On The Distribution of Penultimate Activations of Classification Networks

Minkyo Seo[1] **Yoonho Lee**[3] **Suha Kwak**[2]

[1]Department of Computer Science and Engineering, POSTECH, Pohang, South Korea
[2]Graduate School of AI, POSTECH, Pohang, South Korea
[3]AITRICS, Seoul, South Korea

Abstract

This paper studies probability distributions of penultimate activations of classification networks. We show that, when a classification network is trained with the cross-entropy loss, its final classification layer forms a *Generative-Discriminative pair* with a generative classifier based on a specific distribution of penultimate activations. More importantly, the distribution is parameterized by the weights of the final fully-connected layer, and can be considered as a generative model that synthesizes the penultimate activations without feeding input data. We empirically demonstrate that this generative model enables stable knowledge distillation in the presence of domain shift, and can transfer knowledge from a classifier to variational autoencoders and generative adversarial networks for class-conditional image generation.

1 INTRODUCTION

Deep neural networks have achieved remarkable success in image classification [10, 13]. In most of these networks, an input image is first processed by multiple layers of neurons, whose final output, called *penultimate activations*, is in turn fed to the last fully connected layer that conducts classification. These networks are typically trained in an end-to-end manner by minimizing the *cross-entropy loss*. The penultimate activations are the deepest image representation of the networks and have proven to be useful for various purposes besides classification such as image retrieval [45], semantic segmentation [26], and general image description of unseen classes [35].

This paper studies the penultimate activations of classification networks through a generative model. We derive this model by exploiting a dual relationship between the activations and the final classification layer weights, which results from the common practice of applying softmax to the output of a linear layer. Because of this, penultimate activations are determined entirely by their preceding layers, and they interact with the final classification layer in a predetermined way. Using this fixed interaction, we can approximately recover information about penultimate activations using only the final layer's weights.

Because this generative model only uses final layer weights, it yields a compact representation of inter-class affinity with many potential applications. It can serve as a lightweight knowledge transfer protocol, especially compared to previous frameworks requiring feeding data through networks. Additionally, because our representation of class relations does not directly use data, it is more robust to domain shift and is potentially suitable for transferring knowledge in privacy-sensitive scenarios. This representation's simple structure also enables transfer to learning modalities beyond supervised classification, such as serving as an effective data-driven prior for class-conditional generative models. Furthermore, our model encodes information beyond the decision boundaries in the activation space, which has various potential applications, including anomaly detection and uncertainty estimation.

We experimentally demonstrate that our generative model of penultimate activations can be used for practical applications such as Knowledge Distillation (KD) [1, 12] and class-conditional image generation [4, 16, 24]. For KD, our model allows us to distill knowledge from a teacher network without feeding images forward through the teacher by generating its activations directly; this new approach to KD is complementary to the standard one [12] and more robust against domain shift between teacher and student. We also show that our model of penultimate activations in a trained classifier can be used as a data-dependent prior for a class-conditional image generation model, resulting in higher-quality synthetic images compared to those of vanilla models.

The remainder of this paper is organized as follows. In sec-

Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI 2021), PMLR 161:1141–1151.

tion 2, we analyze penultimate activations of classification networks and derive their probabilistic model. After reviewing previous work related to our model of penultimate activations in section 3, we apply our model to KD and conditional image generation in section 4. We then conclude this paper with a discussion about limitations and future directions of our method in section 5.

2 DISTRIBUTIONS OF PENULTIMATE ACTIVATIONS

Consider a standard neural network that classifies data of c different classes with ground-truth labels i. We assume a balanced dataset where $p(i) = \frac{1}{c}$ for all classes $i = 1, \ldots, c$. We denote penultimate activations of the network by $\mathbf{a} \in \mathbb{R}^d$ and weights of the final fully-connected layer for classification by $\mathbf{W} \in \mathbb{R}^{d \times c}$; this layer produces logits $\mathbf{W}^\top \mathbf{a} \in \mathbb{R}^c$. We denote columns of $\mathbf{W}$ by $\mathbf{w}_1, \ldots, \mathbf{w}_c \in \mathbb{R}^d$, and represent projections of $\mathbf{a}, \mathbf{w}_1, \cdots, \mathbf{w}_c$ onto the unit hypersphere $\mathbb{S}^{d-1}$ using upper bars:

$$\overline{\mathbf{a}} \overset{\text{def}}{=} \frac{\mathbf{a}}{\|\mathbf{a}\|}, \quad \overline{\mathbf{w}}_i \overset{\text{def}}{=} \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|}. \tag{1}$$

2.1 ANALYSIS OF CROSS-ENTROPY LOSS

We analyze how the common practice of minimizing cross-entropy loss affects the distribution of penultimate activations. Our analysis reveals a close connection between the cross-entropy loss and a specific distribution of normalized penultimate activations, which can be described using only $\mathbf{W}$, the last classification layer's parameters.

The column vectors $\mathbf{w}_1, \ldots, \mathbf{w}_c$ of $\mathbf{W}$ can be interpreted as c different prototypes, each of which represents a particular class. The network classifies a datapoint by comparing these prototypes against its penultimate activations. This interpretation motivates us to derive a probability distribution of activations of class i using $\overline{\mathbf{w}}_i$. To this end, we first rewrite the cross-entropy loss in terms of penultimate activations:

$$\mathcal{L}_{\text{xent}} = -\log \frac{\exp(\|\mathbf{w}_i\| \|\mathbf{a}\| \, \overline{\mathbf{w}}_i^\top \overline{\mathbf{a}})}{\sum_j \exp(\|\mathbf{w}_j\| \|\mathbf{a}\| \, \overline{\mathbf{w}}_j^\top \overline{\mathbf{a}})}. \tag{2}$$

Eq. (2) resembles Bayes' theorem: Assuming $\|\mathbf{a}\|$ is a constant and $\|\mathbf{w}\|_i$ are the same for all i, $\exp(\|\mathbf{w}_i\| \|\mathbf{a}\| \, \overline{\mathbf{w}}^\top \overline{\mathbf{a}})$ acts as an unnormalized joint probability for $(\overline{\mathbf{a}}, i)$, and the denominator is the sum of all possible cases for class j.

