A Unified Knowledge Distillation Framework for Deep Directed Graphical Models

Yizhuo Chen *
William and Mary & UIUC
yizhuoc@illinois.edu

Kaizhao Liang
UIUC
kl2@illinois.edu

Zhe Zeng
UCLA
zhezeng@cs.ucla.edu

Shuochao Yao
George Mason University
shuochao@gmu.edu

Huajie Shao †
William and Mary
hshao@wm.edu

Abstract

Knowledge distillation (KD) is a technique that transfers the knowledge from a large teacher network to a small student network. It has been widely applied to many different tasks, such as model compression and federated learning. However, existing KD methods fail to generalize to general deep directed graphical models (DGMs) with arbitrary layers of random variables. We refer by deep DGMs to DGMs whose conditional distributions are parameterized by deep neural networks. In this work, we propose a novel unified knowledge distillation framework for deep DGMs on various applications. Specifically, we leverage the reparameterization trick to hide the intermediate latent variables, resulting in a compact DGM. Then we develop a surrogate distillation loss to reduce error accumulation through multiple layers of random variables. Moreover, we present the connections between our method and some existing knowledge distillation approaches. The proposed framework is evaluated on four applications: data-free hierarchical variational autoencoder (VAE) compression, data-free variational recurrent neural networks (VRNN) compression, data-free Helmholtz Machine (HM) compression, and VAE continual learning. The results show that our distillation method outperforms the baselines in data-free model compression tasks. We further demonstrate that our method significantly improves the performance of KD-based continual learning for data generation. Our source code is available at https://github.com/YizhuoChen99/KD4DGM-CVPR.

1. Introduction

Knowledge distillation (KD) aims at transferring the knowledge of a large teacher model to a small student model, which tries to mimic the behavior of the teacher model to attain a competitive or superior performance [13, 20]. The goal of this work is to develop a *unified knowledge distillation (KD) framework* for *deep directed graphical models* (DGMs). Applications of the proposed framework include: (i) data-free hierarchical variational autoencoder (VAE) compression [50], (ii) data-free variational recurrent neural networks (VRNN) compression [8], (iii) data-free Helmholtz Machine (HM) compression [49], and (iv) KD based continual learning.

Deep directed graphical models (DGMs) refer to DGMs whose conditional distributions are parameterized by deep neural networks (DNNs), which is in contrast to the regular DGMs with tabular conditional probability. One good example is variational autoencoders (VAEs), whose posterior probability of latent variables is parameterized by DNNs. A general deep DGM may have a complex structure, consisting of an arbitrary number of input variables, target variables, and latent variables. Deep DGMs have been widely used in various applications, such as image generation [53], text generation [5], and video prediction [55].

This work is motivated by the growing popularity of recent over-parameterized deep DGMs with millions of parameters to improve their accuracy in various tasks. However, the large models are very computationally expensive. As a result, it is *not* practical to deploy them on resource-constrained edge devices, such as mobile phones and IoT systems [33]. One possible solution to this problem is KD, which enables a smaller student model to approximate the performance of a large teacher. Recently, KD has been widely applied to many different tasks, such as model compression [20], continual learning [34, 59], and federated learning [30, 36]. To our knowledge, the existing KD methods, however, are only applicable to some specific DGMs, including generative adversarial networks (GANs) [2, 33], auto-regressive models in natural language processing (NLP) [35], and VQ-VAE [48]. *They fail to generalize to the general deep DGMs, especially to those with multiple latent variables or complex*

*Work is completed during internship at William and Mary
†Corresponding author

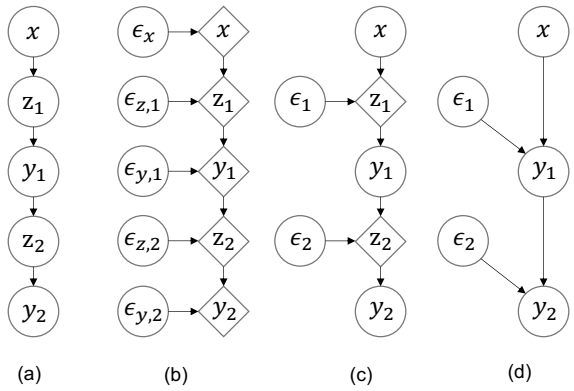

Figure 1. Toy example of DGM in four different forms. Diamonds are deterministic variables and circles are random variables. (a) Original form; (b) Auxiliary form; (c) Our semi-auxiliary form; (d) Compact semi-auxiliary form.

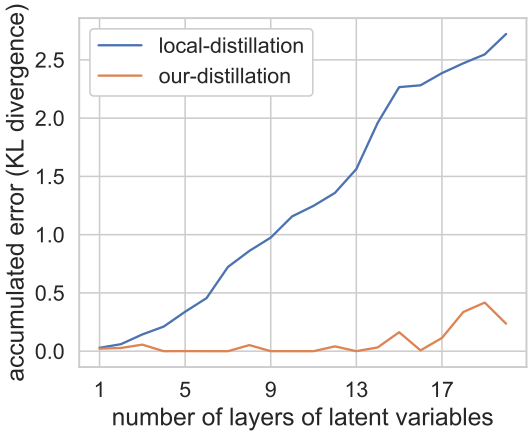

Figure 2. Toy example of accumulated error (KL divergence) between the teacher and student for local distillation and our method. Experimental settings are presented in the last paragraph in Section 3.2.

dependence structures, as illustrated in Fig. 1.

Generalizing knowledge distillation to deep DGMs poses two major challenges. First, distillation by marginalizing all latent variables is generally intractable (as explained in Appendix A). Secondly, distilling each layer locally and independently may suffer from error accumulation, as shown in Fig. 2. We can observe that the accumulated error (i.e., KL divergence) between the teacher and student grows linearly for local distillation. To address these challenges, we propose a novel unified knowledge distillation framework for deep DGMs. Specifically, we first adopt the reparameterization trick [23,24] to convert a DGM into a compact *semi-auxiliary form*. By *semi-auxiliary form*, we mean the latent variables, $\boldsymbol{z}$, in both the student and teacher models are converted into deterministic variables with auxiliary variables, while the input variables and target variables remain unchanged, as shown in Fig. 1 (c). Note that different from the classical reparameterization for VAE model training [25], ours can be applied to both continuous and discrete variables. Then a surrogate distillation loss is derived as a new objective of KD. To mitigate gradient vanishing, we further incorporate a latent distillation loss that penalizes the dissimilarity of latent variables between the teacher and student into our objective. We also present the connections between our approach and some existing KD methods for specific DGMs and show that our method is a proper generalization of these existing methods.

We evaluate the performance of our distillation method on four different tasks: hierarchical VAE compression, VRNN compression, Helmholtz Machine compression, and KD-based continual learning with VAEs. For model compression tasks, the student model distilled by our method in a data-free manner outperforms that trained from scratch and the other baselines. In addition to model compression, we also illustrate that our method can better mitigate the catastrophic forgetting issue than the generative replay approaches in continual learning.

In summary, our contributions include: 1) a new unified KD framework is proposed for general deep DGMs based on reparameterization trick, 2) we derive a novel distillation loss that combines the latent distillation loss and surrogate distillation loss to improve the performance of KD, and 3) evaluation results on multiple benchmark datasets show that our approach can not only achieve high accuracy for deep DGMs compression but also improve the performance of KD-based continual learning.

