On the Learning Property of Logistic and Softmax Losses for Deep Neural Networks

Xiangrui Li, Xin Li, Deng Pan, Dongxiao Zhu*
Department of Computer Science
Wayne State University
{xiangruili, xinlee, pan.deng, dzhu}@wayne.edu

Abstract

Deep convolutional neural networks (CNNs) trained with logistic and softmax losses have made significant advancement in visual recognition tasks in computer vision. When training data exhibit class imbalances, the class-wise reweighted version of logistic and softmax losses are often used to boost performance of the unweighted version. In this paper, motivated to explain the reweighting mechanism, we explicate the learning property of those two loss functions by analyzing the necessary condition (e.g., gradient equals to zero) after training CNNs to converge to a local minimum. The analysis immediately provides us explanations for understanding (1) quantitative effects of the class-wise reweighting mechanism: deterministic effectiveness for binary classification using logistic loss yet indeterministic for multi-class classification using softmax loss; (2) disadvantage of logistic loss for single-label multi-class classification via one-*vs.*-all approach, which is due to the averaging effect on predicted probabilities for the negative class (e.g., non-target classes) in the learning process. With the disadvantage and advantage of logistic loss disentangled, we thereafter propose a novel reweighted logistic loss for multi-class classification. Our simple yet effective formulation improves ordinary logistic loss by focusing on learning hard non-target classes (target *vs.* non-target class in one-*vs.*-all) and turned out to be competitive with softmax loss. We evaluate our method on several benchmark datasets to demonstrate its effectiveness.

Introduction

Deep convolutional neural networks (CNNs) trained with logistic or softmax losses (LGL and SML respectively for brevity), e.g., logistic or softmax layer followed by cross-entropy loss, have achieved remarkable success in various visual recognition tasks (LeCun, Bengio, and Hinton 2015; Krizhevsky, Sutskever, and Hinton 2012; He et al. 2016; Simonyan and Zisserman 2014; Szegedy et al. 2015). The success mainly accredits to CNN's merit of high-level feature learning and loss function's differentiability and simplicity for optimization. When training data exhibit class imbalances, training CNNs with gradient descent is biased towards learning majority classes in the conventional (unweighted) loss, resulting in performance degradation for minority classes. To remedy this issue, the class-wise reweighted loss is often used to emphasize the minority classes that can boost the predictive performance without introducing much additional difficulty in model training (Cui et al. 2019; Huang et al. 2016; Mahajan et al. 2018; Wang, Ramanan, and Hebert 2017). A typical choice of weights for each class is the inverse-class frequency.

A natural question then to ask is *what roles are those class-wise weights playing in CNN training using LGL or SML that lead to performance gain?* Intuitively, those weights make tradeoffs on the predictive performance among different classes. In this paper, we answer this question quantitatively in a set of equations that tradeoffs are on the model predicted probabilities produced by the CNN models. Surprisingly, effectiveness of the reweighting mechanism for LGL is rather different from SML. Here, we view the conventional (e.g., no reweighting) LGL or SML as a special case where all classes are weighted equally.

As these tradeoffs are related to the logistic and softmax losses, answering the above question actually leads us to answering a more fundamental question about their learning behavior: *what is the property that the decision boundary must satisfy when models are trained?* To our best knowledge, this question has not been investigated systematically, despite logistic and softmax losses are extensively exploited in deep leaning community.

While SML can be viewed as a multi-class extension of LGL for binary classification, LGL is a different learning objective when used in multi-class classification (Bishop 2006). From the perspective of learning structure of data manifold as pointed out in (Belkin, Niyogi, and Sindhwani 2006; Bishop 2006; Dong, Zhu, and Gong 2019), SML treats all class labels equally and poses a competition between true and other class labels for each training sample, which may distort data manifold; for LGL, the one-*vs.*-all approach it takes avoids this limitation as it models each target class independently, which may better capture the in-class structure of data. Though LGL enjoys such merits, it is rarely adopted in existing CNN models. The property that LGL and SML decision boundaries must satisfy further reveals the difference between LGL and SML (see Eq. (9), (10) with anal-

*Corresponding author

ysis). If used for the multi-class classification problem, we can identify two issues for LGL. Compared with SML, LGL may introduce data imbalance, which can degrade model performance as sample size plays an important role in determining decision boundaries. More importantly, since the one-*vs.*-all approach in LGL treats all other classes as the negative class, which is of a multi-modal distribution (Li and Zhu 2018; Li, Zhu, and Dong 2018), the averaging effect of the predicted probabilities of LGL can hinder learning discriminative feature representations to other classes that share some similarities with the target class.

Our contribution can be summarized as follows:

- We provide a theoretical derivation on the relation among sample's predicted probability (once CNN is trained), class weights in the loss function and sample size in a system of equations. Those equations explaining the reweighting mechanism are different in effect for LGL and SML.
- We depict the learning property for LGL and SML for classification problems based on those probability equations. Under mild conditions, the expectation of model predicted probabilities must maintain a relation specified in Eq (9).
- We identify that the multi-modality neglect problem in LGL is the main obstacle for LGL in multi-class classification. To remedy this problem, we propose a novel learning objective, in-negative class reweighted LGL, as a competitive alternative for LGL and SML.
- We conduct experiments on several benchmark datasets to demonstrate the effectiveness of our method.

Related Work

With recent explosion in computational power and availability of large scale image datasets, deep learning models have repeatedly made breakthroughs in a wide spectrum of tasks in computer vision (LeCun, Bengio, and Hinton 2015; Goodfellow, Bengio, and Courville 2016). Those advancements include new CNN architectures for image classification(Krizhevsky, Sutskever, and Hinton 2012; He et al. 2016; Simonyan and Zisserman 2014; Szegedy et al. 2015), objective detection and segmentation (Ren et al. 2015; Ronneberger, Fischer, and Brox 2015), new loss functions (Dong, Zhu, and Gong 2019; Zhang and Sabuncu 2018) and effective training techniques to improve CNN performance (Srivastava et al. 2014; Ioffe and Szegedy 2015).

In those supervised learning problems, CNNs are mostly trained with loss functions such as LGL and SML. In practice, class imbalance naturally emerges in real-world data and training CNN models directly on those datasets may lead to poor performance. This phenomenon is referred as the imbalanced learning problem (He and Garcia 2008). To tackle this problem, cost-sensitive method (Elkan 2001; Zhou and Liu 2010) is the widely-adopted approach in current training practices as they don't introduce any obstacles in the backpropagation algorithm. One of the most popular methods is class-wise reweighting loss function based on LGL and SML. For example, (Huang et al. 2016; Wang, Ramanan, and Hebert 2017) reweight each class by its inverse-class frequency. In some long-tailed datasets, a smoothed version of weights is adopted (Mahajan et al. 2018; Mikolov et al. 2013), which emphasizes less on minority classes, such as the square root of inverse-class frequency. More recently, (Cui et al. 2019) proposed a weighting strategy based on the calculation of effective sample size. In the context of learning from noisy data, (Zhang and Sabuncu 2018) provides analysis on the weighted SGL showing close connection to the mean absolute error (MAE) loss. However, what role class-wise weights play in LGL and SML is not explained in previous works. In this paper, we provide a theoretical explication on how the weights control the tradeoffs among model predictions.

If we decompose the multi-class classification as multiple binary classification sub-tasks, LGL can also be used as the objective function via one-*vs.*-all approach (Hastie et al. 2005; Bishop 2006), which is however rarely adopted in existing works of deep learning. Motivated to understand class-wise reweighted LGL and SML, our analysis further leads us to a more profound discovery in the properties of decision boundaries for LGL and SML. Previous work in (Dong, Zhu, and Gong 2019) showed that the learning objective using LGL is quite different from SML as each class is learned independently. They identified the negative class distraction (NCD) phenomenon that might be detrimental to model performance when using LGL in multi-class classification. From our analysis, the NCD problem can be partially explained that LGL treats the negative class (e.g., non-target classes) as a single class and ignores its multi-modality. If there exists one non-target class that share some similarity with the target class, CNN trained with LGL may make less confident predictions for that non-target class (e.g., probability of belonging to the negative class is small) as its predicted probabilities are averaged out due to other non-target classes with confident predictions. Consequently, samples from that specific non-target class can be misclassified into the target class, resulting in large predictive error.

Analysis on LGL and SML

In this section, we provide a theoretical explanation for the class-wise weighting mechanism and depict the learning property of LGL and SML losses.

Notation Let $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ be the set of training samples of size N, where $\boldsymbol{x}_i \in \mathbf{R}^p$ is the p-dimensional feature vector and $y_i = k (k = 0, \cdots, K-1)$ is the true class label, and $S_k = \{(\boldsymbol{x}_i, y_i) : y_i = k\}$ the subset of D for the k-th class. The bold $\boldsymbol{y}_i = (y_i^0, \cdots, y_i^{K-1})$ is used to represent the one-hot encoding for y_i: $y_i^k = 1$ if $y_i = k$, 0 otherwise. $N_k = |S_k| (k = 0, \cdots, K-1)$ is used to represent sample size for the k-th class and hence $\sum_k N_k = N$. The maximum size is denoted as $N_{\max} = \max_{k=0,\cdots,K-1}(N_k)$.

Preliminaries

For classification problem, the probability for a sample $\boldsymbol{x}$ belonging to one class is modeled by logistic (e.g., sigmoid)

for binary classification

$$p(y=1|\boldsymbol{x};\boldsymbol{\theta}) = \frac{1}{1+\exp(-z)},$$
$$p(y=0|\boldsymbol{x};\boldsymbol{\theta}) = 1 - p(y=1|\boldsymbol{x}),$$

and by softmax for multi-class classification

$$p(y=k|\boldsymbol{x};\boldsymbol{\theta}) = \frac{\exp(z_k)}{\sum_{j=0}^{K-1}\exp(z_j)},$$

where all z's are the logits for $\boldsymbol{x}$ modeled by CNN with parameter vector $\boldsymbol{\theta}$. It is worth noting that softmax is equivalent to logistic in binary classification as can be seen from

$$p(y=1|\boldsymbol{x}) = \frac{\exp(z_1)}{\exp(z_0)+\exp(z_1)} = \frac{1}{1+\exp(-(z_1-z_0))}.$$

Hence, without loss of generality, we write class-wise reweighted LGL ($K=2$) and SML ($K\geq 3$) in a unified form as follows

$$L(\boldsymbol{\theta}) = -\sum_{k=0}^{K-1}\lambda_k \sum_{i_k\in S_k}\log f_k(\boldsymbol{\theta};\boldsymbol{x}_{i_k}) = -\sum_{k=0}^{K-1}\lambda_k L_k(\boldsymbol{\theta}), \tag{1}$$

where each $f_k(\boldsymbol{\theta};\boldsymbol{x}_i) = p(y_i=k|\boldsymbol{x}_i)$ is the CNN predicted probability of sample $\boldsymbol{x}_i$ belonging to the k-th class; λs are weight parameters to control each class's contribution in the loss. When all λs are equal, $L(\boldsymbol{\theta})$ is the conventional cross-entropy loss and minimizing it is equivalent to maximizing likelihood. If the training data are imbalanced, a different setup of λs is used, usually classes with smaller sizes are assigned with higher weights. Generally, λs are treated as hyperparameters and selected by cross-validation.

We emphasize here that using logistic function for multi-class ($K\geq 3$) is a different learning objective from softmax in this case as the classification problem is essentially reformulated as K binary classification sub-problems.

Key Equations for Weights λs

Assume that CNN's output layer, after convolutional layers, is a fully connected layer of K neurons with bias terms, then the predicted probability for sample $\boldsymbol{x}$ is given by the softmax activation:

$$f_k(\boldsymbol{x}) = \frac{\exp(\boldsymbol{W_k}\boldsymbol{h_x}+b_k)}{\sum_{j=1}^{K}\exp(\boldsymbol{W}_j\boldsymbol{h_x}+b_j)} \quad (k=0,\cdots,K-1), \tag{2}$$

where $\boldsymbol{h_x}$ is the feature representation of $\boldsymbol{x}$ extracted from convolutional layers, $\boldsymbol{W}_k$ and b_k are parameters of the k-th neuron in the output layer. For notational simplicity, we have dropped $\boldsymbol{\theta}$ in $f_k(\boldsymbol{x})$.

After CNN is trained, we assume that the reweighted SML $L(\boldsymbol{\theta})$ is minimized to local optimum $\boldsymbol{\theta}^*$. By optimization theory, a necessary condition is that the gradient of $L(\boldsymbol{\theta})$ is zero at $\boldsymbol{\theta}=\boldsymbol{\theta}^*$[1]:

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = \boldsymbol{0} \iff \sum_{k=1}^{K}\lambda_k\frac{\partial L_k}{\partial \boldsymbol{\theta}} = \boldsymbol{0}. \tag{3}$$

[1]More strictly, zero is in the subgradient of $L(\boldsymbol{\theta})$ at $\boldsymbol{\theta}^*$. But this doesn't affect the following analysis.

We specifically consider $L_1(\boldsymbol{\theta})$ for the 1-st class with respect to one component η of $\boldsymbol{\theta}$. Then with chain rule, the necessary condition above gives:

$$\begin{aligned}&\lambda_1\frac{\partial L_1}{\partial\eta} + \sum_{k=2}^{K}\lambda_k\frac{\partial L_k}{\partial\eta} = 0 \iff \\ &\lambda_1\sum_{i_1\in S_1}\frac{1}{f_{1,i_1}}\frac{\partial f_{1,i_1}}{\partial\eta} + \sum_{k=2}^{K}\lambda_k\sum_{i_k\in S_k}\frac{1}{f_{k,i_k}}\frac{\partial f_{k,i_k}}{\partial\eta} = 0,\end{aligned} \tag{4}$$

where we use $f_{j,i_k} = f_j(\boldsymbol{x}_{i_k})$ given by Eq. (2).

Let $\boldsymbol{\sigma}(\boldsymbol{z})$ be the softmax function of $\boldsymbol{z}=(z_1,\cdots,z_K)$ with each component $\boldsymbol{\sigma}(z_k) = \exp(z_k)/\sum_i\exp(z_i)$, its derivative is

$$\frac{\partial\boldsymbol{\sigma}(z_k)}{\partial z_i} = \begin{cases}\boldsymbol{\sigma}(z_k)(1-\boldsymbol{\sigma}(z_k)), & i=k\\ -\boldsymbol{\sigma}(z_k)\boldsymbol{\sigma}(z_i), & i\neq k.\end{cases} \tag{5}$$

Denoting $a_{j,i_k} = \boldsymbol{W_j}\boldsymbol{h}_{\boldsymbol{x}_{i_k}} + b_j$ as the j-th logit in Eq. (2) for sample $\boldsymbol{x}_{i_k}$, then $f_{j,i_k} = \boldsymbol{\sigma}(a_{j,i_k})(j=0,\cdots,K-1)$. Again with chain rule and Eq. (5):

$$\begin{aligned}\frac{\partial f_{k,i_k}}{\partial\eta} &= \sum_{j=1}^{K}\frac{\partial f_{k,i_k}}{\partial a_{j,i_k}}\frac{\partial a_{j,i_k}}{\partial\eta}\\ &= f_{k,i_k}(1-f_{k,i_k})\frac{\partial a_{k,i_k}}{\partial\eta} - f_{k,i_k}\sum_{j\neq k}f_{j,i_k}\frac{\partial a_{j,i_k}}{\partial\eta}.\end{aligned} \tag{6}$$

Since Eq. (4) holds valid for any component η of $\boldsymbol{\theta}$, we specifically consider the case when $\eta=b_1$. Therefore we have $\partial a_{1,i_k}/\partial b_1 = 1$ and $\partial a_{j,i_k}/\partial b_1 = 0(j=2,\cdots,K)$. Then Eq. (6) becomes:

$$\frac{\partial f_{k,i_k}}{\partial b_1} = \begin{cases}f_{1,i_1}(1-f_{1,i_1}), & k=1\\ -f_{k,i_k}f_{1,i_k}, & k\neq 1.\end{cases} \tag{7}$$

Plug Eq. (7) back into Eq. (4) and rearrange the terms, we have

$$\lambda_1\sum_{i_1\in S_1}(1-f_{1,i_1}) = \sum_{k=2}^{K}\lambda_k\sum_{i_k\in S_k}f_{1,i_k}. \tag{8}$$

With the same calculations, we can obtain other $K-1$ similar equations, each of which corresponds to one class. Remember f_{j,i_k} is the probability of sample $\boldsymbol{x}_{i_k}$ from the k-th class being predicted into the j-th class, and Eq. (8) reveals the quantitative relation between weights λs, model predicted probabilities and training samples. Notice that CNN is often trained with L_2 regularization to prevent overfitting. If the bias term b_ks are not penalized, Eq. (8) still holds valid. Another possible issue is that the calculation relies on the use of bias terms b_k in the output layer. As using bias increases CNN's flexibility and is not harmful to CNN performance, our analysis is still applicable to a wide range of CNN models trained with cross-entropy loss.

We observe in Eq. (8), $\sum_{i_1\in S_1}(1-f_{1,i_1})/N_1$ (approximately) represents the expected probability of CNN incorrectly predicting a sample of class 1 and $\sum_{i_k\in S_k}f_{1,i_k}/N_k$

the expected probability of CNN misclassifying a sample of class $k (k \neq 1)$ into class 1. If we assume that the training data can well represent the true data distribution that testing data also follow, the learning property of trained CNN shown in Eq. (8) can be generalized to testing data.

More specifically, since the CNN model is a continuous mapping and the softmax output is bounded between 0 and 1, by the uniform law of large numbers (Newey and McFadden 1994), we have the following system of K equations *once CNN is trained*:

$$\begin{cases} \lambda_0 N_0(1-\bar{p}_{0\to 0}) \approx \sum_{k\neq 0} \lambda_k N_k \bar{p}_{k\to 0} \\ \qquad\qquad \vdots \\ \lambda_{K-1} N_{K-1}(1-\bar{p}_{K-1\to K-1}) \approx \sum_{k\neq K-1} \lambda_k N_k \bar{p}_{k\to K-1}, \end{cases} \tag{9}$$

where for indices i and j, $\bar{p}_{i\to j}$ represents the expected probability of CNN predicting a sample from class i into class j:

$$\bar{p}_{i\to j} = \mathrm{E}_{\boldsymbol{x}\sim P(\boldsymbol{x}|y=i)} f_j(\boldsymbol{x}),$$

where $P(\boldsymbol{x}|y=i)$ is the true data distribution for the i-th class.

Binary Case with LGL For binary classification problem $(K=2)$, Eq. (9) gives us the following relation about CNN predicted probabilities:

$$\frac{1-\bar{p}_{0\to 0}}{\bar{p}_{1\to 0}} \approx \frac{\lambda_1 N_1}{\lambda_0 N_0}. \tag{10}$$

- In the conventional LGL where each class is weighted equally ($\lambda_0 = \lambda_1$), Eq. (10) becomes $1-\bar{p}_{0\to 0} = N_1\bar{p}_{1\to 0}/N_0$. If data exhibit severe imbalance, say $N_0 = 10N_1$, then we must have $(\bar{p}_{1\to 0} < 1)$

  $$\bar{p}_{0\to 0} = 1 - \frac{\bar{p}_{1\to 0}}{10} > 0.9.$$

  If $t = 0.5$ is the decision making threshold, this implies that the trained neural network can correctly predict a majority class (e.g., class 0) sample, confidently (at least) with probability 0.9, on average. However, for minority class, the predictive performance is more complex which depends on the trained model and data distribution. For example, if two classes can be well separated and the model made very confident predictions, say $\bar{p}_{0\to 0} = 0.98$, then we must have $\bar{p}_{1\to 1} = 0.8$ for the minority class, implying a good predictive performance on class 1. If $\bar{p}_{0\to 0} = 0.92$, then we have $\bar{p}_{1\to 1} = 0.2$. This means the predicted probability of a minority sample being minority is 0.2 on average. Hence, the classifier must misclassify most minority samples $(0.2 < 0.5)$, resulting in very poor predictive accuracy for minority class.

- If LGL is reweighted using inverse-class frequencies, $\lambda_0 = 1/N_0$ and $\lambda_1 = 1/N_1$, the equation above is equivalent to $\bar{p}_{0\to 0} = 1-\bar{p}_{1\to 0} = \bar{p}_{1\to 1}$. Since predictions are made by $y = \arg\max_i f_i(\boldsymbol{x})$ and $f_1(\boldsymbol{x}) > f_0(\boldsymbol{x})$ means $f_1(\boldsymbol{x}) > 0.5$, we can have a deterministic relation: if either class 0 or 1 can be well predicted (e.g., $\bar{p}_{i\to i} > 0.5$), reweighting by class inverse frequencies can guarantee performance improvement for the minority class. However, the extent of "goodness" depends on the separability of the underlying data distributions of the two classes.

λ_0	$\frac{1}{2}$	$\frac{N_0}{N_0+N_1}$	$\frac{2N_0}{(2N_0+N_1)}$
RHS	10	1	0.5
LHS (Sim1)	10.05 (1.13)	1.00 (0.09)	0.50 (0.04)
LHS (Sim2)	10.12 (0.67)	1.01 (0.05)	0.50 (0.03)

Table 1: Simulation results (along with standard deviation) for Eq. (10) over 100 runs, $\lambda_1 = 1-\lambda_0$. RHS represents theoretical value on the right-hand side of (10); LHS the simulated value on the left hand side.

Simulations for Eq. (10) We conduct simulations under two settings for checking Eq. (10). The imbalance ratio is set to 10 in training data ($N_0 = 1000, N_1 = 10000$), testing data size is $(1000, 1000)$; both training and testing data follow the same data distribution. As the property only relies on the last fully connected hidden layer, we use the following setup:

- Sim1: $P_1(x|y=1) = \mathcal{N}(-1.5,1) + \mathcal{U}(0,0.5)$, $P_2(x|y=0) = \mathcal{N}(1.5,1) + \mathcal{U}(-0.5,0)$. Logistic regression is fitted. $\mathcal{N}$ and $\mathcal{U}$ represents normal and uniform distribution respectively.

- Sim2: $P_1(\boldsymbol{x}|y=1) = \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1)$, $P_0(\boldsymbol{x}|y=0) = \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\sigma}_0)$, where $\boldsymbol{\mu}_1 = (0,0,0)$, $\boldsymbol{\mu}_0 = (1,1,1)$, $\boldsymbol{\sigma}_1 = 1.2\boldsymbol{I}$, $\boldsymbol{\sigma}_0 = \boldsymbol{I}$. A one-hidden-layer forward neural network of layer size $(3,10,1)$ with sigmoid activation.

Table 1 shows simulation results under three λ settings. We see from the Table that simulated values match with the theoretical values accurately, demonstrating the correctness of Equation (10).

Multi-class Case with SML Because $\sum_k \bar{p}_{i\to k} = 1$ and Eq. (9) has $K(K-1)$ variables with only K equations, we can't exactly solve it quantitatively for a relation among those $\bar{p}_{i\to j}$'s when $K > 2$. For the special case when weights are chosen as the inverse-class frequencies $\lambda_k = 1/N_k$, considering for class 1, we have $(1-\bar{p}_{1\to 1}) \approx \sum_{k\neq 1} \bar{p}_{k\to 1}$. Multi-class classification $(K > 2)$ does not have a deterministic relation as in the binary case, as predictions are made by $y = \arg\max_i f_i(\boldsymbol{x})$ and we don't have a decisive threshold for decision making (like the 0.5 in binary case). Our findings match the results in (Zhou and Liu 2010) in the sense that class-wise reweighting for multi-class is indeterministic. However, our results are solely based on the mathematical property of the backpropagation algorithm from optimization theory whereas (Zhou and Liu 2010) is based on decision theory.

Learning property of LGL and SML As the class-wise reweighting mechanism is explained in Eq. (9), those equations also reveal the property of decision boundaries for LGL and SML. For comparison, the decision boundary of support vector machine (SVM) (Cortes and Vapnik 1995) is determined by those support vectors that maximize the margin and those samples with larger margin have no effects on the position of decision boundary. On the contrary, all samples have their contribution to the decision boundary in LGL and SML so that their averaged probabilities that the model produces must satisfy Eq. (9). In particular for

the binary case, we can see that if classes are balanced, the model must make correct predictions with equal confidence for the positive and negative classes, on average; whereas for imbalanced data, the decision boundary will be pushed towards the minority class in a position with Eq (10) always maintained. Another observation is that if the expectation of model predicted probabilities doesn't match with its mode (e.g skewed distribution), the magnitude of tradeoff between performance of the majority and minority class depends on the direction of skewness. If the distribution of the majority class skews away from the decision boundary, upweighting minority class will boost model performance at a small cost of performance degradation for the majority class than if it skews towards the decision boundary. This implies that estimating the shape of data distribution in the latent feature space and choosing the weights accordingly would be very helpful to improve model overall performance.

In-negative Class Reweighted LGL

In this section, we focused on LGL for multi-class classification via one-*vs.*-all approach. In addition to the theoretical merits of LGL mentioned in the introduction section that LGL is capable of better capturing the structure of data manifold than SML, the guarantee of achieving good performance after properly reweighting (e.g., Eg.(10)) is also desirable as the one-*vs.*-all approach naturally introduces data imbalance issue.

Multi-modality Neglect Problem In spite of those merits of LGL, it also introduces the multi-modality neglect problem for multi-class classification. Since the expectation of model predicted probability must satisfy Eq (10) for LGL, the averaging effect might be harmful for model performance. In the one-*vs.*-all approach, the negative class consists of all the remaining non-target classes, which follows a multi-modal distribution (one modality for each non-target class). LGL treats all non-target classes equally in the learning process. If there is a hard non-target class that shares non-trivial similarity with the target class, its contribution in LGL might be averaged out by other easy non-target classes. In other words, those easy non-target classes (e.g., correctly predicted as the negative class with high probabilities) would compensate the predicted probability of the hard non-target class so that the probabilistic relation in Eq (10) is maintained. Consequently, model could incorrectly predict samples from the hard non-target class into the target class, inducing large predictive error for that class. This phenomenon is not desirable as we want LGL to pay more attention on the separation of the target-class with that hard class, meanwhile maintain the separation from the remaining easy non-target classes.

To this end, we propose an improved version of LGL to reweight each non-target class's contribution within the negative class. Specifically, for the target class k (e.g., positive class, labeled as $y = 1$) and all non-target classes (e.g., negative class, labeled as $y = -1$), a two-level reweighting mechanism is applied in LGL, which we term as **in-negative-class reweighted LGL (LGL-INR)**:

$$L_k^{\text{INR}}(\boldsymbol{\theta}) = -\frac{1}{N_k}\sum_{\boldsymbol{x}\in S_k}\log p(y=1|\boldsymbol{x};\boldsymbol{\theta}) - \sum_{j=0,j\neq k}^{K-1}\lambda_j\frac{1}{N_j}\sum_{\boldsymbol{x}\in S_j}\log(1-p(y=1|\boldsymbol{x};\boldsymbol{\theta})), \tag{11}$$

where $p(y=1|\boldsymbol{x};\boldsymbol{\theta})$ is the predicted probability of sample $\boldsymbol{x}$ belonging to the positive class and λ_j is the weight for class j as a sub-class of the negative class.

