IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 10, 2020, accepted November 24, 2020, date of publication November 30, 2020,
date of current version December 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3041416

Neural Image Compression and Explanation

XIANG LI AND SHIHAO JI, (Member, IEEE)
Department of Computer Science, Georgia State University, Atlanta, GA 30302, USA
Corresponding author: Shihao Ji (sji@gsu.edu)

ABSTRACT Explaining the prediction of deep neural networks (DNNs) and semantic image compression are two active research areas of deep learning with a numerous of applications in decision-critical systems, such as surveillance cameras, drones and self-driving cars, where interpretable decision is critical and storage/network bandwidth is limited. In this article, we propose a novel end-to-end Neural Image Compression and Explanation (NICE) framework that learns to (1) explain the predictions of convolutional neural networks (CNNs), and (2) subsequently compress the input images for efficient storage or transmission. Specifically, NICE generates a sparse mask over an input image by attaching a stochastic binary gate to each pixel of the image, whose parameters are learned through the interaction with the CNN classifier to be explained. The generated mask is able to capture the saliency of each pixel measured by its influence to the final prediction of CNN; it can also be used to produce a mixed-resolution image, where important pixels maintain their original high resolution and insignificant background pixels are subsampled to a low resolution. The produced images achieve a high compression rate (e.g., about 0.6× of original image file size), while retaining a similar classification accuracy. Extensive experiments across multiple image classification benchmarks demonstrate the superior performance of NICE compared to the state-of-the-art methods in terms of explanation quality and semantic image compression rate. Our code is available at: https://github.com/lxuniverse/NICE.

INDEX TERMS Explainable AI, sparsity learning, data compression, deep neural networks.

I. INTRODUCTION

Deep neural networks (DNNs) have become the de-facto performing technique in the field of computer vision [1], natural language processing [2], and speech recognition [3]. Given sufficient data and computation, they require only limited domain knowledge to reach state-of-the-art performance. However, the current DNNs are largely black-boxes with many layers of convolution, non-linearities, and gates, optimized solely for competitive performance, and our understanding of the reasoning of DNNs is rather limited. DNNs' predictions may be backed up by a claimed high accuracy on benchmarks. However, it is human's nature not to trust them unless human experts are able to verify, interpret, and understand the reasoning of the system. Therefore, the usage of DNNs in real world decision-critical applications, such as surveillance cameras, drones, autonomous driving, medicine and legal, still must overcome a trust barrier. To address this problem, researchers have developed many different approaches to explain the reasonings of DNNs [4]–[15]. Intuitively, interpretable explanations should be concise and coherent such that they are easier for human to comprehend. However, most existing approaches do not take these requirements into account as manifested by the opaqueness and redundancies in their explanations [4], [6], [9], [10].

On the other hand, over 70% of internet traffic today is the streaming of digital media, and this percentage keeps rising over years [16]. It has been challenging for classic compression algorithms, such as JPEG and PNG, to adapt to the growing demand. Recently, there is an increasing interest of using machine learning (ML) based approaches to improve the compression of images and videos [17]–[20]. Rather than using manually engineered basis functions for compression, these ML-based techniques learn semantic structures and basis functions directly from training images and achieve impressive performance compared to the classic compression algorithms.

Usually, neural explanation and semantic image compression are addressed independently by two different groups of researchers. In light of the similarity between sparse explanation to image classification and sparse representation for image compression, in this article we propose a deep learning based framework that integrates neural explanation and semantic image compression into an end-to-end

The associate editor coordinating the review of this manuscript and approving it for publication was Mostafa Rahimi Azghadi.

training pipeline. With this framework, we can train a sparse mask generator to generate a concise and coherent mask to explain the prediction of CNN; subsequently, this sparse mask can be used to generate a mixed-resolution image with a very high compression rate, superior to the existing semantic compression algorithms. This Neural Image Compression and Explanation (NICE) framework is critical to many real world decision-critical systems, such as surveillance cameras, drones and self-driving cars, that heavily rely on the deep learning techniques today. For these applications, the outputs of NICE: prediction, sparse mask / explanation, and the compressed mixed-resolution image can be stored or transmitted efficiently for decision making, decision interpretation and system diagnosis.

The main contributions of the paper are:

- We propose a deep learning based framework that unifies neural explanation and semantic image compression into an end-to-end trainable pipeline, which produces prediction, sparse explanation and compressed images at the same time;
- The proposed L_0-regularized sparse mask generator is trained in a weakly supervised manner without resorting to expensive dense pixel-wise annotations, and outperforms many existing explanation algorithms that heavily rely on backpropagation;
- The proposed mixed-resolution image compression achieves a higher compression rate compared to the existing semantic compression algorithms, while retaining a similar classification accuracy with the original images.
- The proposed method is very efficient compared to the backpropagation-based alternatives, such as Saliency Map [4] and CAM [21], as our method only requires forward propagation of the generator network. Experiments show that NICE is about 23× faster than Saliency Map, 16.5× faster than CAM and 2.8× faster than RTIS [9]. This makes our method widely deployable in real-time applications.

II. RELATED WORK

Our work is related to two active research areas of deep learning: neural explanation and semantic image compression. We therefore review them next.

A. NEURAL EXPLANATION

In order to interpret DNN's prediction and gain insights of their operations, a variety of neural explanation methods have been proposed in recent years [4]–[15], [21]–[24]. These methods can be categorized based on whether it is designed to explain the entire model behavior (global interpretability) or a single prediction (local interpretability) [25]. The goal of global interpretability is to identify predictor variables that best explain the overall performance of a trained model. This class of methods are crucial to inform population level decision for rule extraction or knowledge discovery [14], [15]. Local interpretability aims to produce interpretable explanations for each individual prediction and the interpretability occurs locally. Local interpretability is by far the most explored area of explainable AI [4], [8], [12], [21], [23], [24]. The primary idea is to measure a change of the final prediction with respect to changes of input or getting feature attribution for the final prediction. Different local explanation methods implement this idea in different ways. For example, occlusion-based explanation methods remove or alter a fraction of input data and evaluate its impact to the final prediction [5], [9]–[11]. Gradient-based methods compute the gradient of an output with respect to an input sample by using backpropagation to locate salient features that are responsible to the prediction [6], [7], [13], [22]. Other local interpretability methods explain data instances by approximating the decision boundary of a DNN with an inherently interpretable model around the predictions. For example, LIME [8] and SHAP [12] sample perturbed instances around a single data sample and fit a linear model to perform local explanations. RTIS [9] extracts features from a DNN classifier and feeds extracted features and target label to an U-Net like generator to generate saliency maps for local explanations. L2X [23] learns a stochastic map based on mutual information that selects instance-wise informative features. Built on top of L2X, VIBI [24] selects instance-wise key features that are maximally compressed about an input and informative about a decision based on an information-bottleneck principle.

NICE falls in the category of local interpretability and aims to produce concise and coherent local explanations similar to Saliency Map [4], RTIS [9] and VIBI [24]. But our method achieves briefness and comprehensiveness explicitly through an L_0-norm regularization and a smoothness constraint, optimized via stochastic binary optimization.