2. Preliminaries

• **Directed Graphical Models (DGMs)** DGMs such as Bayesian Networks [9] belong to an expressive class of probabilistic graphical models [26], in which the joint distribution is factorized into the product of many conditional distributions according to a directed acyclic graph (DAG) that captures variable conditional dependencies. In this work, we primarily study knowledge distillation for deep DGMs, especially for those with complicated dependency structures.

For deep DGMs, we are interested in modeling the conditional distribution $p_\theta(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x})$ for *target variables* $\boldsymbol{y}$ and *latent variables* $\boldsymbol{z}$ given *input variables* $\boldsymbol{x}$, parameterized by θ. Specifically, when there are no input variable, i.e., $\boldsymbol{x} = \emptyset$, we actually model the joint distribution $p_\theta(\boldsymbol{y}, \boldsymbol{z})$. Let $Pa(\cdot)$ denote the parent random variables of a certain variable defined by the DAG of a DGM. Then the conditional distribution $p_\theta(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x})$ has its factorized form below,

$$p_\theta(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x}) = \prod_j p_\theta(\boldsymbol{y}_j|Pa(\boldsymbol{y}_j), \boldsymbol{x}) \prod_i p_\theta(\boldsymbol{z}_i|Pa(\boldsymbol{z}_i), \boldsymbol{x}), \tag{1}$$

where $\boldsymbol{y}_j$ denotes the jth target variable in $\boldsymbol{y}$, $\boldsymbol{z}_i$ denotes the ith latent variable in $\boldsymbol{z}$. Without loss of generality, we

assume for two variables $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$, if $\boldsymbol{y}_j$ is an ancestor of $\boldsymbol{y}_i$, then it holds that $i > j$.

• **Knowledge Distillation (KD)** KD aims to transfer the knowledge of a large teacher model to a smaller student model. One commonly used vanilla distillation method is to encourage the student to mimic the output of the teacher model [20]. Given an empirical distribution of the training data $p_{data}(\boldsymbol{x})$, its distillation loss is an expected dissimilarity measure between the output of the student and that of the teacher. A general form of distillation loss is given by

$$\mathcal{L}_{kd} = \mathbb{E}_{p_{data}(\boldsymbol{x})}\left[d(p_\phi(\boldsymbol{y}|\boldsymbol{x}), p_\theta(\boldsymbol{y}|\boldsymbol{x}))\right], \tag{2}$$

where $p_\phi(\boldsymbol{y}|\boldsymbol{x})$ and $p_\theta(\boldsymbol{y}|\boldsymbol{x})$ denote the output conditional distribution of the teacher and student models, respectively. ϕ and θ denote their corresponding parameters. $d(\cdot,\cdot)$ is a dissimilarity measure between two probability distributions. Kullback-Leibler (KL) divergence [7, 20], for example, is one of some typical choices.

The above Eq. (2) and its extended version have been applied to different DGMs, such as vanilla neural networks [7], GANs [2], and fully-visible auto-regressive models (e.g., Transformer) [22, 35]. Take fully-visible auto-regressive models as an example. When we set $d(\cdot,\cdot)$ to KL divergence, Eq. (2) can be factorized as

$$\begin{aligned}\mathcal{L}_{kd} &= \sum_j \mathbb{E}_{p_{data}(\boldsymbol{x})}\mathbb{E}_{p_\phi(\boldsymbol{y}_{<j}|\boldsymbol{x})} KL_j(\boldsymbol{y}_{<j}, \boldsymbol{x}),\\ KL_j(\boldsymbol{y}_{<j}, \boldsymbol{x}) &= KL(p_\phi(\boldsymbol{y}_j|\boldsymbol{y}_{<j}, \boldsymbol{x}) \parallel p_\theta(\boldsymbol{y}_j|\boldsymbol{y}_{<j}, \boldsymbol{x})).\end{aligned} \tag{3}$$

A tractable estimation to $\mathcal{L}_{kd}$ above can be obtained by Monte Carlo method. It can also be viewed as distilling each conditional distribution in a local and independent manner, as shown in Fig. 3 (d).

However, to our best knowledge, the existing KD approaches are only designed for some specific DGMs. They fail to be applied to a general DGM with multiple latent variables or complicated dependence structures. Hence, the question is, how can we distill the knowledge from the teacher to the student given a general DGM structure?

Intuitively, there are two naive methods: (i) *marginalized distillation*: it marginalizes all latent variables to get $p(\boldsymbol{y}|\boldsymbol{x}) = \int p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x})\mathrm{d}\boldsymbol{z}$. However, the integration $\int p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x})\, d\boldsymbol{z}$ is generally intractable and thus the loss as in Eq. (2) is also intractable. (ii) *local distillation*: it treats latent variables $\boldsymbol{z}$ equally as target variables and then distills each conditional distribution for both $\boldsymbol{z}$ and $\boldsymbol{y}$ locally and independently, as shown in Fig. 3d. The local distillation may suffer from error accumulation through multiple layers if each conditional distribution in the student slightly deviates from the teacher. Fig. 2 shows a toy example that the accumulated error of local distillation, i.e., the KL-divergence between teacher and student, increases linearly as the number of layers of latent variables rises. For detailed discussions, please refer to Appendix A.

• **Reparameterization Trick** Reparameterization trick [23, 24], also called the auxiliary form of a DGM, is originally proposed to backpropagate through a random node that is not differentiable during training. Its basic idea is introduced as follows. Given a conditional distribution of random variable $\boldsymbol{z}_i$ in a DGM, $p(\boldsymbol{z}_i|Pa(\boldsymbol{z}_i), \boldsymbol{x})$, we convert it to a deterministic variable by adding an *auxiliary variable* $\boldsymbol{\epsilon}_i$ to its dependence, as shown in Figure 1 (c). Here $\boldsymbol{\epsilon}_i$ is a root node of the DGM with an independent marginal distribution of $p(\boldsymbol{\epsilon}_i)$. By choosing appropriate $p(\boldsymbol{\epsilon}_i)$ and deterministic transformation $g(\cdot)$ [23], we can have $\boldsymbol{z}_i = g(Pa(\boldsymbol{z}_i), \boldsymbol{x}, \boldsymbol{\epsilon}_i)$, where $\boldsymbol{z}_i$ is determined by its parent variables $Pa(\boldsymbol{z}_i)$, input variables $\boldsymbol{x}$, and the corresponding auxiliary variable $\boldsymbol{\epsilon}_i$. $\boldsymbol{\epsilon}_i$ serves as the source of stochasticity of $\boldsymbol{z}_i$.

In this work, note that we do not primarily use the reparameterization trick for model training. Rather, we leverage it to convert the latent variables $\boldsymbol{z}$ in DGMs to deterministic variables so that we can effectively distill knowledge from a compact form of DGM. Note that different from the classical reparameterization for model training that requires continuous latent variables, ours can be applied to a wider range of variables $\boldsymbol{z}$, including both continuous and discrete variables. Besides, the transformation function $g(\cdot)$ [25] in our framework can be either differentiable or non-differentiable. Hence, our method can be applied to much more DGMs than the classical one. Below, we will elaborate this general idea in more detail.