The first reweighting is at the level of positive *vs.* negative class. If we require $\sum_j \lambda_j = 1$, using inverse-frequencies will maintain the balance between the positive and negative class, as one-*vs.*-all is likely to introduce class imbalance. The second level of reweighting is within the negative class: we upweight the contribution of a hard sub-class by assigning a larger λ, making LGL-IGR focus more on the learning for that class.

Choice of λs When there are a large number of classes, treating all λs as hyperparameters and selecting the optimal values are not feasible in practice as we generally don't have the prior knowledge about which classes are hard. Instead, we adopt a strategy that assigns the weights during the training process. For each non-target class $j(j \neq k)$, let S_j^{MB} be the subset of S_j in the mini-batch, we use the mean predicted probability

$$\bar{p}_j = \frac{1}{|S_j^{\text{MB}}|}\sum_{\boldsymbol{x}\in S_j^{\text{MB}}} p(y=1|\boldsymbol{x},\boldsymbol{\theta})$$

as the class-level hardness measurement. A larger $\bar{p}_j$ implies class j is harder to separate from the target class k. We then transform those $\bar{p}_j$'s using softmax to get λ_j:

$$\lambda_j = \frac{\exp(\beta\bar{p}_j)}{\sum_{i\neq k}\exp(\beta\bar{p}_i)},$$

where $\beta \geq 0$ is the temperature that can smooth ($0 \leq \beta \leq 1$) or sharpen ($\beta > 1$) each non-target class's contribution (Chorowski et al. 2015). LGL-INR adaptively shifts its learning focus to those hard classes, meanwhile keep attentive on those easy classes. Note that this strategy only introduces one extra parameter in LGL-INR.

With the competition mechanism imposed by $\sum \lambda_j = 1$, LGL-INR can be viewed as a smoothed learning objective between the one-*vs.*-one and one-*vs.*-all approach: when $\beta = 0$, $\lambda_j = 1/K - 1$, all non-target classes are weighted equally, which is the in-negative-class balanced LGL using inverse-class frequencies; when β is very large, λ_j concentrates on the hardest class (e.g., $\lambda_j \approx 1$) and LGL-INR approximately performs one-*vs.*-one classification. We don't specifically fine-tune the optimal value of β and $\beta = 1$ works well in our experiments.

Experiments

We evaluate LGL-INR on several benchmark datasets for image classification. Note that in our experiments, applying LGL in multi-class classification naturally introduces

Model	Architecture
CNN2C	CV(C20K5S1)-MP(K2S2)-CV(C50K5S1)-MP(K2S2)-800-10
CNN5C	CV(C32K3S1)-BN-CV(C64K3S1)-BN-CV(C128K3S1)-MP(K2S2)-CV(C256K3S1)-BN-CV(C512K3S1)-MP(K8S1)-512-10

Table 2: CNN architectures used for MNIST-type datasets. C-channel represents number, K-kernel size, S-stride, BN-batch normalization and MP-max pooling

data imbalance which is handled in our LGL-INR formulation. Our primary goal here is to demonstrate that LGL-INR can be used as a drop-in replacement for LGL and SML with competitive or even better performance, rather than outperform the existing best models using extra training techniques. For fair comparison, all loss functions are evaluated in the same test setting. Code is made publicly available at *https://github.com/Dichoto/LGL-INR*.

Experiment Setup

Dataset We perform experiments on four MNIST-type datasets, MNIST, Fashion-MNIST (FMNIST) (Xiao, Rasul, and Vollgraf 2017), Kuzushiji-MNIST (KMNIST) (Clanuwat et al. 2018) and CIFAR10. FMNIST and KMNIST are intended as drop-in replacements for MNIST which are harder than MNIST. Both datasets are gray-scale images consisting of 10 classes of clothing and Japanese character respectively. CIFAR10 consists of colored images of size 32×32 from 10 objects.

Model setup We test three loss functions on each dataset with different CNN architectures. For MNIST-type datasets, two CNNs with simple configurations are used. The first one (CNN2C) has two convolution layers and the other one (CNN5C) has 5 convolution layers with batch normalization (Ioffe and Szegedy 2015). For CIFAR10, we use MobilenetV2 (Howard et al. 2017) and Resnet-18 (He et al. 2016) with publicly available implementations.

Implementation details All models are trained with the standard stochastic gradient descent (SGD) algorithm. The training setups are as follows. For MNIST-type data, the learning rate is set to 0.01, the momentum is 0.5, batch size 64, number of epoch is 20. We don't perform any data augmentation. For CIFAR data, we train the models with 100 epochs and set batch size to 64. The initial learning rate is set to 0.1, and divide it by 10 at 50-th and 75-th epoch. The weight decay is 10^{-4} and the momentum in SGD is 0.9. Data augmentation includes random crop and horizontal flip. We train all models without pretraining on large-scale image data. Model performance is evaluated by the top-1 accuracy rate and we report this metric on the testing data from the standard train/test split of those datasets for fair performance evaluation. For LGL-INR, we report the results using $\beta = 1$.

Predictive Results

Table 3 and Table 4 shows the classification accuracy using LGL, SML and LGL-INR on the MNIST-type and CIFAR10 dataset respectively. From the table, we can observe

Model	Loss	MNIST	FMNIST	KMNIST
CNN2C	LGL	99.15	89.44	94.37
	SML	99.09	**91.15**	95.13
	LGL-INR	**99.29**	**91.15**	**96.43**
CNN5C	LGL	99.36	92.35	96.35
	SML	99.47	93.15	96.39
	LGL-INR	**99.63**	**93.54**	**97.46**

Table 3: Predictive top-1 accuracy rate (%) on the standard testing data of MNIST-type datasets.

Loss	MobilenetV2	Resnet18
LGL	92.40	91.55
SML	91.11	91.32
LGL-INR	**93.34**	**93.68**

Table 4: Predictive top-1 accuracy rate (%) on the standard testing data of CIFAR10 using different models.

that for all three loss functions, model with larger capacity yields higher accuracy. On MNIST-type data, LGL yields overall poorer performance than SML. This is because in those datasets, some classes are very similar to each other (like shirt *vs.* coat in FMNIST) and the negative class consists of 9 different sub-classes. Hence the learning focus of LGL may get distracted from the hard sub-classes due to the averaging behavior of LGL as shown in Eq (9). However, SML doesn't suffer this problem as all negative sub-classes are treated equally. On CIFAR10, LGL achieves better accuracy than SML. This is possibly due to the lack of very similar classes as in MNIST-type data. This observation demonstrates LGL's potential as a competitive alternative to SML in some classification tasks.

On the other hand, LGL-INR adaptively pays more attention on the hard classes while keeps its separation from easy classes. This enables LGL-IRN to outperform LGL and SML notably. Comparing LGL-IRN with LGL, we see that the multi-modality neglect problem deteriorates LGL's ability of learning discriminative features representation, which can be relieved by the in-negative class reweighing mechanism; comparing LGL-IRN with SML, focusing on learning hard classes (not restricted to classes similar to the target class) is beneficial. Also, the adaptive weight assignment in the training process doesn't require extra effort on the weight selection, making our method widely applicable.

Further Analysis

We check the predictive behavior of LGL-INR in detail by looking at the confusion matrix on testing data. Here, we use CNN2C and KMNIST dataset as an example. Fig. 1 show the results. We observe that for LGL, Class 1 and 2 have the lowest accuracy among 10 classes. By shifting LGL's learning focus on hard classes, LGL-INR significantly improves model performance on class 1 and 2. This is within our expectation backed by the theoretical depiction of LGL's learning property. SML does not have the multi-modality neglect problem as each class is treated equally in the learning process, yet it also does not pay more attention to the hard

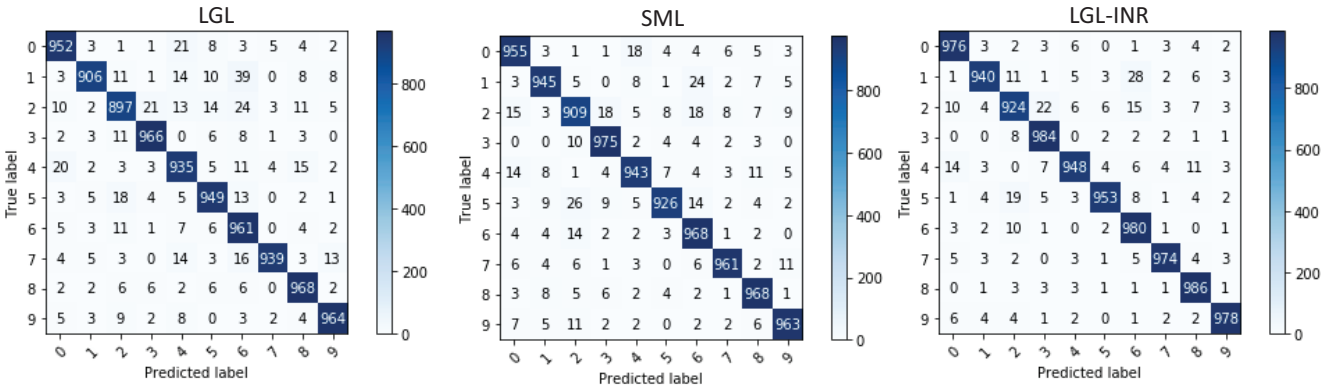


Figure 1: Confusion matrix on KMNIST testing data for LGL, SML and LGL-INR. Model: CNN2C. See Table 3 for overall accuracy. Notably, LGL-INR outperforms LGL in all 10 classes and SML in 9 classes except Class 1 (LGL-INR 940 *vs.* SML 945), in terms of per-class accuracy.

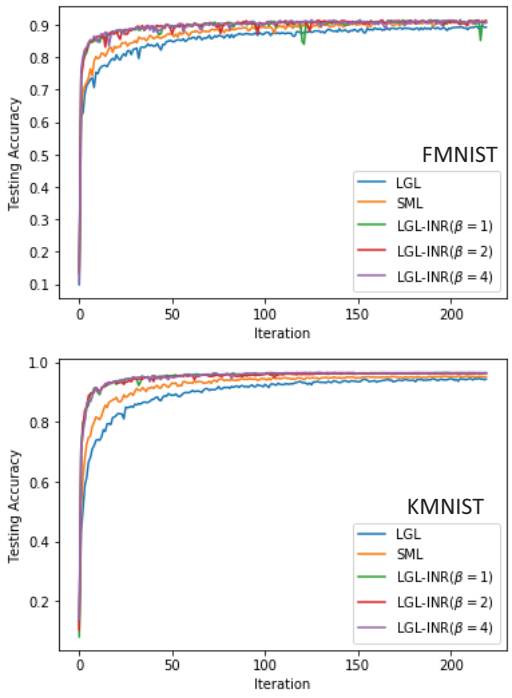


Figure 2: Testing top-1 accuracy on FMNIST and MNIST.

classes. This makes LGL-INR advantageous: LGL-INR outperforms SML on 9 classes out of 10. For example, class 0 have 18 samples misclassified into class 4 whereas only 6 are misclassified in LGL-INR.

Figure 2 displays the training accuracy curve for LGL, SML and LGL-INR on FMNIST and KMNIST. Under the same training protocol, LGL-INR achieves slightly faster convergence rate than SML and LGL with comparative (FMNIST) or better (KMNIST) performance, implying that focusing on learning hard classes may facilitate model training process.

β	1	2	4
Accuracy	96.43	96.29	96.43

Table 5: Accuracy of different β values on KMNIST. Model: CNN2C.

We also check the sensitivity of the temperature parameter β in LGL-INR weighting mechanism. Mathematically, a large or small value for β is not desirable as the LGL-INR is reduced to an approximate one-*vs.*-one or a class-balanced learning objective. We test $\beta = 1, 2, 4$ on KMNIST. As shown in Table 5 and Fig. 2, model performance is not sensitive to β in this range, making LGL-INR a competitive alternative to LGL or SML without introducing much hyper-parameter tuning.

Conclusion

In this paper, motivated to explain the class-wise reweighting mechanism in LGL and SML, we theoretically deprived a system of probability equations that depicts the learning property of LGL and SML, as well as explains the roles of those class-wise weights in the loss function. By examining the difference in the effects of the weight mechanism on LGL and SML, we identify the multi-modality neglect problem is the major obstacle that can negatively affect LGL's performance in multi-class classification. We remedy this shortcoming of LGL with a in-negative-class reweighting mechanism. The proposed method shows its effectiveness on several benchmark image datasets. For future works, we plan to incorporate the estimation of data distribution and use the reweighting mechanism of LGL-INR at the sample level in the model training process to further improve the efficacy of the reweighting mechanism.

Acknowledgement

This work is supported by the National Science Foundation under grant no. IIS-1724227.

References

Belkin, M.; Niyogi, P.; and Sindhwani, V. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research* 7(Nov):2399–2434.

Bishop, C. M. 2006. *Pattern recognition and machine learning*. springer.

Chorowski, J. K.; Bahdanau, D.; Serdyuk, D.; Cho, K.; and Bengio, Y. 2015. Attention-based models for speech recognition. In *Advances in neural information processing systems*, 577–585.

Clanuwat, T.; Bober-Irizar, M.; Kitamoto, A.; Lamb, A.; Yamamoto, K.; and Ha, D. 2018. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*.

Cortes, C., and Vapnik, V. 1995. Support-vector networks. *Machine learning* 20(3):273–297.

Cui, Y.; Jia, M.; Lin, T.-Y.; Song, Y.; and Belongie, S. 2019. Class-balanced loss based on effective number of samples. *arXiv preprint arXiv:1901.05555*.

Dong, Q.; Zhu, X.; and Gong, S. 2019. Single-label multi-class image classification by deep logistic regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3486–3493.

Elkan, C. 2001. The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, 973–978. Lawrence Erlbaum Associates Ltd.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. *Deep learning*. MIT press.

Hastie, T.; Tibshirani, R.; Friedman, J.; and Franklin, J. 2005. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer* 27(2):83–85.

He, H., and Garcia, E. A. 2008. Learning from imbalanced data. *IEEE Transactions on Knowledge & Data Engineering* 9:1263–1284.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Huang, C.; Li, Y.; Change Loy, C.; and Tang, X. 2016. Learning deep representation for imbalanced classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5375–5384.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *nature* 521(7553):436.

Li, X., and Zhu, D. 2018. Robust feature selection via l2, 1-norm in finite mixture of regression. *Pattern Recognition Letters* 108:15–22.

Li, X.; Zhu, D.; and Dong, M. 2018. Multinomial classification with class-conditional overlapping sparse feature groups. *Pattern Recognition Letters* 101:37–43.

Mahajan, D.; Girshick, R.; Ramanathan, V.; He, K.; Paluri, M.; Li, Y.; Bharambe, A.; and van der Maaten, L. 2018. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 181–196.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Newey, W. K., and McFadden, D. 1994. Large sample estimation and hypothesis testing. *Handbook of econometrics* 4:2111–2245.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 234–241. Springer.

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

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15(1):1929–1958.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9.

Wang, Y.-X.; Ramanan, D.; and Hebert, M. 2017. Learning to model the tail. In *Advances in Neural Information Processing Systems*, 7029–7039.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Zhang, Z., and Sabuncu, M. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in neural information processing systems*, 8778–8788.

Zhou, Z.-H., and Liu, X.-Y. 2010. On multi-class cost-sensitive learning. *Computational Intelligence* 26(3):232–257.

Improving Adversarial Robustness via Probabilistically Compact Loss with Logit Constraints

Xin Li*, Xiangrui Li*, Deng Pan*, Dongxiao Zhu†
Department of Computer Science, Wayne State University, Detroit, MI 48202
{xiangruili, xinlee, pan.deng, dzhu}@wayne.edu

Abstract

Convolutional neural networks (CNNs) have achieved state-of-the-art performance on various tasks in computer vision. However, recent studies demonstrate that these models are vulnerable to carefully crafted adversarial samples and suffer from a significant performance drop when predicting them. Many methods have been proposed to improve adversarial robustness (e.g., adversarial training and new loss functions to learn adversarially robust feature representations). Here we offer a unique insight into the predictive behavior of CNNs that they tend to misclassify adversarial samples into the most probable false classes. This inspires us to propose a new Probabilistically Compact (PC) loss with logit constraints which can be used as a drop-in replacement for cross-entropy (CE) loss to improve CNN's adversarial robustness. Specifically, PC loss enlarges the probability gaps between true class and false classes meanwhile the logit constraints prevent the gaps from being melted by a small perturbation. We extensively compare our method with the state-of-the-art using large scale datasets under both white-box and black-box attacks to demonstrate its effectiveness. The source codes are available at https://github.com/xinli0928/PC-LC.

Introduction

Convolutional neural networks (CNNs) have achieved significant progress for various challenging tasks in computer vision, including image classification (Li et al. 2020), semantic segmentation (He et al. 2017), and image generation (Goodfellow et al. 2014). Despite their success, CNNs are highly vulnerable to adversarial samples (Szegedy et al. 2013). With imperceptibly small perturbation added to a clean image, adversarial samples can drastically change models' prediction, resulting in a significant drop in CNN's predictive performance. This phenomenon poses a serious threat to security-critical applications of deep learning, such as autonomous driving (Al-Qizwini et al. 2017), surveillance system (Sreenu and Durai 2019), and medical imaging system (Li and Zhu 2020). Furthermore, studies have shown that adversarial robustness is also a key property to obtain human interpretation in computer vision and other application fields (Noack et al. 2019; Pan et al. 2020). Therefore, improving models' adversarial robustness is critical to build trustworthy Artificial Intelligence systems to prevent unforeseen hazardous situations.

To improve CNN's adversarial robustness, many methods have been proposed. One strategy is to modify the inputs during inference time via noise removal (Meng and Chen 2017), super-resolution (Mustafa et al. 2019b) and JPEG compression (Das et al. 2017) to diminish the impact of perturbation, but can be easily evaded by strong attacks (Athalye, Carlini, and Wagner 2018). Another type of strategy (Tramèr et al. 2017; Kurakin, Goodfellow, and Bengio 2016; Sinha, Namkoong, and Duchi 2017; Zhang et al. 2019; Shafahi et al. 2019) is based on adversarial training (Goodfellow, Shlens, and Szegedy 2014) that can effectively increase the model's robustness by utilizing crafted adversarial examples as data augmentation. However, it is computationally expensive and compromise model classification performance on clean images (Tsipras et al. 2018). Other than modifying data, some techniques directly enhance model robustness by altering network architectures (Taghanaki et al. 2019; Mustafa et al. 2019a), or constructing ensembles of networks (Tramèr et al. 2017; Pang et al. 2019b). However, they require additional processes and are not flexible to be adopted to other models.

While previous works have successfully improved CNN robustness against adversarial attacks, the connection of CNN predictions between adversarial and clean samples is not known. In this paper, we investigate this connection in the typical setting when CNNs are trained with the cross-entropy (CE) loss. When an adversarial sample successfully fools the trained CNN with small perturbations, it tends to be misclassified into the first several most probable false classes when predicting the original clean sample, i.e., the classes with larger predicted probabilities. This consistent pattern of CNN's predictive behaviors is intuitive and potentially implies a deeper connection between the CNN feature learning and its adversarial robustness.

The tendency of misclassifying adversarial samples enlightens us that the adversarial robustness of CNN can be benefited from the training that focuses on the differentiation between sample's true class and its first several most probable false classes. Hence, in this paper, we propose a novel training objective, termed as Probabilistically Compact (PC) loss with logit constraints, which can improve adversarial

*These authors contributed equally.
†Corresponding author.

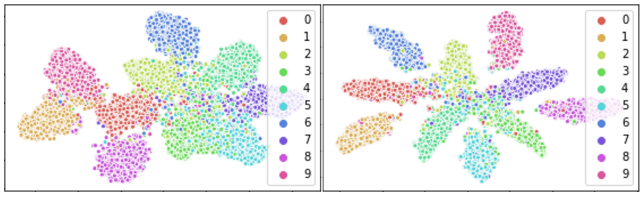


Figure 1: T-SNE visualization of the penultimate layer of ResNet-56 trained with CE loss (left) and PC loss (right) on CIFAR-10.

robustness and achieve comparable classification accuracy without extra training procedure and computational burden.

Unlike CE loss which focuses only on maximizing the output probability of the true class, PC loss aims at maximizing probability gaps between the true class and the most probable false classes. Meanwhile, the logit constraints suppress logit which ensures that the gaps is not only large, but also difficult to be crossed. Consequently, this formulation helps to widen the gaps between different classes in feature space, and ensures that it is difficult for an adversary to fool the trained model with small perturbations. We demonstrate the synergistic effect of the gaps at both probability and feature levels using benchmark datasets. For example, the average probability gaps between the true class and the most probable false class of ResNet-56 on CIFAR-100 test data are 0.527 (CE loss) vs. 0.558 (PC loss), respectively. And for Tiny ImageNet test data the gaps become 0.131 (CE loss) vs. 0.231 (PC loss). These results demonstrate that our PC loss can directly enlarge the probability gap of prediction and the effect is more pronounced for more challenging dataset (Tiny ImageNet). As shown in Figure 1, ResNet-56 trained with our PC loss has clear margin boundaries and samples of each classes are evenly distributed around the center with a minimal overlap on CIFAR-10 test data.

Our main contributions are summarized as follows: (1) We offer an unique insight into the predictive behavior of CNN on adversarial samples that the former tends to misclassify the latter into the first several most probable classes. (2) We formulate the problem by proposing a new loss function, i.e., PC loss with logit constraints to improve CNN's adversarial robustness, where these two components are systematically integrated and simultaneously optimized during training process. (3) Our PC loss can be used as a drop-in replacement of the CE loss to supervise CNN training without extra procedure nor additional computational burden for improving adversarial robustness. Experimental results show that when trained with our method, CNNs can achieve significantly improved robustness against adversarial samples without compromising performance on predicting clean samples.

Related Work

For generating adversarial samples, various computational methods have been developed. Fast Gradient Sign Method (FGSM) (Goodfellow, Shlens, and Szegedy 2014) and its variant Basic Iterative Method (BIM) efficiently generates adversarial samples by perturbing the pixels according to the gradient of the loss function (Kurakin, Goodfellow, and Bengio 2016). Projected Gradient Descent (PGD) (Madry et al. 2017) introduces a random starting point at each iteration in FGSM within a specified l_∞ norm-ball to enhance the attack effect. Momentum iterative method (MIM) (Dong et al. 2018) uses momentum to help iterative gradient-based methods to avoid sticking into local maximum thus further boosts their attacking performance. As an optimization-based attack, the Carlini and Wagner (C&W) (Carlini and Wagner 2017) approach uses binary search mechanism to find the minimal perturbation for a successful attack. SPSA (Uesato et al. 2018) is a gradient-free method which approximates gradient to generate attacks and defeats many defenses. It outperforms gradient-based attacks when the loss surface is hard to optimize.

To counter adversarial attack and enhance model robustness, various defensive techniques have been proposed. Among these proposed defenses, one line of those approaches (Kurakin, Goodfellow, and Bengio 2016; Sinha, Namkoong, and Duchi 2017; Zhang et al. 2019; Shafahi et al. 2019) are based on adversarial training (Goodfellow, Shlens, and Szegedy 2014) and can achieve effective robustness against different adversarial attacks, where the training dataset is augmented with adversarial samples. However, these methods have trade-off between accuracy on clean images and adversarial robustness (Tsipras et al. 2018) and are computationally expensive in adversarial samples generation (Zhang et al. 2019). To reduce the computational burden, Shafahi et al. (Shafahi et al. 2019) propose a training algorithm, which improves the efficiency of adversarial training by updating both model parameters and image perturbation in one backward pass.

Another line of defending strategy against adversaries, other than augmenting the training dataset, is to learn feature representations with adversarial robustness by using model ensembles or altering network architectures (Taghanaki et al. 2019; Mustafa et al. 2019a; Tramèr et al. 2017; Pang et al. 2019b). For example, (Taghanaki et al. 2019) augment CNNs with the radial basis function kernel to further transform features via kernel trick to improve the class separability in feature space and reduce the effect of perturbation. (Mustafa et al. 2019a) propose a prototype objective function, together with multi-level deep supervision. Their method ensures the feature space separation between classes and shows significant improvement of robustness. (Pang et al. 2019b) obtain a strong ensemble defense by introducing a new regularizer to encourage diversity among models within the ensemble system, which encourages the feature representation from the same class to be close. Although these approaches avoid high computationally cost of adversarial training, they have to modify the network or require extra training process, which limits their flexibility to be adapted to different tasks.

More efficient approaches are designing new loss functions to improve model adversarial robustness. By explicitly imposing regularization on latent features, CNNs are encouraged to learn feature representations with more inter-class separability and intra-class compactness (Pang et al.

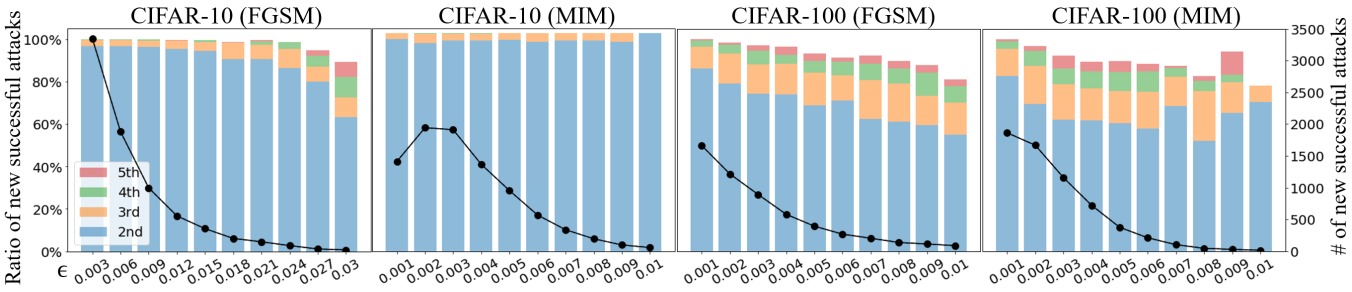


Figure 2: Empirical investigation on the predictive behavior of CNN on adversarial samples from CIFAR-10 and CIFAR-100. The line (black, right y-axis) represents the number of increased successful attacks when ϵ is increased from its previous grid value. Each bar (left y-axis) represents the percentage of misclassification for the increased successful attacks, measuring number of adversarial samples are misclassified into the 2nd, 3rd, 4th and 5th most probable classes. FGSM and MIM are attack methods.

2019a; Mustafa et al. 2019a). For example, (Pang et al. 2019a) propose a Max-Mahalanobis center (MMC) loss to learn discriminative features. They first calculate Max-Mahalanobis (Pang, Du, and Zhu 2018) centers for each class and then encourage the features to gather around the centers using Center Loss (Wen et al. 2016). However, the assumption of geometrical compactness for latent features (in terms of Euclidean distance or L_2-norm) may not hold due to inherent intra-class variations in the data and usually requires suitable assumptions on distribution of the latent features. Differently, our PC loss with logit constraints is motivated by the predictive behavior of CNN on adversarial samples from the probability perspective, which avoids this issue by learning probabilistically compact features without geometric assumptions. The work that is closest to ours is (Chen et al. 2019b) that encourages the predicted probabilities of false classes to be equally distributed, whereas our PC loss directly enlarges the gap of probabilities between true class and the first several most probable false classes.

