.-rest classification, it is also being adapted to solve multi-label classification [7]. For example, the softmax with cross-entropy loss is trained to push the prediction of a multi-label sample to be an equally weighted multi-hot vector. However, as softmax forces the sum of the output predictions to be one, a softmax classifier cannot provide credible confidence scores for multi-label classification problems. Alternatively, one may transform a multi-label classification problem into a ranking problem [8], [9], [23]–[25]. Given training data, the objective is to learn a model able to rank the most relevant labels above the irrelevant ones. However, as such a strategy does not directly solve a classification problem it cannot provide model confidence of the predictions. In this paper, we are interested in building a multi-label classifier that can provide reasonable model confidence scores.

V. Experiments

In this section, we evaluate the quasibinary classifier on several classification problems, with a variety of benchmark datasets and a Resnet18 [26] backbone. Although the evaluation settings may depend on the specific problem, the training settings are mostly the same. To avoid repeated description, we first detail this common training setting for all models.

Common training protocol. All model parameters are randomly initialized, following He *et al*. [3]. We use the SGD optimizer with momentum set to 1e-4. The initial learning rate is set as 0.1 and decreased by a factor of 10 after 50% and 75% of the epochs. We train all the models for 200 epochs with a batch size of 64. For the quasibinary classifier, we set the momentum of the moving average used to track the normalization factor during training as 0.9.

A. One-vs.-rest image classification

Setup. First, we evaluate the performance of the quasibinary classifier on the common image classification problem with one ground truth label per image. While we are particularly interested in the comparison with the binary classifier, we also compare with the softmax classifier, known to be the better solution for this problem. We choose four datasets, *i.e.* ***CIFAR10*** [27], ***CIFAR100*** [27], ***Tiny-ImageNet*** and ***ImageNet*** [28] with $\{10, 100, 200, 1000\}$ classes respectively. The total number of images for each of the three datasets are 60K, 60K, 110K, and 1200K, and for all datasets the images are equally distributed over each class. Except for the ImageNet experiments where we use standard 224×224 input size, we always use a 32×32 input size in order to share the same network architecture. The top-1 error rate on the test sets is reported.

TABLE I: **One-vs.-rest image classification.** Comparison of top-1 error rate on CIFAR10, CIFAR100, Tiny ImageNet and ImageNet, with the total number of classes being 10, 100, 200 and 1000. Binary classifiers are good for small amounts of classes, but have difficulty to converge as the number of classes increases. In that case our quasibinary classifier does much better. For up to 200 classes it is even comparable with the softmax classifier.

	CIFAR10	**CIFAR100**	**Tiny-ImageNet**	**ImageNet**
Binary classifier	**4.8**	35.4	×	×
Quasibinary classifier (*Ours*)	4.9	**21.9**	**42.9**	25.4
Softmax classifier	5.2	22.2	43.3	**23.9**

One-vs.rest image classification results. We report results with models trained with a Resnet18 [26] backbone in Table I. Similar results were obtained with VGG16 and DenseNets, see supplementary material. The performance differences between the binary classifier and our quasibinary classifier are subtle on CIFAR10, when there are only a small number of classes. Already for 100 classes, on CIFAR100, our quasibinary classifier does much better. When increasing the number of classes further the binary classifier fails to converge. This is partially due to the fact that each binary classifier models the likelihood prediction independently. We further suspect the optimization difficulty of binary classifiers on large class dataset is due to the sigmoid activation function, which is known to saturate easily [29]. Interestingly, the quasibinary classifier is even competitive with the softmax classifier for classification problems up to 200 classes. For the more challenging ImageNet setting with 1000 classes, the softmax classifier performs better, as expected.