The von Mises-Fisher (vMF) distribution, a well-known distribution in directional statistics, exactly takes this form of joint probability and is defined as

$$\text{vMF}(\mathbf{x}; \mu, \kappa) \overset{\text{def}}{=} C(\kappa) \exp(\kappa \mu^\top \mathbf{x}), \tag{3}$$

where $\mu \in \mathbb{S}^{d-1}$ is the mean direction, $\kappa \in [0, \infty)$ is a concentration term, and $C(\kappa)$ is a normalizing constant.

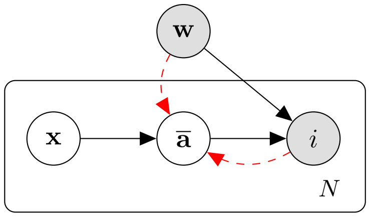


Figure 1: Plate notation representation of our model's structure. We assume that the network is trained to map $\mathbf{x}$ to i, and exploit the structure of the last layer to approximately reconstruct activations using only weights $\mathbf{w}$ and label i.

We write the cross-entropy in Eq. (2) in terms of the vMF distributions:

$$\mathcal{L}_{\text{xent}} = -\log \frac{\text{vMF}(\overline{\mathbf{a}}; \overline{\mathbf{w}}_i, \|\mathbf{w}_i\| \|\mathbf{a}\|)}{\sum_j \text{vMF}(\overline{\mathbf{a}}; \overline{\mathbf{w}}_j, \|\mathbf{w}_j\| \|\mathbf{a}\|)}. \tag{4}$$

This motivates the following generative model which jointly models penultimate activations and labels:

$$q(\overline{\mathbf{a}}, i) = q(i) q(\overline{\mathbf{a}} | i) = \frac{1}{c} \text{vMF}(\overline{\mathbf{a}}; \overline{\mathbf{w}}_i, \|\mathbf{w}_i\| \|\mathbf{a}\|). \tag{5}$$

We see that the model $q(\overline{\mathbf{a}}, i)$ forms a *Generative-Discriminative pair* [25] with the predictive distribution of the classification network:

$$\begin{aligned} &\arg\max \log p(i | \overline{\mathbf{a}}) = \arg\min \mathcal{L}_{\text{xent}} \\ &\approx \arg\max \log \frac{q(\overline{\mathbf{a}}, i)}{\sum_j q(\overline{\mathbf{a}}, j)} = \arg\max \log q(i | \overline{\mathbf{a}}). \end{aligned} \tag{6}$$

Eq. (6) shows that $q(\overline{\mathbf{a}}, i)$ is closely related to the prediction of the classification network, and suggests that $q(\overline{\mathbf{a}} | i)$ can be used as an approximation to the true posterior $p(\overline{\mathbf{a}} | i)$. Our modeling procedure is shown in Figure 1. Note that while the network uses data x during training, we do not assume access to data when approximately inferring activations $\overline{\mathbf{a}}$. By experiments in various domains, we will demonstrate that this simple model can transfer a substantial amount of information about the learned activation space.

For Eq. (6) to be an exact identity, the concentration parameter $\|\mathbf{w}_i\| \|\mathbf{a}\|$ of each vMF component $q(\overline{\mathbf{a}} | i)$ must be equal. In the next section, we empirically verify to what extent this is true in trained classification networks. In addition, to sample from $q(\overline{\mathbf{a}}, i)$ using only the final layer parameters $\mathbf{W}$, we treat $\|\mathbf{a}\|$ as a concentration hyperparameter and tune it using cross-validation on the downstream task.

2.2 EMPIRICAL VERIFICATION OF OUR MODEL

In the previous section, we have suggested that the normalized activations $\overline{\mathbf{a}}$ for each class follow the von Mises-Fisher distribution $q(\overline{\mathbf{a}} | i)$. We qualitatively verify this claim by visualizing penultimate activations of a classification network trained on the MNIST dataset [19] and the vMF distributions derived from its final classification layer in Figure 2. The classification network consists of 4 convolution layers followed by the final fully connected layer that produces 2-dimensional penultimate activations (*i.e.*, $\mathbf{a}, \mathbf{w}_1, \ldots, \mathbf{w}_c \in \mathbb{R}^2$).

Figure 2 shows that in the early stages of training, normalized penultimate activations are not well aligned with vMF distributions. However, as training progresses, they become grouped for each class and follow their corresponding vMF distributions. This is in line with our analysis, in which we claimed that the normalized penultimate activations follow vMF distributions if the network is trained by minimizing the cross-entropy loss.

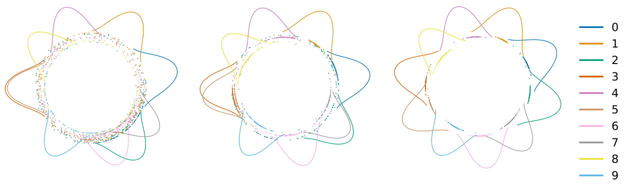
2.3 ARE DIRECTIONAL STATISTICS SUFFICIENT FOR CLASSIFICATION?