Proposed Method

Notation Let $D = (\boldsymbol{x}_i, y_i)_{i=1}^N$ be the set of training samples of size N, where $\boldsymbol{x}_i \in \mathbf{R}^p$ is the p-dimensional feature vector and $y_i = k (k = 1, \cdots, K)$ is the true class label, and $S_k = \{(\boldsymbol{x}_i, y_i) : y_i = k\}$ the subset of D for the k-th class. The bold $\boldsymbol{y}_i = (y_i^1, \cdots, y_i^K)$ is used to represent the one-hot encoding for y_i: $y_i^k = 1$ if $y_i = k$, 0 otherwise.

Cross-entropy (CE) loss Assume that CNN's output layer, after convolutional layers, is a fully connected layer of K neurons with bias terms, then the predicted probability for sample $\boldsymbol{x}$ being classified into k-th class is calculated using the softmax activation (the k-th logit $a_k = \boldsymbol{W}_k \boldsymbol{h}_{\boldsymbol{x}} + b_k$):

$$f_k(\boldsymbol{x}) = p(y = k|\boldsymbol{x}) = \frac{\exp(a_k)}{\sum_{j=1}^K \exp(a_j)} \quad (k = 1, \cdots, K), \tag{1}$$

where $\boldsymbol{h}_{\boldsymbol{x}}$ is the feature representation of $\boldsymbol{x}$, $\boldsymbol{W}_k$ and b_k are parameters of the k-th neuron in the output layer. Then CE loss, which is equivalent to the maximum likelihood approach, is given as follows:

$$L(\boldsymbol{\theta}) = -\sum_{k=1}^K \sum_{i_k \in S_k} \log f_k(\boldsymbol{\theta}; \boldsymbol{x}_{i_k}), \tag{2}$$

where $\boldsymbol{\theta}$ is the vector of trainable model parameters.

Motivation: Predictive Behavior of CNN on Adversarial Samples

Previous studies have shown that when trained to optimum, *i.e.* $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$, CNNs can misclassify adversarial samples that are only slightly different from the original clean samples. This vulnerability has recently inspired many methods for generating adversarial samples (attack), defending adversarial attacks and detecting adversarial samples. Here, we take a different perspective on the attacks and empirically investigate if there is a systematic tendency on how CNNs misclassify adversarial samples.

Specifically, for a testing (clean) sample $(\boldsymbol{x}, y)$, the (untargeted) attack seeks a small perturbation $\boldsymbol{\epsilon}$ that leads to the misclassification of $\boldsymbol{x}$ when the perturbation is added to $\boldsymbol{x}$:

$$\min_{\boldsymbol{\epsilon}} ||\boldsymbol{\epsilon}||_p, \text{s.t. } y' = \arg\max_k f_k(\boldsymbol{x} + \boldsymbol{\epsilon}) \text{ and } y \neq y', \tag{3}$$

where $||\cdot||_p$ is the norm such as L_1, L_2 and L_∞. When Eq. (3) is optimized and the attack succeeds, a natural question to ask is *"are there any connections between y' and y for the trained CNN misclassifying $\boldsymbol{x} + \boldsymbol{\epsilon}$ into class y'?"* Note that $y = \arg\max_k f_k(\boldsymbol{x})$.

Intuitively, we could expect that y' is likely to be the most probable class except the true class label y, *i.e*, the class corresponding to the 2nd largest value of CNN predicted probabilities. This conjectures that solving Eq. (3) is equivalent to solve

$$\min_{\boldsymbol{\epsilon}} ||\boldsymbol{\epsilon}||_p, \text{s.t. } \arg\max_k f_k(\boldsymbol{x}+\boldsymbol{\epsilon}) = \arg_{\#2}\max_k f_k(\boldsymbol{x}), \tag{4}$$

where $\arg_{\#2}\max$ represents the operation of taking the 2nd largest value[1].

[1]We may relax it to the first several most probable classes such as 3rd and 4th.

Here we provide an analysis as our motivation behind this conjecture. Assuming that the CNN is Lipschitz continuous (Fazlyab et al. 2019), then we have the inequality:

$$||\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{\epsilon})-\boldsymbol{f}(\boldsymbol{x})||_p \leq l||(\boldsymbol{x}+\boldsymbol{\epsilon})-\boldsymbol{x}||_p = l||\boldsymbol{\epsilon}||_p, \quad (5)$$

where l is the Lipschitz constant and $\boldsymbol{f}(\cdot) = (f_1(\cdot), \cdots, f_K(\cdot))$. The Lipschitz continuity implies that the change of the ouput is bounded by the change of the input, which is the small perturbation in adversarial attacks. To misclassify $\boldsymbol{x}+\boldsymbol{\epsilon}$, the possible minimal value of the LHS in Eq. (5) is to seek an $\boldsymbol{\epsilon}$ such that $f_j(\boldsymbol{x}+\boldsymbol{\epsilon}) \geq f_y(\boldsymbol{x}+\boldsymbol{\epsilon})$, where y is true class and j is 2nd most probable class. To see this, consider the following two cases:

- Case 1. $f_y(\boldsymbol{x}+\boldsymbol{\epsilon}) \geq f_y(\boldsymbol{x})$. To misclassify $\boldsymbol{x}+\boldsymbol{\epsilon}$, the possible minimal value $||\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{\epsilon})-\boldsymbol{f}(\boldsymbol{x})||_p$ is to reduce $f_{k'}(\boldsymbol{x})$ ($k' \neq y, j$) to compensate $f_j(\boldsymbol{x}+\boldsymbol{\epsilon})$ so that $f_j(\boldsymbol{x}+\boldsymbol{\epsilon}) \geq f_y(\boldsymbol{x}+\boldsymbol{\epsilon})$.
- Case 2. $f_y(\boldsymbol{x}+\boldsymbol{\epsilon}) < f_y(\boldsymbol{x})$. The possible minimal value is that $f_y(\boldsymbol{x}+\boldsymbol{\epsilon}) = f_y(\boldsymbol{x}) - (\frac{f_y(\boldsymbol{x})-f_j(\boldsymbol{x})}{2})$, $f_j(\boldsymbol{x}+\boldsymbol{\epsilon}) = f_j(\boldsymbol{x}) + (\frac{f_y(\boldsymbol{x})-f_j(\boldsymbol{x})}{2})$ and all other $f_{k'}(\boldsymbol{x})$ ($k' \neq y, j$) remain unchanged.

Those two cases may not be achievable in practice, but provide a lower bound on $||\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{\epsilon})-\boldsymbol{f}(\boldsymbol{x})||_p$. The same analysis can be further relaxed to the 3rd and 4th most probable classes. Observing Eq. (5), solving Eq. (3) provides an upper-bound for the LHS of Eq. (5). With Lipschitz continuity, we hence conjecture that CNN tends to misclassify adversarial samples into classes that have large predicted probabilities when predicting the original clean samples.

To verify our conjecture, we perform an empirical study on CIFAR-10 and CIFAR-100 datasets. FGSM and MIM are used as the adversarial attack algorithms and generate adversarial samples for the standard testing data of CIFAR-10 and CIFAR-100. We do not solve Eq. (3) for each testing samples as it is computationally expensive for the test data of size 10,000. Instead, we take a fine grid of perturbation values and summarize the misclassification of newly successful attacks when the perturbation $\boldsymbol{\epsilon}$ is increased from $\boldsymbol{\epsilon}_m$ to $\boldsymbol{\epsilon}_{m+1} = \boldsymbol{\epsilon}_m + \Delta$ (Δ is the value of increment). Figure 2 displays the summary of the misclassification results. From the figure, we can see that for CIFAR-10, every time the perturbation is increased, the newly successful attacks are mostly misclassified into the 2nd most probable class for the clean samples. For CIFAR-100, the misclassification follows a similar trend considering it has 100 classes. We also notice that as the perturbation gets larger in FGSM, more newly successful attacks are classified into the 3rd, 4th and 5th most probable classes of predicting clean samples. A possible reason is that the large perturbation results in overshoot in the misclassification as the difference between 2nd most probable class and 3rd most probable class is small when a clean sample needs large perturbation to be adversarial. Different from FGSM, as perturbation increases, MIM always maintains a high percentage of classifying newly successful adversarial samples into the 2nd most probable class of predicting clean samples for CIFAR-10, due to its iterative procedure in generating adversarial attacks. Overall, Figure 2 empirically agrees with our analysis that motivates our proposed PC loss.

Probabilistically Compact Loss

The predictive behavior of CNN on adversarial samples in the last section inspires us that to improve model robustness to adversaries, CNN needs to focus on the differentiation between the true class and the first several most probable classes. In terms of predicted probability, CNN robustness is benefited from the large gap between true class $f_y(\boldsymbol{x})$ and false class $f_{y'}(\boldsymbol{x})$ ($y' \neq y$). Indeed, (Neyshabur et al. 2017) shows that the gap $f_y(\boldsymbol{x}) - \max_{y'} f_{y'}(\boldsymbol{x})$ can be used to measure the generalizability of deep neural networks.

With the aforementioned motivation, we propose the PC loss to improve CNN's adversarial robustness as follows:

$$L_{pc}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{y' \neq y_i, i \in D} \max(0, f_{y'}(\boldsymbol{x}_i) + \xi - f_{y_i}(\boldsymbol{x}_i)), \quad (6)$$

where N is the number of training samples, $\xi > 0$ is the probability margin treated as hyperparameter. Here, we include all non-target classes in the formulation and penalize any classes for each training sample that violate the margin requirement for two considerations: (1) if one of the most probable classes satisfies the margin requirement, all less probable classes will automatically satisfy this requirement and hence have no effect in PC loss; (2) since the first several most probable classes are unknown and can change during the training process, it is necessary to maintain the margin requirement for all classes.

Compared with previous works (Mustafa et al. 2019a; Pang et al. 2019a; Taghanaki et al. 2019) that improve adversarial robustness via explicitly learning features with large intra-class compactness, PC loss avoids assumptions on the feature space. Instead, PC loss only encourages the feature learning that leads to probabilistic intra-class compactness by imposing a probability margin ξ.

In training CNN with PC loss, the latter is differentiable and hence can be optimized with stochastic gradient descent. The gradient of PC loss can be calculated as (w.r.t. logit a_y)

$$\frac{\partial(f_{y'} + \xi - f_y)}{\partial a_y} = -f_y(1 - f_y + f_{y'}), \quad (7)$$

where the gradient is computed for the softmax function. For other logits ($a_{y'}$), the gradients can be similarly computed.

The Logit Constraints

In last section we introduce PC loss to enhance adversarial robustness by enlarging the probability gaps. In this section, we further propose logit constraints as a complement of PC loss, which suppress logit to ensure that the gaps are not only large, but also difficult to be crossed. Here we explain the necessity of using both parts together. We use $|\cdot|$ to denote $||\cdot||_2$ for simplicity. The conclusion can be extended to other L_p norms. Assume there is a clean image $\boldsymbol{x}$, and a corrupted image $\boldsymbol{x}+\boldsymbol{\epsilon}$ attacked by some adversarial algorithm, where $|\boldsymbol{\epsilon}| < \tau$, with $\tau > 0$ be a small constant.

We take the log probability for simplicity, i.e., PC loss is equivalently enlarging $\log f_y(\boldsymbol{x}) - \log f_j(\boldsymbol{x})$ between true

class y and most probable false class j. Given a perturbation $\boldsymbol{\epsilon}$, the corresponding log probabilities can be estimated via first order approximation

$$\log f_y(\boldsymbol{x}+\boldsymbol{\epsilon}) = \log f_y(\boldsymbol{x}) + \boldsymbol{\epsilon}\cdot\nabla_{\boldsymbol{x}}\log f_y(\boldsymbol{x}), \quad (8)$$

$\log f_j(\boldsymbol{x}+\boldsymbol{\epsilon})$ can also be approximated in the same manner. To prevent $\log f_y(\boldsymbol{x}+\boldsymbol{\epsilon}) - \log f_j(\boldsymbol{x}+\boldsymbol{\epsilon}) < 0$ (i.e., false prediction with perturbation $\boldsymbol{\epsilon}$), we should solve $\min_{\boldsymbol{\theta}} \boldsymbol{\epsilon}\cdot(\nabla_{\boldsymbol{x}}\log f_j(\boldsymbol{x}) - \nabla_{\boldsymbol{x}}\log f_y(\boldsymbol{x}))$.

Lets denote vector $\boldsymbol{b} = \nabla_{\boldsymbol{x}}\log f_j(\boldsymbol{x}) - \nabla_{\boldsymbol{x}}\log f_y(\boldsymbol{x})$. As the attackers can always choose the worst $\hat{\boldsymbol{\epsilon}}$ that maximizes $\boldsymbol{\epsilon}\cdot\boldsymbol{b}$ by letting $\hat{\boldsymbol{\epsilon}}$ in the same direction as $\boldsymbol{b}$, i.e. $\boldsymbol{\epsilon}\cdot\boldsymbol{b} \le |\hat{\boldsymbol{\epsilon}}|\cdot|\boldsymbol{b}|$. Our goal becomes to minimize the upper bound $|\hat{\boldsymbol{\epsilon}}|\cdot|\boldsymbol{b}|$

$$\begin{aligned}\min_{\boldsymbol{\theta}} |\hat{\boldsymbol{\epsilon}}||\boldsymbol{b}| &\Longrightarrow \min_{\boldsymbol{\theta}} |\boldsymbol{b}| \\ &\Longrightarrow \min_{\boldsymbol{\theta}} |\nabla_{\boldsymbol{x}}\log f_j(\boldsymbol{x}) - \nabla_{\boldsymbol{x}}\log f_y(\boldsymbol{x})|.\end{aligned} \quad (9)$$

Hence to prevent the prediction changes after perturbation $\boldsymbol{\epsilon}$, we should minimize $|\nabla_{\boldsymbol{x}}(\log f_y(\boldsymbol{x}) - \log f_j(\boldsymbol{x}))|$. Let a_k denote the logit for the kth class softmax output, observe that

$$\begin{aligned}\nabla_{\boldsymbol{x}}\log f_y - \nabla_{\boldsymbol{x}}\log f_j &= \frac{\nabla_{\boldsymbol{x}} f_y}{f_y} - \frac{\nabla_{\boldsymbol{x}} f_j}{f_j} \\ &= \nabla_{\boldsymbol{x}} a_y - \nabla_{\boldsymbol{x}} a_j,\end{aligned} \quad (10)$$

because $\nabla_{\boldsymbol{x}} f_y = -\sum_k f_k f_y \nabla_{\boldsymbol{x}} a_k + f_y \nabla_{\boldsymbol{x}} a_y$, and the same holds for $\nabla_{\boldsymbol{x}} f_j$. We can equivalently change our objective to $\min_{\boldsymbol{\theta}} |\nabla_{\boldsymbol{x}}(a_y - a_j)|$. We can estimate $|\nabla_{\boldsymbol{x}}(a_y(\boldsymbol{x}) - a_j(\boldsymbol{x}))|$ using

$$\begin{aligned}|\nabla_{\boldsymbol{x}}(a_y - a_j)| \approx |&(a_y(\boldsymbol{x}) - a_j(\boldsymbol{x})) \\ &-(a_y(\boldsymbol{x}+\hat{\boldsymbol{\epsilon}}) - a_j(\boldsymbol{x}+\hat{\boldsymbol{\epsilon}}))|/|\hat{\boldsymbol{\epsilon}}|,\end{aligned} \quad (11)$$

where we denote $|\boldsymbol{\epsilon}| < |\hat{\boldsymbol{\epsilon}}| = \tau$ that upper bounds $|\boldsymbol{\epsilon}|$. Note that an adversarial attack tends to minimize $a_y(\boldsymbol{x}+\hat{\boldsymbol{\epsilon}}) - a_j(\boldsymbol{x}+\hat{\boldsymbol{\epsilon}})$ so that $a_y(\boldsymbol{x}+\hat{\boldsymbol{\epsilon}}) - a_j(\boldsymbol{x}+\hat{\boldsymbol{\epsilon}}) < a_y(\boldsymbol{x}) - a_j(\boldsymbol{x})$. And a robust model should instead prevent $a_y(\boldsymbol{x}+\boldsymbol{\epsilon}) - a_j(\boldsymbol{x}+\boldsymbol{\epsilon}) < 0$ to ensure a correct prediction when attacked, so we have the following inequality for a robust model under attack

$$0 < a_y(\boldsymbol{x}+\hat{\boldsymbol{\epsilon}}) - a_j(\boldsymbol{x}+\hat{\boldsymbol{\epsilon}}) < a_y(\boldsymbol{x}) - a_j(\boldsymbol{x}). \quad (12)$$

Then Eq. (11) can be upper bounded by

$$|\nabla_{\boldsymbol{x}}(a_y - a_j)| < |a_y(\boldsymbol{x}) - a_j(\boldsymbol{x})|/|\hat{\boldsymbol{\epsilon}}|. \quad (13)$$

Substitute this inequality back to Eq. (10), we get a logit constraint condition to ensure model robustness

$$|\nabla_{\boldsymbol{x}}(\log f_y - \log f_j)| < |a_y(\boldsymbol{x}) - a_j(\boldsymbol{x})|/|\hat{\boldsymbol{\epsilon}}| < C, \quad (14)$$

where C is an arbitrary positive constant thresholding robustness, hence we can optimize PC loss subject to the above condition

$$\min_{\boldsymbol{\theta}} L_{pc}(\boldsymbol{\theta}), \text{s.t.} |a_y(\boldsymbol{x};\boldsymbol{\theta}) - a_j(\boldsymbol{x};\boldsymbol{\theta})| < C' \text{ for } \forall \boldsymbol{x}, \quad (15)$$

where $C' = |\hat{\boldsymbol{\epsilon}}|C$. It is equivalent to write the above minimization problem with a multiplier λ

$$\min_{\boldsymbol{\theta},\boldsymbol{\lambda}} \Big(L_{pc}(\boldsymbol{\theta}) + \frac{\lambda}{N}\sum_{\boldsymbol{x}\in D}(d_{yj} - C')\Big), \quad (16)$$

where N is the number of samples, and D is the set of training samples. λ is treated as a hyper-parameter in training, $d_{yj} = \max(0, a_y(\boldsymbol{x};\boldsymbol{\theta}) - a_j(\boldsymbol{x};\boldsymbol{\theta}))$. As shown in Eq. (16), PC loss and logit constraints are systematically integrated and simultaneously optimized during training process to enhance model adversarial robustness.

Experiments

In this section, we evaluate our proposed PC loss with logit constraints along with analysis that our method does not rely on the 'gradient masking' that provides a false sense of security (Athalye, Carlini, and Wagner 2018).

Datasets and models: We analyze seven benchmark datasets: MNIST, KMNIST, Fashion-MNIST (FMNIST), CIFAR-10, CIFAR-100, Street-View House Numbers (SVHN), and Tiny Imagenet. We scale all pixel values to $[0, 1]$ following the preprocessing procedure in (Mustafa et al. 2019a; Pang et al. 2019b). For gray-scale image datasets (K/F/MNIST), we use a LeNet-5 model (LeCun et al. 1998), and for color image datasets (CIFAR-10, CIFAR-100, SVHN, Tiny Imagenet), we use a VGG-13 model (Simonyan and Zisserman 2014). All these models are trained using Adam optimizer with a initial learning rate of 0.01 and a batch size of 256. For our method, we first warm up the training process for T epochs ($T = 50$ for K/F/MNIST and $T = 150$ for other datasets) using CE loss, and then train the model using our method shown in Eqs. (6) and (16) ($\xi = 0.995, \lambda = 0.05$) for another T epochs whereas we directly train the baseline using CE loss for $2T$ epochs.

Attack types In the adversarial setting, there are two main threat models: white-box attacks where the adversary possesses complete knowledge of target model, including its architecture, training method and learned parameters, and black-box attacks where the adversary does not have access to the information about trained classifier but is aware of the classification task. We evaluate the robustness of our proposed method against both white-box and black-box attacks.

Results

Performance on white-box attacks Following the attack settings in (Chen et al. 2019a), we crafted adversarial examples in a non-targeted way with respect to allowed perturbation ϵ for gradient-based attacks, i.e., FGSM, BIM, PGD and MIM. The number of iterations is set to 10 for BIM and 40 for MIM and PGD while perturbation of each step is 0.01. For parameters of optimization-based attack C&W, the maximum iteration steps are set to 100, with a learning rate of 0.001, and the confidence is set to 0.

The results (Table 1) demonstrate that our proposed PC loss with logit constraints outperforms the CE loss under white-box attacks while maintaining the comparable level of performance on the clean image classification. The improvement is even more significant on stronger attacks.

Besides comparing to the standard CE loss, we also compare our defense approach with a closely related Guided Complement Entropy (GCE) approach (Chen et al. 2019a). To ensure a fair comparison we use the exactly same models (LeNet-5 for MNIST and ResNet56 for CIFAR-10) and parameters (max iterations of C&W is 1000) as in the GCE paper. In Table 2, it is evident that our method outperforms GCE in the vast majority of settings.

Performance on black-box attacks The performance under black-box setting is critical to substantiate adversarial robustness since it is closer to the real-world scenario where

Attacks	Param.	MNIST		KMNIST		FMNIST		Param.	CIFAR-10		SVHN	
		CE	Ours	CE	Ours	CE	Ours		CE	Ours	CE	Ours
Clean	-	99.2	99.2	95.5	95.4	90.1	90.2	-	91.6	91.2	94.9	94.7
FGSM	0.1	71.5	**80.5**	26.2	**62.8**	17.5	**58.0**	0.04	4.3	**53.1**	8.7	**39.5**
	0.2	51.6	**76.3**	1.8	**39.7**	9.5	**43.3**	0.12	11.7	**30.3**	4.5	**24.2**
	0.3	31.8	**65.0**	1.2	**34.1**	7.6	**31.8**	0.2	11.5	**18.7**	2.6	**17.1**
BIM	0.1	52.8	**72.0**	55.8	**82.5**	0.0	**18.7**	0.04	0.0	**29.0**	1.3	**26.2**
	0.2	4.5	**48.6**	28.7	**73.3**	0.3	**8.4**	0.12	0.0	**21.0**	0.0	**18.2**
	0.3	1.5	**39.5**	16.8	**60.5**	0.0	**6.4**	0.2	0.0	**20.0**	0.0	**17.6**
PGD	0.1	49.0	**72.3**	31.3	**62.4**	0.0	**15.7**	0.04	0.0	**27.6**	0.0	**27.6**
	0.2	3.3	**50.2**	3.9	**39.9**	0.0	**7.0**	0.12	0.0	**14.6**	0.0	**22.0**
	0.3	0.8	**39.7**	2.0	**33.2**	0.0	**4.6**	0.2	0.0	**7.5**	0.0	**21.0**
MIM	0.1	49.8	**73.8**	26.0	**65.4**	0.0	**14.8**	0.04	0.0	**34.3**	0.0	**29.2**
	0.2	5.0	**54.0**	4.2	**45.4**	0.0	**6.3**	0.12	0.0	**32.7**	0.0	**27.6**
	0.3	1.5	**43.3**	2.0	**36.9**	0.0	**4.5**	0.2	0.0	**32.4**	0.0	**26.0**
CW	0.0	42.2	**78.0**	19.5	**57.3**	0.2	**21.8**	0.0	0.0	**30.2**	0.0	**36.2**

Table 1: Accuracy (%) on K/F/MNIST, CIFAR-10 and SVHN under white-box setting. For CW, the parameter is the confidence.

Attacks	Param.	MNIST			Param.	CIFAR-10		
		CE	GCE*	Ours		CE	GCE*	Ours
FGSM	0.1	71.5	**87.7**	80.5	0.04	12.7	41.2	**58.4**
	0.2	51.6	62.7	**76.3**	0.12	10.3	14.8	**17.3**
	0.3	31.8	47.2	**65.0**	0.2	7.0	11.8	**12.0**
BIM	0.1	52.8	61.9	**72.0**	0.04	0.0	**19.6**	16.6
	0.2	4.5	34.5	**48.6**	0.12	0.0	3.0	**3.4**
	0.3	1.5	33.5	**39.5**	0.2	0.0	2.0	**2.6**
PGD	0.1	49.0	51.9	**72.3**	0.04	0.0	5.9	**10.2**
	0.2	3.3	9.6	**50.2**	0.12	0.0	1.9	**3.5**
	0.3	0.8	2.2	**39.7**	0.2	0.0	1.6	**2.7**
MIM	0.1	49.8	61.2	**73.8**	0.04	0.0	15.4	**16.0**
	0.2	5.0	39.8	**54.0**	0.12	0.0	**13.1**	11.6
	0.3	1.5	38.8	**43.3**	0.2	0.0	**12.7**	11.2
C&W	0.0	0.0	25.6	**30.1**	0.0	0.0	0.8	**3.3**

Table 2: Accuracy (%) between GCE and our method on MNIST and CIFAR10 under white-box setting. *Results are directly from (Chen et al. 2019a).

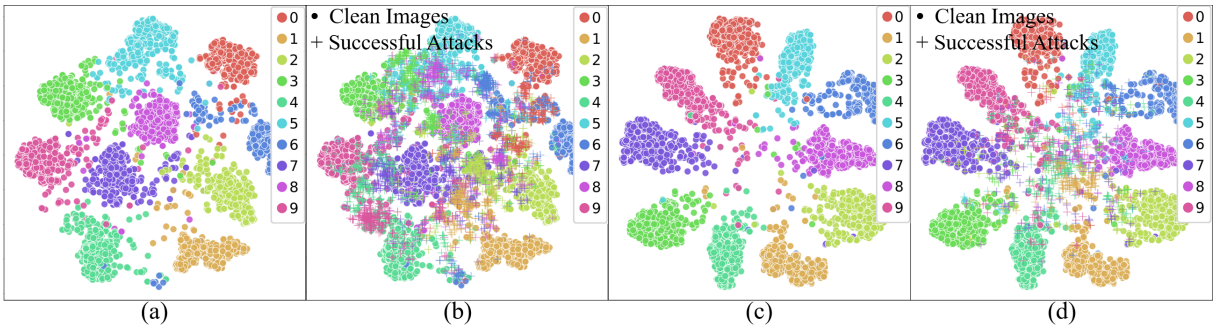
an adversary has no access to the trained classifier. During inference time, black-box adversary uses a substitute model trained on the same dataset to generate adversarial samples to attack the target model. In our cases, we use a 3-layer CNN as the substitute model for LeNet-5 and ResNet-56 for VGG-13 to generate black-box attacks. Similar to (Pang et al. 2019b), we adopt PGD and MIM, the two most commonly used attack methods under the black-box setting. We then further evaluate our defense method using a gradient-free attack approach, i.e., SPSA, as in (Carlini et al. 2019), which performs numerical approximation on the gradients using test data. The learning rate of SPSA is set to 0.01, and the step size is $\delta = 0.01$ (Uesato et al. 2018). As shown in Table 3, the model trained with our PC loss improves robustness against the black-box attacks.

Larger-scale experiments on CIFAR-100 and Tiny ImageNet We also evaluate our method on larger and more complex CIFAR-100 and Tiny ImageNet datasets under both white-box attacks (PGD, MIM) and black-box attack (SPSA). Similar to (Pang et al. 2019b), we reduce the perturbation budget to the range of [0.005, 0.015] and attack iterations to 10 due to the increased data complexity and scale. As shown in Table 4, our method significantly improves the model's adversarial robustness compared to the CE loss and GCE while maintaining the comparable level of performance on the clean image classification. Recall our observation that the most probable false classes are more vulnerable to attacks. GCE flattens the probabilities on false classes and thus enlarges the gap between true class and the most probable false class to increase model's robustness. However, when dataset become complex with more classes, this gap is smaller due to generally lower output probability for the true class, resulting a limited robustness improvement. On the other hand, our method directly maximizes the probability gap and thus is more suitable for large scale datasets.