The sparse mask generator of NICE is also related to a large body of research on semantic segmentation [26]–[32]. In particular, our sparse mask generator is trained to maximize the final classification accuracy of the mixed-resolution images without resorting to expensive dense pixel-wise annotations. Therefore, it can be considered as a weakly supervised *binary* segmenation algorithm that detects salient regions of an image. This is different to the existing semantic segmentation algorithms that employ different levels of supervision, such as full pixel-wise annotation [26], [27], image-level labels [29], bounding boxes [33], scribbles [34], points [31], or adversarial loss [32]. Since the main goal of NICE is to provide a competitive or improved neural explanation, in our experiments we mainly compare NICE with deep explanation methods instead of segmentation algorithms.

B. SEMANTIC IMAGE COMPRESSION

Classic image compression algorithms, such as JPEG [35] and PNG [36], have hard-coded procedures / components to compress images. For example, the JPEG compression first employs a discrete cosine transform (DCT) over each 8×8 image block, followed by quantization to represent the frequency coefficients as a sequence of binaries. The DCT can be seen as a generic feature extractor with

a fixed set of basis functions that are irrespective of the distribution of the input images. Compared to standard image compression algorithms, the ML-based approaches [17], [19], [20], [37]–[41] can automatically discover semantic structures and learn basis functions from training images to achieve even higher compression rate. All of these ML-based approaches follow a similar structure of autoencoder, where an encoder is used to extract feature representation from images and a decoder is responsible to reconstruct images from the quantized representations. The main differences among these ML-based approaches are the architectures of encoder and decoder. While the majority of these algorithms [17], [19], [20], [37], [40], [41] employ CNNs as the encoder and decoder, some others explore recurrent networks such as LSTM and GRU [38], [39].

To the best of our knowledge, all of these methods are not sufficiently content-aware, except the work [18] from Prakash *et al.* which is probably the most relevant work to ours. While Prakash *et al.* adopt CAM [21] as the semantic region detector, we develop a principled L_0-regularized sparse mask generator to detect the semantic regions and further compress images with mixed resolutions. We will compare NICE with [18] when we present results of semantic image compression.

III. THE NICE FRAMEWORK

Given a training set $D = \{(\boldsymbol{x}_i, y_i), i = 1, 2, \cdots, N\}$, where $\boldsymbol{x}_i$ denotes the i-th input image and y_i denotes the corresponding target, a neural network is a function $h(\boldsymbol{x}; \boldsymbol{\theta})$ parameterized by $\boldsymbol{\theta}$ that fits to the training data D with the goal of achieving good generalization to unseen test data. To optimize $\boldsymbol{\theta}$, typically the following empirical risk minimization (ERM) is adopted:

$$\mathcal{R}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(h\left(\boldsymbol{x}_i; \boldsymbol{\theta}\right), y_i\right), \tag{1}$$

where $\mathcal{L}(\cdot)$ denotes the loss over training data D, such as the cross-entropy loss for classification or the mean squared error (MSE) for regression. The goal of this article is to develop an approach that can explain the prediction of a neural network $h(\boldsymbol{x}; \theta)$ in response to an input image $\boldsymbol{x}$; meanwhile, to reduce storage or network transmission cost of the image, we'd like to compress the image $\boldsymbol{x}$ based on the above derived explanation such that the compressed image $\tilde{\boldsymbol{x}}$ has the minimal file size while retaining a similar classification accuracy as the original image $\boldsymbol{x}$.

To meet these interdependent goals, we develop a Neural Image Compression and Explanation (NICE) framework that integrates explanation and compression into an end-to-end trainable pipeline as illustrated in Fig. 1. In this framework, given an input image, a mask generator under the L_0-norm and smoothness constraints generates a sparse mask that indicates salient regions of the image. The generated mask is then used to transform the original input image to a mixed-resolution image that has a high resolution in the salient regions and a low resolution in the background.

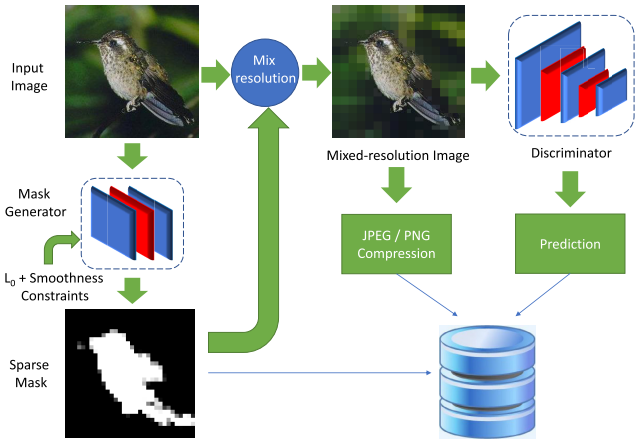


FIGURE 1. Overall architecture of NICE.

To evaluate the quality of sparse mask generator and the compressed image, at the end of the pipeline a discriminator network (e.g., CNN) classifies the generated image for prediction. Finally, the prediction, sparse mask and compressed image can be stored or transmitted efficiently for decision making, interpretation and system diagnosis. The whole pipeline is fully differentiable and can be trained end-to-end by backpropagation. We will introduce each of these components next.

A. SPARSE NEURAL EXPLANATION

To correctly classify an image, a state-of-the-art CNN classifier does not need to analyze all the pixels in an image. Partially, this is because not all the pixels in an image are equally important for image recognition. For example, although the background pixels may provide some useful clues to recognize an object, it is the pixels on the object that play a decisive role for recognition. Based on this understanding, we'd like to learn a set of random variables (one for each pixel of an image) such that the variables on object pixels receive high values while the variables on background pixels receive low values. In other words, we want to learn a binary segmentation model that can partition pixels into object pixels and background pixels. To make our segmentation discriminative, we require the output of our model to be sparse/concise such that only the most important or influential pixels receive high values, and the remaining pixels receive low values. Furthermore, we expect the segmentation to be smooth/coherent within a small continuous region since most of natural objects usually have smooth appearances. We therefore request our neural explanation model to produce explanations that are concise and coherent. We will materialize these two requirements mathematically.

We model our neural explanation by attaching a binary random variable $z \in \{0, 1\}$ to each pixel of an image:

$$\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i \odot \boldsymbol{z}_i, \quad \boldsymbol{z}_i \in \{0, 1\}^P, \tag{2}$$

where $\boldsymbol{z}_i$ denotes a binary mask for image $\boldsymbol{x}_i$, and $\odot$ is an element-wise product. Furthermore, we define z_i^j the binary variable for pixel j of image $\boldsymbol{x}_i$. We assume both image $\boldsymbol{x}_i$ and its mask $\boldsymbol{z}_i$ have the same spatial dimension of $m \times n$ or

P pixels. After training, we wish z_i^j takes value 1 if pixel j is on object and 0 otherwise.