Our approach to modeling normalized activations $q(\overline{\mathbf{a}}|i)$ implicitly assumes that the directional vectors $\overline{\mathbf{a}}$ and $\overline{\mathbf{w}}_i$ hold sufficient information for classification.

This assumption is empirically verified by quantifying how much the accuracy of trained classification networks drops when normalizing both $\mathbf{a}$ and $\mathbf{w}_i$. To this end, we choose 9 networks pre-trained for the ImageNet classification task [32] and measure their performance on the ImageNet validation set. As summarized in Table 1, the performance drop by the normalization is marginal, especially when the network has more capacity.

We additionally provide an informal argument based on degrees of freedom for the sufficiency of directional statistics. First, the norm of $\mathbf{a}$ has no effect on the ranking of logits, thus not affecting the decision boundary in terms of logit ordering. On the other hand, the norm of $\mathbf{w}_i$ could change the decision boundary, but it has little influence on large networks. Note that the direction vector $\overline{\mathbf{w}}$ accounts for $d-1$ of the d degrees of freedom of the prototype vector $\mathbf{w} \in \mathbb{R}^d$. Therefore, the fraction of the information that $\overline{\mathbf{w}}$ holds about $\mathbf{w}$ roughly converges towards 1 as d increases ($\lim_{d\to\infty} \frac{d-1}{d} = 1$). Since the dimension d is typically very large in standard classification networks, we argue that the statistics of $\overline{\mathbf{w}}$ should hold sufficient information.

Furthermore, the sufficiency of directional statistics has been confirmed indirectly by the widespread use of hypersphere embedding in the face recognition literature [7, 21] and the hardness metric based only on angles between activation and weight vectors [3]. The fact that hypersphere embedding works in such domains supports our claim that classification can be done only with directional information. Our analysis further suggests that even standard (unnormalized) networks may store most of their information in directional statistics.



(a) Epoch 1 (b) Epoch 2 (c) Epoch 30

Figure 2: Visualization of penultimate activations and vMF distributions derived from a classification network trained on the MNIST dataset. $\overline{\mathbf{a}}$ of datapoints are represented by dots, and vMF distributions $q(\overline{\mathbf{a}}|i)$ are drawn by solid lines.

3 RELATED WORK

3.1 UNDERSTANDING DEEP NEURAL NETWORKS

Understanding what neural networks learn about data has been a fundamental problem in deep learning. Previous methods have analyzed classification networks by optimizing an image to maximally activate a specific neuron [6, 42] or maximize the predicted probability of a specific class [36]. A similar technique has been used to visualize the entire feature map of an image [23]. For the same purpose, areas that most contribute to classification are estimated for each class through a weighted average of each activation channel [46]. Our work also belongs to this line of research in that our model explicitly exhibits the learned class relations in a classification network.

While these previous methods offer high-level insights into the characteristics of deep neural network classifiers, they do not provide a way of using their insights to facilitate the training of other models. Meanwhile, Yin *et al.* [41] propose to synthesize class-conditional images using knowledge of a trained classifier and exploits the generated images for knowledge distillation. In contrast, our generative model can synthesize class-conditional *activations* without such a costly image synthesis procedure. We experimentally show that our method can extract the relationship between classes while being stable under distribution shift and transfer to different modalities such as generative modeling.

3.2 GENERATIVE-DISCRIMINATIVE PAIRS

Generative classifiers model the joint probability $p(x, y)$ while discriminative ones model the conditional probability $p(y|x)$. If two models belong to the same parametric family but respectively use the generative and discriminative criteria, the two are said to form a Generative-Discriminative pair [25, 31]. In this context, the work by Lee *et al.* [20] is the most similar to ours conceptually. They propose a generative model of activations that forms a generative-discriminative pair with a given classifier, and apply the

Table 1: Performance of various classification networks before and after normalizing $\mathbf{a}$ and $\mathbf{w}_i$ in top-1 accuracy on the ImageNet validation set. R: ResNet [10], D: DenseNet [13], S: ShuffleNet [22], RX: ResNeXt [40].

	R-18	R-50	R-101	R-152	D-121	D-201	S-v2	RX-50	RX-101
Original	69.8	76.2	77.4	78.3	74.7	77.2	69.4	77.6	79.3
Normalized	67.1	74.7	76.2	77.5	72.5	75.5	68.4	76.9	78.9
Drop rate	-3.9%	-1.9%	-1.5%	-1.1%	-2.9%	-2.2%	-1.5%	-0.9%	-0.6%

model to detecting out-of-distribution samples and adversarial attacks. We also derive a generative model that approximately forms a generative-discriminative pair with any classification networks, yet apply our model to transferring learned class relations to other networks and tasks.

3.3 LEARNING WITH VMF DISTRIBUTIONS

As the vMF distribution is one of the simplest distributions for directional data, mixtures of vMFs have been widely used for clustering directional data [2, 8]. For Bayesian inference of neural network weights, vMF distributions are used to model the directional statistics of the weights that are decomposed into radial and directional components [28]. Also, vMF embedding spaces have been studied for deep metric learning [9] since such hypersphere embedding spaces are more desirable than conventional Euclidean spaces when their dimension is large. Kumar *et al.* [18] used vMF distributions to reduce the large computations involved in normalizing the softmax for a set of words.

Our use of directional statistics differs from these previous methods: we use it as a tool for explaining the behavior of standard classification models rather than for specialized purposes like building a compact embedding space and computation reduction.

4 APPLICATIONS

This section demonstrates that our generative model of penultimate activations can be applied to two practical applications, KD [1, 12, 30] and class-conditional image generation [4, 27].