Combining with adversarial training To demonstrate our method's compatibility and synergy with other adversarial defense techniques, we investigate the performance of our method in combination with adversarial training. Our goal is not to beat adversarial training, instead we attempt to show our method can be combined with it to further improve adversarial robustness. During training, we augment the dataset with adversarial samples generated using FGSM with perturbation range of $[0.1, 0.3]$ for gray-scale image datasets (K/F/MNIST), and 5-step PGD with perturbation range of $[0.0, 0.1]$ for color image dataset (CIFAR-10). The ratio of adversarial examples and clean images in each training mini-batch remains $1:1$. For gray-scale image datasets, table 5 shows that integrating our method with adversarial training further improves the model's adversarial robustness under both white-box (PGD, BIM, MIM, CW) and black-box attack settings (SPSA). Furthermore, our PC loss with adversarial training outperforms GCE with adversarial training, which demonstrates our method has better compatibility with other defense techniques. It is worth mentioning that our method alone outperforms the fast version adversarial training on gray-scale datasets, which generates adversarial training examples by one-step FGSM attack. And for color image dataset (CIFAR-10), we augment the dataset with adversarial examples crafted by more advanced PGD attacks. The result shows the same trend, and the performance gain is more pronounced on this more challenging dataset.

Attacks	Param.	MNIST		KMNIST		FMNIST		Param.	CIFAR-10		SVHN	
		CE	Ours	CE	Ours	CE	Ours		CE	Ours	CE	Ours
PGD	0.1	96.6	**98.1**	90.8	**92.5**	65.5	**74.0**	0.04	19.5	**42.2**	43.6	**48.8**
	0.2	84.3	**92.8**	76.7	**85.0**	50.4	**57.1**	0.12	13.2	**38.0**	19.5	**28.1**
	0.3	61.1	**85.6**	59.1	**77.7**	47.8	**53.5**	0.2	16.7	**35.6**	13.2	**24.5**
MIM	0.1	96.4	**98.1**	90.4	**92.2**	62.4	**72.3**	0.04	17.2	**42.0**	40.1	**44.2**
	0.2	84.2	**94.7**	74.9	**83.0**	43.3	**54.2**	0.12	1.3	**16.3**	13.3	**21.7**
	0.3	56.7	**81.2**	48.4	**63.8**	30.5	**36.0**	0.2	0.3	**11.2**	10.0	**16.2**
SPSA	0.3	72.9	**95.7**	50.2	**78.0**	4.3	**39.8**	0.3	0.0	**45.3**	4.0	**58.0**

Table 3: Accuracy (%) on K/F/MNIST, CIFAR-10 and SVHN under black-box setting.



(a) (b) (c) (d)

Figure 3: T-SNE visualization of the penultimate layer of the model trained by CE loss (a,b) and our PC loss (c,d) on MNIST dataset. (a,c) display only clean images whereas (b,d) also include successful attacks generated with FGSM ($\epsilon = 0.3$).

Attacks	Param.	CIFAR-100			Tiny ImageNet		
		CE	GCE	Ours	CE	GCE	Ours
Clean	-	40.2	64.5	**67.7**	**38.2**	32.8	37.7
PGD	0.005	11.4	24.4	**56.7**	11.3	8.9	**24.8**
	0.010	2.0	14.8	**54.6**	2.8	2.6	**19.2**
	0.015	0.4	9.1	**52.6**	0.8	1.0	**15.8**
MIM	0.005	8.7	21.9	**55.8**	7.9	7.3	**23.5**
	0.010	1.4	12.3	**52.8**	1.9	2.1	**18.0**
	0.015	0.3	7.4	**48.9**	0.6	1.4	**14.6**
SPSA	0.015	3.9	11.5	**22.1**	6.3	7.3	**16.2**

Table 4: Accuracy (%) on CIFAR-100 and Tiny ImageNet between CE loss, GCE and our new PC loss.

Feature Space Visualization In order to visually dissect the advantages of PC loss over the CE loss, we also inspect the feature space of trained models using t-SNE on MNIST datasets. As shown in Figure 3a, the model trained with CE loss has a large portion of clean images lay across the boundaries between different classes thus easily to be manipulated to become adversarial samples. On the contrary, for the model trained with our PC loss with logit constraints, in Figure 3c, the samples of each class have clear boundaries and are evenly distributed around the center with a minimal overlap. Note that the samples locate near the center are 'hard samples' for a classifier even without attacks.

Looking into the successful attacks (labeled with '+') in Figure 3, we find the predictive behavior of CNN on adversarial samples is consistent to our hypothesis. In Figure 3b, for a model (LeNet-5) trained with CE loss, adversarial samples are mostly located to the nearest classes corresponding to the most probable false classes. For example, many adversarial attacks generated based on class 5 are located within the class 8 of clean images and *vice versa*. While in Figure 3d, for a model trained with PC loss, due to the large margin between classes, the adversarial samples are harder to cross the boundaries with the only exception that the adversarial samples are distributed near the center of the feature space where hard samples are usually located.

Identifying Gradient Masking

Previous defense strategies (Buckman et al. 2018; Xie et al. 2017) rely on the effect of gradient masking, which was considered as a false sense of security (Athalye, Carlini, and Wagner 2018). Briefly, these defenses deteriorate the gradient information to make gradient-based attack methods hard to generate effective adversarial examples. However, these defenses can be easily defeated by black-box or gradient-free attackers. We show that our method does not rely on gradient masking on the basis of characteristics defined in (Athalye, Carlini, and Wagner 2018; Carlini et al. 2019). (1) Iterative attacks have better performance than one-step attack: Our results in Table 1 indicate that the iteration-based attacks (BIM, MIM, PGD) are more successful in generating adversarial attacks than single step method (FGSM). (2) Robustness against Black-box attacks is higher than white-box attacks: When model's gradients information is manipulated by the defender, the attacker can recover the gradient with black-box attacks and perform more successful attacks than using white-box attacks (Papernot et al. 2017). However, the results in Tables 1 & 3 demonstrate that our method

Attacks	Param.	MNIST				KMNIST				FMNIST				Param.	CIFAR-10	
		CE+AT	GCE+AT	Ours+AT	Ours	CE+AT	GCE+AT	Ours+AT	Ours	CE+AT	CGE+AT	Ours+AT	Ours		CE+AT	Ours+AT
BIM	0.3	27.0	28.3	**86.1**	39.5	58.4	8.1	**65.9**	60.5	0.1	4.8	**18.6**	4.9	0.04	17.3	**38.7**
PGD	0.3	3.2	26.8	**72.3**	39.7	37.2	0.5	**48.7**	33.2	0.0	2.7	**13.4**	2.8	0.04	11.4	**33.7**
MIM	0.3	10.8	27.7	**78.8**	43.3	22.3	15.7	**51.9**	36.9	0.0	1.4	**9.1**	2.2	0.04	9.0	**33.3**
C&W	0.0	75.5	69.4	**96.4**	78.0	47.9	48.2	**67.1**	57.3	4.0	21.5	**29.8**	21.8	0.0	0.0	**33.7**
SPSA	0.3	77.3	56.6	**97.1**	95.7	72.0	69.9	**79.0**	78.0	14.2	30.0	**41.6**	39.8	0.3	6.5	**39.6**

Table 5: Accuracy (%) on K/F/MNIST and CIFAR-10 with adversarial training under both white- and black-box attacks.

is more effective against black-box attacks and thus does not obfuscate gradients. (3) Increasing perturbation budget will increase attack success: As shown in the Table 1, increase of perturbations monotonically enhances the attacks. With a large budget ($\epsilon = 0.3$), the success rate is close to 100%.

Conclusion

We propose a novel PC loss with logit constraints inspired by the predictive behavior of CNN on adversarial samples. A CNN trained with our PC loss can achieve impressive robustness against adversarial samples without compromising performance on clean images nor requires additional procedures/computing, making it scalable to large-scale datasets. In addition, our PC loss is flexible and compatible with other defense methods, e.g., as a drop-in replacement of CE loss to supervise adversarial training. In future work, we plan to extensively investigate the connection of predictions between adversarial and clean samples in more general settings.

Acknowledgements

This work is supported by the National Science Foundation under grants CNS-2043611 and IIS-1724227.

Ethical Impact

During the past few years, CNNs have been successfully implemented in various computer vision tasks such as autonomous driving, surveillance, and medical imaging diagnosis. Some of these systems have been already deployed and are integrated in our daily lives. Nevertheless recent studies have revealed the vulnerabilities of CNNs against adversarial attacks. Without adversarially robust defense strategies, the users of these systems can be exposed to excessive hazardous situations such as car accidents, burglaries, and inaccurate diagnosis. Therefore improving the robustness of CNNs is critical to build trustworthy AI systems to benefit humanity and society.

In this paper, we propose a novel loss function that can significantly improve CNN robustness against adversarial attacks and thus help to mitigate the risks at deployment of AI systems. The high efficiency of our approach makes it not only adaptive to most classification tasks with various model architectures in different fields without any extra procedure, but also compatible with other defense techniques to further improve the CNN's robustness. As such, our method has a strong potential to become a central pillar of the next-generation trustworthy AI system.

References

Al-Qizwini, M.; Barjasteh, I.; Al-Qassab, H.; and Radha, H. 2017. Deep learning algorithm for autonomous driving using GoogLeNet. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, 89–96. IEEE.

Athalye, A.; Carlini, N.; and Wagner, D. 2018. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420* .

Buckman, J.; Roy, A.; Raffel, C.; and Goodfellow, I. 2018. Thermometer Encoding: One Hot Way To Resist Adversarial Examples. In *The International Conference on Learning Representations (ICLR)*.

Carlini, N.; Athalye, A.; Papernot, N.; Brendel, W.; Rauber, J.; Tsipras, D.; Goodfellow, I.; Madry, A.; and Kurakin, A. 2019. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705* .

Carlini, N.; and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, 39–57. IEEE.

Chen, H.-Y.; Liang, J.-H.; Chang, S.-C.; Pan, J.-Y.; Chen, Y.-T.; Wei, W.; and Juan, D.-C. 2019a. Improving adversarial robustness via guided complement entropy. In *Proceedings of the IEEE International Conference on Computer Vision*, 4881–4889.

Chen, H.-Y.; Wang, P.-H.; Liu, C.-H.; Chang, S.-C.; Pan, J.-Y.; Chen, Y.-T.; Wei, W.; and Juan, D.-C. 2019b. Complement objective training. *arXiv preprint arXiv:1903.01182* .

Das, N.; Shanbhogue, M.; Chen, S.-T.; Hohman, F.; Chen, L.; Kounavis, M. E.; and Chau, D. H. 2017. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *arXiv preprint arXiv:1705.02900* .

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 9185–9193.

Fazlyab, M.; Robey, A.; Hassani, H.; Morari, M.; and Pappas, G. 2019. Efficient and accurate estimation of lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems*, 11423–11434.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* .

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236* .

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11): 2278–2324.

Li, X.; Li, X.; Pan, D.; and Zhu, D. 2020. On the Learning Property of Logistic and Softmax Losses for Deep Neural Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 4739–4746.

Li, X.; and Zhu, D. 2020. Robust Detection of Adversarial Attacks on Medical Images. In *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, 1154–1158. IEEE.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* .

Meng, D.; and Chen, H. 2017. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 135–147.

Mustafa, A.; Khan, S.; Hayat, M.; Goecke, R.; Shen, J.; and Shao, L. 2019a. Adversarial defense by restricting the hidden space of deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 3385–3394.

Mustafa, A.; Khan, S. H.; Hayat, M.; Shen, J.; and Shao, L. 2019b. Image super-resolution as a defense against adversarial attacks. *IEEE Transactions on Image Processing* 29: 1711–1724.

Neyshabur, B.; Bhojanapalli, S.; McAllester, D.; and Srebro, N. 2017. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, 5947–5956.

Noack, A.; Ahern, I.; Dou, D.; and Li, B. 2019. Does Interpretability of Neural Networks Imply Adversarial Robustness? *CoRR* abs/1912.03430. URL http://arxiv.org/abs/1912.03430.

Pan, D.; Li, X.; Li, X.; and Zhu, D. 2020. Explainable Recommendation via Interpretable Feature Mapping and Evaluation of Explainability. In Bessiere, C., ed., *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, 2690–2696. ijcai.org. doi:10.24963/ijcai.2020/373. URL https://doi.org/10.24963/ijcai.2020/373.

Pang, T.; Du, C.; and Zhu, J. 2018. Max-mahalanobis linear discriminant analysis networks. *arXiv preprint arXiv:1802.09308* .

Pang, T.; Xu, K.; Dong, Y.; Du, C.; Chen, N.; and Zhu, J. 2019a. Rethinking Softmax Cross-Entropy Loss for Adversarial Robustness. *arXiv preprint arXiv:1905.10626* .

Pang, T.; Xu, K.; Du, C.; Chen, N.; and Zhu, J. 2019b. Improving adversarial robustness via promoting ensemble diversity. *arXiv preprint arXiv:1901.08846* .

Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z. B.; and Swami, A. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, 506–519.

Shafahi, A.; Najibi, M.; Ghiasi, M. A.; Xu, Z.; Dickerson, J.; Studer, C.; Davis, L. S.; Taylor, G.; and Goldstein, T. 2019. Adversarial training for free! In *Advances in Neural Information Processing Systems*, 3353–3364.

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

Sinha, A.; Namkoong, H.; and Duchi, J. 2017. Certifiable distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571* 2.

Sreenu, G.; and Durai, M. S. 2019. Intelligent video surveillance: a review through deep learning techniques for crowd analysis. *Journal of Big Data* 6(1): 48.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* .

Taghanaki, S. A.; Abhishek, K.; Azizi, S.; and Hamarneh, G. 2019. A kernelized manifold mapping to diminish the effect of adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 11340–11349.

Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; and McDaniel, P. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204* .

Tsipras, D.; Santurkar, S.; Engstrom, L.; Turner, A.; and Madry, A. 2018. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152* .

Uesato, J.; O'Donoghue, B.; Oord, A. v. d.; and Kohli, P. 2018. Adversarial risk and the dangers of evaluating against weak attacks. *arXiv preprint arXiv:1802.05666* .

Wen, Y.; Zhang, K.; Li, Z.; and Qiao, Y. 2016. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, 499–515. Springer.

Xie, C.; Wang, J.; Zhang, Z.; Ren, Z.; and Yuille, A. 2017. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991* .

Zhang, D.; Zhang, T.; Lu, Y.; Zhu, Z.; and Dong, B. 2019. You only propagate once: Accelerating adversarial training via maximal principle. In *Advances in Neural Information Processing Systems*, 227–238.

Explainable Recommendation via Interpretable Feature Mapping and Evaluation of Explainability

Deng Pan , **Xiangrui Li** , **Xin Li** and **Dongxiao Zhu***

Department of Computer Science
Wayne State University, USA
{pan.deng, xiangruili, xinlee, dzhu}@wayne.edu

Abstract

Latent factor collaborative filtering (CF) has been a widely used technique for recommender system by learning the semantic representations of users and items. Recently, explainable recommendation has attracted much attention from research community. However, trade-off exists between explainability and performance of the recommendation where metadata is often needed to alleviate the dilemma. We present a novel feature mapping approach that maps the uninterpretable general features onto the interpretable aspect features, achieving both satisfactory accuracy and explainability in the recommendations by simultaneous minimization of rating prediction loss and interpretation loss. To evaluate the explainability, we propose two new evaluation metrics specifically designed for aspect-level explanation using surrogate ground truth. Experimental results demonstrate a strong performance in both recommendation and explaining explanation, eliminating the need for metadata. Code is available from https://github.com/pd90506/AMCF.

1 Introduction

Since the inception of the Netflix Prize competition, latent factor collaborative filtering (CF) has been continuously adopted by various recommendation tasks due to its strong performance over other methods [Koren *et al.*, 2009], which essentially employs a latent factor model such as matrix factorization and/or neural networks to learn user or item feature representations for rendering recommendations. Despite much success, latent factor CF approaches often suffer from the lack of interpretability [Zhang and Chen, 2018]. In a contemporary recommender system, explaining why a user likes an item can be as important as the accuracy of the rating prediction itself [Zhang and Chen, 2018].

Explainable recommendation can improve transparency, persuasiveness and trustworthiness of the system [Zhang *et al.*, 2019]. To make intuitive explanation for recommendations, recent efforts have been focused on using metadata such as user defined tags and topics from user review texts or item descriptions[Lu *et al.*, 2018; Chen *et al.*, 2018] to illuminate users preferences. Other works such as [Hou *et al.*, 2019; He *et al.*, 2015; Zhang *et al.*, 2014] use *aspects* to explain recommendations. Although these approaches can explain recommendation using external metadata, the interpretability of the models themselves and the *interpretable features* enabling the explainable recommendations have still not been systematically studied and thus, are poorly understood. It is also worth mentioning that the challenges in explainable recommendation not only lie in the modeling itself, but also in the lack of a gold standard for evaluation of explainability.

Here we propose a novel feature mapping strategy that not only enjoys the advantages of strong performance in latent factor models but also is capable of providing explainability via interpretable features. The main idea is that by mapping the *general features* learned using a base latent factor model onto interpretable *aspect features*, one could explain the outputs using the aspect features without compromising the recommendation performance of the base latent factor model. We also propose two new metrics for evaluating the quality of explanations in terms of a user's general preference over all items and the aspect preference to a specific item. Simply put, we formulate the problem as: 1) how to find the interpretable aspect basis; 2) how to perform interpretable feature mapping; and 3) how to evaluate explanations.

We summarize our main contributions as follows: 1) We propose a novel feature mapping approach to map the general uninterpretable features to interpretable aspect features, enabling explainability of the traditional latent factor models without metadata; 2) Borrowing strength across aspects, our approach is capable of alleviating the trade-off between recommendation performance and explainability; and 3) We propose new schemes for evaluating the quality of explanations in terms of both general user preference and specific user preference.

2 Related Work

There are varieties of strategies for rendering explainable recommendations. We first review methods that give explanations in light of aspects, which are closely related to our work. We then discuss other recent explainable recommendation works using metadata and knowledge in lieu of aspects.

*Corresponding author

2.1 Aspect Based Explainable Recommendation

Aspects can be viewed as explicit features of an item that could provide useful information in recommender systems. An array of approaches have been developed to render explainable recommendations at the aspect level using metadata such as user reviews. These approaches mostly fall into three categories: 1) Graph-based approaches: they incorporate aspects as additional nodes in the user-item bipartite graph. For example, TriRank [He *et al.*, 2015] extract aspects from user reviews and form a user-item-aspect tripartite graph with smoothness constraints, achieving a review-aware top-N recommendation. ReEL [Baral *et al.*, 2018] calculate user-aspect bipartite from location-aspect bipartite graphs, which infer user preferences. 2) Approaches with aspects as regularizations or priors: they use the extracted aspects as additional regularizations for the factorization models. For example, AMF [Hou *et al.*, 2019] construct an additional user-aspect matrix and an item-aspect matrix from review texts, as regularizations for the original matrix factorization models. JMARS [Diao *et al.*, 2014] generalize probabilistic matrix factorization by incorporating user-aspect and movie-aspect priors, enhancing recommendation quality by jointly modeling aspects, ratings and sentiments from review texts. 3) Approaches with aspects as explicit factors: other than regularizing the factorization models, aspects can also be used as factors themselves. [Zhang *et al.*, 2014] propose an explicit factor model (EMF) that factorizes a rating matrix in terms of both predefined explicit features (i.e. aspects) as well as implicit features, rendering aspect-based explanations. Similarly, [Chen *et al.*, 2016] extend EMF by applying tensor factorization on a more complex user-item-feature tensor.

2.2 Beyond Aspect Explanation

There are also other approaches that don't utilize aspects to explain recommendations. For example, [Lee and Jung, 2018] give explanations in light of the movie similarities defined using movie characters and their interactions; [Wang *et al.*, 2019] propose explainable recommendations by exploiting knowledge graphs where paths are used to infer the underlying rationale of user-item interactions. With the increasingly available textual data from users and merchants, more approaches have been developed for explainable recommendation using metadata. For example, [Chen *et al.*, 2019b; Costa *et al.*, 2018; Lu *et al.*, 2018] attempt to generate textual explanations directly whereas [Wu *et al.*, 2019; Chen *et al.*, 2019a] give explanations by highlighting the most important words/phrases in the original reviews.

Overall, most of the approaches discussed in this section rely on metadata and/or external knowledge to give explanations without interpreting the model itself. In contrast, our Attentive Multitask Collaborative Filtering (AMCF) approach maps uninterpretable general features to interpretable aspect features using an existing aspect definition, as such it not only gives explanations for users, but also learns interpretable features for the modelers. Moreover, it is possible to adopt any latent factor models as the base model to derive the general features for the proposed feature mapping approach.

3 The Proposed AMCF Model

In this section, we first introduce the problem formulation and the underlying assumptions. We then present our AMCF approach for explainable recommendations. AMCF incorporates aspect information and maps the latent features of items to the aspect feature space using an attention mechanism. With this mapping, we can explain recommendations of AMCF from the aspect perspective. An **Aspect** $\boldsymbol{s}$ [Bauman *et al.*, 2017] is an attribute that characterizes an item. Assuming there are totally m aspects in consideration, if an item has aspects $s_{i_1}, ..., s_{i_k}$ simultaneously, an item can then be described by $\mathcal{I}_i = \{s_{i_1}, s_{i_2}, ..., s_{i_k}\}$, $k \leq m$. We say that an item i has aspect s_j, if $s_j \in \mathcal{I}_i$.

3.1 Problem Formulation

Inputs. The inputs consist of 3 parts: the set of users U, the set of items V, and the set of corresponding multi-hot aspect vectors for items, denoted by S.

Outputs. Given the user-item-aspect triplet, e.g. user i, item j, and aspect multi-hot vector s_j for item j, our model not only predicts the review rating, but also the user general preference over all items and the user specific preference on item j in terms of aspects, i.e., which aspects of the item j that the user i is mostly interested in.

3.2 Rationale

The trade-off between model interpretability and performance states that we can either achieve high interpretability with simpler models or high performance with more complex models that are generally harder to interpret [Zhang and Chen, 2018]. Recent works [Zhang and Chen, 2018; He *et al.*, 2015; Zhang *et al.*, 2014] have shown that with adequate metadata and knowledge, it is possible to achieve both explainability and high accuracy in the same model. However, those approaches mainly focus on explanation of the recommendation, rather than exploiting the interpretability of the models and features, and hence are still not interpretable from modeling perspective. Explainability and interpretability refer to "why" and "how" a recommendation is made, respectively. Many above-referenced works only answer the "why" question via constraints from external knowledge without addressing "how". Whereas our proposed AMCF model answers both "why" and "how" questions, i.e., our recommendations are made based on the attention weights (why) and the weights are learned by interpretable feature decomposition (how). To achieve this, we assume that an interpretable aspect feature representation can be mathematically derived from the corresponding general feature representation. More formally:

Assumption 1. *Assume there are two representations for the same prediction task:* $\boldsymbol{u}$ *in complex feature space* $\mathcal{U}$ *(i.e. general embedding space including item embedding and aspect embedding), and* $\boldsymbol{v}$ *in simpler feature space* $\mathcal{V}$ *(i.e. space spanned by aspect embeddings), and* $\mathcal{V} \subset \mathcal{U}$*. We say that* $\boldsymbol{v}$ *is the projection of* $\boldsymbol{u}$ *from space* $\mathcal{U}$ *to space* $\mathcal{V}$*, and there exists a mapping* $\boldsymbol{M}(\cdot, \boldsymbol{\theta})$*, such that* $\boldsymbol{v} = \boldsymbol{M}(\boldsymbol{u}, \boldsymbol{\theta})$*, with* θ *as a hyper-parameter.*

This assumption is based on the widely accepted notion that a simple local approximation can give good interpretation of a complex model in that particular neighborhood [Ribeiro *et al.*, 2016]. Instead of selecting surrogate interpretable simple models (such as linear models), we map the general complex features to the simpler interpretable aspect features, then render recommendation based on those general complex features. We give explanations using interpretable aspect features, achieving the best of both worlds in keeping the high performance of the complex model as well as gaining the interpretability of the simpler model. In this work, the *interpretable simple features* are obtained based on *aspects*, hence we call the corresponding feature space as *aspect space*. To map the complex general features onto the interpretable aspect space, we define the aspect projection.

Definition 1. ***(Aspect Projection)*** *Given Assumption 1, we say $\boldsymbol{v}$ is an aspect projection of $\boldsymbol{u}$ from general feature space $\mathcal{U}$ to aspect feature space $\mathcal{V}$ (Figure 1).*

To achieve good interpretability and performance in the same model, from Definition 1 and Assumption 1, we need to find the mapping $\boldsymbol{M}(\cdot, \boldsymbol{\theta})$. Here we first use a latent factor model as the base model for explicit rating prediction, which learns general features, as shown in Figure 2 (left, L_{pred}), where we call the *item embedding* $\boldsymbol{u}$ as the general complex feature learned by the base model. Then the remaining problem is to derive the mapping from the non-interpretable general features to the interpretable aspect features.

3.3 Aspect Embedding

To design a simple interpretable model, its features should be well aligned to our interest, e.g. the *aspects* is a reasonable choice. Taking movie genre as an example: if we use 4 genres (Romance, Comedy, Thriller, Fantasy) as 4 aspects, the movie *Titanic*'s aspect should be represented by $(1, 0, 0, 0)$ because it's romance genre, and the movie *Cinderella*'s aspect is $(1, 0, 0, 1)$ because it's genre falls into both romance and fantasy.

From Assumption 1 and Definition 1, to make the feature mapping from a general feature $\boldsymbol{u}$ to an aspect feature $\boldsymbol{v}$, we need to first define the aspect space $\mathcal{V}$. Assuming there are m aspects in consideration, we represent the m aspects by m latent vectors in general space $\mathcal{U}$, and use these m aspect vectors as the basis that spans the aspect space $\mathcal{V} \subset \mathcal{U}$. These aspects' latent vectors can be learned by neural embedding or other feature learning methods, with each aspect corresponding to an individual latent feature vector. Our model uses embedding approach to extract m aspect latent vectors of n-dimension, where n is the dimension of space $\mathcal{U}$. In Figure 2, the vertical columns in red $(\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_m)$ represent m aspect embeddings in the general space $\mathcal{U}$, which is obtained by embedding the aspect multi-hot vectors from input.

3.4 Aspect Projection of Item Embedding

In Assumption 1, $\boldsymbol{u}$ is the general feature representation (i.e. the item embedding) in space $\mathcal{U}$, and $\boldsymbol{v}$ is the interpretable aspect feature representation in space $\mathcal{V}$. The orthogonal projection from the general space $\mathcal{U}$ to the aspect space $\mathcal{V}$ is denoted by $\boldsymbol{M}$, i.e. $\boldsymbol{v} = \boldsymbol{M}(\boldsymbol{u})$.