We regard $\boldsymbol{z}_i$ as our explanation to the prediction of $h(\boldsymbol{x}_i; \boldsymbol{\theta})$ and learn $\boldsymbol{z}_i$ by minimizing the following L_0-norm regularized loss function:

$$\begin{aligned}\mathcal{R}(\boldsymbol{\theta}, z) &= \frac{1}{N}\sum_{i=1}^{N}\left(\mathcal{L}\left(h\left(\boldsymbol{x}_i \odot \boldsymbol{z}_i; \boldsymbol{\theta}\right), y_i\right) + \lambda||\boldsymbol{z}_i||_0\right) \\ &= \frac{1}{N}\sum_{i=1}^{N}\left(\mathcal{L}\left(h\left(\boldsymbol{x}_i \odot \boldsymbol{z}_i; \boldsymbol{\theta}\right), y_i\right) + \lambda\sum_{j=1}^{P} 1_{\left[z_i^j \neq 0\right]}\right), \quad (3)\end{aligned}$$

where $\mathbf{1}_{[c]}$ is an indicator function that is 1 if the condition c is satisfied, and 0 otherwise. Here, we insert (2) into (1) and add an L_0-norm on the elements of $\boldsymbol{z_i}$, which *explicitly* measures number of non-zeros in $\boldsymbol{z}_i$ or the sparsity of $\boldsymbol{z}_i$. By doing so, we'd like the masked image achieves the similar classification accuracy as the original image, while using as fewer pixels as possible. In other words, the sparse mask $\boldsymbol{z}_i$ can produce a concise explanation to the prediction of the classifier (i.e., the first requirement). To optimize (3), however, we note that both the first term and the second term of (3) are not differentiable w.r.t. $\boldsymbol{z}$. Therefore, further approximations need to be considered.

We can approximate this optimization problem via an inequality from stochastic variational optimization [42]. Specifically, given any function $\mathcal{F}(\boldsymbol{z})$ and any distribution $q(\boldsymbol{z})$, the following inequality holds

$$\min_{\boldsymbol{z}} \mathcal{F}(\boldsymbol{z}) \leq \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z})}[\mathcal{F}(\boldsymbol{z})], \quad (4)$$

i.e., the minimum of a function is upper bounded by the expectation of the function. With this result, we can derive an upper bound of (3) as follows.

Since $z_i^j, \forall j \in \{1, \cdots, P\}$ is a binary random variable, we assume z_i^j is subject to a Bernoulli distribution with parameter $\pi_i^j \in [0, 1]$, i.e. $z_i^j \sim \text{Ber}(z; \pi_i^j)$. Thus, we can upper bound $\min_{\boldsymbol{z}} \mathcal{R}(\boldsymbol{\theta}, \boldsymbol{z})$ by the expectation

$$\begin{aligned}\tilde{\mathcal{R}}(\boldsymbol{\theta}, \boldsymbol{\pi}) = \frac{1}{N}\sum_{i=1}^{N}\Big(&\mathbb{E}_{q(\boldsymbol{z}_i|\boldsymbol{\pi}_i)}\Big[\mathcal{L}\left(h\left(\boldsymbol{x}_i \odot \boldsymbol{z}_i; \boldsymbol{\theta}\right), y_i\right)\Big] \\ &+ \lambda\sum_{j=1}^{P}\pi_i^j\Big). \quad (5)\end{aligned}$$

Now the second term of (5) is differentiable w.r.t. the new model parameters $\boldsymbol{\pi}$. However, the first term is still problematic since the expectation over a large number of binary random variables $\boldsymbol{z}_i \in \{0, 1\}^P$ is intractable, so is its gradient.

1) THE HARD CONCRETE GRADIENT ESTIMATOR

Fortunately, this kind of binary latent variable models has been investigated extensively in the literature. There exist a numerous of gradient estimators to this problem, including REINFORCE [43], Gumble-Softmax [44], [45], REBAR [46], RELAX [47] and the hard concrete estimator [48], among which the hard concrete estimator is the one that is easy to implement and demonstrates superior performance in our experiments. We therefore resort to this gradient estimator to optimize (5). Specifically, the hard concrete gradient estimator employs a reparameterization trick to approximate the original optimization problem of (5) by a close surrogate loss function

$$\begin{aligned}&\hat{\mathcal{R}}(\boldsymbol{\theta}, \log\boldsymbol{\alpha}) \\ &= \frac{1}{N}\sum_{i=1}^{N}\Bigg(\mathbb{E}_{\boldsymbol{u}_i\sim\mathcal{U}(0,1)}\Big[\mathcal{L}\left(h\left(\boldsymbol{x}_i \odot g(f(\log\boldsymbol{\alpha}_i, \boldsymbol{u}_i)); \boldsymbol{\theta}\right), y_i\right)\Big] \\ &\quad + \lambda\sum_{j=1}^{P}\sigma\left(\log\alpha_i^j - \beta\log\frac{-\gamma}{\zeta}\right)\Bigg) \\ &= \mathcal{L}_D(\boldsymbol{\theta}, \log\boldsymbol{\alpha}) + \lambda\mathcal{L}_C(\log\boldsymbol{\alpha}), \quad (6)\end{aligned}$$

with

$$\begin{aligned}f(\log\boldsymbol{\alpha}_i, \boldsymbol{u}_i) = \sigma\big((&\log\boldsymbol{u}_i - \log(1-\boldsymbol{u}_i) \\ &+ \log\boldsymbol{\alpha}_i)/\beta\big)(\zeta - \gamma) + \gamma, \quad (7)\end{aligned}$$

and

$$g(\cdot) = \min(1, \max(0, \cdot)), \quad (8)$$

where $\sigma(t) = 1/(1 + \exp(-t))$ is the sigmoid function, $\mathcal{L}_D$ measures how well the classifier fits to training data D, $\mathcal{L}_C$ measures the expected number of non-zeros in $\boldsymbol{z}$, and $\beta = 2/3$, $\gamma = -0.1$ and $\zeta = 1.1$ are the typical parameters of the hard concrete distribution. Function $g(\cdot)$ is a hard-sigmoid function that bounds the stretched concrete distribution between 0 and 1. For more details on the hard concrete gradient estimator, we refer the readers to [48]. With this reparameterization, the surrogate loss function (6) is differentiable w.r.t. its parameters.

2) SMOOTHNESS REGULARIZATION

The L_0-regularized objective function developed above enforces the sparsity/conciseness of an explanation. To improve the coherence of an explanation, we introduce an additional smoothness constraint on the mask:

$$\begin{aligned}\mathcal{L}_S(\log\boldsymbol{\alpha}) &= \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{q(\boldsymbol{z}_i|\log\boldsymbol{\alpha}_i)}\Bigg[\sum_{m,n=1}^{w,h}\Big(\left|z_i^{m,n} - z_i^{m-1,n}\right| \\ &\quad + s\left|z_i^{m,n} - z_i^{m,n-1}\right| + \left|z_i^{m,n} - z_i^{m-1,n-1}\right| \\ &\quad + \left|z_i^{m,n} - z_i^{m-1,n+1}\right|\Big)\Bigg] \\ &\approx \frac{1}{N}\sum_{i=1}^{N}\sum_{m,n=1}^{w,h}\Big(\left|y_i^{m,n} - y_i^{m-1,n}\right| + \left|y_i^{m,n} - y_i^{m,n-1}\right| \\ &\quad + \left|y_i^{m,n} - y_i^{m-1,n-1}\right| + \left|y_i^{m,n} - y_i^{m-1,n+1}\right|\Big), \\ &\quad (9)\end{aligned}$$

where $y_i^{m,n}$ is the expectation of random variable $z_i^{m,n}$ under the hard concrete distribution $q(\boldsymbol{z}_i|\log\boldsymbol{\alpha}_i)$, which can be calculated as:

$$y = \mathbb{E}_{q(z|\log\alpha)}[z] = \sigma\left(\log\alpha - \beta\log\frac{-\gamma}{\zeta}\right). \quad (10)$$

Note that this smoothness constraint penalizes the discrepancy of z among its four neighborhoods, and thus a coherence explanation is preferred (i.e., the second requirement). To avoid notational clutter, in (9) some of the boundary conditions are not rigorously checked, but we hope they will be apparent given the context. With this additional regularization, our final objective is then a composition of three terms

$$\mathcal{L}(\boldsymbol{\theta}_d, \log \boldsymbol{\alpha}) = \mathcal{L}_D + \lambda_1 \mathcal{L}_C + \lambda_2 \mathcal{L}_S, \tag{11}$$

where λ_1 and λ_2 are the regularization hyperparameters that balance the data loss $\mathcal{L}_D$, the capacity loss $\mathcal{L}_C$ and the smoothness loss $\mathcal{L}_S$. It is worthy noting that from now on we denote the parameters of classifier (discriminator) $\boldsymbol{\theta}_d$ to distinguish it from the parameters of generator $\boldsymbol{\theta}_g$ that will be introduced next.