4.1 CLASS-WISE KNOWLEDGE DISTILLATION

This section describes how the generative model of activations can be used to develop a new algorithm for KD, and validates its effectiveness.

4.1.1 Algorithm Details

KD is the task of distilling knowledge from a teacher network T to a student network S [12]. Unlike most of the existing methods, our model enables KD without feeding data forward through T by directly generating activations of a certain class. In detail, our model is used to approximate the average prediction of T per class, which is represented as the probability of T's prediction y given class i and estimated by

$$p_T(y|i) = \int p_T(\overline{\mathbf{a}}|i) p_T(y|\overline{\mathbf{a}})\, d\overline{\mathbf{a}} \approx \frac{1}{N} \sum_{j=1}^{N} p_T(y|\overline{\mathbf{a}}_j), \tag{7}$$

where we employ Monte Carlo integration since the exact integral is intractable. Also, each $\overline{\mathbf{a}}_j$ is an *i.i.d.* sample from vMF$(\overline{\mathbf{w}}_i, \kappa)$, where κ is set to 80 for all experiments by inspecting the empirical norm of the feature distribution on a teacher model.

The estimated $p_T(y|i)$ in Eq. (7) quantifies the relationship between two classes y and i that is captured by T, and is employed as a target for KD in our method. Recall that for teacher network p_T and student network p_S, the standard KD loss [12] is

$$\mathcal{L}_{\text{KD}} = -\mathbb{E}_{\substack{i,\mathbf{x} \sim p(i,\mathbf{x}) \\ \mathbf{y} \sim p_T(y|\mathbf{x})}} \left[\log p_S(y|\mathbf{x})\right], \tag{8}$$

where y denotes prediction and $\mathbf{x}$ and i are data and label, respectively. This loss is designed to minimize the KL divergence between $p_T(y|\mathbf{x})$ and $p_S(y|\mathbf{x})$ for each data $\mathbf{x}$. Unlike this data-wise KD, our approach is a Class-wise KD (CKD) whose objective is

$$\mathcal{L}_{\text{CKD}} = -\mathbb{E}_{\substack{i,\mathbf{x} \sim p(i,\mathbf{x}) \\ \mathbf{y} \sim p_T(y|i)}} \left[\log p_S(y|\mathbf{x})\right], \tag{9}$$

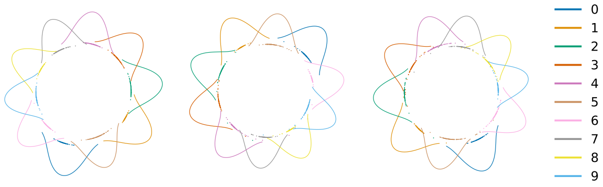
where the categorical distribution $p_T(y|i)$ is given by Eq. (7). Note again that while the standard KD objective in Eq. (8) requires a forward pass through the teacher network T to compute $p_T(y|\mathbf{x})$, ours in Eq. (9) utilizes the pre-computed distribution $p_T(y|i)$ without exploiting T during training of S. This property of CKD is useful especially when it is hard to conduct forward propagation through T (*e.g.*, online learning of S with limited memory and computation power) or if there is domain shift between training datasets for T and S as demonstrated by experiments in section 4.1.4. The overall procedures of the standard KD and our CKD are described in Algorithm 1 and 2, respectively, where the main differences between them are colored in red.

Algorithm 1 Knowledge Distillation [12]

```
Require: teacher network $\mathbf{x} \mapsto p_T(y|\mathbf{x})$
Require: student network $\mathbf{x} \mapsto p_S(y|\mathbf{x})$
while not converged do
    $\mathbf{x}, i \sim p(\mathbf{x}, i)$
    $p_T \Leftarrow p_T(y|\mathbf{x})$
    $p_S \Leftarrow p_S(y|\mathbf{x})$
    $\mathcal{L}_{\text{KD}} \Leftarrow -p_T \cdot \log p_S$
end while
```

Algorithm 2 Class-wise Knowledge Distillation (ours)

```
Require: teacher network $\mathbf{x} \mapsto p_T(y|\mathbf{x})$
Require: student network $\mathbf{x} \mapsto p_S(y|\mathbf{x})$
$p_T(y|i) \Leftarrow \frac{1}{N}\sum_{j=1}^{N} p_T(y|\bar{\mathbf{a}}_j)$
while not converged do
    $\mathbf{x}, i \sim p(\mathbf{x}, i)$
    $p_T \Leftarrow p_T(y|i)$
    $p_S \Leftarrow p_S(y|\mathbf{x})$
    $\mathcal{L}_{\text{CKD}} \Leftarrow -p_T \cdot \log p_S$
end while
```



(a) Teacher (b) Student (KD) (c) Student (CKD)

Figure 3: Visualization of penultimate activations and their vMF distributions of teacher and student networks on the MNIST dataset. $\bar{\mathbf{a}}$ of datapoints are represented by dots and vMF distributions $q(\bar{\mathbf{a}}|i)$ are drawn by solid lines.

4.1.2 Qualitative Analysis on the Effect of CKD

To investigate which kind of information of T is transferred to S through KD or CKD, we qualitatively examine penultimate activations and their generative models of the three networks on the MNIST dataset. In this experiment, T consists of 4 convolution layers followed by the final fully connected layer and produces 2-dimensional penultimate activations. S has the same architecture but with half the number of convolution kernels. From the visualization results in Figure 3, we observe that T and S have the same cyclic order of classes in the space of their penultimate activations. This demonstrates that Eq. (9) encourages S to follow the inter-class relationships captured by T.