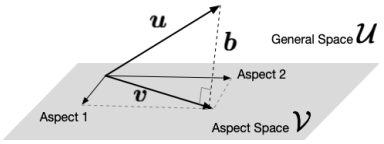


Figure 1: An illustration of interpretable feature mapping. $\boldsymbol{u}$ is an uninterpretable feature in general space $\mathcal{U}$, and $\boldsymbol{v}$ is the interpretable projection of $\boldsymbol{u}$ in the interpretable aspect space $\mathcal{V}$. Here $\boldsymbol{b}$ indicates the difference between $\boldsymbol{u}$ and $\boldsymbol{v}$.

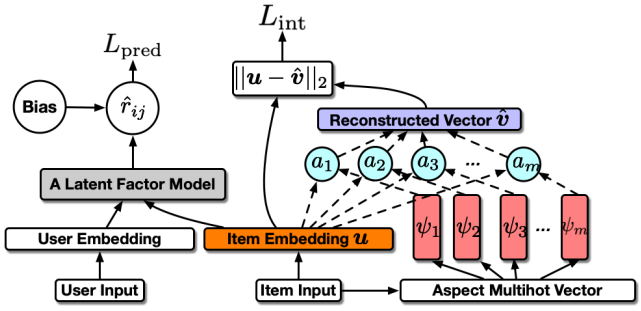


Figure 2: The training phase: explainable recommendation via interpretable feature mapping.

From the perspective of learning disentangled representations, the item embedding $\boldsymbol{u}$ can be disentangled as $\boldsymbol{u} = \boldsymbol{v} + \boldsymbol{b}$ (Figure 1), where $\boldsymbol{v}$ encodes the aspect information of an item and $\boldsymbol{b}$ is the item-unique information. For example, movies from the same genre share similar artistic style ($\boldsymbol{v}$) yet each movie has its own unique characteristics ($\boldsymbol{b}$). With this disentanglement of item embeddings, we can explain recommendation via capturing user's preference in terms of aspects.

Let's assume that we have m linearly independent and normalized *aspect* vectors $(\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_m)$ in space $\mathcal{U}$, which span subspace $\mathcal{V}$. For any vector $\boldsymbol{v} = \boldsymbol{M}(\boldsymbol{u})$ in space $\mathcal{V}$, there exists an unique decomposition such that $\boldsymbol{v} = \sum_{i=1}^{m} v_i \boldsymbol{\psi}_i$. The coefficients can be directly calculated by $v_i = \boldsymbol{v} \cdot \boldsymbol{\psi}_i = \boldsymbol{u} \cdot \boldsymbol{\psi}_i$, $(i = 1, ..., m$, $\boldsymbol{\psi}_i$ is normalized). Note that the second equality comes from the fact that $\boldsymbol{v}$ is the orthogonal projection of $\boldsymbol{u}$ on space $\mathcal{V}$.

Generally speaking, however, $(\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_m)$ are not orthogonal. In this case, as long as they are linearly independent, we can perform Gram-Schmidt orthogonalization process to obtain the corresponding orthogonal basis. The procedure can be simply described as follows: $\tilde{\boldsymbol{\psi}}_1 = \boldsymbol{\psi}_1$; and $\tilde{\boldsymbol{\psi}}_i = \boldsymbol{\psi}_i - \sum_{j=1}^{i-1} \langle \boldsymbol{\psi}_i, \tilde{\boldsymbol{\psi}}_j \rangle \tilde{\boldsymbol{\psi}}_j$, where $\langle \boldsymbol{\psi}_i, \tilde{\boldsymbol{\psi}}_j \rangle$ denotes inner product. We can then calculate the unique decomposition as in the orthogonal cases. Assume the resulting decomposition is $\boldsymbol{v} = \sum_{i=1}^{m} \tilde{v}_i \tilde{\boldsymbol{\psi}}_i$, the coefficients corresponding to the original basis $(\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_m)$ can then be calculated by: $v_i = \tilde{v}_i - \sum_{j=i+1}^{m} \langle \boldsymbol{\psi}_i, \tilde{\boldsymbol{\psi}}_j \rangle$; and $v_m = \tilde{v}_m$.

Hence, after the aspect feature projection and decomposition, regardless of orthogonal or not, we have the following unique decomposition in space $\mathcal{V}$: $\boldsymbol{v} = \sum_{i=1}^{m} v_i \boldsymbol{\psi}_i$.

Aspect Projection via Attention. As described above, any interpretable aspect feature $\boldsymbol{v}$ can be uniquely decomposed as $\boldsymbol{v} = \sum_{i=1}^{m} v_i \boldsymbol{\psi}_i$, which is similar to the form of atten-

tion mechanism. Therefore, instead of using Gram-Schmidt orthogonalization process, we utilize attention mechanism to reconstruct $\boldsymbol{v}$ directly. Assume we can obtain an attention vector $\boldsymbol{a} = (a_1, ..., a_m)$, which can be used to calculate $\hat{\boldsymbol{v}} = \sum_{i=1}^{m} a_i \boldsymbol{\psi}_i$, with the fact that the decomposition is unique, our goal is then to minimize the distance $||\hat{\boldsymbol{v}} - \boldsymbol{v}||_2$ to ensure that $a_i \approx v_i$.

However, as the interpretable aspect feature $\boldsymbol{v}$ is not available, we cannot minimize $||\hat{\boldsymbol{v}} - \boldsymbol{v}||_2$ directly. Fortunately, the general feature $\boldsymbol{u}$ is available (obtained from a base latent factor model), with the fact that $\boldsymbol{v}$ is the projection of $\boldsymbol{u}$, i.e. $\boldsymbol{v} = \boldsymbol{M}(\boldsymbol{u})$, we have the following lemma:

Lemma 1. *Provided that $\boldsymbol{v}$ is the projection of $\boldsymbol{u}$ from space $\mathcal{U}$ to space $\mathcal{V}$, where $\mathcal{V} \subset \mathcal{U}$, we have*

$$\arg\min_{\boldsymbol{a}} ||\hat{\boldsymbol{v}} - \boldsymbol{v}||_2 = \arg\min_{\boldsymbol{a}} ||\hat{\boldsymbol{v}} - \boldsymbol{u}||_2,$$

where $\hat{\boldsymbol{v}} = \sum_{i=1}^{m} a_i \boldsymbol{\psi}_i$, $\boldsymbol{a} = (a_1, ..., a_m)$, and $||\cdot||_2$ denotes l_2 norm.

Proof. Refer to the illustration in Figure 1, and denote the difference between $\boldsymbol{u}$ and $\boldsymbol{v}$ as $\boldsymbol{b}$, i.e. $\boldsymbol{u} = \boldsymbol{v} + \boldsymbol{b}$. Hence

$$\arg\min_{\boldsymbol{a}} ||\hat{\boldsymbol{v}} - \boldsymbol{u}||_2 = \arg\min_{\boldsymbol{a}} ||\hat{\boldsymbol{v}} - \boldsymbol{v} - \boldsymbol{b}||_2.$$

Note that $\boldsymbol{b}$ is perpendicular to $\hat{\boldsymbol{v}}$ and $\boldsymbol{v}$, the right hand side can then be written as

$$\arg\min_{\boldsymbol{a}} ||\hat{\boldsymbol{v}} - \boldsymbol{v} - \boldsymbol{b}||_2 = \arg\min_{\boldsymbol{a}} \sqrt{(||\hat{\boldsymbol{v}} - \boldsymbol{v}||_2^2 + ||\boldsymbol{b}||_2^2)},$$

as $\boldsymbol{b}$ is not parameterized by $\boldsymbol{a}$, we then get

$$\arg\min_{\boldsymbol{a}} ||\hat{\boldsymbol{v}} - \boldsymbol{v}||_2 = \arg\min_{\boldsymbol{a}} ||\hat{\boldsymbol{v}} - \boldsymbol{u}||_2.$$

□

From the above proof, we know that attention mechanism is sufficient to reconstruct $\boldsymbol{v} \approx \hat{\boldsymbol{v}} = \sum_{i=1}^{m} a_i \boldsymbol{\psi}_i$ by minimizing $||\hat{\boldsymbol{v}} - \boldsymbol{u}||_2$. Note that from the perspective of disentanglement $\boldsymbol{u} = \boldsymbol{v} + \boldsymbol{b}$, the information in $\boldsymbol{b}$, i.e., the item specific characteristics, is not explained in our model. Intuitively, the item specific characteristics are learned from the metadata associated with the item.

3.5 The Loss Function

The loss function for finding the feature mapping $\boldsymbol{M}(\cdot)$ to achieve both interpretability and performance of the recommender model has 2 components:

- L_{pred} prediction loss in rating predictions, corresponding to the loss function for the base latent factor model.
- L_{int} interpretation loss to the general feature $\boldsymbol{u}$. This loss is to quantify $||\hat{\boldsymbol{v}} - \boldsymbol{u}||_2$.

We calculate the rating prediction loss component using RMSE: $L_{pred} = \sqrt{\frac{1}{N} \sum_{(i,j) \in \text{Observed}} (r_{ij} - \hat{r}_{ij})^2}$, where $\hat{r}_{ij}$ represents the predicted item ratings. We then calculate the interpretation loss component as the average distance between $\boldsymbol{u}$ and $\hat{\boldsymbol{v}}$: $L_{int} = \frac{1}{N} \sum_{(i,j) \in \text{Observed}} ||\hat{\boldsymbol{v}} - \boldsymbol{u}||_2$. The loss component L_{int} encourages the interpretable feature $\tilde{\boldsymbol{v}}$ obtained from the attentive neural network to be a good approximation of the aspect feature representation $\boldsymbol{v}$ (Lemma 1). Hence the overall loss function is $L = L_{pred} + \lambda L_{int}$, where λ is a tuning parameter to leverage importance between the two loss components.

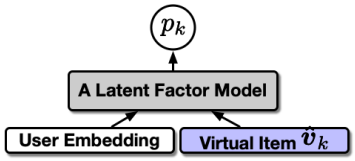


Figure 3: The explanation phase: a virtual item vector $\hat{\boldsymbol{v}}_k$ is calculated to represent a specific aspect.

Gradient Shielding Trick. To ensure that interpretation doesn't compromise the prediction accuracy, we allow forward propagation to both L_{int} and L_{pred} but refrain the back-propagation from L_{int} to the item embedding $\boldsymbol{u}$. In other words, when learning the model parameters based on back-propagation gradients, the item embedding $\boldsymbol{u}$ is updated only via the gradients from L_{pred}.

3.6 User Preference Prediction

Thus far we attempt to optimize the ability to predict user preference via aspect feature mapping. We call the user overall preference as *general preference*, and the user preference on a specific item as *specific preference*.

General preference. Figure 3 illustrates how to make prediction on user general preference. Here we define a virtual item $\hat{\boldsymbol{v}}_{\boldsymbol{k}}$, which is a linear combination of aspect embeddings. For general preference, we let $\hat{\boldsymbol{v}}_k = \boldsymbol{\psi}_k$ to simulate a pure aspect k movie, the resulting debiased (discarded all bias terms) rating prediction $\hat{p}_k$ indicates the user's preference on such specific aspect (e.g., positive for 'like', negative for 'dislike'). Formally: $\hat{p}_k = f(\boldsymbol{q}, \hat{\boldsymbol{v}}_k)$, where $\boldsymbol{q}$ is the user embedding, $\hat{\boldsymbol{v}}_k = \boldsymbol{\psi}_k$ is the aspect k's embedding, and $f(\cdot)$ is the corresponding base latent factor model without bias terms.

Specific preference. Figure 3 also shows our model's ability to predict user preference on a specific item, as long as we can find how to represent them in terms of aspect embeddings. Fortunately, the attention mechanism is able to help us find the constitution of any item in terms of aspect embeddings using the attention weights. That is, for any item, it is possible to rewrite the latent representation $\boldsymbol{u}$ as a linear combination of aspect embeddings: $\boldsymbol{u} = \hat{\boldsymbol{v}} + \boldsymbol{b} = \sum_k a_k \boldsymbol{\psi}_k + \boldsymbol{b}$, where a_k and $\boldsymbol{\psi}_k$ are the k-th attention weight and the k-th aspect feature, respectively. The term $\boldsymbol{b}$ reflects interpretation loss. For aspect k of an item, we use $\hat{\boldsymbol{v}}_k = a_k \boldsymbol{\psi}_k$ to represent the embedding of a virtual item which represents the aspect k property of the specific item. Hence, the output $\hat{p}_k$ indicates the specific preference on aspect k of a specific item.

Model Interpretability. From specific preference, a latent general feature can be decomposed into the linear combination of interpretable aspect features, which would help interpret models in a more explicit and systematic manner.

4 Experiments and Discussion

We design and perform experiments to demonstrate two advantages of our AMCF approach: 1) comparable rating predictions; 2) good explanations on why a user likes/dislikes an item. To demonstrate the first advantage we compare the rating prediction performance with baseline approaches of *rat-*

Dataset	# of ratings	# of items	# of users	# of genres
MovieLens 1M	1,000,209	3,706	6,040	18
MovieLens 100k	100,000	1,682	943	18
Yahoo Movie	211,333	11,915	7,642	29

Table 1: Summary statistics of the data sets.

ing prediction only methods. The demonstration of the second advantage, however, is not a trivial task since currently no gold standard for evaluating explanation of recommendations except for using real customer feedback[Chen *et al.*, 2019b; Gao *et al.*, 2019]. Hence it's necessary to develop new schemes to evaluate the quality of explainability for both general and specific user preferences.

4.1 Datasets

MovieLens Datasets. This data set [Harper and Konstan, 2016] offers very complete movie genre information, which provides a perfect foundation for genre (aspect) preference prediction, i.e. determining which genre a user likes most. We consider the 18 movie genres as aspects.

Yahoo Movies Dataset. This data set from Yahoo Lab contains usual user-movie ratings as well as metadata such as movie's title, release date, genre, directors and actors. We use the 29 movie genres as the aspects for movie recommendation and explanation. Summary statistics are shown in Table 1.

Pre-processing. We use multi-hot encoding to represent genres of each movie or book, where 1 indicates the movie is of that genre, 0 otherwise. However, there are still plenty of movies with missing genre information, in such cases, we simply set them as none of any listed genre, i.e., all zeros in the aspect multi-hot vector: $(0, 0, ..., 0)$.

4.2 Results of Prediction Accuracy

We select several strong baseline models to compare rating prediction accuracy, including non-interpretable models, such as SVD [Koren *et al.*, 2009], Neural Collaborative Filtering (NCF) [He *et al.*, 2017] and Factorization Machine (FM) [Rendle, 2010], and an interpretable linear regression model (LR). Here the LR model is implemented by using aspects as inputs and learning separate parameter sets for different individual users. In comparison, our AMCF approaches also include SVD, NCF or FM as the base model to demonstrate that the interpretation module doesn't compromise the prediction accuracy. Note that since regular NCF and FM are designed for implicit ratings (1 and 0), we replace their last sigmoid output layers with fully connected layers in order to output explicit ratings.

In terms of robustness, we set the dimension of latent factors in the base models to $20, 80$, and 120. The regularization tuning parameter λ is set to 0.05, which demonstrated better performance compared to other selections. It is worth noting that the tuning parameters of the base model of our AMCF approach are directly inherited from the corresponding non-interpretable model. We compare our AMCF models with baseline models as shown in Table 2. It is clear that AMCF achieves comparable prediction accuracy to their non-interpretable counterparts, and significantly outperforms the interpretable LR model.

4.3 Evaluation of Explainability

Despite the recent efforts have been made to evaluate the quality of explanation by defining explainability precision (EP) and explainability recall (ER)[Peake and Wang, 2018; Abdollahi and Nasraoui, 2016], the scarcity of ground truth such as a user's true preference remains a significant obstacle for explainable recommendation. [Gao *et al.*, 2019] make an initial effort in collecting ground truth by surveying real customers, however, the labor intense, time consuming and sampling bias may prevent its large-scale applications in a variety of contexts. Other text-based approaches [Costa *et al.*, 2018; Lu *et al.*, 2018] can also use natural language processing (NLP) metrics such as Automated Readability Index (ARI) and Flesch Reading Ease (FRE). As we don't use metadata such as text reviews in our AMCF model, user review based explanation and evaluation could be a potential future extension to our model.

Here we develop novel quantitative evaluation schemes to assess our model's explanation quality in terms of general preferences and specific preferences, respectively.

General Preference

Let's denote the ground truth of user general preferences as $\boldsymbol{p}_i$ for user i, and the model's predicted preference for user i is $\hat{\boldsymbol{p}}_i$, we propose measures inspired by *Recall@K* in recommendation evaluations.

Top M recall at K. (TM@K): Given the M most preferred aspects of a user i from $\boldsymbol{p}_i$, top M recall at K is defined as the ratio of the M aspects located in the top K highest valued aspects in $\hat{\boldsymbol{p}}_i$. For example, if $\boldsymbol{p}_i$ indicates that user i's top 3 preferred aspects are *Adventure, Drama*, and *Thriller*, while the predicted $\hat{\boldsymbol{p}}_i$ shows that the top 5 are *Adventure, Comedy, Children, Drama, Crime*, the top 3 recalls at 5 (T3@5) is then $2/3$ whereas top 1 recall at 3 (T1@3) is 1.

Bottom M recall at K. (BM@K): Similarly defined as above, except that it measures the most disliked aspects.

As the ground truth of user preferences are usually not available, some reasonable approximations are needed. Hence we propose a method to calculate the so-called surrogate ground truth. First we define the weights $\boldsymbol{w}_{ij} = (r_{ij} - b_i^u - b_j^v - \bar{r})/A$, where the weight $\boldsymbol{w}_{ij}$ is calculated by nullifying user bias b_i^u, item bias b_j^v, and global average $\bar{r}$, and A is a constant indicating the maximum rating (e.g. $A = 5$ for most datasets). Note that user bias b_i^u and item bias b_j^v can be easily calculated by $b_i^u = (\frac{1}{|V_i|}\sum_{j \in V_i} r_{ij}) - \bar{r}$, and $b_j^v = (\frac{1}{|U_j|}\sum_{i \in U_j} r_{ij}) - \bar{r}$. Here V_i represents the sets of items rated by user i, and U_j represents the sets of users that have rated item j. With the weights we calculate user i's preference on aspect t using the following formula: $\boldsymbol{p}_i^t = \sum_{j \in V_i} \boldsymbol{w}_{ij} s_j^t$, where $s_j^t = 1$ if item j has aspect t, 0 otherwise. Hence a user i's overall preference can be represented by an l_1 normalized vector $\boldsymbol{p}_i = (\boldsymbol{p}_i^1, ..., \boldsymbol{p}_i^t, ..., \boldsymbol{p}_i^T)/||\boldsymbol{p}_i||_1$. As our model can output a user preference vector directly, we evaluate the explainability by calculating the average of TM@K and BM@K. The evaluation results are reported in Table 3. We observe that the explainability of AMCF is significantly better than random interpretation, and is comparable to the strong interpretable baseline LR model with a much

Dataset	LR	SVD			AMCF(SVD)			NCF			AMCF(NCF)			FM			AMCF(FM)		
		20	80	120	20	80	120	20	80	120	20	80	120	20	80	120	20	80	120
ML100K	1.018	0.908	0.908	**0.907**	0.907	0.909	**0.907**	0.939	0.939	0.936	0.937	0.939	0.934	0.937	0.933	0.929	0.940	0.936	0.931
ML1M	1.032	0.861	0.860	0.853	0.860	0.858	**0.851**	0.900	0.895	0.892	0.902	0.889	0.889	0.915	0.914	0.913	0.915	0.914	0.915
Yahoo	1.119	1.022	1.021	1.014	1.022	1.022	**1.010**	1.028	1.027	1.028	1.027	1.026	1.025	1.042	1.042	1.039	1.044	1.042	1.041

Table 2: Performance comparison of rating prediction using different data sets in terms of RMSE. Texts in the parentheses indicate the base CF models that we choose for AMCF; and the numbers [20, 80, 120] indicate the dimension of the latent factors for the models.

Dataset	Model	T1@3	B1@3	T3@5	B3@5	$score_s$
ML100K	AMCF	0.500	0.481	0.538	0.553	**0.378**
	LR	**0.628**	**0.668**	**0.637**	**0.675**	0.371
	Rand	0.167	0.167	0.278	0.278	0
ML1M	AMCF	0.461	0.403	0.513	0.489	**0.353**
	LR	**0.572**	**0.565**	**0.598**	**0.620**	0.322
	Rand	0.167	0.167	0.278	0.278	0
Yahoo	AMCF	0.413	0.409	0.422	0.440	0.224
	LR	**0.630**	**0.648**	**0.628**	**0.565**	**0.235**
	Rand	0.103	0.103	0.172	0.172	0

Table 3: Preferences outputs: TM@K/BM@K represent Top/ Bottom M recall at K, and $score_s$ represents the specific preference. The Rand rows show the theoretical random preference outputs. Here AMCF takes SVD with 120 latent factors as the base model.

better prediction accuracy. Thus our AMCF model successfully integrates the strong prediction performance of a latent factor model and the strong interpretability of a LR model.

Specific Preference

Our approach is also capable of predicting a user's preference on a specific item, i.e. $\hat{\boldsymbol{p}}_{ij}$, showing which aspects of item j are liked/disliked by the user i. Compared to user general preference across all items, the problem of which aspect of an item attracts the user most (specific preference) is more interesting and more challenging. There is no widely accepted strategy to evaluate the quality of single item preference prediction (except for direct customer survey). Here we propose a simple yet effective evaluation scheme to illustrate the quality of our model's explanation on user specific preference. With the overall preference $\boldsymbol{p}_i$ of user i given above, and assuming $\boldsymbol{s}_j$ is the multi-hot vector represents the aspects of item j, we say the element-wise product $\boldsymbol{p}_{ij} = \boldsymbol{p}_i \odot \boldsymbol{s}_j$ reflects the user's specific preference on item j.

Note that we should not use the TM@K/BM@K scheme as in general preference evaluation, both $\boldsymbol{p}_{ij}$ and predicted $\hat{\boldsymbol{p}}_{ij}$'s entries are mostly zeros, since each movie is only categorized into a few genres. Hence the quality of specific preference prediction is expressed using a similarity measure. We use $s(\boldsymbol{p}_{ij}, \hat{\boldsymbol{p}}_{ij})$ to represent the cosine similarity between $\boldsymbol{p}_{ij}$ and $\hat{\boldsymbol{p}}_{ij}$, and the score for specific preference prediction is defined by averaging over all user-item pairs in the test set: $score_s = \frac{1}{N}\sum_{ij} s(\boldsymbol{p}_{ij}, \hat{\boldsymbol{p}}_{ij})$. We report the results of specific user preferences in the $score_s$ column of Table 3. As the LR cannot give specific user preferences directly, we simply apply $\hat{\boldsymbol{p}}_{ij} = \hat{\boldsymbol{p}}_i \odot \boldsymbol{s}_j$ where $\hat{\boldsymbol{p}}_i$ represents the general preference predicted by LR.

An insight. Assume that for a specific user i, our AMCF model can be simply written as $\hat{r}_{ij} = f_i(\boldsymbol{u}_j)$ to predict the rating for item j. Note that our AMCF model can decompose the item in terms of aspects. Lets denote these aspects as $\{\psi_1, ...\psi_m\}$. Then the prediction can be approximated by $\hat{r}_{ij} \approx f_i(\sum_{k=1}^{m} a_{jk}\psi_k)$, where a_{jk} denote the k-th attention weights for item j. In the case of LR, the rating is obtained by $\hat{r}_{ij} = g_i(\sum_{k=1}^{m} b_k x_k)$, where g_i is the LR model for user i, b_k is the k-th coefficient of it, and x_k represents the indicator of aspect k, $x_k = 1$ when the item has aspect k, $x_k = 0$ otherwise. The similarity between AMCF formula and LR formula listed above indicates that the coefficients of LR and the preference output of AMCF share the same intrinsic meaning, i.e., both indicate the importance of aspects.

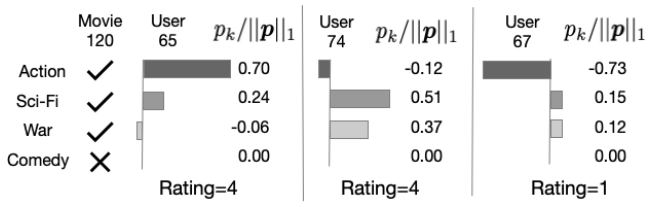


Figure 4: Examples of explainable recommendations. We l_1-normalize the preference vector $\boldsymbol{p}$ to make the comparison fair.

An example. For specific explanation, given a user i and an item j, our AMCF model predicts a vector $\boldsymbol{p}$, representing the user i's specific preference on an item j in terms of all predefined aspects. Specifically, the magnitude of each entry of $\boldsymbol{p}$ (i.e. $|p_i|$) represents the impact of a specific aspect on whether an item liked by a user or not. For example, in Figure 4, the movie 120 is high-rated by both users 65 and 74, however, with differential explanations: the former user preference is more on the *Action* genre whereas the latter is more on *Sci-Fi* and *War*. On the other hand, the same movie is low-rated by user 67 mainly due to the dislike of *Action* genre.

5 Conclusion

Modelers tend to better appreciate the interpretable recommender systems whereas users are more likely to accept the explainable recommendations. In this paper, we proposed a novel interpretable feature mapping strategy attempting to achieve both goals: systems interpretability and recommendation explainability. Using extensive experiments and tailor-made evaluation schemes, our AMCF method demonstrates strong performance in both recommendation and explanation.

Acknowledgements

This work is supported by the National Science Foundation under grant no. IIS-1724227.

References

[Abdollahi and Nasraoui, 2016] Behnoush Abdollahi and Olfa Nasraoui. Explainable matrix factorization for collaborative filtering. In *Proceedings of the 25th WWW*, pages 5–6. International WWW Conferences Steering Committee, 2016.

[Baral *et al.*, 2018] Ramesh Baral, XiaoLong Zhu, SS Iyengar, and Tao Li. Reel: R eview aware explanation of location recommendation. In *Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization*, pages 23–32. ACM, 2018.

[Bauman *et al.*, 2017] Konstantin Bauman, Bing Liu, and Alexander Tuzhilin. Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews. In *Proceedings of the 23rd ACM SIGKDD*, pages 717–725. ACM, 2017.

[Chen *et al.*, 2016] Xu Chen, Zheng Qin, Yongfeng Zhang, and Tao Xu. Learning to rank features for recommendation over multiple categories. In *Proceedings of the 39th International ACM SIGIR*, pages 305–314. ACM, 2016.

[Chen *et al.*, 2018] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 WWW*, pages 1583–1592. International WWW Conferences Steering Committee, 2018.

[Chen *et al.*, 2019a] Xu Chen, Yongfeng Zhang, and Zheng Qin. Dynamic explainable recommendation based on neural attentive models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 53–60, 2019.

[Chen *et al.*, 2019b] Zhongxia Chen, Xiting Wang, Xing Xie, Tong Wu, Guoqin Bu, Yining Wang, and Enhong Chen. Co-attentive multi-task learning for explainable recommendation. In *IJCAI*, June 2019.

[Costa *et al.*, 2018] Felipe Costa, Sixun Ouyang, Peter Dolog, and Aonghus Lawlor. Automatic generation of natural language explanations. In *Proceedings of the 23rd International Conference on Intelligent User Interfaces Companion*, page 57. ACM, 2018.