3. Modeling

In this section, we first introduce the semi-auxiliary form of DGMs using reparameterization trick. Then we propose a new surrogate loss function and latent distillation loss for our KD method.

3.1. Semi-auxiliary Form

As discussed in Section 2, the two naive methods, marginalized distillation and local distillation, do not work well due to intractable distillation loss or error accumulation. In order to address these issues, we propose a novel idea that converts DGM to its *semi-auxiliary form* based on reparameterization trick. Specifically, we convert all the latent variables, $\boldsymbol{z}$, in both the teacher and student to deterministic variables with auxiliary variables, while keeping target variables $\boldsymbol{y}$ and input variables $\boldsymbol{x}$ unchanged. This is because our ultimate goal is to encourage student to mimic the output (target variable) of the teacher based on input variables. Hence, we can omit the deterministic (latent) variables in a DGM, yielding a compact semi-auxiliary form that only consists of target variables, input variables and auxiliary variables. In this way, each target variable has a tractable and direct dependence on input variables or prior target variables, as shown in Fig. 1 (c).

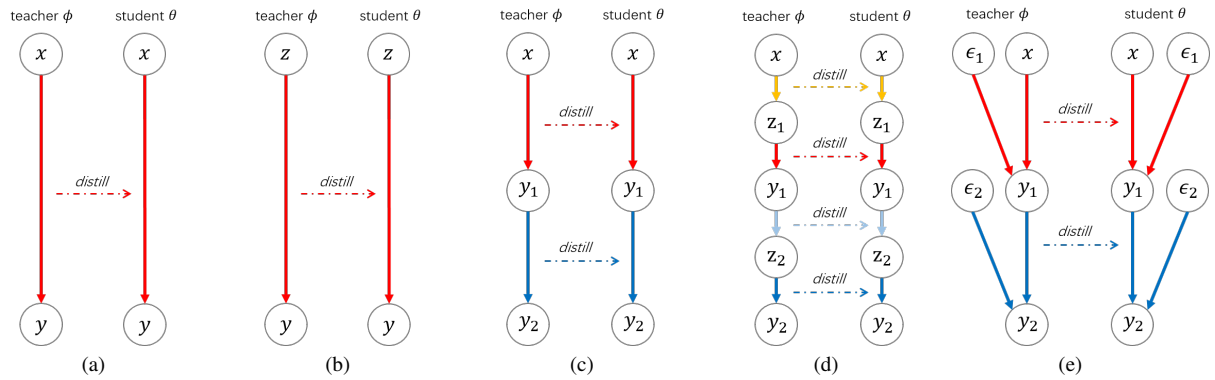

Figure 3. Examples of different distillation methods. Each pair of conditional distributions marked with same color represents an independent distillation component. (a) Distillation on vanilla neural network. (b) GAN distillation. (c) Distillation on a 2-layer fully-visible autoregressive DGM. (d) Local distillation on the original DGM. (e) Our distillation method with semi-auxiliary form.

Fig. 1 illustrates a toy example of converting DGM to its semi-auxiliary form. Fig. 1 (a) is the original form of a DGM, and its corresponding auxiliary form is shown in Fig. 1 (b). In this paper, we only assign auxiliary variables to latent variables $\boldsymbol{z}$, as shown in Fig. 1 (c). We can observe from it that latent variables $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are deterministic when their ancestors are given. We thus can omit the deterministic variables, leading to a compact semi-auxiliary form, as shown in Fig. 1 (d).

3.2. Surrogate Distillation Loss

After obtaining the semi-auxiliary form of a DGM in Fig. 1 (d), our distillation loss becomes the dissimilarity between $p_\phi(\boldsymbol{y}|\boldsymbol{\epsilon}, \boldsymbol{x})$ and $p_\theta(\boldsymbol{y}|\boldsymbol{\epsilon}, \boldsymbol{x})$. We call it *surrogate distillation loss*. Our goal is to minimize the surrogate distillation loss w.r.t. student's parameters θ below.

$$\mathcal{L}_{sd} = \mathbb{E}_{p_\phi(\boldsymbol{\epsilon})p_{data}(\boldsymbol{x})}\left[d(p_\phi(\boldsymbol{y}|\boldsymbol{\epsilon}, \boldsymbol{x}), p_\theta(\boldsymbol{y}|\boldsymbol{\epsilon}, \boldsymbol{x}))\right], \quad (4)$$

where $\boldsymbol{\epsilon}$ denotes a set of auxiliary variables ϵ. The expectation of dissimilarity is taken over both empirical data distribution $p_{data}(\boldsymbol{x})$ and auxiliary variable distribution $p_\phi(\boldsymbol{\epsilon})$. The expectation can be estimated using Monte Carlo method. Note that $p_\phi(\boldsymbol{\epsilon})$ is generally chosen to be simple and fixed distribution with no parameters, such as unit Gaussian or standard uniform distribution. Thus, it implies that teacher's $p_\phi(\boldsymbol{\epsilon})$ is equivalent to student's $p_\theta(\boldsymbol{\epsilon})$, so there is no need to distill $p_\phi(\boldsymbol{\epsilon})$ to $p_\theta(\boldsymbol{\epsilon})$. An illustration is given in Fig. 3e.

Proposition 3.1. *The surrogate distillation loss as defined in Eq. (4) is an upper bound of the distillation loss as defined in Eq. (2) when the dissimilarity measure is chosen to be KL divergence.*

We provide the detailed proof in Appendix B.

Next, we discuss the advantages of our method over the two naive methods mentioned above. Firstly, the proposed method bypasses the intractable computation in marginalized distillation. While marginalized distillation measures $p(\boldsymbol{y}|\boldsymbol{x})$ which is intractable in general, we instead measure $p(\boldsymbol{y}|\boldsymbol{\epsilon}, \boldsymbol{x})$ which is easily tractable by function composition with no need of integral or sum operation. Here $p(\boldsymbol{y}|\boldsymbol{\epsilon}, \boldsymbol{x})$ is tractable because $p(\boldsymbol{y}|\boldsymbol{\epsilon}, \boldsymbol{x}) = \prod_i p(\boldsymbol{y}_i|\boldsymbol{\epsilon}_{\leq i}, \boldsymbol{y}_{<i}, \boldsymbol{x})$.

Secondly, our method can make the DGMs shallower than local distillation, mitigating error accumulation through multiple layers of latent variables. As illustrated in Fig. 3d and 3e, we can see our method only constraints the target variables $\boldsymbol{y}$ in the student while local distillation constrains both latent variables $\boldsymbol{z}$ and target variables $\boldsymbol{y}$. As a result, local distillation may suffer from error accumulation issue. Fig. 2 shows a toy example of comparing the accumulated error, i.e., KL divergence between the teacher and student, for our method and local distillation. In this illustrative experiment, we let student mimic the output distribution of the teacher with L-layers latent variables for $L \in 1, \ldots, 20$. Each layer of the teacher follows the Gaussian distribution $p(\boldsymbol{z}_{i+1}|\boldsymbol{z}_i) = \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{z}_i), 0.01\boldsymbol{I})$, where $\boldsymbol{\mu}(\boldsymbol{z}_i)$ is $\boldsymbol{z}_i^{1.1}$ for $\boldsymbol{z}_i \geq 0$ and $-(-\boldsymbol{z}_i)^{1.1}$ for $\boldsymbol{z}_i < 0$. $p(\boldsymbol{z}_1)$ is a uniform distribution $U[-1, 1]$. The student is parameterized by neural networks with proper residual structure [18]. Then *density ratio estimation* [43, 51] is used to measure the KL divergence between the teacher and student. We can observe from Fig. 2 that the accumulated error (KL divergence) grows linearly w.r.t the number of layers for local distillation because each layer of the student deviates from the teacher to an extent. In contrast, the accumulated error (KL divergence) of our method increases slowly.