4.1.3 Network Compression through KD

The effectiveness of CKD is first evaluated on the CIFAR-100 dataset [17] in the scenario of network compression. We adopt WRN-40-2 as T and WRN-16-2 as S, both of which are introduced by [44], and follow the experimental protocol for network compression proposed in [38]. Follwing [12], we set the temperature for the KD loss to 4 for all experiments. The results of CKD are quantified and compared to other distillation methods in Table 2. CKD outperforms most previous methods such as RKD [29], SP [39], and VID [1]. These results demonstrate that CKD is capable of extracting useful knowledge from T. In addition, CKD and the standard KD [12] are complementary to each other and the performance is further enhanced by integrating them.

Table 2: Top-1 test accuracy of the student networks on the CIFAR-100 dataset. Results are averaged over 5 runs.

Method	Accuracy
Teacher (WRN-40-2)	75.61
Student (WRN-16-2)	73.26
FitNet [30]	73.58
KD [12]	74.92
AT [43]	74.08
RKD [29]	73.35
SP [39]	73.83
VID [1]	74.11
CRD [38]	75.48
CKD (ours)	74.32
CKD + KD (ours)	75.21

4.1.4 KD in the Presence of Domain Shift

Most KD techniques assume that T and S are trained with the same dataset or, at least, on the same domain. However, this assumption does not always hold in real-world settings, *e.g.*, when the dataset used to train T is not available due to privacy issues or when we train S using streaming data that may be corrupted by various noises. In those cases, the quality of knowledge extracted from T in a data-wise manner may be degraded since T assumes a data distribution different from what S observes.

We argue that our CKD is more robust against such a domain shift issue since it performs KD without taking input data explicitly. We evaluate CKD and compare it to the standard KD [12] on the CIFAR-100 dataset [17] while simulating domain shift. Specifically, we consider two different types of domain shift: photometric transform and downsampling. For the photometric transform, we randomly alter brightness, contrast, and saturation of input image with five different degrees $\in \{0, 0.2, 0.4, 0.6, 0.8\}$ of alteration, where 0 means we use the original image without noise. Also, for image downsampling, we reduce input image resolution with three different rates ($\times 0.75$, $\times 0.5$, $\times 0.25$) using nearest-neighbor interpolation. T is trained on the original dataset while S

Table 3: Top-1 test accuracy of the student networks on the CIFAR-100 dataset with various degrees of photometric transform and image downsampling. Results are averaged over 5 runs.

	Photometric Transform					Downsampling			
	0.0	0.2	0.4	0.6	0.8	×1.0	×0.75	×0.5	×0.25
Label	73.26	73.29	72.97	72.39	71.53	73.26	70.15	63.64	49.00
KD	**74.92**	**74.34**	71.92	65.68	51.35	**74.92**	68.52	45.84	20.27
CKD (ours)	74.32	74.18	**73.98**	**73.61**	**72.47**	74.32	**71.24**	**64.55**	**49.72**

is trained on its domain-shifted versions. We use the same architectures as in section 4.1.3, using WRN-40-2 as T and WRN-16-2 as S.

Experimental results are summarized in Table 3. CKD consistently enhances the performance of the baseline using only ground-truth labels ("Label"). On the other hand, the standard KD ("KD") deteriorates when the domain shift is significant. We believe this result is mainly because the standard KD strongly depends on the data distribution. On the other hand, the knowledge captured by CKD is still useful in the presence of domain shift since it extracts inter-class relationships directly from the weights of the final classification layer rather than relying on the data.

4.2 CONDITIONAL IMAGE GENERATION WITH HYPERSPHERICAL VAE

We use our generative model of penultimate activations to enhance class-conditional generative models. Such models generate data $\mathbf{x}$ using a latent variable $\mathbf{z}$ together with a class label i. While previous methods typically use the concatenated vector $[\mathbf{z}; i]$ as an input to the decoder network, we propose to instead use our learned model of activations as the distribution of $\mathbf{z}$ given the label: $p(\mathbf{z}|i) = q(\bar{\mathbf{a}}|i)$. In this section, this idea is applied to Hyperspherical Variational Auto-Encoder (HVAE) [4], a latent variable model which performs inference using a vMF latent distribution.

This section first describes HVAE and its variant for conditional image generation, then illustrates how our model of activations is integrated with HVAE and improves the quality of generated images. The efficacy of our method is demonstrated on the MNIST dataset.

Baseline 1: Hyperspherical VAE (HVAE). HVAE is a latent variable model, which first computes the latent variable $\mathbf{z} \in \mathbb{S}^d$ of a given datapoint $\mathbf{x}$ using an encoder $q(\mathbf{z}|\mathbf{x})$, then reconstructs $\mathbf{x}$ from $\mathbf{z}$ by a stochastic decoder $p(\mathbf{x}|\mathbf{z})$. HVAE assumes that $p(\mathbf{z})$ is a uniform distribution on the unit hypersphere $\mathbb{S}^d$. Accordingly, the encoder of HVAE is trained by maximizing the following lower bound of the evidence, usually called the ELBO:

$$\mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] - D_{KL}(q(\mathbf{z}|\mathbf{x}) \mid\mid p(\mathbf{z})). \tag{10}$$