B. Zero-label image classification

Setup. Next, we evaluate the performance of the quasibinary classifier in modeling zero-label image data. In particular, our goal is to build a model that is able to separate out data samples that belong to neither of the predefined classes (out-of-distribution classes), while maintaining the classification performance on the in-distribution classes. Similar to [6], [11], we use out-of-distribution data for training. Previous works [5], [6], [10], [11] construct out-of-distribution samples by either taking natural image samples from other datasets different from the training source or by synthesizing noise images. As those out-of-distribution samples are usually easy to distinguish from the source dataset, and consequently most methods achieve near perfect performance, we construct a new dataset based on CIFAR100.

In **CIFAR60+40** the zero-label out-of-distribution data is created to have more confusion with the in-distribution data. The original CIFAR100 dataset has 20 coarse classes where each of them is subdivided into 5 fine-grained classes. Thus, a

TABLE II: **Zero-label image classification** on CIFAR60+40. Binary classifiers perform reasonable, but do not excel. Softmax based methods obtain good performance on either in-distribution accuracy [6] or out-of-distribution MMC [11], but cannot do both well. Our quasibinary classifier achieves good performance on all measures.

	IN Accuracy ↑	**OUT** MMC ↓	**BOTH** AU-ROC↑
Binary classifiers [16], [18], [30]	77.8 %	14.7 %	0.901
Softmax + $\mathcal{L}_{\text{Uniform}}$ [6]	**80.7** %	59.8 %	0.800
Softmax + $\mathcal{L}_{\text{MaxConf}}$ [11]	45.2 %	7.4 %	0.764
Quasibinary classifier	80.6 %	**6.9** %	**0.913**

TABLE III: **Multi-label image classification.** Although the softmax returns good predictions in multi-label settings, their calibrated confidence scores (ECE) suffer compared to the proposed quasibinary classifiers, even when softened with a temperature. This indicates that softmax classifiers make unjustifiably overconfident predictions.

	MS-COCO		**NUS-WIDE**	
	F_1↑	ECE(%)↓	F_1↑	ECE(%)↓
Binary classifier [16], [18], [30]	51.2	26.8	40.7	23.6
Softmax [31]	**54.7**	32.2	43.2	25.8
Softmax w/ temperature [32]	**54.7**	31.4	43.2	24.6
Quasibinary classifier	**54.7**	**2.8**	**43.5**	**3.3**

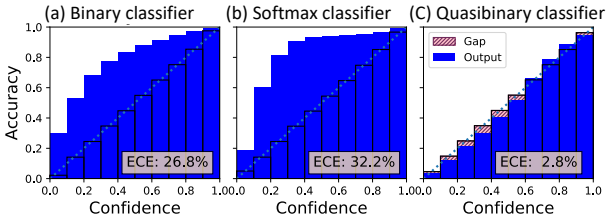


Fig. 3: **Reliability diagrams for multi-label classification** on MS-COCO, where being diagonal means a perfect calibrated confidence prediction (with ECE=0). Our quasibinary classifier achieves the best confidence calibration, leading to more credible confidence scores.

total of 100 fine-grained classes are considered in CIFAR100. We take the identical image data from CIFAR100 for CIFAR60+40, however, for each coarse class, 2 out of the 5 fine-grained classes the labels are removed, meaning they become our zero-label out-of-distribution samples. As a result, images for 60 classes are retained as labeled in-distribution (IN) and the images from the remaining 40 classes are treated as zero-label out-of-distribution (OUT). Splits will be made available.

Baselines. We compare the quasibinary classifier with three methods for modeling out-of-distribution data: 1) binary classifiers; 2) softmax classifier with $\mathcal{L}_{\text{Uniform}}$ loss [6] that minimizes KL-divergence of predicted probability distribution for OUT samples *w.r.t.*a uniform distribution; and 3) softmax classifier with $\mathcal{L}_{\text{MaxConf}}$ loss [11] that suppresses the predicted probability dimension with the maximum confidence for OUT samples.

Evaluation. We consider three metrics: 1) Accuracy on IN samples, 2) Mean of Maximum Confidence (MMC) [11] for OUT samples and 3) AU-ROC measure to evaluate how good a model can differentiate OUT from IN.