[Diao *et al.*, 2014] Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J Smola, Jing Jiang, and Chong Wang. Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars). In *Proceedings of the 20th ACM SIGKDD*, pages 193–202. ACM, 2014.

[Gao *et al.*, 2019] Jingyue Gao, Xiting Wang, Yasha Wang, and Xing Xie. Explainable recommendation through attentive multi-view learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, March 2019.

[Harper and Konstan, 2016] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):19, 2016.

[He *et al.*, 2015] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1661–1670. ACM, 2015.

[He *et al.*, 2017] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th WWW*, pages 173–182. International WWW Conferences Steering Committee, 2017.

[Hou *et al.*, 2019] Yunfeng Hou, Ning Yang, Yi Wu, and S Yu Philip. Explainable recommendation with fusion of aspect information. *WWW*, 22(1):221–240, 2019.

[Koren *et al.*, 2009] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 8:30–37, 2009.

[Lee and Jung, 2018] O-Joun Lee and Jason J Jung. Explainable movie recommendation systems by using story-based similarity. In *IUI Workshops*, 2018.

[Lu *et al.*, 2018] Yichao Lu, Ruihai Dong, and Barry Smyth. Why i like it: multi-task learning for recommendation and explanation. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 4–12. ACM, 2018.

[Peake and Wang, 2018] Georgina Peake and Jun Wang. Explanation mining: Post hoc interpretability of latent factor models for recommendation systems. In *Proceedings of the 24th ACM SIGKDD*, pages 2060–2069. ACM, 2018.

[Rendle, 2010] Steffen Rendle. Factorization machines. In *2010 IEEE International Conference on Data Mining*, pages 995–1000. IEEE, 2010.

[Ribeiro *et al.*, 2016] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD*, pages 1135–1144. ACM, 2016.

[Wang *et al.*, 2019] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5329–5336, 2019.

[Wu *et al.*, 2019] Libing Wu, Cong Quan, Chenliang Li, Qian Wang, Bolong Zheng, and Xiangyang Luo. A context-aware user-item representation learning for item recommendation. *ACM Transactions on Information Systems (TOIS)*, 37(2):22, 2019.

[Zhang and Chen, 2018] Yongfeng Zhang and Xu Chen. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192*, 2018.

[Zhang *et al.*, 2014] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR*, pages 83–92. ACM, 2014.

[Zhang *et al.*, 2019] Yongfeng Zhang, Jiaxin Mao, and Qingyao Ai. Www'19 tutorial on explainable recommendation and search. In *Companion Proceedings of WWW*, pages 1330–1331. ACM, 2019.

Explaining Deep Neural Network Models with Adversarial Gradient Integration

Deng Pan , **Xin Li** and **Dongxiao Zhu***
Department of Computer Science, Wayne State University, USA
{pan.deng, xinlee, dzhu}@wayne.edu

Abstract

Deep neural networks (DNNs) have became one of the most high performing tools in a broad range of machine learning areas. However, the multi-layer non-linearity of the network architectures prevent us from gaining a better understanding of the models' predictions. Gradient based attribution methods (e.g., Integrated Gradient (IG)) that decipher input features' contribution to the prediction task have been shown to be highly effective yet requiring a reference input as the anchor for explaining model's output. The performance of DNN model interpretation can be quite inconsistent with regard to the choice of references. Here we propose an Adversarial Gradient Integration (AGI) method that integrates the gradients from adversarial examples to the target example along the curve of steepest ascent to calculate the resulting contributions from all input features. Our method doesn't rely on the choice of references, hence can avoid the ambiguity and inconsistency sourced from the reference selection. We demonstrate the performance of our AGI method and compare with competing methods in explaining image classification results. Code is available from https://github.com/pd90506/AGI.

1 Introduction

Recently, deep neural networks (DNNs) has attracted much attention in machine learning community due to its state-of-the-art performance on various tasks such as image classification [Li *et al.*, 2020], sentiment analysis [Qiang *et al.*, 2020] and item recommendation [Pan *et al.*, 2020]. Despite the successes, interpreting a complex DNN still remains an open problem, hindering its wide deployment in safety and security-critical domains. A trustworthy DNN model should not only demonstrates a high performance in its detection and prediction, but also needs to explainable [Adadi and Berrada, 2018]. DNNs are complex nonlinear functions parameterized by model weights; understanding how the information flows from input to output remains a major challenge in Explainable Artificial Intelligence (XAI) research.

*Corresponding author

In general there are two directions towards interpreting DNNs, i.e., gradient based methods, and local approximation methods. Some gradient based methods calculate input feature importance by exploiting its gradient with respect to the model inputs. For example, Saliency Map (SM) [Simonyan *et al.*, 2013] uses gradient directly, Guided Backpropagation [Springenberg *et al.*, 2014] only propagates non-negative gradients, and Integraded Gradients (IG) [Sundararajan *et al.*, 2017] integrates gradients from a reference to input. Class Activation Mapping (CAM) based methods [Zhou *et al.*, 2016; Selvaraju *et al.*, 2017; Chattopadhay *et al.*, 2018] capture the gradient with respect to intermediate layers of convolutional feature maps. As for the local approximation methods, extensive research have been done to explain the local neighborhood behaviors of a complex model by approximating it with a simple yet interpretable model [Ribeiro *et al.*, 2016; Shrikumar *et al.*, 2017; Lundberg and Lee, 2017], such as linear model and decision tree. Other methods such as [Fong and Vedaldi, 2017; Datta *et al.*, 2016; Li *et al.*, 2016] attempt to perturb the inputs to identify the part of inputs that are most responsible for a prediction in the neighborhood.

Within gradient models, although CAM based methods give promising results in various applications, a major limitation is that it applies only to Convolutional Neural Network (CNN) architectures. SM works on non-CNN models, however, it only captures the local gradient information at the inputs, which can be misleading due to the high non-linearity of DNNs. To overcome these limitations, methods such as IG [Sundararajan *et al.*, 2017] are proposed to not only consider the gradients at the input, but also the cumulative gradients along the path from a reference to the input example.

One key issue of IG (as well as other methods such as DeepLIFT [Shrikumar *et al.*, 2017]) is that it explicitly requires a reference (or baseline) to make interpretation. This could result in inconsistent interpretations with regard to different references. Although finding a reasonable reference is not infeasible for easier tasks (e.g. MNIST classification), it could become problematic when the underlying tasks are complicated. As described by the authors of IG paper : "a reference should convey a complete absence of signal". If both white noise image and black image convey no signal, why prefer the latter to the former in simpler tasks? Does it hold true for more complex tasks? In the DeepLIFT paper, a

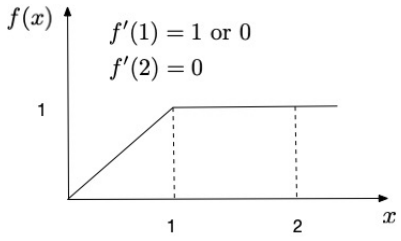


Figure 1: Gradients alone may cause misleading interpretation.

blurred version of the original input is used as the reference as opposed to a black image for CIFAR-10 data. As such, the choice of reference can be *ad hoc* and lack of rigorous justification.

To tackle this major issue, we attempt to eliminate the requirement of reference by utilizing the false class gradients to find adversarial examples for the DNN classifier. A simple intuition is that adversarial examples are well-defined and easy to find given the input and the DNN model. In contrast, the choice of a reference can be more subjective and *ad hoc*. We derive our formulation based on the observation that sum of gradients from all classes equal to 0, as such, the false class gradients are equivalent to the gradient of the true class.

We summarize our main contributions as follows: 1) we propose a novel Adversarial Gradient Integration (AGI) method for a more consistent DNN model interpretation eliminating the need for a reference; 2) we establish the connection between the true class' and the false classes' gradients; and 3) we explain a DNN model's prediction via discriminating against the false classes, instead of focusing only on correct classification of the true class.

2 Preliminaries

To describe our approach, we first review gradient-based SM [Simonyan *et al.*, 2013] and IG [Sundararajan *et al.*, 2017] methods, and show their limitations such as gradient vanishing issue (Figure 1) and *ad hoc* choice of reference issue. In section 3 and 4, we derive our AGI method and show that it has various attractive properties that can overcome the limitation of the previous methods.

2.1 Saliency Map

Saliency Map [Simonyan *et al.*, 2013] is a pioneering visualization method for model interpretation, which simply calculates the gradient of classification output with respect to the input images. Formally, $M_{\text{Saliency}} = \nabla_x f^t(x)$, where t represents the true label, and $f^t(x)$ represents the output value corresponding to the true class label. This method captures the local gradient information at the input, however, it can result in misleading interpretations since local gradient information may not faithfully represent the global attribution. For example, in Figure 1, the gradient at $x = 2$ is 0, however, we could not say that the contribution from x is none. Moreover, when $x = 1$, the gradient may be inconsistent, i.e., 1 or 0, depending on how we define the gradient. These critical issues need to be addressed.

2.2 Integrated Gradients

Different remedial methods have been developed to address the inconsistency among the local gradients shown in Figure 1. For example, DeepLIFT [Shrikumar *et al.*, 2017] calculates the difference between $x = 0$ and $x = 2$ rather than using the gradient at one point. Another method to mitigate the drawbacks of SM is to integrate the gradient from $x = 0$ to $x = 2$ to average the effects. This strategy not only avoids the gradient vanishing issue, but also prevents the obstruction of some singular points (i.e., where gradient is not continuous), as long as the model is integrable within the range. In fact, this is similar to the idea of IG proposed by [Sundararajan *et al.*, 2017]. The formulation of IG is

$$\text{IG}_j(x) ::= \left(x_j - x'_j\right) \times \int_{\alpha=0}^{1} \frac{\partial f\left(x' + \alpha \times (x - x')\right)}{\partial x_j} d\alpha, \tag{1}$$

where j denotes the index of jth input feature, x'_j represents a reference. Although IG successfully addresses the issues of SM method, we point out the following two limitations: 1) a predefined path is needed to integrate from a reference to the original input. IG takes a straight line specified by $\gamma(\alpha) = x' + \alpha \times (x - x')$ in the input space as the integrating path; and 2) a manually selected reference is required because integration must have a starting point.

3 Problem Formulation

Looking into IG's first limitation: *the predefined integrating path*, one motivation of picking the straight line from a reference to the input is because it is the shortest path in the input space. Intuitively, to effectively discriminate the input from a reference point, an integration method should indeed pick the shortest path (if there are any detour, it may then contain information that discriminates from other examples). However, the problem is what we are interested is the learned feature space from the penultimate layer of DNNs, instead of the input space. Hence the shortest path needs to be the one in the learned feature space. As the mapping from input space to feature space is highly non-linear and complex, the corresponding curve, i.e., shortest curve, in the input space is most likely not a straight line.

The problem to solve becomes how can we find such a curve in the input space that may correspond to the shortest path in the feature space? Since operations from the penultimate layer to the output layer is usually linear, the shortest curve from the input point to a reference should correspond to a straight line in the feature space (Figure 2). Thanks to backpropagation method, as long as we find the gradient along the direction of the straight line in feature space, we are guaranteed to obtain the corresponding path in the input space. Assuming the last layer has parameter set W and the penultimate layer's output is $\phi(x)$, the output of the last layer is then $y = W\phi(x)$. Taking the derivative with respect to $\phi(x)$, we have $y' = W$, which corresponds to a constant gradient field. The steepest descent algorithm is guaranteed to find a straight line in a constant gradient field. Using chain rule in backpropagation, we obtain the gradient field in the input space, and hence the corresponding curve.

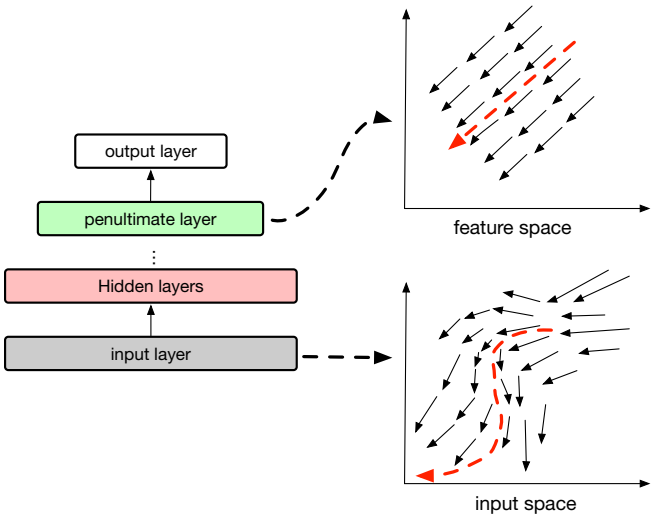


Figure 2: The input space and feature space correspond to the input layer and penultimate layer, respectively. Black solid arrows represent the gradient directions in the corresponding spaces. The red dashed curve represents the steepest ascent/descent path. A straight line in feature space corresponds to a curve in the input space.

With respect to the second limitation: *the choice of reference*, from the above discussion, we actually have already observed the redundancy of the reference: although there should be a destination at the end of the straight line, we don't have to pre-define it. In such a case, the steepest descent algorithm can not only lead us to a candidate reference point, but also help us finding the shortest path. Hence both reference and integration path can be obtained with the model and inputs provided, and is input specific. Figure 2 illustrates the shortest paths in the feature space (right upper panel) and the input space (right lower panel). Note that the assumption of linearity from penultimate layer to output layer is not required since we can use steepest descent algorithm regardless.

The property of steepest descent appears to be flawless, however, the caveat arises from the inconsistency. i.e., ideally, just like the choice of the reference, the descent should lead us to a point where all class outputs being equal. Unfortunately, this usually isn't possible. The reason is that although we can control the descent of the true class, we have no control of which false class will ascend along the path. In fact, descending from the true class could very likely result in ascending to one of the false class instead of evenly distributed to all false class [Goodfellow *et al.*, 2014].

4 Adversarial Gradient Integration

Here we formally describe the proposed AGI method to overcome the aforementioned limitations, i.e., the inconsistency in automatically finding the reference.

4.1 Perspective: Discrimination v.s. Classification

The discussion we have thus far motivates us to define a more consistent and systematic concept than the reference for IG based model explanation. Here we propose using adversarial example in lieu of reference. Unlike the latter whose definition can be vague thus *ad hoc*, a targeted adversarial example is clearly defined as the closest perturbed example to the original input such that it changes prediction of the true class to the targeted class(es).

To understand the rationale behind our choice, let's imagine in a classification task, instead of considering a prediction as *classifying the input as the true class*, why not view it as *discriminating the input to all the false classes*? Therefore, rather than interpreting *what makes the model to classify the true class*, we may equivalently interpret *what makes the model to discriminate the true class from the false classes*.

With the perspective of discrimination in mind, how could we leverage adversarial examples to reinforce the role that a reference plays? Recall in the discussion of gradient SM method, we have the property that all class probabilities are summed to 1, and we have the overall zero gradient, leading us to establish the connection between true class gradient and adversarial gradients as below:

$$\nabla_x \sum_i f^i(x) = 0 \Longleftrightarrow \nabla_x f^t(x) = -\sum_{i \neq t} \nabla_x f^i(x). \quad (2)$$

It means that the gradient of the true class t w.r.t the input is equivalent to the summation of negative gradient of the false classes w.r.t the input. This inspires us that in the neighborhood of the input, the gradient contribution to the true class label and the adversarial class labels are equivalent.

Lets focus on one adversarial false class i, whose gradient we call as an adversarial gradient toward class i. If we follow the direction of the adversarial gradient, and perturb the input via ascending the gradient direction (i.e., $x \leftarrow x + \epsilon \cdot \nabla f^i(x)$, the steepest ascent), over multiple steps, we may eventually approach a point where the perturbed input becomes an adversarial example $x'_{(i)}$, which gives false prediction of class i instead of t.

In this case, lets assume that there is a path denoted by $\gamma(\alpha)$, what we are interested is how the true class prediction $f^t(x)$ changes along the path. And what is its role in making model interpretation? To understand these, let's first define the path gradient integration (PGI):

Definition 1. *Assume f denotes the prediction model, f^t is the true class output of the model. Let x be the original input, and x' is another point in the input space. If there is a path $\gamma(\alpha)$ from x' to x, with $x' = \gamma(0)$ and $x = \gamma(1)$, the path gradient integration for the jth input feature is defined by*

$$PGI_j = \int_{\alpha=0}^{1} \nabla_{\gamma_j} f^t(\gamma(\alpha)) \cdot \frac{\partial \gamma_j(\alpha)}{\partial \alpha} d\alpha. \quad (3)$$

Definition 1 essentially defines a path integration from a starting point to the input point, following the path $\gamma(\alpha)$. If we define the starting point to be the choice of reference x', and take a straight line path from reference x' to the input x, i.e. $\gamma(\alpha) = x' + \alpha \times (x - x')$, Definition 1 becomes an alternative formulation of IG [Sundararajan *et al.*, 2017]:

$$IG_j = \int_{\alpha=0}^{1} \nabla_{\gamma_j} f^t(\gamma(\alpha)) \cdot \frac{\partial \gamma_j(\alpha)}{\partial \alpha} d\alpha, \quad (4)$$
$$\text{where } \gamma(\alpha) = x' + \alpha \times (x - x').$$

But what if we use an adversarial example x_i' as the starting point rather than a reference point? Following the interpretation of IG, which states that the result of IG represents the attribution of the input features to the prediction, similarly, we can interpret the integration from the adversarial example as the attribution of the input feature to *discriminate* the true class t from a false class i.

4.2 Adversarial Gradient Integration (AGI)

Integrating along a straight line from adversarial example x_i' to x is not ideal. The shortest path in the input space doesn't account for the shortest path in learned feature space (as we discussed in Section 3). Hence here the steepest ascent path is chosen as the integration path.

Definition 2. *Given all assumptions from Definition 1. Let f^i be a false class output, $\gamma(\alpha)$ be the path obtained by steepest ascending f^i, and x_i' be the corresponding adversarial example at the end of the path, the adversarial gradient integration of the jth input feature is defined by*

$$AGI_j = \int_{\alpha=0}^{1} \nabla_{\gamma_j} f^t(\gamma(\alpha)) \cdot \frac{\partial \gamma_j(\alpha)}{\partial \alpha} d\alpha, \quad \gamma(\alpha)\text{: a path obtained by the steepest ascent, } x' = \gamma(0) \text{ and } x = \gamma(1). \tag{5}$$

Recall that in [Sundararajan *et al.*, 2017], the authors propose that an attribution method needs to satisfy three axioms, i.e., sensitivity, implementation invariance, and completeness. We argue AGI satisfies all of them due to the nature of path integration. From Definition 2, we can easily find that IG and AGI differ only in two aspects: 1) AGI integrates over the curve of steepest ascent whereas IG integrates over a straight line from the input space; and 2) AGI starts from an adversarial example, while IG starts from a manually selected reference point. The differences essentially are summarize to two benefits of AGI: 1) it gives more intuitive shortest path in the learned feature space, and 2) no need to manually select the reference point, preventing the derived inconsistency.

4.3 From IG to AGI

Follow the interpretation by IG: the IG result shows the contribution of individual input features to the true class t. We interpret AGI similarly as: the AGI result shows the attribution of individual input feature to discriminate t from a false class i. Now, what if we sum AGIs from all false classes? Inspired by Eq. 2, we assume

$$\sum_i AGI_i \sim -IG, \tag{6}$$

which essentially says that the attribution to all discriminations should be equivalent to attribution to classification. The rationale is that although we usually say that *a model classifies the input as something*, another perspective can be, in contrast, that *a model discriminates something from other things*. For example, LeNet discriminates one digit class from other digit classes in MNIST dataset. Hence we can interpret a classification by summing over all interpretations of AGIs, which essentially interprets the discrimination between all adversarial classes and the true class.

Algorithm 1: Individual$AGI(f, \boldsymbol{x}, i, \epsilon, m)$

Input : Classifier f, input $\boldsymbol{x}$, adversarial class i
step size ϵ, max number of steps m;
Output: Individual AGI for class i: AGI_i;
$AGI_i \leftarrow 0$;
$j \leftarrow 0$;
while $\arg\max_\ell f^\ell(\boldsymbol{x}) \neq i$ *and* $j < m$ **do**
  $\boldsymbol{d} \leftarrow \epsilon \cdot \text{sign}(\frac{\nabla_{x_j} f^i(x)}{|\nabla_x f^i(x)|})$; // Adv. direction
  $AGI_i \leftarrow AGI_i - \nabla_{x_j} f^t(x) \cdot \boldsymbol{d}$;
  $\boldsymbol{x} \leftarrow \boldsymbol{x} + \boldsymbol{d}$; // ascending
  j++;
end

4.4 Finding the Steepest Ascending Path

In order to obtain AGI for one false class i, two gradients need to be calculated: $\nabla_x f^i(x)$ and $\nabla_x f^t(x)$. Note that because the path is defined by steepest ascent, then in Eq. 5, we have

$$\frac{\partial \gamma(\alpha)}{\partial \alpha} d\alpha = -\frac{\nabla_x f^i(x)}{|\nabla_x f^i(x)|} d\alpha, \tag{7}$$

the minus sign here is because the direction is opposite (we ascend from x to x_i', while integration is done from x_i' to x). The formulation then becomes

$$AGI_j = \int_{\text{til adv}} -\nabla_{x_j} f^t(x) \cdot \frac{\nabla_{x_j} f^i(x)}{|\nabla_x f^i(x)|} d\alpha, \tag{8}$$

which integrates along the path until $\arg\max_\ell f^\ell(x) = i$ Here we may face the issue that gradient ascending may encounter local maxima that prevents it from ascending further. We adopt the approach proposed by [Madry *et al.*, 2017] that uses the signed gradient instead of the original gradient to make sure it is easier to surpass the decision boundary. In addition, we set a maximum step size m to prevent the path from infinite looping when trapped in local maxima. The algorithm for computing individual AGI is given by Algorithm 1 (the subscript j is omitted for conciseness).

4.5 Individual AGI Aggregation

As for aggregating all individual AGIs, when the number of classes are small, we can simply sum up all AGIs. However, when there are a large amount of classes, such as 1000 classes in ImageNet dataset, calculating all AGIs for 999 false classes become computational prohibitive. In order to alleviate the excessive computational burden, we randomly select a reasonable amount of false classes by sampling from all candidate classes. Although this sampling procedure may lose information from the unselected classes, the resulting AGI can still capture the essential information because the discriminating tasks actually share a fair amount of semantic information. For example, the information for discriminating Labrador from Cat may be similar to discriminating Shepherd from Cat. Hence we argue that sampling a subset of classes is adequate to obtain a satisfying model interpretation and we also provide experimental justifications in Section 5.3 and Section 5.4. The algorithm for calculating AGI is given by Algorithm 2.

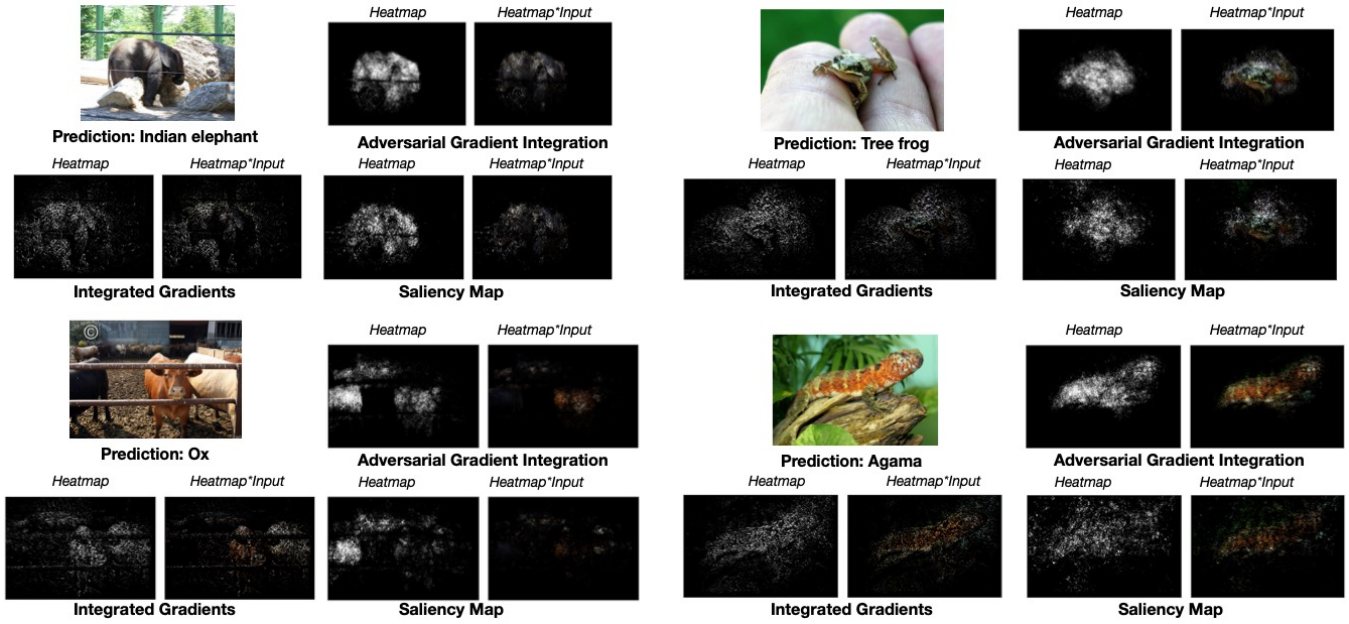


Figure 3: Examples of heatmap interpretations of predictions on Inception V3 using AGI (ours), IG, and Gradient SM. Each method presents both the output heatmap as well as heatmap×Input. Unlike IG and SM whose output heatmaps are usually sparse, AGI's heatmaps are more focused on the target area. This property enables more confident interpretations compared to other methods.

Algorithm 2: $AGI(f, \boldsymbol{x}, \epsilon, k, m)$

Input : Classifier f, input $\boldsymbol{x}$, step size ϵ, subsampling size k, max number of steps m;
Output: AGI;
$AGI \leftarrow 0$;
$S \leftarrow$ Sampling k false classes ;
for i *in* S **do**
  $AGI \leftarrow AGI + IndividualAGI(f, x, i, \epsilon, m)$
end

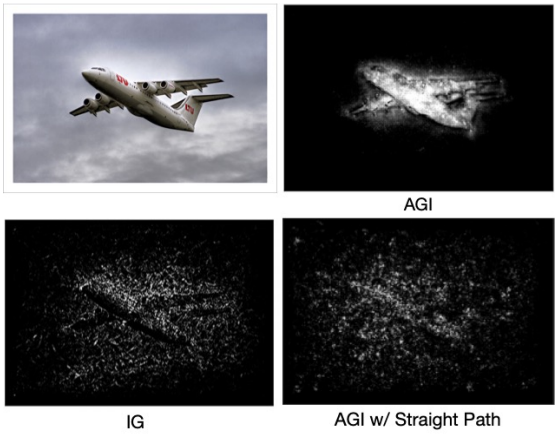


Figure 4: Comparison of heatmap sparsity. Here AGI w/ straight path represents that we replace the integration path of AGI by a straight line in the input space. The results show that inappropriate integration path may result in sparsity.