3.3. Latent Distillation Loss

The surrogate distillation loss in Eq.(4) can sufficiently achieve a satisfactory performance for knowledge distillation. Nevertheless, there are still two limitations of the proposed method for some special DGMs. Firstly, it fails to back-propagate the gradients when there exist discrete latent variables. Secondly, it might suffer from gradient vanishing when the network structure with multiple latent variables is very deep and complex.

To deal with these issues, we propose to penalize the dissimilarity of latent variables $\boldsymbol{z}$ in the teacher and student model. The resulting latent distillation loss for latent variables $\boldsymbol{z}_i$ is given by

$$\begin{aligned}\tilde{\mathcal{L}}_{z,i} &= \mathbb{E}_{p_\phi(\boldsymbol{\epsilon})p_{data}(\boldsymbol{x})}\left[d\left[p_\phi(\boldsymbol{z}_i|\boldsymbol{\epsilon},\boldsymbol{x}), p_\theta(\boldsymbol{z}_i|\boldsymbol{\epsilon},\boldsymbol{x})\right]\right] \\ &= \mathbb{E}_{p_\phi(\boldsymbol{\epsilon})p_{data}(\boldsymbol{x})}\left[d(p_\phi(\boldsymbol{z}_i|\boldsymbol{\epsilon}_{\leq i},\boldsymbol{x}), p_\theta(\boldsymbol{z}_i|\boldsymbol{\epsilon}_{\leq i},\boldsymbol{x}))\right],\end{aligned} \tag{5}$$

where $\boldsymbol{\epsilon}_{\leq i}$ is a set of all the ancestral auxiliary variables of $\boldsymbol{z}_i$. $\boldsymbol{z}_i$ is deterministic when $\boldsymbol{\epsilon}_{\leq i}$ and $\boldsymbol{x}$ are given. In order to better penalize the dissimilarity of latent variables $\boldsymbol{z}_i$, one good choice is to *solely convert the current* $\boldsymbol{z}_i$ (not $\boldsymbol{z}_{<i}$) back to its original form by removing $\boldsymbol{\epsilon}_i$ from its dependence. Then we have the following latent distillation loss

$$\mathcal{L}_{z,i} = \mathbb{E}_{p_\phi(\boldsymbol{\epsilon})p_{data}(\boldsymbol{x})}\left[d(p_\phi(\boldsymbol{z}_i|\boldsymbol{\epsilon}_{<i},\boldsymbol{x}), p_\theta(\boldsymbol{z}_i|\boldsymbol{\epsilon}_{<i},\boldsymbol{x}))\right]. \tag{6}$$

The proposed latent distillation loss can benefit our optimization process. When latent variables are continuous, the latent distillation loss may provide shallower and supplementary supervisory signals to hasten the convergence. When latent variables are discrete, it can deal with the back-propagation cutting off problem.

3.4. Final Target-Free Loss

Combining the above surrogate distillation loss with latent distillation loss, our final distillation loss is given by

$$\mathcal{L}_{our} = \mathcal{L}_{sd} + \lambda \sum_i \mathcal{L}_{z,i}, \tag{7}$$

where λ is a hyper-parameter that controls the importance of latent distillation loss. Similar to [20], our loss in Eq. (7) is a *target-free* distillation loss, which means target data is not required for calculating it. Eq. (7) can also be applied to DGMs with no input variables (i.e. $\boldsymbol{x} = \emptyset$). In this case, Eq. (7) can be computed in a completely *data-free* manner. We summarize our KD method for a general DGM in Algorithm 1 in Appendix C.

3.5. Connections to Other Distillation Methods

We present the connections between our method and some existing knowledge distillation methods.

• **Vanilla KD and Sequence-Level KD.** When there is no latent variable in a DGM, our distillation method in Eq. (4) is naturally reduced to Eq. (2) by removing the dependence on auxiliary variables $\boldsymbol{\epsilon}$, which is a typical vanilla KD [20]. Also, when the DGM is a fully visible auto-regressive model, by removing dependence on $\boldsymbol{\epsilon}$ and letting $d(\cdot,\cdot)$ be KL divergence, Eq. (4) can be reduced to Eq. (3), which is the Monte Carlo approximation of intractable sequence-level distillation loss [22]. Please note that further simplification has been made in [22] to enhance the practicability.

• **Feature Based KD.** Feature based knowledge distillation uses the intermediary representations of a teacher network to supervise a student network [47]. When the teacher and student share the same size of latent features, the feature distillation loss can be written as

$$\mathcal{L}_f = r_f(f_\phi(\boldsymbol{x}), f_\theta(\boldsymbol{x})), \tag{8}$$

where $f_\phi(\boldsymbol{x})$ and $f_\theta(\boldsymbol{x})$ are intermediary deterministic features of the teacher and student, respectively. $r_f(\cdot,\cdot)$ is a distance between two vectorized feature maps. A vanilla neural network with multiple intermediate features can be viewed as a DGM with multiple deterministic latent variables. Deterministic variables can be viewed as following the degenerate distributions. In general, for $p \geq 1$, p-Wasserstein distance $(\inf \mathbb{E}[r(\boldsymbol{a}_1,\boldsymbol{a}_2)^p])^{\frac{1}{p}}$ between two degenerate distributions located at $\mathbf{a}_1$ and $\mathbf{a}_2$ is equivalent to $r(\mathbf{a}_1,\mathbf{a}_2)$. Thus, by viewing the intermediate features as latent variables following degenerate distributions, and choosing $d(\cdot,\cdot)$ in our latent distillation loss $\mathcal{L}_{z,i}$ as Wasserstein distance $(\inf \mathbb{E}[r_f(\boldsymbol{z}_\phi,\boldsymbol{z}_\theta)^p])^{\frac{1}{p}}$, our latent distillation loss is reduced to feature distillation loss in Eq. (8).

• **GAN Distillation.** GAN distillation in [2, 33] also incorporates the idea of feature distillation into their model, which is given by

$$\mathcal{L}_{gan} = r_o(G_\phi(\boldsymbol{z}), G_\theta(\boldsymbol{z})) + r_f(f_\phi(\boldsymbol{z}), f_\theta(\boldsymbol{z})). \tag{9}$$

The first term above is the output distillation loss of the generator and the second one is the intermediary feature distillation loss. Similar to feature distillation loss, by viewing the intermediate features and generator output as latent and target variables following degenerate distributions, and choosing $d(\cdot,\cdot)$ in $\mathcal{L}_{sd}$ and $\mathcal{L}_{z,i}$ as p-Wasserstein distance $(\inf \mathbb{E}[r_o(\boldsymbol{y}_\phi,\boldsymbol{y}_\theta)^p])^{\frac{1}{p}}$ and $(\inf \mathbb{E}[r_f(\boldsymbol{z}_\phi,\boldsymbol{z}_\theta)^p])^{\frac{1}{p}}$ respectively, our final distillation loss in Eq. (7) can be reduced to GAN distillation loss as well.