After training, we get $\log \boldsymbol{\alpha}$ for each input image $\boldsymbol{x}$. At testing time, we employ the following estimator to generate a sparse mask:

$$\hat{\boldsymbol{z}} = \min(\mathbf{1}, \max(\mathbf{0}, \sigma\left((\log \boldsymbol{\alpha})/\beta\right)(\zeta - \gamma) + \gamma)), \tag{12}$$

which is the sample mean of $\boldsymbol{z}$ under the hard concrete distribution $q(\boldsymbol{z}|\log \boldsymbol{\alpha})$.

B. SEMANTIC IMAGE COMPRESSION

Upon receiving the sparse mask $\hat{\boldsymbol{z}}$ from above, we can use it to generate a mixed-resolution image for semantic image compression, as shown in Fig. 1. Suppose that we have an input image $\boldsymbol{x}$ and a sparse mask $\hat{\boldsymbol{z}} \in [0, 1]^P$, a mixed-resolution image can be generated by

$$\tilde{\boldsymbol{x}} = M(\boldsymbol{x}, \hat{\boldsymbol{z}}) = \boldsymbol{x} \odot \hat{\boldsymbol{z}} + \boldsymbol{x}_b \odot (1 - \hat{\boldsymbol{z}}), \tag{13}$$

where $\boldsymbol{x}_b$ is a low resolution image that can be generated by subsampling original image $\boldsymbol{x}$ with a block size of $b \times b$, which can be efficiently implemented by average pooling with a $b \times b$ filter and a stride of b. Here b is a tunable hyperparameter that trades off between the image compression rate and the classification accuracy of the classifier. In other words, the larger b is, the lower resolution images will be generated and thus a lower classification accuracy, and vice-versa. As we can see, when $b = 1$ the mixed-resolution image $\tilde{\boldsymbol{x}}$ is equal to the original image $\boldsymbol{x}$; when we use the image size as b, the mixed-resolution image $\tilde{\boldsymbol{x}}$ becomes a masked image with a constant value as background. When b is a value between these two extremes, we can generate mixed-resolution images of different levels of quality.

C. SPARSE MASK GENERATOR

The learning of sparse mask $\boldsymbol{z}$ discussed above is transductive, by which we can learn a mask for each image in training set D. However, this approach cannot generate masks for new images that are not in the training set D. A more desirable approach is inductive, which can be implemented through a generator $G(\boldsymbol{x}; \boldsymbol{\theta}_g)$ such that it can produce a sparse mask given any image $\boldsymbol{x}$ as input. We model this generator as a neural network parameterized by $\boldsymbol{\theta}_g$.

To integrate this generator into an end-to-end training pipeline, we model this generator to output $\log \boldsymbol{\alpha}$ given an input image $\boldsymbol{x}$; we can then sample a sparse mask $\boldsymbol{z}$ from the hard concrete distribution $q(\boldsymbol{z}|\log \boldsymbol{\alpha})$, i.e., $\boldsymbol{x} \xrightarrow{G(\cdot;\boldsymbol{\theta}_g)} \log \boldsymbol{\alpha} \xrightarrow{\text{sample}} \boldsymbol{z}$. With this reparameterization, the overall loss function (11) becomes $\mathcal{L}(\boldsymbol{\theta}_d, \boldsymbol{\theta}_g)$, which can be minimized by optimizing the generator network $\boldsymbol{\theta}_g$ and the discriminator network $\boldsymbol{\theta}_d$ jointly with backpropagation. In the experiments, we employ a CNN as our sparse mask generator as CNN is the de-facto technique today for image related analysis.

TABLE 1. Network architectures of the generators and discriminators used in the experiments. Layer abbreviations used in the table: [C: Convolution; R: Relu; M: MaxPooling; Up: UpSample].

Dataset	Generator	Discriminator
MNIST	C(1,1,3,1,1)	LeNet5-Caffe
CIFAR10	C(1,1,5,1,0) + M(2) + Up(2)	VGG11 + FC(512, 10)
Caltech256	C(3,1,3,1,1) + R + M(2) C(1,1,3,1,1) + R + M(2) C(1,1,3,1,1) + M(2) + Up(8)	ResNet18 FC(512, 256)

IV. EXPERIMENTS

To evaluate the performance of NICE, we conduct extensive experiments on three image classification benchmarks: MNIST [49], CIFAR10 [50] and Caltech256 [51].[1] Since NICE is a neural explanation and semantic compression algorithm, we compare NICE with the state-of-the-art algorithms in neural explanation and semantic compression. For neural explanation, we compare NICE with Saliency Map [4], RTIS [9] and CAM [21] via visualization and the post-hoc classification. For semantic image compression, we compare NICE with the CAM-based method proposed in [18], a state-of-the-art semantic compression algorithm that is the most relevant to ours.

A. IMPLEMENTATION DETAILS

1) IMAGE CLASSIFICATION BENCHMARKS

MNIST [49] is a gray-level image dataset containing 60,000 training images and 10,000 test images of the size 28×28 for handwritten digits classification. CIFAR10 [50] contains 10 classes of RGB images of the size 32×32, in which 50,000 images are for training and 10,000 images are for test. Caltech256 [51] is a high-resolution RGB image dataset containing 22,100 images from 256 classes of man-made and natural objects, such as plants, animals and buildings, etc. Since MNIST and CIFAR10 are low-resolution images, we use them mainly to demonstrate NICE's performance on neural explanation. For the high-resolution images of Caltech256, we demonstrate NICE's performance on neural explanation and semantic image compression.

2) NETWORK ARCHITECTURES AND TRAINING DETAILS

The network architectures of the sparse mask generators and CNN classifiers (discriminators) used in the MNIST, CIFAR10 and Caltech256 experiments are provided in Table 1.

We pretrain three CNN classifiers (discriminators) on the three image classification benchmarks: MNIST,

[1] http://www.vision.caltech.edu/Image_Datasets/Caltech256/

CIFAR10 and Caltech256 and achieve the classification accuracies of 99%, 90.8% and 78.3%, respectively. These classifiers are the target CNNs we aim to explain. The architectures of the generators are tuned by us through extensive architecture search. The hyperparameters λ_1 and λ_2 in the overall loss (11) are tuned on validation set to balance the classification accuracy and sparsity/smoothness of the masks.