5 Experiments

In this section, we perform experiments attempting to answer the following questions: 1) does AGI output meaningful interpretations for classifying the true class? 2) does class subsampling compromise the performance? 3) does individual AGI give reasonable interpretation for discriminating the true class against a false class? and 4) does AGI pass sanity checks?

5.1 Experimental Setup

The model to be interpreted includes InceptionV3 [Szegedy *et al.*, 2015], ResNet152 [He *et al.*, 2015] and VGG19 [Simonyan and Zisserman, 2014]. All experiments are conducted using ImageNet dataset.

In terms of baseline DNN interpretation methods, we use SM [Simonyan *et al.*, 2013] and IG[Sundararajan *et al.*, 2017] as baselines for qualitative and quantitative comparisons of interpretation quality. Additionally, Guided-Backpropagation [Springenberg *et al.*, 2014] is selected for comparison with AGI in the sanity check experiments. Regarding parameter settings, we set the step size $\epsilon = 0.05$, and the class subsampling size for ImageNet to 20. As for the reference for IG method, we use the default choice in the original paper (i.e., black image).

Additional data processing to optimize the heatmap visualization have also been used. We reassign all heatmap attributions less than $q = \text{Percentile}(80\%)$ to be q (lower bound), and all values larger than $u = \text{Percentile}(99\%)$ to be u (upper bound), then normalize them within $[0, 1]$. The lower bound is set because that we only want to focus on the area with relatively high attributions, a low attribution is likely caused by

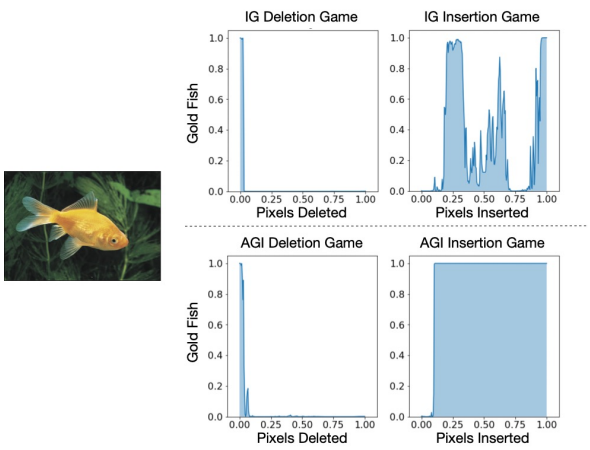


Figure 5: An example of insertion game and deletion game. The insertion (deletion) score is obtained by calculating the area under the curve of the insertion (deletion) game.

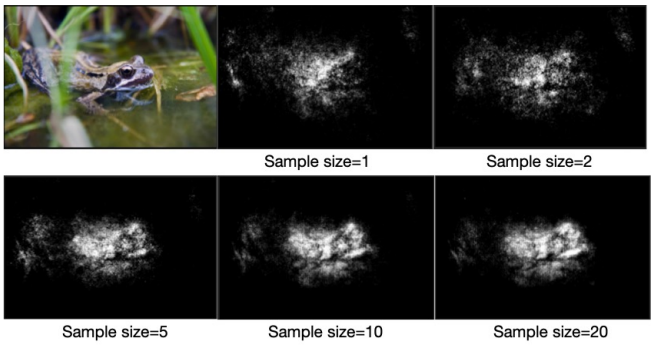


Figure 6: By increasing the number of classes in subsampling, the quality of explanation improves markedly up to $n = 10$ then stabilizes afterwards.

background. The upper bound is set to avoid extremely high values that can potentially undermine the visualization (This procedure is applied to all methods).

Our implementation for generating the steepest ascent curve is inspired from the PGD attack algorithm [Madry *et al.*, 2017]. For InceptionV3, setting the max ascending step = 20, and sample size = 20, it will cost $\approx$ 15 seconds to interpret a single 224 $\times$ 224 color image on a computer with Nvidia GTX 1080 GPU. However, the path-finding procedure for sampled negative classes can be paralleled to speed up the running time, making the overall process less than 1 second.

5.2 Qualitative Evaluation

Figure 3 shows examples of different interpretation methods explaining predictions made by InceptionV3 (Additional examples and experiments on Resnet152 and VGG19 can be found in the supplementary materials). A key observation from the experiments is that IG's output heatmap is far more sparse than AGI's (by sparse, we mean high attribution values are sparsely distributed in the map instead of concentrating on the target objects). We argue that this phenomenon is sourced from two aspects: First, IG uses a black image as the

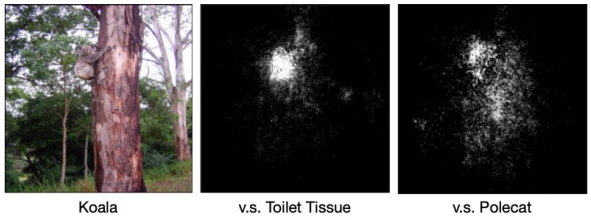


Figure 7: An example on choice of discriminating class on interpretation. Only Koala itself is highlighted when discriminating against Toilet Tissue whereas part of the tree trunk is also highlighted together with Koala when discriminating against Polecat.

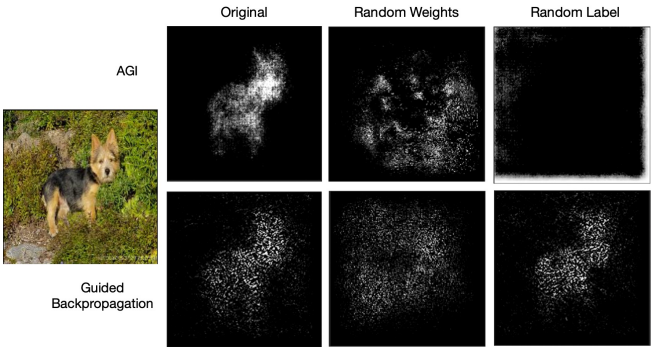


Figure 8: An example on sanity checks. The upper panel shows the original heatmap, heatmap with randomized model, and heatmap with randomized data label of the AGI, respectively. The lower panel shows the counterparts obtained by guided backpropagation method.

default reference whereas AGI doesn't. Black reference image may be good for those cases where objects are light colored, but can fail when the target object is dark. For example, the heatmap of Indian Elephant generated by IG (Figure 3) doesn't highlight the body of the elephant which is in fact under the shadow whereas that generated by our AGI has no such issue. Second, IG uses shortest integration path (straight line) in input space whereas AGI uses shortest path in the learned feature space. To demonstrate the relationship between the straightline integration and sparsity, we replace our AGI's integration path with the straight line, i.e., instead of integrating over the curve of steepest ascent, we integrate from adversarial examples to the input example over a straight line like IG. The results in Figure 4 show that an inappropriate integration path may cause heatmap sparsity.

5.3 Quantitative Evaluation

In addition to the qualitative examples presented above, we also conduct quantitative experiments to validate our method using insertion scores and deletion scores [Petsiuk *et al.*, 2018]. Starting from a blank image, the insertion game successively inserts pixels from highest to lowest attribution scores and makes predictions. We draw a curve that represents the prediction values, the area under the curve (AUC) is then defined as the insertion score. The higher the insertion score, the better the quality of interpretation. Similarly, starting from the original image, the deletion score is obtained by successively deleting the pixels from the high-

Metrics	Deletion Score						Insertion Score					
Methods	IG	SM	AGI-1	AGI-5	AGI-10	AGI-20	IG	SM	AGI-1	AGI-5	AGI-10	AGI-20
InceptionV3	**0.032**	0.036	0.043	0.043	0.046	0.048	0.294	0.537	0.408	0.503	0.532	**0.561**
ResNet152	**0.030**	0.056	0.044	0.052	0.056	0.060	0.262	0.407	0.405	0.475	0.489	**0.503**

Table 1: Deletion score: the lower the better; Insertion score: the higher the better. Here AGI-n represents the corresponding AGI method with n subsampled false classes. The benefit of increasing the size of class subsampling becomes diminishing.

est to lowest attribution scores. The lower the deletion score, the better the quality of interpretation. An example of such process is shown in Figure 5. Table 1 shows the average scores over 1000 test examples in ImageNet by InceptionV3 and ResNet152.

While IG, SM and AGI all have sufficiently small deletion scores (Note that deletion score become less indicative when getting extremely small due to the existence of adversarial effects in DNNs [Petsiuk *et al.*, 2018]). AGI has much larger insertion scores than the competitors (Table 1). Since the latter is an indicator of the ability to detect the target object, we conclude that AGI outperforms IG and SM in terms of detecting the meaningful objects. Observing the trend from AGI-1 to AGI-20, the insertion score converges when the subset of selected false classes become sufficiently large. In the case of InceptionV3, from AGI-1 to AGI-10, the insertion score gain per additional class is ~ 0.014 whereas the score gain per additional class become ~ 0.003 from AGI-10 to AGI-20. This demonstrates that a relatively small subset of false classes is sufficient to obtain a good interpretation.

5.4 Class Subsampling

A major computational burden of our AGI method is to calculate individual AGI for all false classes. As we argued before, random subsampling doesn't affect the results as long as the sample size is sufficiently large. Although individual AGIs may be different for different false classes, the aggregated AGI converges when sample size increases. As such, the information from the sampled AGIs is sufficient to generalize to all other classes. Figure 6 substantiates our claim: the results indeed become more clear when sample size increases from 1 to 10 but doesn't change too much afterwards. This observation indicates that different discriminating tasks may share a fair amount of input attributions, hence a small subset can be sufficiently representative for the overall interpretation. Note that it is possible to utilize the semantic information to help selecting the subset of classes, which could render better results. We didn't utilize it for subsampling in this paper mainly because that the random selection method can already output promising results. However, we do point out that utilizing semantic information for subsampling process may potentially reduce the sampling size.

5.5 Discrimination from Other Classes

One of our main contributions is that we decompose the interpretation of classifying the true label into the sum of interpretation of discriminating against false labels. To demonstrate our claim, we conduct experiments to interpret model discrimination using individual AGIs (Algorithm 1). The individual AGI should represent the attributions that discriminate true class from the specific false class.

Figure 7 shows an example from ImageNet dataset. To discriminate Koala from Toilet Tissue, we can observe that only the body of Koala from the image is highlighted in the heatmap. However, when attempting to discriminate it from a Polecat, the attribution from the trunk become more prominent, which means the latter is used to reinforce explanation. The underlying reason behind it could be that Polecats don't live on trees, while Koala do.

5.6 Sanity Checks for Image Interpretation

As pointed out by [Adebayo *et al.*, 2018], visual interpretation could be misleading, as some previous interpretation methods are just edge detection instead of genuine interpretation. In case when an interpretation method is independent of either the prediction model or of the data generating process, a sanity check is required for validating its correctness.

Here we perform two tests for the sanity check, 1) a model parameter randomization test: the interpretation of model with learned parameters should be substantially different from the model with random parameters; and 2) a data randomization test: the same input with different labeling should result in different interpretations. Figure 8 shows that our AGI method passes both tests (first row) since after both model randomization and data randomization, the outputs are significantly different from the original ones. While Guided-Backpropagate [Springenberg *et al.*, 2014] (second row), on the other hand, doesn't pass the data randomization test, as randomizing data label should has significant heatmap differences from the correct label (the heatmap corresponding to 'original' and 'random' labels).

6 Conclusion

In this paper, motivated by the limitations of two well-established DNN interpretation methods, SM and IG, we propose a novel attribution method, i.e., AGI, which doesn't require a manually selected reference, nor a predefined integration path. As such it can be applied to automatically and consistently explain DNNs' predictions. Through extensive experiments, our AGI method significantly outperforms the competing methods in both qualitative and quantitative experiments. Our AGI method can be broadly applied to explain a wide range of DNN models' predictions.

Acknowledgements

This work is supported by the National Science Foundation under grants CNS-2043611 and IIS-1724227.

Ethical Impact

DNN models have been increasingly deployed in many security and safety-critic real world settings. Despite the im-

pressive performance, they are mostly black-box models that usually fail to give insight on why and how they make predictions. As trustworthiness and transparency become more salient issues, interpretable DNN methods are highly desirable and their wide adoption is expected to accelerate our current pace of leveraging AI for social good.

References

[Adadi and Berrada, 2018] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[Adebayo *et al.*, 2018] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31:9505–9515, 2018.

[Chattopadhay *et al.*, 2018] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 839–847. IEEE, 2018.

[Datta *et al.*, 2016] Anupam Datta, Shayak Sen, and Yair Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, pages 598–617. IEEE, 2016.

[Fong and Vedaldi, 2017] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3429–3437, 2017.

[Goodfellow *et al.*, 2014] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[He *et al.*, 2015] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. corr abs/1512.03385 (2015), 2015.

[Li *et al.*, 2016] Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016.

[Li *et al.*, 2020] Xiangrui Li, Xin Li, Deng Pan, and Dongxiao Zhu. On the learning property of logistic and softmax losses for deep neural networks. In *AAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 4739–4746. AAAI Press, 2020.

[Lundberg and Lee, 2017] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774, 2017.

[Madry *et al.*, 2017] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[Pan *et al.*, 2020] Deng Pan, Xiangrui Li, Xin Li, and Dongxiao Zhu. Explainable recommendation via interpretable feature mapping and evaluation of explainability. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 2690–2696. ijcai.org, 2020.

[Petsiuk *et al.*, 2018] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018.

[Qiang *et al.*, 2020] Yao Qiang, Xin Li, and Dongxiao Zhu. Toward tag-free aspect based sentiment analysis: A multiple attention network approach. In *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*, pages 1–8. IEEE, 2020.

[Ribeiro *et al.*, 2016] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[Selvaraju *et al.*, 2017] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[Shrikumar *et al.*, 2017] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3145–3153. JMLR. org, 2017.

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

[Simonyan *et al.*, 2013] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[Springenberg *et al.*, 2014] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[Sundararajan *et al.*, 2017] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365*, 2017.

[Szegedy *et al.*, 2015] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. corr abs/1512.00567 (2015), 2015.

[Zhou *et al.*, 2016] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.

Counterfactual Interpolation Augmentation (CIA): A Unified Approach to Enhance Fairness and Explainability of DNN

Yao Qiang , **Chengyin Li** , **Marco Brocanelli** and **Dongxiao Zhu***
Department of Computer Science, Wayne State University, USA
{yao, cyli, brok, dzhu}@wayne.edu

Abstract

Bias in the training data can jeopardize fairness and explainability of deep neural network prediction on test data. We propose a novel bias-tailored data augmentation approach, Counterfactual Interpolation Augmentation (CIA), attempting to debias the training data by d-separating the spurious correlation between the target variable and the sensitive attribute. CIA generates counterfactual interpolations along a path simulating the distribution transitions between the input and its counterfactual example. CIA as a pre-processing approach enjoys two advantages: First, it couples with either plain training or debiasing training to markedly increase fairness over the sensitive attribute. Second, it enhances the explainability of deep neural networks by generating attribution maps via integrating counterfactual gradients. We demonstrate the superior performance of the CIA-trained deep neural network models using qualitative and quantitative experimental results. Our code is available at: https://github.com/qiangyao1988/CIA

1 Introduction

Deep neural network (DNN) trained with biased data is known to learn and exploit the spurious correlation between the target variable and the sensitive attribute (e.g., color, gender, and race) as a shortcut for prediction [Kim *et al.*, 2019; Geirhos *et al.*, 2020]. However, the spurious correlation may only reflect dataset-specific biases or sampling artifacts rather than the causal mechanism between the intended feature and target variable. As a result, the DNN's output may be biased against the protected groups defined by the sensitive attribute. For example, a facial recognition model performs poorly for female with darker skin compared to other gender/race groups [Buolamwini and Gebru, 2018]. Developing bias mitigation techniques to alleviate the adverse effect has attracted increasing attention in recent years.

Extensive approaches have been developed to mitigate bias in DNN's prediction. Many methods attempt to remove sensitive information from the learned features during the training process [Madras *et al.*, 2018; Kim *et al.*, 2019; Li *et al.*, 2020]. However, the adversarial training and disentangled representation learning approaches are limited because they potentially remove some useful information related to the sensitive attribute, thus compromising the model performance on the target task. [Kim *et al.*, 2021] aim to debias and increase the quality of the training set via data augmentation. Despite its initial success, they augment data through linearly interpolating the latent features from the discriminative models, limiting their capability to generate a set of legitimate and manifold data augmentations. Clearly, generative models that learn the distribution of features provide a promising solution.

While many existing approaches ensure fairness, explainability arises as another salient challenge. Besides selecting appropriate metrics (e.g., demographic parity, equality-of-odds) for fairness evaluation, researchers attempt to apply model explanation techniques to help understand whether a DNN model makes fair decisions [Qiang *et al.*, 2020; Pan *et al.*, 2020; Tong and Kagal, 2020]. Among others, feature attribution methods (e.g., IG [Sundararajan *et al.*, 2017]) calculating the attribution of each input feature as its importance have gained great success. Nevertheless, the computing process may be misled by the sensitive attribute, resulting in incorrect explanations as shown in Figure 1(d), due to the arbitrary choices of the baseline and integral path.

To address the above problems, we design a bias-tailored counterfactual interpolation augmentation (CIA) approach to 1) mitigate bias in the training set, and 2) develop fair and explainable DNN models using the counterfactual interpolations generated from CIA. Our unified approach is illustrated in Figure 1. Here we mitigate bias in the training set through the lens of counterfactual fairness [Kusner *et al.*, 2017; Pfohl *et al.*, 2019]. The counterfactual causal inference is modeled using a conditional variational auto-encoder (CVAE) [Sohn *et al.*, 2015], which generates the counterfactual interpolations by interpolating the sensitive attribute along a constructed path simulating the distribution transitions between the sensitive groups. We then inject the bias-tailored counterfactual interpolations into the biased training set to intervene the spurious causal effect. Therefore, DNN models trained with CIA tend to learn the features that are truly causal to the target variables, resulting in fair outputs.

Similar to the attribution methods, the counterfactual ex-

*Corresponding Author

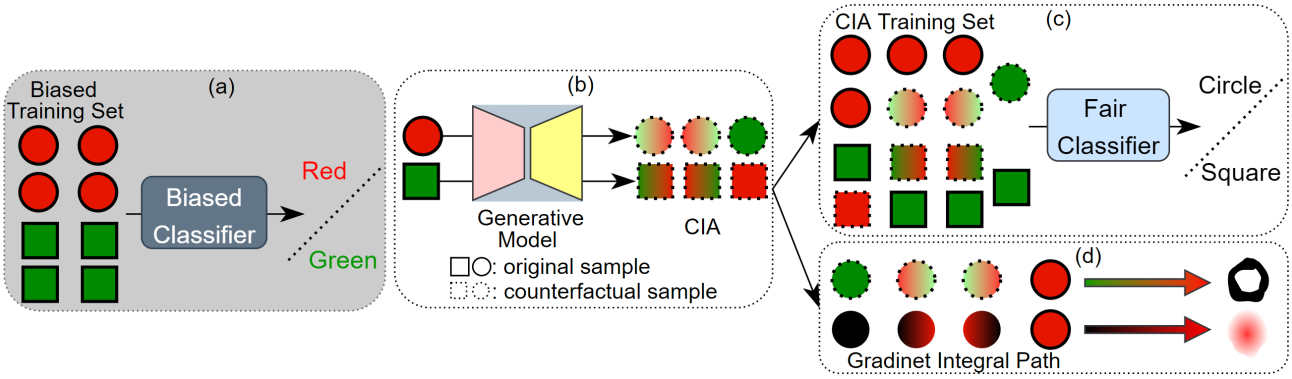


Figure 1: An illustrative example. (a) The target variable (shape) is spuriously correlated with the sensitive attribute (color) in the biased training set. A biased classifier undesirably learns and leverages the spurious correlations for prediction. (b) Our CIA generates bias-tailored counterfactual interpolation augmentation to mitigate bias in the training set and to enhance fair explanation. (c) CIA enables training a fair classifier to learn discriminative features for shape classification. (d) In the first row, CIA generates a meaningful explanation for classifying the target (shape). In the second row, a baseline interpolation generates explanation of the target (shape) confounded by the sensitive attribute (color). Best viewed in color.

planation can give powerful insights into what is important to the underlying decision process leveraging the counterfactual examples, which are in contrast with the original input by making some artificial modifications on the features of interest [Kusner *et al.*, 2017; Wachter *et al.*, 2017]. Here we develop a new DNN model explanation method that integrates gradients along the interpolated path simulating the distribution transitions from the counterfactual example to the input. Since the gradient integration focuses on the intended attributes and does not get distracted by the sensitive attribute, our method can generate more meaningful explanations by dissolving the negative impacts from the sensitive attribute.

We summarize our contributions as follows:

- First, we propose CIA, a novel data augmentation strategy to increase DNN fairness via de-correlating the target from the sensitive attribute in training data.
- Second, we design an DNN model explanation method that leverages the generated counterfactual interpolations from CIA for gradients integration. We demonstrate this work as a unified approach to enhance both fairness and explainability of the DNN.
- Third, we experimentally show that CIA minimizes the detrimental effects of bias using two benchmark datasets. The experiment results demonstrate several quantitative and qualitative benefits of our DNN model explanation approach.

2 Related Works

2.1 Debiasing Learning

Fairness has attracted increasing attention since DNN often exhibits bias towards/against certain protected groups, e.g., as defined by sensitive attributes, such as gender and race [Madras *et al.*, 2018]. The existing fairness-aware prediction methods can be categorized into pre-processing, in-processing, and post-processing approaches.

Pre-processing. Pre-processing approaches attempt to debias and increase the quality of a training set through data augmentation. [Zhang and Sang, 2020] propose to balance data distribution for visual debiasing by adding supplementary adversarial examples. This method relies on selecting adversarial attacks and an auxiliary task classifier to generate adversarial examples. [Kim *et al.*, 2021] first divide the training images into bias-guiding and bias-contrary samples based on the assumption that the bias attributes are easy-to-learn. Then, they generate the bias-swapped image augmentations containing the bias attributes from the bias-contrary images while preserving bias-irrelevant ones in the bias-guiding images. [Chuang and Mroueh, 2021] present fair mixup as a new data augmentation method to generate interpolated samples between the sensitive groups. Fairness is achieved by regularizing the trained DNN model on the path of the generated interpolations with fairness constraints. However, their approach only performs data augmentation by leveraging the latent features from the discriminative models limiting their ability to generate a set of legitimate and manifold instances. Differently, generative models are designed to characterize the probability density of observations in the latent space, leading to a better description of the training dataset. Consequently, they can generate the manifold data augmentations via sampling from the learned latent feature distributions.

In-processing. In-processing approaches aim to remove the sensitive information from the learned features during the training process. Some approaches enforce constraints for specific fairness metrics (e.g., demographic parity and equality-of-odds) via an auxiliary regularization term, either adding constraints to disentangle the association between model predictions and sensitive attributes [Nam *et al.*, 2020] or updating objective function to minimize the performance difference between certain groups [Sagawa *et al.*, 2019]. The problem is that the models may behave differently at inference time even though such fairness constraints are satisfied during training. Adversarial training is enabled through the min-max objective: maximizing the classifier's ability to pre-

dict the target variable while minimizing the adversary's capability to predict the sensitive attribute [Madras *et al.*, 2018; Kim *et al.*, 2019]. Nevertheless, this process can compromise the model performance on the main classification task. [Hong and Yang, 2021] design a novel fairness constraint loss, Bias-Contrastive, utilizing the constractive learning to encourage the proximity between the training examples with the same target class but different bias class in the feature space. Performance of this constractive learning method heavily relies on the choice of positive and negative samples.

Post-processing. Post-processing approaches calibrate or modify the predictions according to the sensitive attribute at inference time [Hardt *et al.*, 2016]. These methods require access to the sensitive attribute, they are not feasible for real-world applications due to the salient security and privacy concerns.

Here we advocate for the pre-processing approach, which is agnostic to the choice of fairness algorithms and DNN model architectures. Moreover, it generates debiased data that can be used for training DNNs to further increase their fairness. Our training data debiasing strategy falls under the umbrella of causal fairness [Kusner *et al.*, 2017] aiming to enforce the model to concentrate more on task-relevant causal features while getting rid of the superficial correlations. Moreover, our strategy is expected to enhance the quality of the existing model explanation as described below.

2.2 Integrated Gradients and Variants

Gradient-based feature attribution techniques interpret DNN in terms of the gradient, i.e., the partial derivative of the output with respect to the input, as a sensitivity measurement of the network for each input element [Sundararajan *et al.*, 2017; Erion *et al.*, 2021]. Integrated Gradients (IG) [Sundararajan *et al.*, 2017] applies integrated gradients along a linear path from a baseline to the input avoiding the well-known problem of gradient saturation, i.e., the gradients may not reflect feature importance [Miglani *et al.*, 2020]. The formulation of IG is:

$$\mathrm{IG}_i(x) = (x_i - x_i') \cdot \int_0^1 \frac{\partial F(x_i' + \alpha \cdot (x_i - x_i'))}{\partial x_i} d\alpha, \quad (1)$$

where x is the input, x' is the baseline representing a missing or neutral input (e.g., a black or random noise image). $F(\cdot)$ denotes the prediction output. The linear integral path is denoted as: $\gamma(\alpha) = x' + \alpha \times (x - x')$, where $\alpha \in [0, 1]$. To compute this integral efficiently, authors propose a Riemann summation approximation.

Recent studies demonstrate that the choice of baseline heavily impacts the quality of feature attributions [Haug *et al.*, 2021]. [Sturmfels *et al.*, 2020] present several alternatives: (1) Maximum distance baseline; (2) Blurred baseline [Xu *et al.*, 2020]; (3) Gaussian baseline; and (4) Uniform baseline. [Izzo *et al.*, 2020] propose the neutral baseline lying on the decision boundary of the predictive model. More recently, [Pan *et al.*, 2021] develop Adversarial Gradient Integration, which releases the choice of the baseline by integrating the gradients from adversarial examples to the target input. [Kapishnikov *et al.*, 2021] introduce Guided IG, which aligns the model's prediction and the input to explicitly reduce the noise in resulting attributions with a condition path.

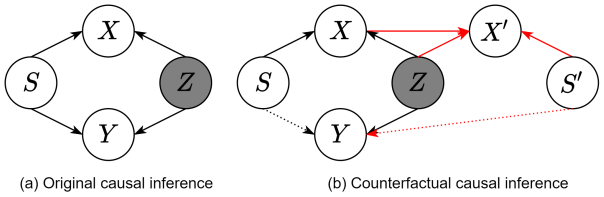


Figure 2: Structure of the hypothesized causal graphs. (a) Unobserved latent variables Z and sensitive attribute S are two confounders that jointly generate the observed data X and the outcome Y. (b) Add another confounder S' to generate the counterfactual example X'.

We point out a key issue that the existing choices of the baseline critically impact attribution quality, leading to unfair explanations in the debiasing learning scenario. To overcome this, we develop a new attribution based technique, which integrates gradients along the counterfactual interpolated path to achieve a higher explanation quality.

3 Counterfactual Interpolation Augmentation

3.1 Notations

Let $\mathcal{X} = \{x_i, y_i, s_i\}, i \in 1, ..., N$ be the training set, where x_i is the input, y_i denotes the target label, and s_i represents the sensitive attribute. For ease of notation, we consider binary sensitive attributes in the following sections. z is the latent space feature. x' and s' denote the counterfactual samples of x and s, respectively. We use capital letters to denote the random variables.