3.6. Applications

We evaluate the performance of our method on four representative applications selected as below.

• **Data-Free Hierarchical VAE Compression**. Model compression is one of the most representative application of KD. Therefore, we first apply our method to compress a popular generative model, Hierarchical VAE in a data-free manner. Recent studies show that VAEs generate better with over-parametrization [53]. However, it is challenging to deploy them to edge devices with limited computing resources. In this work, we apply our method to compress a large 5-layer hierarchical VAE model [50] to smaller models, as illustrated in Fig. 4a.

• **Data-Free VRNN Compression**. Then our method is applied to compress a sequence generative model with more complicated structure, VRNN [11], as shown in Fig. 4b. VRNN has as many latent and target variables as the length of the sequence in dataset.

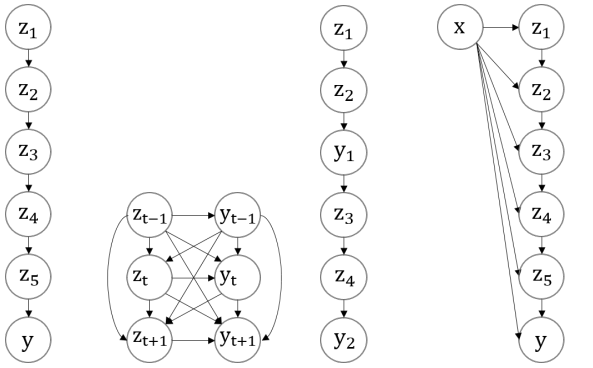

Figure 4. Structures of DGMs used in our experiments. (a) Hierarchical VAE. (b) VRNN. (c) HM. (d) Hierarchical VAE for continual learning.

• **Data-Free HM Compression.** Next, we adopt our method to compress a model with discrete latent variables, Helmholtz Machine (HM) [10]. The classical HM only has one target variable and each layer is parametrized by a linear transformation. In order to demonstrate the applicability of our method, we extend it to a 5-layer deep HM with two targets, as shown in Fig. 4c.

• **KD-based VAE Continual Learning.** In addition to model compression, we evaluate the performance of our method on another popular KD based application, continual learning [59]. Our goal is to model a new distribution while retaining the ability of modeling a learned old distribution without access to the old dataset. Prior work on continual learning [45, 57] mainly resort to generative replay strategy, in which we generate a set of fake samples from VAE model learned on old training datasets, and mix them with new training dataset to train the new VAE model. In fact, our experiments will show that generative replay is inferior to our distillation method for VAE continual learning because of the blurry nature of VAE generation.

4. Experiments

In this section, we first present a toy experiment to demonstrate that our method can also mitigate error accumulation issue for DGM with discrete variables. Then we evaluate the performance of our method on the four applications as mentioned in Section 3.6. We carry out extensive experiments on five benchmark datasets: Old Faithful Geyser [17], IAM online handwriting [38], SVHN [42], CIFAR10 [28], and CelebA [37]. The detailed data splitting for training and test is described in Appendix D. In addition, we present the detailed model configurations and hyperparameter settings of the teacher and student in Appendix E.

4.1. Toy Example for DGMs with Discrete Variables

In this experiment, we modify the illustrative toy experiment in Section 3.2 to demonstrate that, the error accumulation issue still exists in DGMs with discrete variables, and our method can still successfully mitigate it. Specifically, we change each layer of the teacher model in Section 3.2 to discrete variables following the binomial distribution $p(\boldsymbol{z}_{i+1}|\boldsymbol{z}_i) = \mathcal{B}(1000, \boldsymbol{p}(\boldsymbol{z}_i))$, where $\boldsymbol{p}(\boldsymbol{z}_i) = (\frac{\boldsymbol{z}_i}{1000})^{1.2}$. $p(\boldsymbol{z}_1)$ is changed to uniform distribution $U[0, 1]$. The experiment result is shown in Fig. 5. We can observe that, similar as its continuous counterpart, the accumulated error (KL divergence) grows as the number of layers increases for local distillation method. In contrast, our method can significantly reduce the accumulated error to a stable low level.

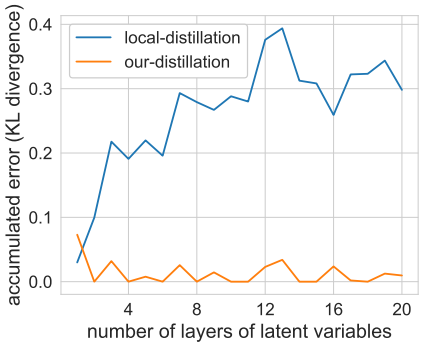

Figure 5. Toy example of accumulated error (KL divergence) between the teacher and student for local distillation and our method in a DMG with discrete random variables.

4.2. Evaluation on Data-Free Hierarchical VAE Compression.

Next, we apply our distillation method to compress large 5-layer hierarchical VAE [50] models. We compare the performance of the student model using our method and two baselines: training from scratch and local distillation [1]. Experiments are carried out on three benchmark datasets: SVHN, CIFAR10, and CelebA, with varying sizes of the student model. We adopt four widely-used metrics, Frechet Inception Distance (FID, lower is better) [6, 19], Earth Mover Distance (EMD, lower is better, also known as Wasserstein Distance) [56], Maximum Mean Discrepancy (MMD, lower is better) [16], and 1-Nearest-Neighbor Accuracy (1NN, lower is better) [39], to evaluate the generative performance of our method. To demonstrate the similarity of the teacher and student, we further calculate the FID, EMD, MMD and 1NN between the student and its corresponding teacher for different methods. Table 1 illustrates the comparison of different methods averaged over three random seeds on SVHN, Cifar10, and CelebA datasets. We can observe that our method consistently outperforms the baselines in all the metrics, which means the proposed method can help students better imitate their teachers. Importantly, our data-free distillation method outperforms the student model trained from scratch, which suggests that directly optimizing a capacity-limited student VAE may not learn a decent model that can generate high-fidelity samples. Conversely, our method can

help the student VAE learn better performance even without accessing training data.

Moreover, we demonstrate that our method is stable and robust to the change of hyper-parameter λ in Fig. 7 in the appendix. For qualitative evaluation, we also show the samples generated from different methods on CelebA in Fig. 6 in Appendix F.

4.3. Evaluation on Data-free VRNN Compression & HM Compression.

Next, we evaluate the performance of our method on the other two tasks: data-free VRNN Compression and data-free HM Compression. First, we adopt our distillation method to compress a complicated deep sequence generative model, VRNN. Following the prior works [8, 14], we adopt qualitative measure to show the generated strokes by our method and the baselines in Fig 8 in Appendix G. It can be observed that our method can generate more readable and clearer strokes than the baselines. The student VRNN distilled by our method in Fig 8 (c) has comparable generative performance to that of the teacher in Fig 8 (b). Hence, we can conclude that the proposed method is able to achieve good performance for data-free VRNN compression.