Baseline 2: HVAE Conditioned by Concatenation (HVAE-C). A straightforward way to extend HVAE to take class label i into account is to concatenate i to the end of the latent vector $\mathbf{z}$. We call this conditioned HVAE model HVAE-C. Whereas HVAE assumes that $\mathbf{x}$ is generated from $\mathbf{z}$ alone, *i.e.*, $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$, HVAE-C assumes that $\mathbf{x}$ is generated from both $\mathbf{z}$ and i, *i.e.*, $p(\mathbf{x}, \mathbf{z}, i) = p(i)p(\mathbf{z})p(\mathbf{x}|\mathbf{z}, i)$. We train HVAE-C by maximizing the following lower bound of the evidence considering i:

$$\mathbb{E}_{q(\mathbf{z}|\mathbf{x},i)}\left[\log p(\mathbf{x}|i, \mathbf{z}) + \log p(i)\right] - D_{KL}(q(\mathbf{z}|\mathbf{x}) \mid\mid p(\mathbf{z})). \tag{11}$$

Ours: HVAE Conditioned by Learned Prior (HVAE-L). Recall from section 2.1 that one can utilize the weights of the last fully connected layer of a classification network to model a distribution of penultimate activations for a specific class i. We employ this activation distribution conditioned on class i as a learned prior for $\mathbf{z}$ of HVAE. This method is similar to HVAE-C in that the class information is involved in the process of generating $\mathbf{x}$, but the two models differ in the way to integrate the information. Unlike HVAE-C, HVAE-L generates $\mathbf{x}$ from $\mathbf{z}$ alone, yet the distribution of $\mathbf{z}$ is determined by class label i, *i.e.*, $p(\mathbf{x}, \mathbf{z}, i) = p(i)p(\mathbf{z}|i)p(\mathbf{x}|\mathbf{z})$. Accordingly, it is trained by optimizing the following objective:

$$\mathbb{E}_{q(\mathbf{z}|\mathbf{x},i)}\left[\log p(\mathbf{x}|\mathbf{z}) + \log p(i)\right] - D_{KL}(q(\mathbf{z}|\mathbf{x}) \mid\mid p(\mathbf{z}|i)). \tag{12}$$

Also, the above objective differs from that of Eq. (10) since the two models assume different generation procedures.

We compare our model (HVAE-L) against the two baselines (HVAE and HVAE-C) on the MNIST image generation task. Our experimental setup, including network architecture and hyperparameters, follows that of [4]. Specifically, both of the encoder and decoder of these models consist of three fully connected layers, whose output dimensions are $768 - 256 - 128 -$ dimension of $\mathbf{z} - 128 - 256 - 768$. In addition, we ensure that the dimensionality of the latent vector $\mathbf{z}$ is the same for all the models. The prior distribution $p(\mathbf{z}|i)$ of HVAE-L is derived from an MNIST classification network, whose architecture is the same with that of the encoder; the concentration parameter κ of the prior distribution is set to

Table 4: Comparison on the MNIST generative modeling task. Results are averaged over 5 runs.

	Log-Likelihood				ELBO			
Dimension of $\mathbf{z}$	3	5	10	20	3	5	10	20
HVAE	−122.0	−107.4	−93.6	−90.9	−124.0	−111.3	−98.0	−96.3
HVAE-C	−124.9	−110.7	−93.3	−89.7	−127.6	−114.3	−97.6	−95.3
HVAE-L (ours)	**−119.4**	**−105.2**	**−90.5**	**−87.9**	**−123.0**	**−109.1**	**−95.1**	**−93.3**

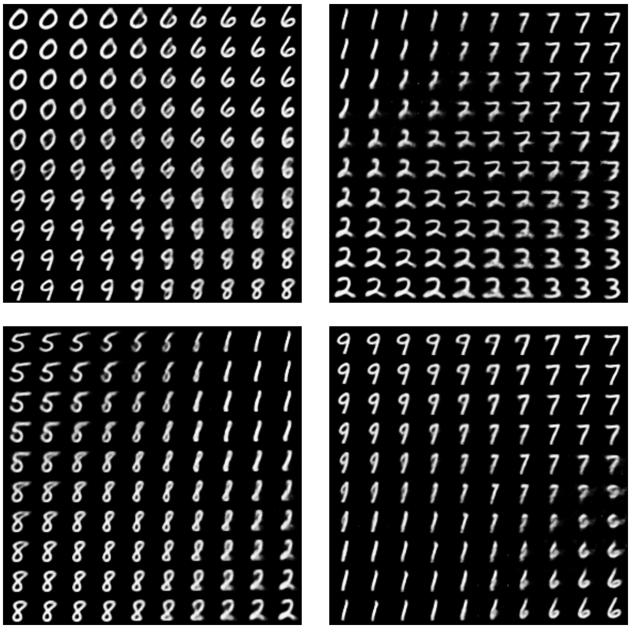


Figure 4: Visualization of the latent space of HVAE-L by interpolating between centers of different classes.

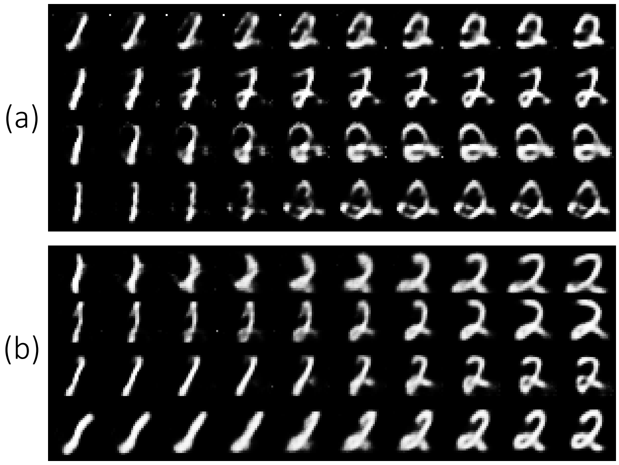


Figure 5: Visualization of the latent spaces. (a) HVAE-C. (b) HVAE-L. For HVAE-C, input latent vectors are sampled through interpolation between class codes of 1 and 2 while fixing $\mathbf{z}$. On the other hand, input latent vectors for HVAE-L are computed by interpolation between two points sampled from $p(\mathbf{z}|i = 1)$ and $p(\mathbf{z}|i = 2)$, respectively.