In the MNIST experiments, different λ_1s are used to generate sparse masks with different percentages of non-zeros (sparse explanations). λ_2 is set to 0 for all the MNIST experiments as the algorithm can generate coherent explanations without the smoothness constraint. The block size of the low resolution image $\boldsymbol{x}_b$ is set to 28, which means a constant background is used to generate the mixed-resolution images. We use the Adam optimizer [52] with a learning rate of 0.001 and a decay rate of 0.1 at every 5 epochs.

In the CIFAR10 experiments, the block size of the low resolution image $\boldsymbol{x}_b$ is set to 32, thus a constant background image is used to generate the mixed-resolution images. We set $\lambda_1 = 3$ and $\lambda_2 = 0.01$ and train the pipeline by using the Adam optimizer with a learning rate of 0.001 and a decay rate 0.1 at every 5 epochs.

In the Caltech256 experiments, we split the dataset into a training set of 16,980 images and a test set of 5,120 images,[2] where 5,120 images in training set is first used as validation set for architecture search and hyperparameter tuning and later the full 16,980 training images are used to train the final pipeline. The images are resized to 256×256 as inputs. We set $b = 256$ to generate the lowest resolution images $\boldsymbol{x}_b$, and set $\lambda_1 = 5$ and $\lambda_2 = 0.01$ and train the pipeline by using the SGD optimizer with a learning rate of 0.001 and a cosine decay function.

On different datasets, we experiment with different optimizers. The best performing one is selected based on its performance on validation set. It turns out that SGD works better on Caltech256, while Adam works better on MNIST and CIFAR10.

B. EXPLAINING CNN's PREDICTIONS

We first demonstrate NICE on explaining the predictions of the target CNNs we pretrained above. To do so, we incorporate the target CNN as discriminator into the pipeline (Fig. 1), and freeze its parameters $\boldsymbol{\theta}_d$ and only update the parameters of generator $\boldsymbol{\theta}_g$ by optimizing the overall loss (11). The sparse mask $\boldsymbol{z}$ generated by the generator serves as the explanation to CNN's prediction since the mask indicates the salient region that has strong influence to the final prediction.

1) MNIST

We train the NICE pipeline on the MINST dataset to explain the prediction of the target LeNet5 classifier we pretrained above. Fig. 2 illustrates example sparse explanations generated by NICE with different λ_1s (when $\lambda_2 = 0$). As we can see, when λ_1 increases, the amount of non-zeros in the mask $\boldsymbol{z}$ decreases and NICE can produce sparser explanations to the final predictions. When $\lambda_1 = 1$, the explanations are almost identical to the input images, and when $\lambda_1 = 30$, the masks identify sparser but more influential regions for the final predictions. As a comparison, we also include the explanation results produced by Saliency Map [4] and RTIS [9].[3] While NICE highlights coherent regions over digits as explanations, Saliency Map, a backpropagation-based approach, identifies discontinued regions as explanations, which are quite blurry and difficult to understand. RTIS can yield coherent regions as explanations but the regions identified are overly smooth. Apparently, the explanations produced by NICE are more concise, coherent and match well with how humans explain their own predictions.

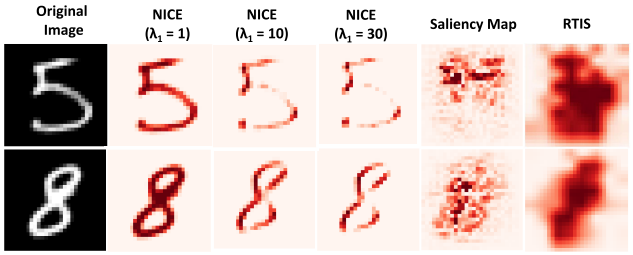


FIGURE 2. The sparse masks generated by NICE, Saliency Map [4] and RTIS [9] on the MNIST dataset. The dark red color represents high values (close to 1), indicating strong influence to the final decisions. By adjusting λ_1 of NICE, we can control the sparsity of the explanations.

2) CIFAR10

We also train the NICE pipeline on the CIFAR10 dataset to explain the target VGG11 classifier we pretrained above. Fig. 3 compares the explanations produced by Saliency Map [4], RTIS [9] and NICE on some CIFAR10 images [3]. The RTIS results are directly cited from the RTIS paper, and we apply Saliency Map and NICE on the same set of CIFAR10 images selected by the RTIS paper. Due to the low resolution of the images, it's very challenging to generate reliable explanations. As we can see, the explanations generated by NICE are more concise and the boundaries of salient regions are much sharper than those of Saliency Map and RTIS. The superior performance of NICE is most likely due to the L_0-norm regularization that *explicitly* promotes the sparsity of an explanation.

3) Caltech256

Similarly, we train the NICE pipeline on the Caltech256 dataset to explain the predictions of the ResNet18 classifier we pretrained above. Fig. 4 demonstrates the sparse masks produced by NICE for different images in Caltech256. As we can see, the generated explanations are very concise and coherent, i.e., the sparse masks are mainly concentrated on the object regions, which align very well with our reasoning on these images. Additionally, the generated sparse masks also provide intuitive explanations when the classifier makes mistakes. For example, as shown in Fig. 4(e), the classifier

[2] 20 images per class are included in the test set.

[3] CAM [21] does not perform well on small images, and we observe no published work provides CAM's results on MNIST and CIFAR10. We therefore ignore its results here as well.

FIGURE 3. Comparison of explanations generated by Saliency Map [4], RTIS [9] and NICE on some CIFAR10 images. The RTIS results are from the RTIS paper. Compared to Saliency Map and RTIS, the explanations generated by NICE are more concise and the boundaries of salient regions are much sharper.

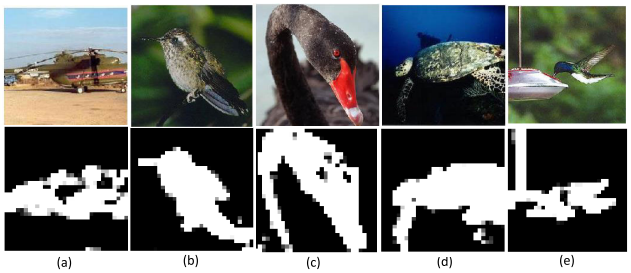


FIGURE 4. The sparse masks generated by NICE on Caltech256 images. The predictions are correct to (a, b, c, d) and incorrect to (e). Even though the prediction is incorrect, the sparse mask (e) provides an intuitive explanation why the discriminator predicts an image of "humming bird" as "bread maker".

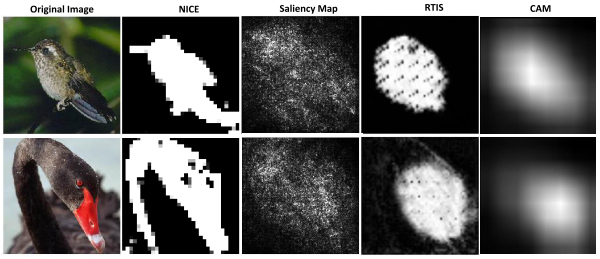


FIGURE 5. The sparse masks generated by NICE, Saliency Map [4], RTIS [9] and CAM [21] on the Caltech256 dataset. NICE highlights the whole body of object as the explanation instead of edges or scattered pixels as identified by Saliency Map, or overly-smooth regions as identified by RTIS and CAM.

incorrectly predicts an image of ''humming bird'' as ''bread maker''. The corresponding sparse explanation highlights the influential regions contributing the most to the classifier's prediction. Clearly, the classifier utilizes both the regions of the humming bird and the bird-feeder for the prediction, and the combination of the two regions confuses the classifier and leads to the incorrect classification. Such an explanation is very useful for system diagnosis: it uncovers the vulnerabilities and flaws of the classifier, and can help to improve the performance of the system.