In addition, our method is applied to compress Helmholtz Machine (HM) with *discrete latent variables*. The experiment results, as illusrated in Fig. 7 in Appendix H, show that it can still achieve good performance as the teacher for HM compression. For more details, please refer to the experimental results in Appendix H.

4.4. Evaluation on KD-based Continual Learning

We also evaluate the performance of the proposed method on KD-based VAE continual learning using CelebA dataset. For each group of experiments, one of forty ground-truth attributes is selected. Based on this attribute, we divide the whole dataset into two parts for continual learning. Specifically, we first train a VAE M_{old} on the first part of images, and then we learn another VAE M_{new} on the second part by jointly minimizing a *new distribution standard training loss* L_{new} and an *old knowledge preserving loss* L_{pre} under the guidance of M_{old}. We conduct the experiment for three different attributes.

We compare our method with local distillation and other two generative replay approaches: CURL [46] and LGM [45]. All 4 methods have the same L_{new} but different L_{pre}. Table 2 illustrates the comparison of different methods on CelebA. We can observe from Columns Old that our method performs much better than the baselines on preserving old knowledge. It is because local distillation method has error accumulation issue, and CURL and LGM can only preserve old knowledge by retraining on the generated low-quality images. Besides, we can see from Columns New that our method is comparable to LGM and CURL on learning new distribution, because these methods only impose necessary L_{pre}. However, local distillation does not work well on new distributions, indicating that its L_{pre} is too large yet ineffective.

5. Related Work

In the past few years, there is a large amount of work on knowledge distillation (KD) for different DGMs. Most of existing KD approaches focus on vanilla DNNs or multi-target DNNs [7, 20, 40, 47, 58]. For instance, Chen et al. [7] proposed a weighted cross entropy loss for multi-class object detection models using KD. Zagoruyko et al. [40] developed an attention based mechanism to transfer knowledge from the teacher CNN to the student for image recognition tasks. Tf-FD [31] is a self-feature-distillation method consisting of intra-layer (from prominent features to redundant features) and inter-layer (from deep semantic-rich features to shallow features) distillation. SHAKE [32] proposed to exploit the benefit of mutual distillation at a low computational cost by introducing shadow head(s) for (multiple) teacher(s). The models considered in these KD methods can be viewed as DGMs with only one stochastic layer, consisting of one input variable and one or multiple conditionally independent target variables.

Some researchers also study KD for auto-regressive models without latent variables [21, 35, 44, 52]. Among them, sequence-level knowledge distillation (SeqKD) [22] is a promising strategy that supervises student with the teacher's sequence distribution over the space of all possible sequences. However, these methods do not generalize well to a general DGM with latent variables.

Besides, recent studies apply KD to compress deep generative models with one layer of latent variable. To reduce the number of parameters used in GANs, researchers [2, 33] devised new knowledge distillation methods for compressing GANs. [29] distilled the learned representation from VAE models to GAN for high-fidelity synthesis. [48] leveraged VQ-VAE with KD to develop a non-autoregressive machine translation model. However, these works only consider models with one layer of latent variable.

In summary, existing KD methods are mainly focused on specific DGMs, but fail to generalize to the general deep DGMs, especially to those with multiple layers of random variables or complex dependence structures. Different from prior work, we developed a novel unified KD framework for a general deep DGM using reparameterization trick. In fact, our method serves as a bridge, which can help previous advanced KD methods generalize to more general DGMs.

6. Conclusion

This paper proposed a new unified KD framework for deep directed graphical models (DGMs). Specifically, we

Table 1. Comparison of different methods for hierarchical VAE compression on SVHN, Cifar10 and CelebA datasets. The results are averaged over 3 different random seeds. XXX-T means that this metric XXX is calculated between the student and its corresponding teacher.