Results. We report results in Table II. The quasibinary classifier and softmax with $\mathcal{L}_{\text{Uniform}}$ loss [6] obtain a similar accuracy on IN samples, but softmax with $\mathcal{L}_{\text{Uniform}}$ loss fails to suppress the MMC for OUT samples. Compared to the softmax classifier with $\mathcal{L}_{\text{MaxConf}}$ loss [11], the quasibinary classifier obtains a slightly better MMC performance on OUT samples, but a much better accuracy is achieved on IN samples (80.6% vs 45.2%). What is more, since [6], [11] optimize the softmax classifier towards a uniform distribution prediction for OUT samples, they both have a theoretical upper bound for MMC performance that is equal to 1/#classes. In theory, the quasibinary classifier and binary classifier can both assign 0% confidence to OUT data. However, in practice, we observe binary classifiers obtain a suboptimal performance on the accuracy for IN data and MMC for OUT data. We conclude our quasibinary classifier is successful in modeling zero-label samples.

C. Multi-label image classification

Setup. Last, we evaluate the quasibinary classifier for multi-label image classification on ***NUS-WIDE*** [33] and ***MS-COCO*** [34]. NUS-WIDE consist of 260K Flickr images with 81 classes selected from high frequency user tags. MS-COCO consists of 120K images that are labeled by 80 object classes. The average number of labels per image for the two datasets are 2.41 and 2.94, respectively. We follow Gong *et al*. [8] to remove 60K invalid images in NUS-WIDE and split the rest into a 150K training set and a 50K test set. For MS-COCO, we use the original training/validation splits, namely 82K for training and 40K for testing. We compare three classifiers: 1) binary classifiers [16], [18], [30], 2) Softmax classifier [7], [9], [12] and 3) quasibinary classifier. We follow the setting of Li *et al*. [9] to fine-tune a VGG16 [35] backbone for 20 epochs. During test time, the top-4 most confident predictions are outputted as suggested by Li *et al*. [9].

Evaluation. For evaluation, we first follow the conventional evaluation protocol to report macro F_1 score, which assesses the quality of the confidence score for ranking. However, we are more interested in how credible the predicted confidence scores are. Thus, we also report the Expected Calibration Error (ECE) [20], which indicates how well the predicted confidence score indicates the actual likelihood of the model to make a correct prediction. Notice that ECE is mostly used to evaluate the reliability of the top-1 confidence in single-label classification problem. In our setting, we evaluate the top-4 confidence reliability.

Results. We report results in Table III. Regarding the F_1 score, the performance of the quasibinary classifier and softmax are similar, while the binary classifier is worse. This is mainly because binary classifiers do not use the $\#labels$

as a prior during training. Considering the ECE score, we observe the quasibinary classifier improves the reliability of the confidence predictions over binary classifiers and softmax considerably. Further, we show in Fig. 3 the 10-bins reliability diagrams [20] as a visualization of the ECE score, where being diagonal means the accuracy of test samples in each bin aligns with received confidence score, *i.e*. perfect calibrated confidence prediction (with ECE=0). We again observe the quasibinary classifier achieves the best performance. Although modulating the temperature for softmax helps with over-confident scores in one-vs-rest classification (data not shown), when multiple labels are present the returned scores are still overconfident. The reason is that even with multiple labels the sum of outputs remains 1 and more than one logits must still share the "max" score. Last, we show qualitative results in Fig. 4. We observe that only the quasibinary classifier is able to assign high confidence to multiple correct labels at the same time. Most interestingly, we observe quasibinary classifier helps to retrieve part of the missing annotations with high confidence, e.g. "sunset" and "clouds" for the second image of NUS-WIDE examples. We conclude that the quasibinary classifier models multi-label classification problems with credible confidence scores.