3.2 Counterfactual Causal Inference

Counterfactual fairness [Kusner *et al.*, 2017] requires the same distribution of predictions for each sample in the factual world where $S = s$ and in counterfactual world where $S = s'$, for all $s' \neq s \in \mathcal{S}$. It refrains the sensitive attribute from being the cause of a change in the model prediction.

Definition 1. (Counterfactual Fairness) [Kusner *et al.*, 2017] *A classifier $\hat{Y}$ is counterfactually fair if under any context $X = x$ and $S = s$,*

$$\begin{aligned} &p(\hat{Y}_{S \leftarrow s} = y | X = x, S = s) \\ =&p(\hat{Y}_{S \leftarrow s'} = y | X = x, S = s'), \end{aligned} \quad (2)$$

for all y and for any value s' attainable by S.

However, the counterfactual fairness only requires the predictions to be the same across factual-counterfactual pairs, regardless of whether those pairs share the same value of the target y. Following [Pfohl *et al.*, 2019], we further require the model to be counterfactually fair, conditioning on the factual target y, formally:

$$\begin{aligned} &p(\hat{Y}_{S \leftarrow s} = y | X = x, Y = y, S = s) \\ =&p(\hat{Y}_{S \leftarrow s'} = y | X = x, Y = y, S = s'). \end{aligned} \quad (3)$$

We seek to address the training data bias problem through the lens of causal inference motivated by Definition 1 and Eq.

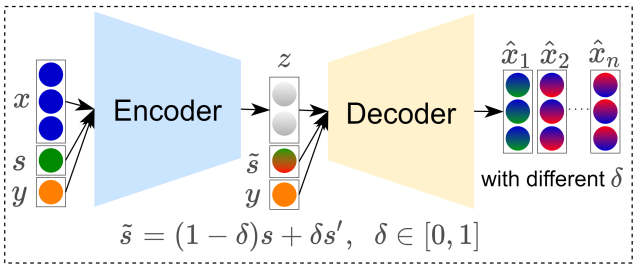


Figure 3: CIA employs a pre-trained CVAE to generate a set of counterfactual interpolations $(\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_n)$ of x conditioned on interpolated sensitive attributes $\tilde{s}$ and y, where s' contrasts with s.

3. However, it is hard to identify the causal mechanisms from limited observational data that may be sampled from a single biased training distribution. It would be a natural decision to help identify the counterfactual causal mechanisms with additional hand-designed counterfactual examples.

Figure 2(a) illustrates the causal graph, modeling the generative process of the original biased dataset $\mathcal{X}$, in which z is drawn from an isotropic Gaussian prior: $z \sim p(Z) = \mathcal{N}(0, I)$, s is drawn form a multinomial distribution with marginals π: $s \sim p(S) = \text{Categorical}(S|\pi)$, and x and y are drawn independently given s and z: $x, y = p(X|Z, S)p(Y|Z, S)$. The data bias problem is caused by the distribution of sensitive attribute $p(S)$, e.g., s is randomly drawn from a multinomial distribution. We model the counterfactual causal inference to generate counterfactual interpolation augmentations illustrated in Figure 2(b). A counterfactual generative process is $x', y = p(X'|Z, S')p(Y|Z, S')$, and here S' is a new confounding variable in contrast with S.

3.3 Generating Counterfactual Interpolations

It is generally impossible to infer the causal structure of the underlying data generating process directly from the observable properties. Therefore, we employ a generative model to capture the causal structure in the presence of an unobserved confounder with observable proxies [Madras *et al.*, 2019].

We first pre-train a generative model (e.g., CVAE) in which the encoder and decoder inputs are conditioned on the sensitive attribute and target variable. Concretely, the encoder learns $q_\phi(z|x, y, s)$, which is equivalent to learning latent feature z of data x with condition s and y. The decoder learns $p_\theta(x|z, y, s)$ decoding the latent feature z with condition s and y to input space. The generative model is trained to minimize the following objective function:

$$\begin{aligned}\mathcal{L}_{\text{CVAE}}(\theta, \phi) = &- \mathbb{E}_{q_\phi(z|x,y,s)} \log p_\theta(x|z, y, s) \\ &+ \text{KL}(q_\phi(z|x, y, s)||p_\theta(z)).\end{aligned} \tag{4}$$

The first term denotes a reconstruction loss encouraging the encoder to map the observed data (x, y, s) into latent feature z and the decoder to reconstruct x from (z, y, s). The second term indicates a regularization making the distribution $q_\phi(z|x, y, s)$ similar to a prior Gaussian distribution $p(z)$ by Kullback–Leibler (KL) divergence

CVAE can generate non-existent manipulated samples as interpolations for real samples along any arbitrary axis. We design an interpolated path moving linearly along the sensitive attribute s as:

$$\tilde{s} = (1 - \delta) \cdot s + \delta \cdot s', \delta \in [0, 1], \tag{5}$$

and inject $\tilde{s}$ into the decoder of the pre-trained CVAE as shown in Figure 3. We generate a set of counterfactual interpolations $(\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_n)$ transiting from the factual example x to its counterfactual example x' along the interpolated path defined in Eq. 5. The variation of δ determines the number of generated interpolations. This interpolated process is applicable regardless of a single sensitive attribute (e.g., color in BiasedMNIST dataset) or multiple sensitive attributes (e.g., gender and age in CelebA dataset).

4 Training and Interpreting Fair DNN

4.1 Training Fair DNN with CIA

By adding the generated counterfactual interpolations $\mathcal{X}_{\text{CIA}}$, we obtain our augmented training dataset $\mathcal{X}_{\text{AUG}} = \mathcal{X} \cup \mathcal{X}_{\text{CIA}}$. A reasonable amount of counterfactual interpolations in $\mathcal{X}_{\text{CIA}}$ alleviate the dataset bias caused by the sensitive attribute in $\mathcal{X}$, thus preventing the model from learning biased representation. Finally, we train our debiased model F_{debias} on $\mathcal{X}_{\text{AUG}}$ with the cross-entropy objective:

$$\mathcal{L}_{\text{class}} = \mathbb{E}_{x \sim \mathcal{X}_{\text{AUG}}} \left[- \sum_c y_c \log F_{\text{debias}}(x) \right], \tag{6}$$

where c is the index of the classes.

4.2 Counterfactual Gradients Integration

IG sums gradients over gradual modifications from a baseline to the original input, essentially distributing the total change in model output across gradual input changes. IG's performance heavily relies on the choice of baseline. An arbitrary choice could negatively impact the explanatory power and lead to meaningless explanations. The explanation generated from IG using a black image baseline without the sensitive attribution cannot correctly reflect feature importance in the debiasing learning scenario as illustrated in Figure 1(d).

We propose a gradient-based feature attribution technique, Counterfactual Gradients Integration (CGI), which leverages the counterfactual interpolations generated from CIA to artificially induce a procedure on how the model attention moves across the gradual changes on the sensitive attribute of the input while computing the final prediction score. Thus, CGI can generate explanations regardless of bias while querying a fair DNN model for gradients.

4.3 Path Integral of CGI

IG pre-defines a straight line as the path integral from the baseline x' to the original input x as $\gamma(\alpha) = x' + \alpha(x - x')$, where $\alpha \in [0, 1]$, i.e., $\gamma(0) = x'$ and $\gamma(1) = x$. The baseline x' represents the absence of features. In CGI, we design the path integral as the interpolated path, transiting from the counterfactual sample x' to the input x for generating counterfactual interpolations in CIA, formally: $\gamma(\delta) = g(x, (1-\delta) \cdot s + \delta \cdot s')$, where $g(\cdot)$ denotes the pre-trained generative model and $\delta \in [0, 1]$. We formulate $\text{CGI}_i(x)$ along the

i-th dimension for an input x and its counterfactual example x' as:

$$\mathrm{CGI}_i(x) = (x_i - x'_i) \int_{\delta=0}^{1} \frac{\partial F(\gamma(\delta))}{\partial \gamma_i(\delta)} \frac{\partial \gamma_i(\delta)}{\partial \delta} d\delta. \quad (7)$$

CGI is obtained by accumulating the gradients along the integration path $\gamma(\delta)$ by varying the δ parameter. The model will encounter interpolations on the sensitive attribute from s' to s during the CGI process.

5 Experiments and Results

5.1 Datasets

BiasedMNIST. Following [Arjovsky *et al.*, 2019], we modify MNIST by introducing color (i.e., red and green) as the sensitive attribute correlating strongly (but spuriously) with the target labels in the training set. A fairness-indifferent DNN model can easily achieve high accuracy by only learning the superficial properties (colors) instead of the inherent properties (shapes) for digit recognition. However, such a biased model can fail at inference time when the spurious correlation between the sensitive attribute and the target shifts or vanishes, for example, randomly coloring the digits.

CelebA. The CelebA is a multi-attribute dataset for face recognition with 40 binary attribute annotations for each image. Following [Nam *et al.*, 2020], we select $HeavyMakeup$ and $HairColor$ as target attributes (y) and $Gender$ as the sensitive attribute (s). There is a significant spurious correlation between the target and the sensitive attributes (i.e., most women have blond hair or wear heavy makeup in this dataset). [Nam *et al.*, 2020] compiled two test datasets: unbiased, by selecting the same number of images for every possible value of the pair (y, s), and bias-conflict, by removing all the samples where y and s have the same values from the unbiased set.

5.2 Implementation Details

Architecture details. We employ the LeNet-5 and a pre-trained VGG-16 as the feature extractor along with two fully connected layers as the classification models for BiasedMNIST and CelebA, respectively. The encoder and decoder in CVAE for BiasedMNIST are multi-layered perceptrons consisting of three hidden layers where the latent feature dimension is set to be 2. For CelebA, the encoder of CVAE has 4 $\times$ Conv2D layers with a 3×3 kernel. The decoder consists 4 $\times$ Conv2DTranspose layers with a 3×3 kernel. A batch normalization layer and Leaky ReLu activation function are added after the Conv2D and Conv2DTranspose layers. The latent feature dimension is set to be 128. We add a fourth channel to each image to encode the sensitive attributes.

Training details. We use Adam optimizer throughout all the experiments in the paper. All models are trained with a learning rate of 0.001 and a batch size of 64. We train the classification models for 5 epochs using the cross-entropy loss. We train CVAEs for 50 and 20 epochs for BiasedMNIST and CelebA, respectively, with binary cross-entropy loss as the reconstruction objective. We generate counterfactual interpolations for the whole training set using the pre-trained CVAE following our CIA approach for BiasedMNIST. Since CelebeA dataset is much larger, with more than 160,000 images in the training set, we randomly select 10,000 samples from the training set and generate their counterfactual interpolations.

5.3 Baseline Methods

LAFTR. [Madras *et al.*, 2018] explore adversarial representation learning ensuring group fairness (e.g., demographic parity, equalized odds, and equal opportunity) to different adversarial objectives.

PriorTraining. [Wang *et al.*, 2021] propose a general framework for learning interpretable fair representations by introducing an interpretable "prior knowledge" during the representation learning process. They add an adversarial loss similar to LAFTR as fairness constraints. Another prior loss is used to ensure the interpretable feature learning.

Group DRO. [Sagawa *et al.*, 2019] aim to minimize "worst-case" training loss over a set of pre-defined groups. The authors expect that models that learn the spurious correlation between sensitive attributes and target variables would perform poorly on groups for which the correlation does not hold. By adding a strong regularization on the worst-case groups, Group DRO can prevent the models from learning pre-specified spurious correlations.

LfL. [Sagawa *et al.*, 2019] propose a failure-based debiasing scheme by training a pair of neural networks simultaneously. The first network is trained to be biased by repeatedly amplifying its "prejudice". They debias the training of the second network by focusing on samples that go against the prejudice of the first network.

5.4 BiasedMNIST Results

We compare our method with the vanilla models (Vanilla, plain training without any debiasing procedure), LAFTR [Madras *et al.*, 2018], and PriorTraining [Wang *et al.*, 2021]. We quantitatively assess the effectiveness of different methods via comparing classification performance on training and test sets. The results are shown in Table **??**. The vanilla model heavily relies on the spurious correlation between color (sensitive attribute) and digit (target), so it fails to learn the digit shape during training, resulting in a large accuracy drop on the test set (79.48 $\rightarrow$ 18.08). LAFTR and PriorTraining apply adversarial training to remove sensitive information from the learned features, which may compromise the model performance on the main classification task. Our CIA debiases the training set using counterfactual interpolations and consequently achieves the highest training and test accuracies. The number of generated counterfactual interpolations benefits the performance of CIA, i.e., CIA-30 achieves the best performance.

We qualitatively compare the explanation performance of our CGI with two baselines, IG and BlurIG [Xu *et al.*, 2020], in Figure 4. IG applies a black image as the baseline for gradients integration whereas BlurIG defines the path integral by successively blurring the original input.

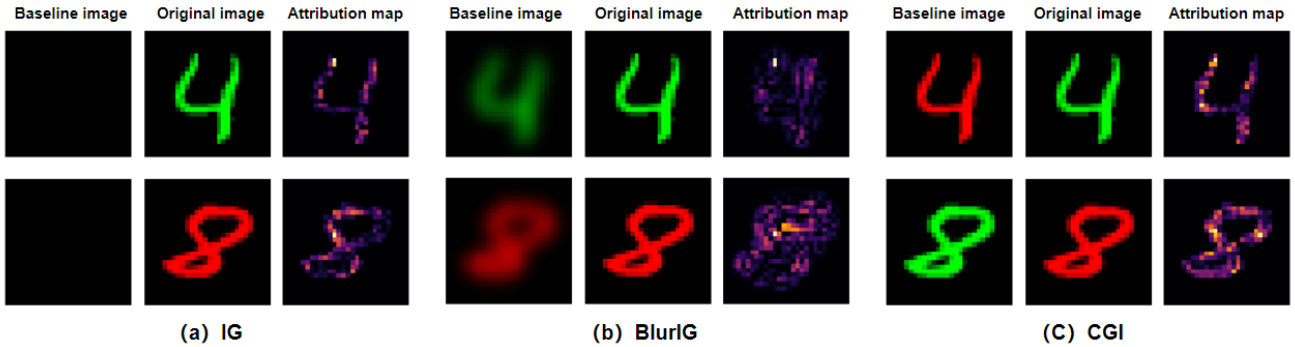


Figure 4: Examples of attribution heatmaps obtained by IG, BlurIG, and CGI. CGI demonstrates to generate higher quality attribution heatmaps with clearer digits shape, less noise, and focuses more densely on the digits (i.e., more bright masks).

Method	Training Acc	Test Acc
Vanilla	79.48	18.08
LAFTR	74.14	75.22
PriorTraining	74.62	75.46
CIA-10	79.64	78.16
CIA-20	79.95	78.23
CIA-30	**79.97**	**78.69**

Table 1: Fairness evaluation on BiasedMNIST. CIA-10, CIA-20 and CIA-30 denotes our CIA method with 10, 20 and 30 generated counterfactual interpolations for each sample, respectively. Best performing results are marked in bold.

5.5 CelebA Results

We compare our method with LfF [Nam *et al.*, 2020] and Group DRO [Sagawa *et al.*, 2019] with results shown in Table **??**. The vanilla model spuriously uses the sensitive attributes for target variable prediction, leading to low accuracies, especially on the bias-conflict sets. Notably, there are large accuracy gaps (i.e., unbiased dataset: 69.14 $\rightarrow$ 85.60 and 61.45 $\rightarrow$ 68.39; bias-conflict dataset: 50.26 $\rightarrow$ 84.17 and 31.56 $\rightarrow$ 50.16) between the vanilla model and our model demonstrating the effectiveness of CIA for bias mitigation. Our model outperforms Group DRO and LfF on most evaluation data sets. We note that CIA is a pre-processing approach that is both algorithm- and model-agnostic. As such, it is compatible with many other in-processing and post-processing fairness algorithms. We mainly demonstrate the advantage of only using CIA coupled with plain training in this work and leave the combination of CIA with other algorithms as our future works.

Figure 5 shows a qualitative example demonstrating CGI is capable of generating higher quality attribution map. Note that there is substantial noise in IG's attribution map due to the arbitrary choice of the baseline. Both CGI and BlurIG have captured the meaningful facial features (e.g., eyes and lips) related to the target attribute $HeavyMakeup$. While CGI's attribution map has higher density masks demonstrating a focus more densely on these facial features.

5.6 Quantitative Performance

We use insertion score and deletion score [Petsiuk *et al.*, 2018] to quantitatively evaluate the interpretation quality of different attribution methods. An attribution method should yield a high insertion score while keeping a low deletion score. We select 1000 samples from BiasedMNIST and 128 samples from CelebA (target variable: $Heavymakeup$) and report the quantitative results in Table **??**. Our CGI outperforms other attribution methods evident by higher insertion and lower deletion scores.

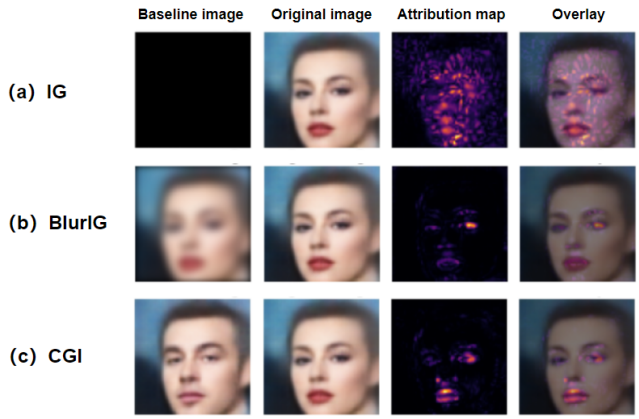


Figure 5: Examples of attribution maps obtained by IG, BlurIG, and CGI. The target attribute is $HeavyMakeup$.

6 Discussion

6.1 Fair Explanation

Although these explanation methods can generate attributions to interpret the model predictions, it is still unclear whether the attributions are generated from the discriminative features or the sensitive attribute since we do not have the ground truth attributions available for evaluation [Zhou *et al.*, 2021]. We illustrate an example from BiasedMNIST in Figure 6 to examine whether these methods are making fair explanations. Both IG and CGI can generate high-quality attribution maps with clear digit shape. While CGI's attribution map clearly shows that the attributions are captured from the digit shape rather than the color. This is because our CGI applies the counterfactual interpolations for gradients integration, which counteracts the effect of the sensitive attribute.

Target	Acc.Type	Vanilla	Group DRO	LfF	**CIA-10**	**CIA-20**	**CIA-30**
HairColor	Unbiased	69.14	85.43	84.24	84.95	85.12	**85.60**
	Bias-conflict	50.26	83.40	81.24	83.16	83.69	**84.17**
HeavyMakeup	Unbiased	61.45	64.88	66.20	67.86	68.04	**68.39**
	Bias-conflict	31.56	**50.24**	45.48	48.07	49.26	50.16

Table 2: Evaluation results on CelebA. *Gender* is the sensitive attribute. The results of Group DRO and LfF are cited from [Nam *et al.*, 2020]. We report the average accuracy over all (y, s) pairs.

Method	BiasedMNIST		CelebA	
	Deletion↓	Insertion↑	Deletion↓	Insertion↑
IG	0.2080	0.5591	0.1038	0.2514
BlurIG	0.2693	0.5014	**0.0638**	0.3016
CGI(ours)	**0.1649**	**0.6253**	0.0746	**0.3264**

Table 3: Quantitative results using deletion score and insertion score.

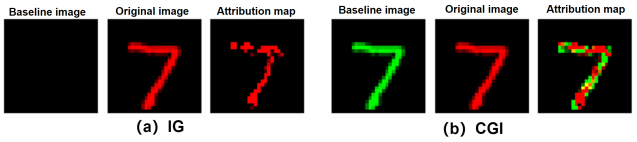


Figure 6: An showcase example demonstrates CGI is capable of generating fair explanation.

6.2 Investigating Saturation Effects

[Miglani *et al.*, 2020] split the area along the integral path as the saturated region where the model outputs changes minimally, and unsaturated region where the model outputs changes substantially. The gradients from the saturated region dominate the calculation of IG. Nevertheless, the integrated gradients of the saturated region seem to be noisier and substantially less faithful than the unsaturated region. Therefore, it is desirable to have a larger unsaturated region to convey feature importance via gradients integration.

Here, we conduct experiment to investigate the saturation regions of IG and CGI. Figure 7 clearly shows that our CGI integrates gradients in a larger unsaturated region than IG does, which contributes proportionately to the computed attribution leading to better explanations as shown in Figure 4. This further demonstrates the effectiveness of our CGI approach in the aspect of gradient saturation effect.

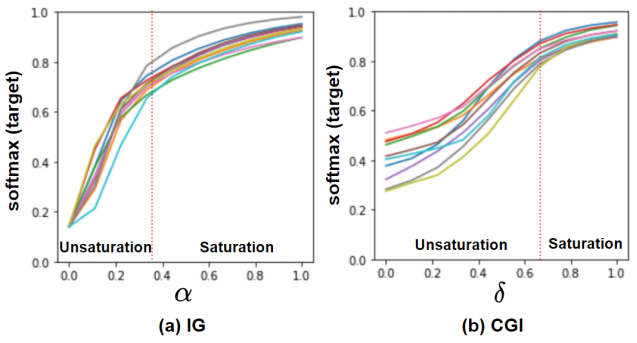


Figure 7: Comparing saturation regions of IG and CGI. We randomly select 10 samples from BiasedMNIST and report the model predictive probability (y-axis) along α and δ (x-axis) integral path.

7 Conclusion

We propose CIA as a pre-processing method to improve DNN's fairness via de-correlating the target variable with the sensitive attribute in training set. CIA generates counterfactual interpolations from a generative model. We further develop a gradient-based feature attribution method leveraging the counterfactual interpolations from CIA to generate high quality and fair explanations. Our experimental results demonstrate the outstanding performance of our approach via quantitative and qualitative evaluations using benchmark datasets. In the future, we will investigate the problem of fair explanation generation with implicit bias mitigation.

Acknowledgments

This work is supported by the National Science Foundation under grant CNS-2043611.

References

[Arjovsky *et al.*, 2019] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

[Buolamwini and Gebru, 2018] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR, 2018.

[Chuang and Mroueh, 2021] Ching-Yao Chuang and Youssef Mroueh. Fair mixup: Fairness via interpolation. *arXiv preprint arXiv:2103.06503*, 2021.

[Erion *et al.*, 2021] Gabriel Erion, Joseph D Janizek, Pascal Sturmfels, Scott M Lundberg, and Su-In Lee. Improving performance of deep learning models with axiomatic attribution priors and expected gradients. *Nature Machine Intelligence*, pages 1–12, 2021.

[Geirhos *et al.*, 2020] Robert Geirhos, Jörn-Henrik Jacobsen, et al. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[Hardt *et al.*, 2016] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29:3315–3323, 2016.

[Haug *et al.*, 2021] Johannes Haug, Stefan Zürn, Peter El-Jiz, and Gjergji Kasneci. On baselines for local feature attributions. *arXiv preprint arXiv:2101.00905*, 2021.

[Hong and Yang, 2021] Youngkyu Hong and Eunho Yang. Unbiased classification through bias-contrastive and bias-balanced learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[Izzo *et al.*, 2020] Cosimo Izzo, Aldo Lipani, Ramin Okhrati, and Francesca Medda. A baseline for shapley values in mlps: from missingness to neutrality. *arXiv preprint arXiv:2006.04896*, 2020.

[Kapishnikov *et al.*, 2021] Andrei Kapishnikov, Subhashini Venugopalan, Besim Avci, Ben Wedin, Michael Terry, and Tolga Bolukbasi. Guided integrated gradients: An adaptive path method for removing noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5050–5058, 2021.

[Kim *et al.*, 2019] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. Learning not to learn: Training deep neural networks with biased data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9012–9020, 2019.

[Kim *et al.*, 2021] Eungyeup Kim, Jihyeon Lee, and Jaegul Choo. Biaswap: Removing dataset bias with bias-tailored swapping augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14992–15001, 2021.

[Kusner *et al.*, 2017] Matt J Kusner, Joshua R Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. *arXiv preprint arXiv:1703.06856*, 2017.

[Li *et al.*, 2020] Xin Li, Xiangrui Li, Deng Pan, and Dongxiao Zhu. Improving adversarial robustness via probabilistically compact loss with logit constraints. *arXiv preprint arXiv:2012.07688*, 2020.

[Madras *et al.*, 2018] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning adversarially fair and transferable representations. In *International Conference on Machine Learning*, pages 3384–3393. PMLR, 2018.

[Madras *et al.*, 2019] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Fairness through causal awareness: Learning causal latent-variable models for biased data. In *FaCCT*, pages 349–358, 2019.

[Miglani *et al.*, 2020] Vivek Miglani, Narine Kokhlikyan, Bilal Alsallakh, Miguel Martin, and Orion Reblitz-Richardson. Investigating saturation effects in integrated gradients. *arXiv preprint arXiv:2010.12697*, 2020.

[Nam *et al.*, 2020] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: Training debiased classifier from biased classifier. *arXiv preprint arXiv:2007.02561*, 2020.

[Pan *et al.*, 2020] Deng Pan, Xiangrui Li, Xin Li, and Dongxiao Zhu. Explainable recommendation via interpretable feature mapping and evaluation of explainability. *arXiv preprint arXiv:2007.06133*, 2020.

[Pan *et al.*, 2021] Deng Pan, Xin Li, and Dongxiao Zhu. Explaining deep neural network models with adversarial gradient integration. In *Thirtieth International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

[Petsiuk *et al.*, 2018] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018.

[Pfohl *et al.*, 2019] Stephen R Pfohl, Tony Duan, Daisy Yi Ding, and Nigam H Shah. Counterfactual reasoning for fair clinical risk prediction. In *Machine Learning for Healthcare Conference*, 2019.

[Qiang *et al.*, 2020] Yao Qiang, Xin Li, and Dongxiao Zhu. Toward tag-free aspect based sentiment analysis: A multiple attention network approach. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[Sagawa *et al.*, 2019] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

[Sohn *et al.*, 2015] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491, 2015.

[Sturmfels *et al.*, 2020] Pascal Sturmfels, Scott Lundberg, and Su-In Lee. Visualizing the impact of feature attribution baselines. *Distill*, 5(1):e22, 2020.

[Sundararajan *et al.*, 2017] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.

[Tong and Kagal, 2020] Schrasing Tong and Lalana Kagal. Investigating bias in image classification using model explanations. *arXiv preprint arXiv:2012.05463*, 2020.

[Wachter *et al.*, 2017] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[Wang *et al.*, 2021] Tianhao Wang, Zana Buçinca, and Zilin Ma. Learning interpretable fair representations. Technical report, Technical report, Harvard University, 2021.

[Xu *et al.*, 2020] Shawn Xu, Subhashini Venugopalan, and Mukund Sundararajan. Attribution in scale and space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9680–9689, 2020.

[Zhang and Sang, 2020] Yi Zhang and Jitao Sang. Towards accuracy-fairness paradox: Adversarial example-based data augmentation for visual debiasing. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 4346–4354, 2020.

[Zhou *et al.*, 2021] Yilun Zhou, Serena Booth, Marco Tulio Ribeiro, and Julie Shah. Do feature attribution methods correctly attribute features? *arXiv preprint arXiv:2104.14403*, 2021.