Fig. 5 illustrates the comparison of NICE with Saliency Map [4], RTIS [9] and CAM [21] on the Caltech256 images. As we can see, our algorithm highlights the whole body of object as the explanation while Saliency Map typically identify edges or scattered pixels as the explanation. RTIS and CAM can identify coherent salient regions of an image, however, those regions are overly smooth and cover large background regions. Moreover, the saliency maps generated by RTIS usually have some black grids in the highlighted parts, which are caused by the upsampling step in the mask generator. Apparently, our explanations are more concise and coherent than those of the competing methods, and can preserve semantic contents of the images with a high accuracy. The superior performance of NICE on identifying semantic regions plays a critical role in semantic image compression as we will demonstrate later.

To evaluate NICE's performance of identifying important pixels from an image, Fig. 6 demonstrates the evolution

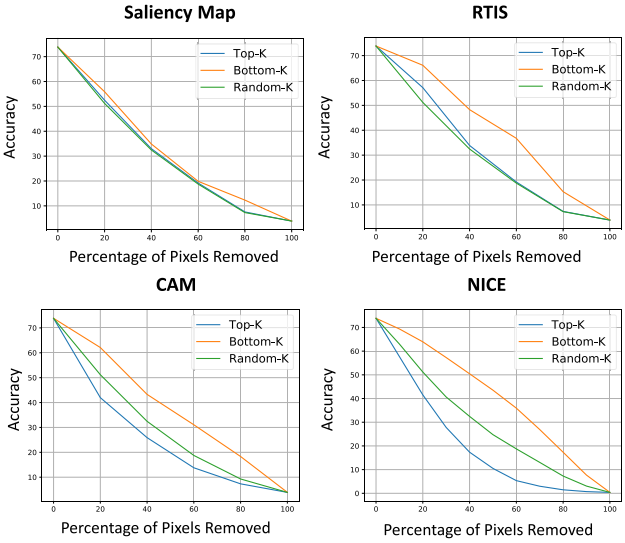


FIGURE 6. The evolution of classification accuracies on the Caltech256 test dataset when different percentages of pixels are filled with random values.

of classification accuracies on the Caltech256 test dataset when different percentages of pixels are filled with random values sampled uniformly from [0, 255] (a.k.a. post-hoc classification evaluation). We compare three different strategies of selecting pixels for random value imputation: (1) Top-K% pixels sorted descending by $\log \alpha_j, \forall j \in \{1, 2, \cdots, P\}$, (2) Bottom-K% pixels sorted descending by $\log \alpha_j$, and (3) uniformly random K% pixels. Similarly, the same post-hoc

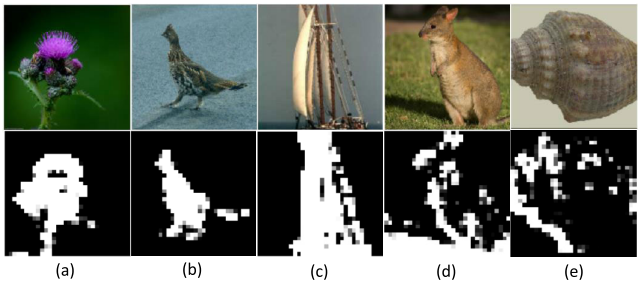


FIGURE 7. Sample ImageNet images and their sparse masks generated by the generator trained on Caltech256. While the ground truth labels of (a, b, c) are included in Caltech256, the ground truth labels of (d, e) are not in Caltech256. NICE is able to generate accurate sparse masks for images in (a, b, c). But when the classes are not in Caltach256 the masks are not very accurate as shown in (d, e).

classification evaluation is performed with Saliency Map, RTIS and CAM. As we can see from Fig. 6, NICE identifies important pixels from images as randomizing their Top-K% values incurs a dramatic accuracy loss compared to random pixel selection or Bottom-K% pixel selection. The results of Saliency Map, RTIS and CAM show insignificant accuracy loss when randomizing their Top-K% pixels, demonstrating the superior performance of NICE on identifying salient regions.

4) TRANSFERABILITY OF SPARSE MASK GENERATOR

The experiments above demonstrate the superiority of the sparse mask generator in generating concise and coherent explanations to target classifier's predictions. This has been verified in the case when the generator is applied to the test images from the same dataset. Since our generator is inductive, it would be interesting to test if a generator trained from one dataset could be applied to images from other datasets, which have similar statistics but yet have some mismatch, i.e., the transferability of generator.

To measure the transferability of generator, we apply the generator trained on Caltech256 to the ImageNet images [53]. Although both Caltech256 and ImageNet contain high-resolution RGB images, ImageNet contains 1000 classes which is 4 times of Caltech256's and the ImageNet images tend to be more complex than those in Caltech256. To feed the ImageNet images to the generator trained on Caltech256, we resize the ImageNet images to 256×256. Fig. 7 illustrates some sample ImageNet images and their sparse masks. Images (a, b, c) are from the classes that included in Caltech256, while images (d, e) are from the classes that are not in Caltech256. It shows that for the classes that overlap with Caltech256, the generator can generate sparse masks that align well with the object in the images, while for the images that are from non-overlapping classes the masks are not very accurate, indicating the transferability of generator is class dependent.

C. SEMANTIC IMAGE COMPRESSION

Finally, we evaluate the semantic image compression performance of NICE on the Caltech256 images. As a comparison, we also use the salient regions generated by Saliency Map, RTIS and CAM for semantic image compression. In this

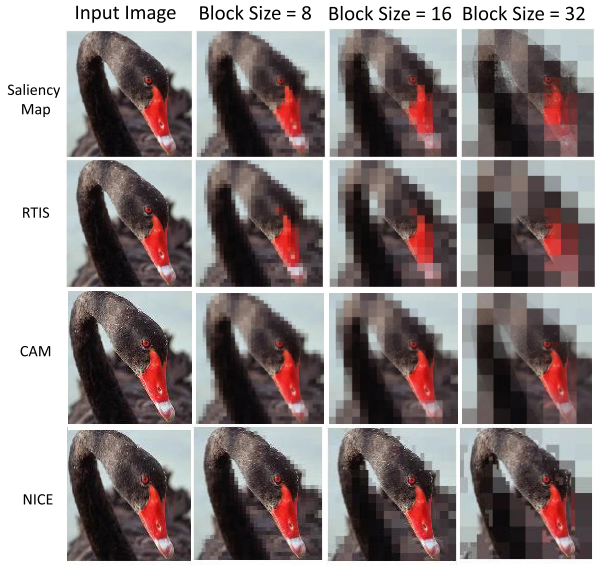


FIGURE 8. The mixed-resolution images generated by NICE, Saliency Map, RTIS and CAM with different block size *b*s.

task, two approaches can be used to train the NICE pipeline: (1) **Discriminator-fixed**: given a pretrained discriminator, we freeze its parameters $\boldsymbol{\theta}_d$ in the pipeline and only update the parameters of generator $\boldsymbol{\theta}_g$ by optimizing the overall loss (11). In this case, the mask generator is trained to generate sparse explanation to the original discriminator. (2) **Discriminator-finetuned**: similar to discriminator-fixed except that the top few layers of the discriminator $\boldsymbol{\theta}_d$ are finetuned. In this case, the discriminator can adjust its parameters to improve its predictions on the mixed-resolution images, and thus higher accuracy and compression rate are expected. Note that due to their specific training methodologies, Saliency Map, RTIS and CAM do not have the flexibility of finetuning their discriminators, limiting their applications to semantic compression tasks.