VI. Conclusion

Softmax and binary classifiers face difficulties in image classification settings beyond the regular multi-class classification. Characteristic examples are multiple class, multiple label problems, where an image may contain more than one object. Another example is the zero-label out-of-distribution problem, where the image may contain none of the relevant labels. To address the limitations of binary and softmax classifers, we introduce the quasibinary classifiers. Quasibinary classifiers define a novel normalization function that is learnable, constant, and shared between classes and data points. This allows them to compute probabilities that are better calibrated and, thus, more directly comparable between classes as well other data points. We show in a variety of settings and datasets that quasibinary classifiers are considerably better in image classification settings where regular binary and softmax classifiers suffer, including zero-label and multi-label image classification. Importantly, we show that quasibinary classifiers yield well calibrated probabilities allowing for direct and reliable comparisons not only between classes but also between data points.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NeurIPS*, 2012.

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *ICCV*, 2015.

[4] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *CVPR*, 2017.

[5] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *ICLR*, 2018.

[6] K. Lee, H. Lee, K. Lee, and J. Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," *ICLR*, 2018.

[7] J. Read, "Scalable multi-label classification," Ph.D. dissertation, 2010.

[8] Y. Gong, Y. Jia, T. Leung, A. Toshev, and S. Ioffe, "Deep convolutional ranking for multilabel image annotation," *arXiv preprint arXiv:1312.4894*, 2013.

[9] Y. Li, Y. Song, and J. Luo, "Improving pairwise ranking for multi-label image classification," in *CVPR*, 2017.

[10] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *ICLR*, 2017.

[11] M. Hein, M. Andriushchenko, and J. Bitterwolf, "Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem," in *CVPR*, 2019.

[12] L. Yao, E. Poblenz, D. Dagunts, B. Covington, D. Bernard, and K. Lyman, "Learning to diagnose from scratch by exploiting dependencies among labels," *arXiv preprint arXiv:1710.10501*, 2017.

[13] K. Napierala and J. Stefanowski, "Types of minority class examples and their influence on learning classifiers from imbalanced data," *JIIS*, 2016.

[14] T. J. Lakshmi and C. S. R. Prasad, "A study on classifying imbalanced datasets," in *ICNSC*, 2014.

[15] J. Nam, J. Kim, E. L. Mencía, I. Gurevych, and J. Fürnkranz, "Large-scale multi-label text classification — revisiting neural networks," in *ECML PKDD*, 2014.

[16] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases," in *CVPR*, 2017.

[17] X. Wang, Y. Peng, L. Lu, Z. Lu, and R. M. Summers, "Tienet: Text-image embedding network for common thorax disease classification and reporting in chest x-rays," in *CVPR*, 2018.

[18] H. Chen, S. Miao, D. Xu, G. D. Hager, and A. P. Harrison, "Deep hierarchical multi-label classification of chest x-ray images," in *MIDL*, 2018.

[19] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "Resampling or reweighting: A comparison of boosting implementations," in *ICTAI*, 2008.

[20] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *ICML*, 2017.

[21] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Traces and emergence of nonlinear programming*, 2014.

[22] J. Devlin, R. Zbib, Z. Huang, T. Lamar, R. Schwartz, and J. Makhoul, "Fast and robust neural network joint models for statistical machine translation," in *ACL*, 2014.

[23] J. Weston, S. Bengio, and N. Usunier, "Wsabie: Scaling up to large vocabulary image annotation," in *IJCAI*, 2011.

[24] M.-L. Zhang and Z.-H. Zhou, "Multilabel neural networks with applications to functional genomics and text categorization," *TKDE*, 2006.

[25] F. Wu, Z. Wang, Z. Zhang, Y. Yang, J. Luo, W. Zhu, and Y. Zhuang, "Weakly semi-supervised deep learning for multi-label image annotation," *IEEE Transactions on Big Data*, 2015.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[27] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[28] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[29] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *AISTATS*, 2010.

[30] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Machine learning*, 2011.

[31] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten, "Exploring the limits of weakly supervised pretraining," in *ECCV*, 2018.