dataset	method	#param	FID (↓)	EMD (↓)	MMD (↓)	1NN (↓)	FID-T (↓)	EMD-T (↓)	MMD-T (↓)	1NN-T (↓)
CelebA	teacher	6.60M	4.95	8.54	0.24	0.89	-	-	-	-
	our	0.44M	**5.38 ± 0.10**	**8.77 ± 0.06**	**0.27 ± 0.01**	**0.92 ± 0.01**	**0.019 ± 0.002**	**6.48 ± 0.04**	**0.12 ± 0.00**	**0.17 ± 0.01**
	local	0.44M	6.23 ± 0.17	9.25 ± 0.11	0.33 ± 0.01	0.95 ± 0.00	0.052 ± 0.006	8.32 ± 0.14	0.26 ± 0.02	0.82 ± 0.02
	scratch	0.44M	6.10 ± 0.31	9.08 ± 0.16	0.33 ± 0.02	0.95 ± 0.01	0.052 ± 0.016	8.34 ± 0.36	0.26 ± 0.05	0.82 ± 0.05
	our	0.12M	**5.96 ± 0.12**	**9.06 ± 0.09**	**0.31 ± 0.01**	**0.95 ± 0.00**	**0.036 ± 0.005**	**8.04 ± 0.11**	**0.23 ± 0.01**	**0.79 ± 0.02**
	local	0.12M	8.95 ± 0.19	11.24 ± 0.16	0.50 ± 0.01	0.99 ± 0.00	0.157 ± 0.018	10.82 ± 0.17	0.47 ± 0.01	0.99 ± 0.00
	scratch	0.12M	8.18 ± 0.15	10.50 ± 0.14	0.45 ± 0.01	0.99 ± 0.00	0.095 ± 0.007	9.98 ± 0.03	0.43 ± 0.00	0.97 ± 0.00
	our	0.04M	**8.20 ± 0.12**	**10.66 ± 0.12**	**0.45 ± 0.01**	**0.99 ± 0.00**	**0.069 ± 0.004**	**9.91 ± 0.06**	**0.40 ± 0.00**	**0.98 ± 0.00**
	local	0.04M	11.08 ± 0.27	12.79 ± 0.22	0.62 ± 0.01	1.00 ± 0.00	0.139 ± 0.015	12.80 ± 0.28	0.64 ± 0.01	1.00 ± 0.00
	scratch	0.04M	9.57 ± 0.14	11.46 ± 0.11	0.55 ± 0.01	1.00 ± 0.00	0.093 ± 0.004	11.19 ± 0.13	0.56 ± 0.02	1.00 ± 0.00
SVHN	teacher	5.39M	4.19	7.98	0.17	0.80	-	-	-	-
	our	0.10M	**4.38 ± 0.05**	**7.94 ± 0.04**	**0.19 ± 0.00**	**0.81 ± 0.00**	**0.028 ± 0.006**	**6.90 ± 0.05**	**0.14 ± 0.01**	**0.47 ± 0.02**
	local	0.10M	5.93 ± 0.50	8.66 ± 0.32	0.30 ± 0.04	0.95 ± 0.02	0.108 ± 0.019	9.29 ± 0.41	0.40 ± 0.04	0.98 ± 0.01
	scratch	0.10M	4.69 ± 0.16	8.04 ± 0.12	0.21 ± 0.01	0.85 ± 0.01	0.037 ± 0.006	7.86 ± 0.10	0.22 ± 0.01	0.80 ± 0.02
	our	0.03M	**4.81 ± 0.06**	**8.10 ± 0.03**	**0.22 ± 0.01**	**0.87 ± 0.01**	**0.031 ± 0.012**	**7.82 ± 0.08**	**0.23 ± 0.01**	**0.82 ± 0.03**
	local	0.03M	6.95 ± 0.35	9.40 ± 0.28	0.37 ± 0.02	0.98 ± 0.01	0.153 ± 0.017	10.39 ± 0.21	0.49 ± 0.02	1.00 ± 0.00
	scratch	0.03M	5.84 ± 0.32	8.62 ± 0.18	0.31 ± 0.02	0.92 ± 0.01	0.080 ± 0.009	9.10 ± 0.22	0.39 ± 0.03	0.95 ± 0.01
	our	0.01M	**6.71 ± 0.38**	**9.22 ± 0.31**	**0.36 ± 0.03**	**0.96 ± 0.01**	**0.055 ± 0.011**	**9.13 ± 0.11**	**0.37 ± 0.02**	**0.98 ± 0.00**
	local	0.01M	8.26 ± 0.37	10.40 ± 0.35	0.45 ± 0.02	1.00 ± 0.00	0.170 ± 0.015	11.25 ± 0.25	0.55 ± 0.01	1.00 ± 0.00
	scratch	0.01M	7.73 ± 0.28	9.95 ± 0.25	0.43 ± 0.01	0.99 ± 0.00	0.063 ± 0.015	10.22 ± 0.18	0.49 ± 0.01	0.99 ± 0.00
Cifar10	teacher	5.39M	4.63	7.57	0.25	0.89	-	-	-	-
	our	0.10M	**5.47 ± 0.23**	**8.16 ± 0.20**	**0.31 ± 0.02**	**0.92 ± 0.01**	**0.024 ± 0.006**	**6.29 ± 0.08**	**0.19 ± 0.01**	**0.48 ± 0.01**
	local	0.10M	6.22 ± 0.05	8.61 ± 0.06	0.37 ± 0.00	0.94 ± 0.00	0.036 ± 0.003	7.58 ± 0.08	0.31 ± 0.01	0.91 ± 0.01
	scratch	0.10M	6.19 ± 0.25	8.54 ± 0.15	0.38 ± 0.02	0.95 ± 0.01	0.034 ± 0.005	7.27 ± 0.12	0.28 ± 0.03	0.83 ± 0.03
	our	0.03M	**6.11 ± 0.16**	**8.59 ± 0.11**	**0.36 ± 0.01**	**0.95 ± 0.01**	**0.036 ± 0.013**	**7.29 ± 0.11**	**0.28 ± 0.01**	**0.82 ± 0.02**
	local	0.03M	7.55 ± 0.09	9.66 ± 0.05	0.45 ± 0.00	0.97 ± 0.00	0.052 ± 0.003	9.08 ± 0.07	0.44 ± 0.01	0.99 ± 0.00
	scratch	0.03M	6.74 ± 0.36	8.95 ± 0.27	0.42 ± 0.03	0.96 ± 0.01	0.037 ± 0.007	7.84 ± 0.29	0.35 ± 0.03	0.93 ± 0.02
	our	0.01M	**7.61 ± 0.91**	**9.65 ± 0.79**	**0.47 ± 0.05**	**0.98 ± 0.01**	**0.045 ± 0.016**	**8.48 ± 0.78**	**0.42 ± 0.05**	**0.96 ± 0.02**
	local	0.01M	10.53 ± 0.74	12.10 ± 0.71	0.64 ± 0.03	1.00 ± 0.00	0.085 ± 0.015	11.38 ± 0.69	0.62 ± 0.02	1.00 ± 0.00
	scratch	0.01M	10.17 ± 1.13	11.67 ± 1.01	0.64 ± 0.06	1.00 ± 0.00	0.067 ± 0.023	10.56 ± 0.99	0.61 ± 0.05	1.00 ± 0.00

Table 2. Comparison of different methods after learning new distributions on CelebA dataset. Columns Old show the measure results between the generated images and real images on old distribution while Columns New show measure results between generated images and real images on new distributions using four different metrics. A(±B) denotes that the metric is increased or decreased by B to A after learning new distributions.

attribute	method	FID (↓)		EMD (↓)		MMD (↓)		1NN (↓)	
		Old	New	Old	New	Old	New	Old	New
Female → Male	our	**4.39(-0.25)**	4.19	**7.76(-0.13)**	7.58	**0.23(-0.02)**	0.21	**0.89(-0.01)**	0.84
	local	4.73(+0.09)	4.25	7.93(+0.03)	7.58	0.25(+0.01)	0.21	0.93(+0.02)	0.86
	CURL	5.57(+0.99)	4.18	8.37(+0.53)	7.60	0.32(+0.08)	0.21	0.96(+0.05)	0.84
	LGM	5.56(+0.92)	4.12	8.37(+0.48)	7.51	0.32(+0.07)	0.20	0.96(+0.06)	0.83
No Beard → Beard	our	**4.27(-0.15)**	4.05	**7.68(-0.07)**	7.42	**0.21(-0.01)**	0.20	**0.89(+0.01)**	0.85
	local	4.63(+0.21)	4.42	7.86(+0.11)	7.56	0.24(+0.02)	0.23	0.90(+0.03)	0.90
	CURL	5.24(+0.81)	4.14	8.16(+0.40)	7.44	0.29(+0.07)	0.21	0.93(+0.06)	0.86
	LGM	5.19(+0.77)	4.12	8.13(+0.38)	7.44	0.29(+0.06)	0.20	0.93(+0.06)	0.88
No Eyeglasses → Eyeglasses	our	**4.43(-0.02)**	4.50	**7.75(-0.03)**	7.85	**0.23(-0.00)**	0.24	**0.87(-0.01)**	0.88
	local	6.51(+2.06)	4.56	8.89(+1.11)	7.79	0.37(+0.14)	0.24	0.98(+0.09)	0.92
	CURL	5.31(+0.85)	4.76	8.22(+0.44)	7.98	0.30(+0.07)	0.25	0.93(+0.04)	0.90
	LGM	5.28(+0.83)	4.73	8.21(+0.43)	7.96	0.29(+0.06)	0.25	0.93(+0.05)	0.90

first adopted reparametrization trick to convert latent variables into deterministic variables with auxiliary variables, resulting in a compact *semi-auxiliary form* of DGM. Then a novel objective that combines the surrogate distillation loss and latent distillation loss was proposed to improve the performance of KD. We further illustrated that our framework is a proper generalization of some existing KD methods. We evaluated the performance of our method on different tasks. The results showed that it can better compress hierarchical VAE, VRNN, HM models in a data-free manner than the baselines. Furthermore, our method can better mitigate the catastrophic forgetting issue in KD based continual learning of VAEs. Finally, we discussed the limitations of the proposed method and future work in Appendix I.