20. All the networks including the classification network are optimized by the Adam optimizer [15] with a learning rate of 1e−3 and mini-batches of 64 images.

The performance of our model is summarized and compared with that of the two baselines in Table 4, where HVAE-L outperforms both baselines. This result demonstrates that $q(\bar{\mathbf{a}}|i)$, the vMF distributions of class-conditional activations can serve as a useful prior for class-conditional image generation of HVAE. Specifically, we conjecture that the improvement by HVAE-L arises from the following properties of the prior. First, the prior is derived from a classification network trained using examples of all classes, thus is aware of the affinity between different classes as well as variations within each class. Second, the prior represents class identity and appearance variation jointly within a single latent space. These two properties allow HVAE-L to exploit the latent space more flexibly and effectively. In contrast, HVAE-C cannot take these advantages since it treats class labels as independent symbols and disentangles them from appearance variations. The advantage of our prior model is also demonstrated qualitatively in Figure 4 and Figure 5, where images generated by HVAE-L are more smoothly and naturally interpolated between different classes in the latent space than those of HVAE-C.

4.3 CONDITIONAL IMAGE GENERATION WITH GANS

In this section, our generative model of penultimate activations is utilized as a class-conditional prior for conditional Generative Adversarial Networks (cGANs) [24]. Most cGANs independently sample a class label i and a latent vector indicating a specific appearance of the class [24, 27]. On the other hand, our cGAN variant samples a single latent vector $\mathbf{z}$, which represents class identity and appearance jointly, from $p(\mathbf{z}|i) = q(\bar{\mathbf{a}}|i)$, the vMF distribution of class-conditional activations derived in section 2.1.

Our method is incorporated with SNGAN [24] implemented by [34], and compared to the original SNGAN on a class-conditional image generation task using the CIFAR10 dataset [17]. The only difference between SNGAN and our variant is that we replace the input Gaussian noise $\mathbf{z}$ of

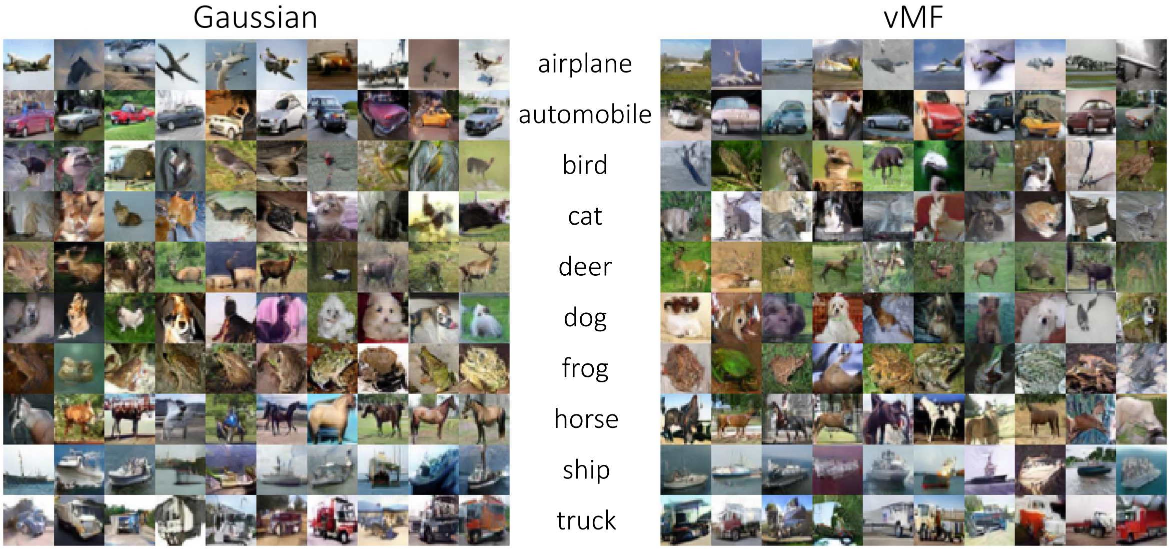


Figure 6: Qualitative conditional image generation results using GAN variants on the CIFAR10 dataset.

Table 5: Comparison on the CIFAR10 generative modeling task. Results are averaged over 5 runs.

$\mathbf{z}$	IS (↑)	FID 5k (↓)
Gaussian	8.406	18.867
vMF	**8.511**	**18.764**

SNGAN with our vMF noise for label i, and the input for conditional batch normalization [5] is still i. The vMF distributions used to sample the latent vectors are derived from the WRN-40-2 network [44] trained on the same dataset. Latent vectors of all methods are 128-dimensional, and we multiply 10 to latent vectors sampled from our vMF models to match their norm to that of baseline methods. We directly follow the evaluation protocol of [34]. The only additional hyperparameter was the concentration parameter κ of the vMF distribution, which we set to 5 based on initial experiments.

The quality of generated images is measured in two different metrics, Inception Score (IS) [33] and Frechet Inception Distance (FID 5k) [11]. IS measures the certainty in class prediction along with the diversity between different classes, while FID 5k quantifies the dissimilarity between activations of real and generated images. Both metrics are based on the ImageNet-pretrained Inception-v3 network [37].