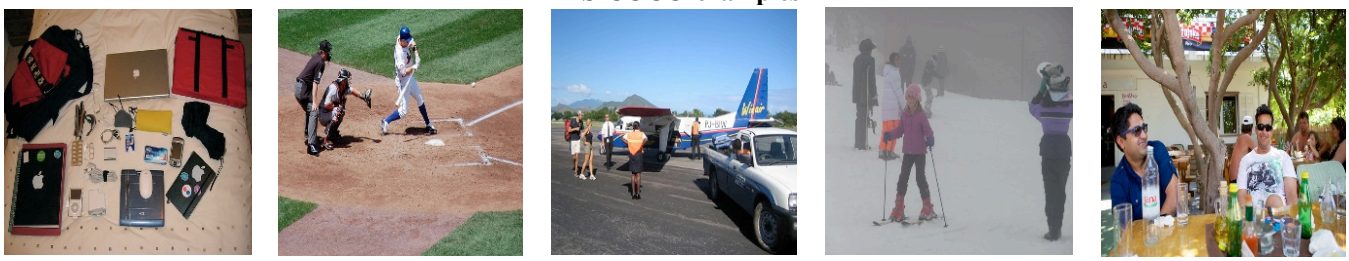
[32] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NIPS*, 2015.

[33] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "Nus-wide: a real-world web image database from national university of singapore," in *CIVR*, 2009.

[34] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, 2014.

[35] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

MS-COCO examples



(a) Binary classifiers

book: 13.4%	**person**: 98.9%	**person**: 65.6%	**person**: 92.5%	**person**: 97.1%
laptop: 13.3%	**baseballglove**: 1.0%	car: 14.3%	**skis**: 7.4%	**dining table**: 0.9%
cell phone: 9.8%	**baseball bat**: 0.1%	**truck**: 13.0%	backpack: 0.1%	**cup**: 0.4%
mouse: 8.5%	**sports ball**: 0.0%	handbag: 1.5%	**snowboard**: 0.0%	**bottle**: 0.3%

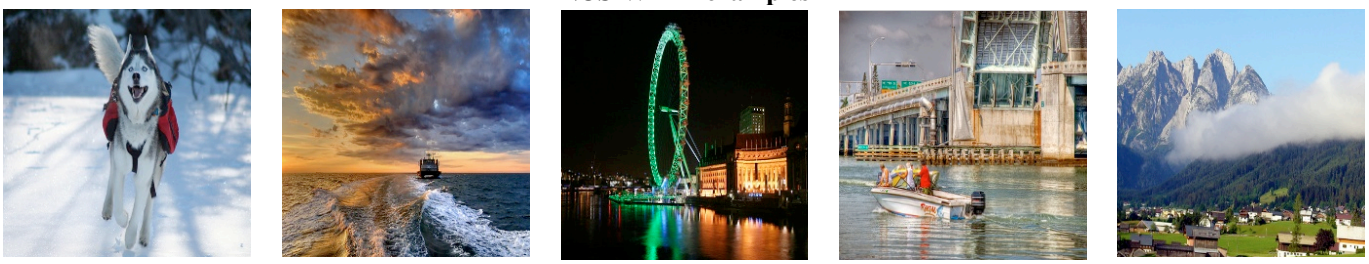
(b) Softmax

laptop: 16.9%	**baseballglove**: 34.4%	**person**: 17.0%	**skis**: 45.9%	**person**: 12.0%
cell phone: 13.5%	**person**: 31.2%	handbag: 12.9%	**person**: 35.6%	**bottle**: 11.5%
book: 12.3%	**sports ball**: 14.9%	car: 12.8%	backpack: 10.8%	**dining table**: 11.2%
bed: 11.3%	**baseball bat**: 13.3%	**truck**: 10.9%	**snowboard**: 2.4%	**cup**: 7.8%