References

[1] Alessandro Achille, Tom Eccles, Loic Matthey, Christopher P Burgess, Nick Watters, Alexander Lerchner, and Irina Higgins. Life-long disentangled representation learning with cross-domain latent homologies. *arXiv preprint arXiv:1808.06508*, 2018. 6

[2] Angeline Aguinaldo, Ping-Yeh Chiang, Alex Gain, Ameya Patil, Kolten Pearson, and Soheil Feizi. Compressing gans using knowledge distillation. *arXiv preprint arXiv:1902.00159*, 2019. 1, 3, 5, 7

[3] Jörg Bornschein and Yoshua Bengio. Reweighted wake-sleep. *arXiv preprint arXiv:1406.2751*, 2014. 15

[4] Jorg Bornschein, Samira Shabanian, Asja Fischer, and Yoshua Bengio. Bidirectional helmholtz machines. In *International Conference on Machine Learning*, pages 2511–2519. PMLR, 2016. 15

[5] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015. 1

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 6

[7] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. *Advances in neural information processing systems*, 30, 2017. 3, 7

[8] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28:2980–2988, 2015. 1, 7

[9] Adnan Darwiche. *Modeling and reasoning with Bayesian networks*. Cambridge university press, 2009. 2

[10] Peter Dayan, Geoffrey E Hinton, Radford M Neal, and Richard S Zemel. The helmholtz machine. *Neural computation*, 7(5):889–904, 1995. 6

[11] Otto Fabius and Joost R Van Amersfoort. Variational recurrent auto-encoders. *arXiv preprint arXiv:1412.6581*, 2014. 5

[12] Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. Sequential neural models with stochastic layers. *arXiv preprint arXiv:1605.07571*, 2016. 15

[13] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021. 1

[14] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013. 7

[15] Karol Gregor, Ivo Danihelka, Andriy Mnih, Charles Blundell, and Daan Wierstra. Deep autoregressive networks. In *International Conference on Machine Learning*, pages 1242–1250. PMLR, 2014. 15

[16] Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola. A kernel method for the two-sample-problem. *Advances in neural information processing systems*, 19, 2006. 6

[17] Wolfgang Karl Härdle et al. *Smoothing techniques: with implementation in S*. Springer Science & Business Media, 1991. 6, 13

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4, 12, 14

[19] 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. *Advances in neural information processing systems*, 30, 2017. 6

[20] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1, 3, 5, 7

[21] Mingkun Huang, Yongbin You, Zhehuai Chen, Yanmin Qian, and Kai Yu. Knowledge distillation for sequence model. In *Interspeech*, pages 3703–3707, 2018. 7

[22] Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*, 2016. 3, 5, 7

[23] Diederik P Kingma. Fast gradient-based inference with continuous latent variable models in auxiliary form. *arXiv preprint arXiv:1306.0733*, 2013. 2, 3

[24] Diederik 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. 2, 3

[25] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2, 3, 17

[26] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009. 2

[27] Rahul G Krishnan, Uri Shalit, and David Sontag. Deep kalman filters. *arXiv preprint arXiv:1511.05121*, 2015. 15

[28] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6, 13

[29] Wonkwang Lee, Donggyun Kim, Seunghoon Hong, and Honglak Lee. High-fidelity synthesis with disentangled representation. In *European Conference on Computer Vision*, pages 157–174. Springer, 2020. 7

[30] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019. 1

[31] Lujun Li. Self-regulated feature learning via teacher-free feature distillation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI*, pages 347–363. Springer, 2022. 7

[32] Lujun Li and Zhe Jin. Shadow knowledge distillation: Bridging offline and online knowledge transfer. In *Advances in Neural Information Processing Systems*. 7

[33] Muyang Li, Ji Lin, Yaoyao Ding, Zhijian Liu, Jun-Yan Zhu, and Song Han. Gan compression: Efficient architectures for interactive conditional gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5284–5294, 2020. 1, 5, 7

[34] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 1

[35] Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. Autoregressive knowledge distillation through imitation learning. *arXiv preprint arXiv:2009.07253*, 2020. 1, 3, 7

[36] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *arXiv preprint arXiv:2006.07242*, 2020. 1

[37] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. 6, 13

[38] Marcus Liwicki and Horst Bunke. Iam-ondb-an on-line english sentence database acquired from handwritten text on a whiteboard. In *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, pages 956–961. IEEE, 2005. 6, 13

[39] David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545*, 2016. 6

[40] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? *arXiv preprint arXiv:1906.02629*, 2019. 7

[41] Radford M Neal. Connectionist learning of belief networks. *Artificial intelligence*, 56(1):71–113, 1992. 15

[42] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 6, 13

[43] XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861, 2010. 4, 12

[44] Aaron Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George Driessche, Edward Lockhart, Luis Cobo, Florian Stimberg, et al. Parallel wavenet: Fast high-fidelity speech synthesis. In *International conference on machine learning*, pages 3918–3926. PMLR, 2018. 7

[45] Jason Ramapuram, Magda Gregorova, and Alexandros Kalousis. Lifelong generative modeling. *Neurocomputing*, 404:381–400, 2020. 6, 7

[46] Dushyant Rao, Francesco Visin, Andrei A Rusu, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Continual unsupervised representation learning. *arXiv preprint arXiv:1910.14481*, 2019. 7, 14

[47] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014. 5, 7

[48] Aurko Roy, Ashish Vaswani, Arvind Neelakantan, and Niki Parmar. Theory and experiments on vector quantized autoencoders. *arXiv preprint arXiv:1805.11063*, 2018. 1, 7

[49] Lawrence K Saul, Tommi Jaakkola, and Michael I Jordan. Mean field theory for sigmoid belief networks. *Journal of artificial intelligence research*, 4:61–76, 1996. 1, 15

[50] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. *Advances in neural information processing systems*, 29:3738–3746, 2016. 1, 5, 6, 14

[51] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. Density-ratio matching under the bregman divergence: a unified framework of density-ratio estimation. *Annals of the Institute of Statistical Mathematics*, 64(5):1009–1044, 2012. 4, 12

[52] Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. Multilingual neural machine translation with knowledge distillation. *arXiv preprint arXiv:1902.10461*, 2019. 7

[53] Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. *arXiv preprint arXiv:2007.03898*, 2020. 1, 5

[54] Eszter Vértes and Maneesh Sahani. Flexible and accurate inference and learning for deep generative models. *arXiv preprint arXiv:1805.11051*, 2018. 15

[55] Bohan Wu, Suraj Nair, Roberto Martin-Martin, Li Fei-Fei, and Chelsea Finn. Greedy hierarchical variational autoencoders for large-scale video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2318–2328, 2021. 1

[56] Qiantong Xu, Gao Huang, Yang Yuan, Chuan Guo, Yu Sun, Felix Wu, and Kilian Weinberger. An empirical study on evaluation metrics of generative adversarial networks. *arXiv preprint arXiv:1806.07755*, 2018. 6

[57] Fei Ye and Adrian G Bors. Learning latent representations across multiple data domains using lifelong vaegan. In *European Conference on Computer Vision*, pages 777–795. Springer, 2020. 6

[58] 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. 7

[59] Mengyao Zhai, Lei Chen, Frederick Tung, Jiawei He, Megha Nawhal, and Greg Mori. Lifelong gan: Continual learning for conditional image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2759–2768, 2019. 1, 6