As a start, we use the sparse masks generated by NICE, Saliency Map, RTIS and CAM to produce a set of mixed-resolution images via (13) for visualization. Fig. 8 illustrates some example mixed-resolution images generated with different algorithms and block size *b*s. As we can see, NICE generated mixed-resolution images are clearly better than those from other algorithms. Thanks to the high accuracy of NICE on identifying salient regions, even when the background regions are subsampled with a block size of 32, the discriminator can still successfully classify these images. As a result, high compression rate and high classification accuracy can be achieved simultaneously.

To quantitatively evaluate the trade-off between semantic compression rate and classification accuracy, we train the NICE pipeline with **Discriminator-fixed** and **Discriminator-finetuned**[4] with $b = 16$ to generate sparse masks for each Caltech256 test image. After training, we generate mixed-resolution images with a different b in $\{1, 2, 4, 8, 16, 32, 64\}$. We then use PNG [36],

[4]For discriminator-finetuned, we set the parameters of Conv-4, Conv-5 and the FC layers of ResNet18 to be trainable and freeze all the other layers.

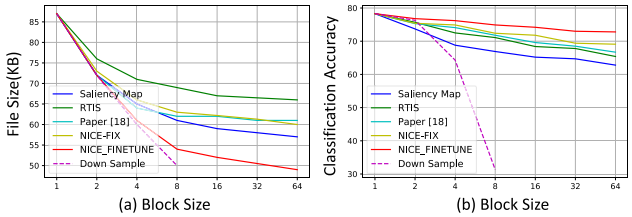


FIGURE 9. The evolution of (a) average file size of the PNG compressed image and (b) classification accuracy as a function of block size *b* of NICE-fixed, NICE-finetuned, Saliency Map, RTIS, CAM (paper [18]) and down sampling.

a standard image compression algorithm, to store the generated mixed-resolution images and report the file sizes.[5] We also classify the mixed-resolution images with the discriminators to report classification accuracies. As a comparison, the same procedure is applied to Saliency Map, RTIS and CAM (paper [18]) for semantic image compression. We also include a baseline that uses down sampling in our compression pipeline to demonstrate the importance of salient region detection for semantic compression. Specifically, we use down sampling to generate low resolution images regardless of salient regions of the images. When testing the accuracy on the down sampled images, we upsample them to original resolution by bilinear interpolation.

Fig. 9 shows the average file size of compressed images and the corresponding classification accuracy as a function of block size *b*. As we can see, when the block size increases, the file size of the compressed images decreases (higher compression rate) and the classification accuracy also decreases (lower classification accuracy), and vis-versa. The classification accuracies of NICE-finetuned are significantly higher than the other four baseline methods, meanwhile it achieves the best compression rate. When the block-size is 8, NICE-finetuned achieves a 1.6× compression rate (87KB vs. 54KB) with a small (3.35%) accuracy drop (78.30% vs. 74.95%), demonstrating the superior performance of NICE on semantic image compression.

Comparing NICE-finetuned with down sampling, the results show that down sampling hurts classification accuracy significantly because it uniformly drops pixels regardless of their saliency. For example, when we down sample images with a factor of 8, it reduces the average file size significantly (slightly better than NICE), but the corresponding classification accuracy is only 30.83%, while NICE still achieves an accuracy of 74.95%. Therefore, the mixed-resolution images produced by NICE can achieve a much better balance between compression rates and final classification accuracies than down sampling.

Note that semantic image compression rate depends on the size of salient regions of an image. Given the large objects in Caltech256, the 1.6× compression rate means NICE uses 60% pixels to achieve a similar classification accuracy.

[5]The reason that we choose PNG [36] instead of JPEG [35] for compression is because PNG is a lossless compression. Thus, the file size reduction of the mixed-resolution images can be 100% attributed to NICE, and the possible artifacts introduced by JPEG, a lossy compression, can be avoided.

TABLE 2. Inference time comparison between NICE and the baseline algorithms. The results are averaged over 100 runs.

Explanation Algorithm	Saliency Map [4]	CAM [21]	RTIS [9]	NICE
Run Time (sec) on 1000 Images mean(std)	13.94(0.55)	9.87(0.48)	1.73(0.02)	0.59(0.01)

To achieve even higher compression rates, other image compression algorithms [20], [54] can be used to compress NICE generated images further since our algorithm is complementary to these compression techniques.

D. INFERENCE TIME COMPARISON

Besides the improved explanation performance of NICE, another advantage of NICE is its superior inference speed over the competing methods. As discussed above, to generate the sparse mask to explain the decision of a target CNN, NICE only needs one forward propagation of the generator network, while backpropagation-based algorithms, like Saliency Map and CAM, requires heavy computation of backpropagation to generate the salient regions. Similar to NICE, RTIS does not need backpropagation at inference time, but it requires to compute the feature maps of intermediate layers of the target CNN as the input of the generator, which is also time consuming. To have a quantitative speed comparison, we calculate the inference times of different explanation algorithms on 1000 images from Caltech256. We run each experiment 100 times on a NVIDIA Tesla V100 GPU and report the average run-times in Table 2. As we can see, NICE is about 23× times faster than Saliency Map and 16.5× faster than CAM, while being 2.8× faster than non-backpropagation based RTIS.

V. CONCLUSION

We propose NICE, a unified end-to-end trainable framework, for neural explanation and semantic image compression. Compared to many existing explanation algorithms that heavily rely on backpropagation, the sparse masks generated by NICE are much more concise and coherent and align well with human intuitions. With the sparse masks, the proposed mixed-resolution image compression further achieves higher compression rates compared to the existing semantic compression algorithms, while retaining similar classification accuracies with the original images. We conduct a series of experiments on multiple image classification benchmarks with multiple CNN architectures and demonstrate its improved explanation quality and semantic image compression rate.

As for future work, we plan to extend the technique developed here to other domains, such as text and bioinformatics for neural explanation and summarization, where interpretable decisions are also critical for the deployment of DNNs.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their comments and suggestions, which helped to improve

the quality of this article. The authors would also gratefully acknowledge the support of VMware Inc., for its university research fund to this research.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: http://arxiv.org/abs/1810.04805

[3] E. Battenberg, J. Chen, R. Child, A. Coates, Y. Gaur, Y. Li, H. Liu, S. Satheesh, D. Seetapun, A. Sriram, and Z. Zhu, "Exploring neural transducers for end-to-end speech recognition," 2017, *arXiv:1707.07413*. [Online]. Available: http://arxiv.org/abs/1707.07413

[4] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2013, *arXiv:1312.6034*. [Online]. Available: http://arxiv.org/abs/1312.6034

[5] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 818–833.