(c) Quasibinary classifiers (*Ours*)

laptop:100.0%	**baseballglove**:100.0%	**person**:100.0%	**person**:100.0%	**person**:100.0%
cell phone: 72.1%	**person**:100.0%	handbag:100.0%	**skis**:100.0%	**bottle**:100.0%
book: 66.5%	**baseball bat**: 74.9%	**truck**:100.0%	backpack: 43.9%	**dining table**: 98.9%
bed: 65.1%	**sports ball**: 63.5%	car: 91.5%	**snowboard**: 11.6%	**cup**: 85.3%

NUS-WIDE examples



(a) Binary classifiers

animal: 78.1%	**water**: 28.5%	buildings: 25.1%	**water**: 71.0%	**sky**: 43.5%
snow: 11.6%	**sky**: 26.8%	**nighttime**: 23.0%	**boats**: 8.1%	**mountain**: 21.1%
dog: 6.1%	clouds: 22.3%	**water**: 22.2%	buildings: 6.5%	**clouds**: 17.1%
person: 0.8%	**ocean**: 14.6%	**reflection**: 5.7%	reflection: 3.6%	**valley**: 11.5%

(b) Softmax classifier

animal: 36.7%	**ocean**: 19.1%	**nighttime**: 25.9%	**water**: 26.9%	**valley**: 20.7%
snow: 28.2%	**sky**: 17.6%	buildings: 16.3%	**boats**: 19.8%	**mountain**: 19.8%
dog: 22.0%	clouds: 15.8%	**water**: 15.4%	buildings: 10.6%	**sky**: 15.4%
sky: 2.1%	**water**: 13.6%	**reflection**: 10.7%	window: 7.9%	**clouds**: 13.7%

(c) Quasibinary classifiers (*Ours*)

dog:100.0%	**water**:100.0%	**water**:100.0%	**boats**:100.0%	**mountain**:100.0%
snow:100.0%	sunset:100.0%	**nighttime**: 96.5%	**water**: 96.6%	**sky**:100.0%
animal:100.0%	**sky**:100.0%	**reflection**: 91.8%	window: 62.9%	**clouds**: 90.3%
sky: 20.4%	clouds:100.0%	buildings: 83.6%	**sky**: 62.5%	**valley**: 81.3%

Fig. 4: **Multi-label image classification with credible confidence.** We show the top-4 most confident predictions and scores for all three classifiers, with the correct prediction being marked in bold font. The images are from NUSWIDE and MS-COCO. While binary classifiers perform sub-optimal and softmax has to enforce the predictions to be summed to one, the quasibinary classifier provides credible confidence scores by design.

Supplementary Material
Quasibinary Classier for Images with Zero and Multiple Labels

Shuai Liao
University of Amsterdam
s.liao@uva.nl

Efstratios Gavves
University of Amsterdam
e.gavves@uva.nl

Changyong Oh
University of Amsterdam
c.oh@uva.nl

Cees G. M. Snoek
University of Amsterdam
c.g.m.snoek@uva.nl

I. Proof of Eq. (6)

Given q_k as the probability for a sample to be the k-th class, we will prove that the sum of all q_k equals the total number of labels (#label) in the sample (Eq. (6) in the main paper), *i.e.*:

$$\sum_k q_k = \#\text{label} \tag{1}$$

As a quick explanation, we first show a simple example. The mathematical proof is followed after.

Example. Let us consider a 3-class problem, *e.g.* $\{A, B, C\}$ and for a specific sample we have #label=2. We can decompose the Bernoulli probabilities $P(A)$, $P(B)$ and $P(C)$ for each class into the summation of joint probabilities from a set of mutually exclusive joint events. The decompositions are:

$$P(A) = P(AB\overline{C}) + P(A\overline{B}C) \tag{2}$$
$$P(B) = P(AB\overline{C}) + P(\overline{A}BC) \tag{3}$$
$$P(C) = P(A\overline{B}C) + P(\overline{A}BC) \tag{4}$$