The quantitative results in Table 5 show that our model outperforms the baseline, particularly in the IS metric. We argue that this improvement comes from the advantages of the learned prior $p(\mathbf{z}|i) = q(\bar{\mathbf{a}}|i)$ that allow the decoder to utilize the latent space more effectively while considering the affinity between classes and class-specific appearance variations, as discussed in section 4.2. The qualitative results in Figure 6 demonstrate that our method tends to generate images with more diverse instances and backgrounds while keeping their class identity.

5 DISCUSSION

Our core contribution is a simple generative model of activations that forms a generative-discriminative pair with the given classification network. We believe our approach provides insight into how even the design of a single layer can impose an inductive bias on a model that guides how and where knowledge is stored. We show that it is possible to exploit such structures to efficiently extract useful information from a trained model. While our derivation in section 2.1 is specific to the typical design of using matrix multiplication in the final layer (i.e., $\mathbf{l} = \mathbf{W}^\top \mathbf{a}$), it could be extended to analyze other multiplicative interactions [14], including Mahalanobis metric learning, gating mechanisms, and self-attention.

Our approach may be useful in many other applications. For example, beyond our mild domain shift setting in section 4.1.4, CKD may also have benefits in a more severe domain adaptation setting where one wishes to transfer information between different domains. Also, the density functions $q(\bar{\mathbf{a}}|i)$ may be used as a decision criterion for anomaly detection, and their relative overlap may help estimate or calibrate the uncertainty of classification networks. We believe these are all exciting directions for future research.

Author Contributions

Minkyo Seo and Yoonho Lee contributed equally to this paper.

This work was done while Minkyo Seo was visiting Kakao as a research intern.

Acknowledgements

This work was supported by Kakao, the NRF grant, and the IITP grant, funded by Ministry of Science and ICT, Korea (No.2019-0-01906 AIGS Program–POSTECH, NRF-2021R1A2C3012728–60%, IITP-2020-0-00842–40%).

References

[1] Sungsoo Ahn, Shell Xu Hu, Andreas Damianou, Neil D Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9163–9171, 2019.

[2] Arindam Banerjee, Inderjit S. Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *J. Mach. Learn. Res.*, 6:1345–1382, December 2005. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1046920.1088718`.

[3] Beidi Chen, Weiyang Liu, Animesh Garg, Zhiding Yu, Anshumali Shrivastava, and Animashree Anandkumar. Angular visual hardness. In *ICML*, 2020.

[4] Tim R Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M Tomczak. Hyperspherical variational auto-encoders. *arXiv preprint arXiv:1804.00891*, 2018.

[5] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. 2016.

[6] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[7] Xing Fan, Wei Jiang, Hao Luo, and Mengjuan Fei. Spherereid: Deep hypersphere manifold embedding for person re-identification. *Journal of Visual Communication and Image Representation*, 60:51–58, 2019.

[8] Siddharth Gopal and Yiming Yang. Von mises-fisher clustering models. In *International Conference on Machine Learning*, pages 154–162, 2014.

[9] Md Hasnat, Julien Bohné, Jonathan Milgram, Stéphane Gentric, Liming Chen, et al. von mises-fisher mixture model-based deep learning: Application to face verification. *arXiv preprint arXiv:1706.04264*, 2017.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[11] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.

[12] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network. *arXiv:1503.02531 [cs, stat]*, March 2015. URL `http://arxiv.org/abs/1503.02531`. arXiv: 1503.02531.

[13] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4700–4708, 2017.

[14] Siddhant M. Jayakumar, Jacob Menick, Wojciech M. Czarnecki, Jonathan Schwarz, Jack Rae, Simon Osindero, Yee Whye Teh, Tim Harley, and Razvan Pascanu. Multiplicative interactions and where to find them. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=rylnK6VtDH`.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[17] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[18] Sachin Kumar and Yulia Tsvetkov. Von mises-fisher loss for training sequence to sequence models with continuous outputs. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=rJlDnoA5Y7`.

[19] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[20] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, pages 7167–7177, 2018.

[21] Hao Liu, Xiangyu Zhu, Zhen Lei, and Stan Z Li. Adaptiveface: Adaptive margin and sampling for face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11947–11956, 2019.

[22] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Proc. European Conference on Computer Vision (ECCV)*, pages 122–138, 2018. ISBN 978-3-030-01264-9.

[23] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[24] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. *arXiv preprint arXiv:1802.05637*, 2018.

[25] Andrew Y Ng and Michael I Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in neural information processing systems*, pages 841–848, 2002.

[26] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 1520–1528, 2015.

[27] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2642–2651. JMLR. org, 2017.

[28] Changyong Oh, Kamil Adamczewski, and Mijung Park. Radial and directional posteriors for bayesian neural networks. *arXiv preprint arXiv:1902.02603*, 2019.

[29] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3967–3976, 2019.

[30] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. FitNets: Hints for Thin Deep Nets. *arXiv:1412.6550 [cs]*, December 2014. URL `http://arxiv.org/abs/1412.6550`. arXiv: 1412.6550.

[31] Y Dan Rubinstein, Trevor Hastie, et al. Discriminative vs informative learning. In *KDD*, volume 5, pages 49–53, 1997.

[32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, pages 1–42, April 2015.

[33] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[34] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. How good is my gan? In *ECCV*, pages 213–229, 2018.

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

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

[37] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826, 2016.

[38] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation, 2019.

[39] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1365–1374, 2019.

[40] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, July 2017.

[41] Hongxu Yin, Pavlo Molchanov, Jose M. Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *CVPR*, 2020.

[42] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[43] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

[44] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[45] Andrew Zhai and Hao-Yu Wu. Classification is a strong baseline for deep metric learning. In *Proc. British Machine Vision Conference (BMVC)*, 2019.

[46] 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.