[6] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS ONE*, vol. 10, no. 7, Jul. 2015, Art. no. e0130140.

[7] C. Gan, N. Wang, Y. Yang, D.-Y. Yeung, and A. G. Hauptmann, "DevNet: A deep event network for multimedia event detection and evidence recounting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2568–2577.

[8] M. Ribeiro, S. Singh, and C. Guestrin, "'Why should i trust you?': Explaining the predictions of any classifier," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Demonstrations*, 2016, pp. 1135–1144.

[9] P. Dabkowski and Y. Gal, "Real time image saliency for black box classifiers," in *Proc. NIPS*, 2017, pp. 6967–6976.

[10] R. C. Fong and A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3429–3437.

[11] J. Li, W. Monroe, and D. Jurafsky, "Understanding neural networks through representation erasure," 2016, *arXiv:1612.08220*. [Online]. Available: http://arxiv.org/abs/1612.08220

[12] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. NIPS*, 2017, pp. 4765–4774.

[13] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proc. ICML*, 2017, pp. 3145–3153.

[14] C. Yang, A. Rangarajan, and S. Ranka, "Global model interpretation via recursive partitioning," in *Proc. IEEE 20th Int. Conf. High Perform. Comput. Commun., IEEE 16th Int. Conf. Smart City; IEEE 4th Int. Conf. Data Sci. Syst. (HPCC/SmartCity/DSS)*, Jun. 2018.

[15] J. Ish-Horowicz, D. Udwin, S. Flaxman, S. Filippi, and L. Crawford, "Interpreting deep neural networks through variable importance," 2019, *arXiv:1901.09839*. [Online]. Available: http://arxiv.org/abs/1901.09839

[16] "White paper: Cisco visual networking index: Forecast and trends, 2017–2022," Tech. Rep., 2019.

[17] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *Proc. ICLR*, 2017, pp. 1–27.

[18] A. Prakash, N. Moran, S. Garber, A. Dilillo, and J. Storer, "Semantic perceptual image compression using deep convolution networks," in *Proc. Data Compress. Conf. (DCC)*, Apr. 2017, pp. 250–259.

[19] N. Johnston, D. Vincent, D. Minnen, M. Covell, S. Singh, T. Chinen, S. J. Hwang, J. Shor, and G. Toderici, "Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4385–4393.

[20] K. Nakanishi, S. ichi Maeda, T. Miyato, and D. Okanohara, "Neural multi-scale image compression," in *Proc. Asian Conf. Comput. Vis. (ACCV)*, 2018, pp. 718–732.

[21] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2921–2929.

[22] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.

[23] J. Chen, L. Song, M. J. Wainwright, and M. I. Jordan, "Learning to explain: An information-theoretic perspective on model interpretation," 2018, *arXiv:1802.07814*. [Online]. Available: http://arxiv.org/abs/1802.07814

[24] S. Bang, P. Xie, H. Lee, W. Wu, and E. Xing, "Explaining a black-box using deep variational information bottleneck approach," 2019, *arXiv:1902.06918*. [Online]. Available: http://arxiv.org/abs/1902.06918

[25] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso, "Machine learning interpretability: A survey on methods and metrics," *Electronics*, vol. 8, no. 8, p. 832, Jul. 2019.

[26] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[27] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2961–2969.

[28] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 801–818.

[29] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille, "Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1742–1750.

[30] P. O. Pinheiro and R. Collobert, "From image-level to pixel-level labeling with convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1713–1721.

[31] A. Bearman, O. Russakovsky, V. Ferrari, and L. Fei-Fei, "What's the point: Semantic segmentation with point supervision," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 549–565.

[32] M. Chen, T. Artières, and L. Denoyer, "Unsupervised object segmentation by redrawing," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2019, pp. 12726–12737.

[33] J. Dai, K. He, and J. Sun, "BoxSup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1635–1643.

[34] D. Lin, J. Dai, J. Jia, K. He, and J. Sun, "ScribbleSup: Scribble-supervised convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3159–3167.

[35] G. K. Wallace, "The jpeg still picture compression standard," *IEEE Trans. Consum. Electron.*, vol. 38, no. 1, pp. 18–36, Feb. 1992.

[36] K. Sayood, *Lossless Compression Handbook*. Amsterdam, The Netherlands: Elsevier, 2002.

[37] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," 2016, *arXiv:1611.01704*. [Online]. Available: http://arxiv.org/abs/1611.01704

[38] G. Toderici, S. M. O'Malley, S. J. Hwang, D. Vincent, D. Minnen, S. Baluja, M. Covell, and R. Sukthankar, "Variable rate image compression with recurrent neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–12.

[39] G. Toderici, D. Vincent, N. Johnston, S. J. Hwang, D. Minnen, J. Shor, and M. Covell, "Full resolution image compression with recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5306–5314.

[40] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," 2017, *arXiv:1703.00395*. [Online]. Available: http://arxiv.org/abs/1703.00395

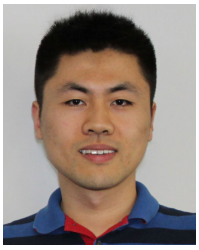
[41] M. Li, W. Zuo, S. Gu, D. Zhao, and D. Zhang, "Learning convolutional networks for content-weighted image compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3214–3223.

[42] T. Bird, J. Kunze, and D. Barber, "Stochastic variational optimization," 2018, *arXiv:1809.04855*. [Online]. Available: http://arxiv.org/abs/1809.04855

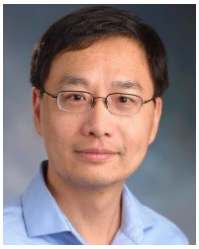
[43] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 229–256, May 1992.

[44] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–13.

[45] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–17.
[46] G. Tucker, A. Mnih, C. J. Maddison, J. Lawson, and J. Sohl-Dickstein, "Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models," in *Proc. NIPS*, 2017, pp. 2627–2636.
[47] W. Grathwohl, D. Choi, Y. Wu, G. Roeder, and D. Duvenaud, "Backpropagation through the void: Optimizing control variates for black-box gradient estimation," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Feb. 2018.
[48] C. Louizos, M. Welling, and D. P. Kingma, "Learning sparse neural networks through l_0 regularization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–13.
[49] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[50] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.
[51] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," Tech. Rep., 2007.
[52] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–15.
[53] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[54] O. Rippel and L. Bourdev, "Real-time adaptive image compression," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 2922–2930.



XIANG LI received the bachelor's degree in electrical engineering from Donghua University, China, in 2013, and the master's degree in pattern recognition from Northeastern University, China, in 2018. He is currently pursuing the Ph.D. degree in computer science with Georgia State University. His research interests include the robustness of deep learning, and security and interpretability of deep neural networks.



SHIHAO JI (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from Duke University, in 2006. After that, he was a Research Associate with Duke for about one and a half years. He is currently an Associate Professor with the Computer Science Department, Georgia State University. Prior to joining GSU, he spent about ten years in industry research labs. His principal research interests include machine learning and deep learning with an emphasis on high-performance computing. His research interest includes developing efficient algorithms that can learn from a variety of data sources (e.g., image, audio, and text) on a large scale and automate decision-making processes in dynamic environments.