Note that #label=2 indicates a sample can only have 2 labels, thus probabilities like $P(A\overline{BC})$, $P(ABC)$ are 0, since they correspond to #label=1 and #label=3. As a result, we have

$$P(A) + P(B) + P(C) \tag{5}$$
$$= 2 \times (P(AB\overline{C}) + P(A\overline{B}C) + P(\overline{A}BC)) \tag{6}$$
$$= 2 \times 1 \tag{7}$$
$$= \#\text{label} \tag{8}$$

Problem definitions. Let us now consider a K-class problem. We define a binary random variable $Y_k \in \Omega = \{0, 1\}$ as the event for a data sample to associate with the k-th label ($Y_k = 1$) or not ($Y_k = 0$). Thus, $P(Y_k)$ follows Bernoulli distribution, and we have

$$P(\Omega) = p(Y_k = 1) + p(Y_k = 0) = 1 \tag{9}$$
$$\text{and } P(Y_k) \in [0, 1], \tag{10}$$

where $\Omega = \{Y_k = 0, Y_k = 1\}$.

If we assume $Y_1, \cdots, Y_K$ are independent w.r.t. to each other, we naturally have

$$0 \leq \sum_k^K P(Y_k) \leq K, \quad \text{where } \{Y_k\} \text{ are independent.}$$

But now, given the constraint of #label $= n$, $Y_1, \cdots, Y_K$ are no longer independent. Essentially, Eq. (6) in the main paper need us to prove:

$$\sum_k^K p(Y_k = 1) = n, \qquad \text{given } \#\text{label=n.} \tag{11}$$

Proof. We denote the event set $\Phi = \Omega^K$ as the joint probability for a sample to have/not have each of the K labels as:

$$J = (Y_1, \cdots, Y_k, \cdots, Y_K) \in \Phi$$

Since each Y_k has a binary choice and they are mutually exclusive, the joint event space Φ is of size $||\Phi|| = 2^K$, and

$$\sum_{J \in \Phi} P(J) = 1 \tag{12}$$

Now, we know a sample only has n labels, *i.e.* #label=n, the non-zero probability of joint event J corresponds to the n-combinations of choosing n classes out of K to have $Y_k = 1$. We denote such a subset of the joint events space as $\Phi\binom{K}{n}$, which has a size of $\binom{K}{n} = \frac{K!}{n!(K-n)!}$.

$$\sum_{J \in \Phi\binom{K}{n}} P(J) = 1, \quad \text{when } \#\text{label=n.} \tag{13}$$

Decomposing binary event $p(Y_k = 1)$ as the summation of the joint probability like equations (2),(3),(4), we observe each $J \in \Phi\binom{K}{n}$ contributes n times in such a decomposition of each binary event $p(Y_k = 1)$ separately. Thus, according to equation (13), we get:

$$\sum_k^K p(Y_k = 1) = n \times \sum_{J \in \Phi\binom{K}{n}} P(J) = n \tag{14}$$

II. More results on One-vs.rest image classification

TABLE I: **One-vs.-rest image classification**. Comparison of top-1 error rate on CIFAR10, CIFAR100 and Tiny ImageNet, with the total number of classes being 10, 100, and 200. Binary classifiers are good for small amounts of classes. Quasibinary classifiers are competitive with softmax.

	CIFAR10	**CIFAR100**	**Tiny-ImageNet**
		ResNet18	
Binary classifiers	**4.8**	35.4	×
Quasibinary classifiers (*Ours*)	4.9	**21.9**	**42.9**
Softmax classifiers	5.2	22.2	43.3
		DenseNet40	
Binary classifiers	8.6	×	×
Quasibinary classifiers (*Ours*)	6.7	32.6	55.0
Softmax classifiers	**6.4**	**31.2**	**53.0**
		VGG16	
Binary classifiers	**6.1**	25.8	×
Quasibinary classifiers (*Ours*)	8.3	**25.5**	48.9
Softmax classifiers	8.0	25.8	**48.7**

×: Failed to converge.