6-2022

Generative flows with invertible attentions

Rhea Sanjay SUKTHANKER

Zhiwu HUANG
Singapore Management University, zwhuang@smu.edu.sg

Suryansh KUMAR

Radu TIMOFTE

Luc VAN GOOL

Generative Flows with Invertible Attentions

Rhea Sanjay Sukthanker[1], Zhiwu Huang[1,2], Suryansh Kumar[1], Radu Timofte[1], Luc Van Gool[1,3]
[1]CVL, ETH Zürich, Switzerland [2]SAVG, SMU, Singapore [3]PSI, KU Leuven, Belgium
srhea@alumni.ethz.ch {zhiwu.huang,sukumar,radu.timofte,vangool}@vision.ee.ethz.ch

Abstract

Flow-based generative models have shown an excellent ability to explicitly learn the probability density function of data via a sequence of invertible transformations. Yet, learning attentions in generative flows remains understudied, while it has made breakthroughs in other domains. To fill the gap, this paper introduces two types of invertible attention mechanisms, i.e.*, map-based and transformer-based attentions, for both unconditional and conditional generative flows. The key idea is to exploit a masked scheme of these two attentions to learn long-range data dependencies in the context of generative flows. The masked scheme allows for invertible attention modules with tractable Jacobian determinants, enabling its seamless integration at any positions of the flow-based models. The proposed attention mechanisms lead to more efficient generative flows, due to their capability of modeling the long-term data dependencies. Evaluation on multiple image synthesis tasks shows that the proposed attention flows result in efficient models and compare favorably against the state-of-the-art unconditional and conditional generative flows.*

1. Introduction

Deep generative models have shown their capability to model complex real-world datasets for various applications, such as image synthesis [10, 15, 26, 42, 45], image super-resolution [29, 53], facial manipulation [7, 9, 19, 38], autonomous driving [50, 60], and others. The widely studied modern generative models include generative adversarial nets (GANs) [3, 15, 23, 56], variational autoencoders (VAEs) [26, 36, 46, 55], autoregressive models [47, 48] and flow-based models [10, 11, 24]. The GAN models implicitly learn the data distribution to produce samples by transforming a noise distribution into the desired space, where the generated data can approximate the real data distribution. On the other hand, VAEs optimize a lower bound on the data's log-likelihood, leading to a suitable approximation of the actual data distribution. Although these two models have achieved great success, neither provides exact data likelihood.

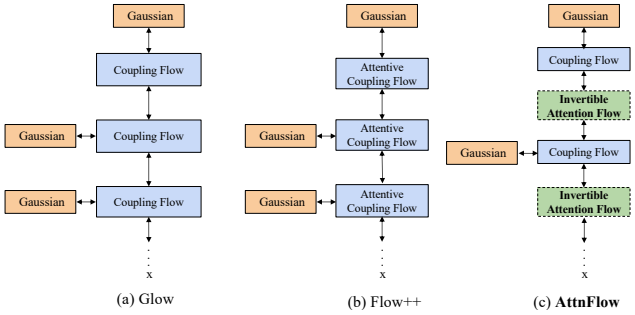


Figure 1. Conceptual comparison of the proposed AttnFlow against two representative generative flows, *i.e.*, (a) Glow [24] and (b) Flow++ [17]. Based on [24], Flow++ introduces the conventional attention mechanism to model short-term dependencies within one split of each feature map in the context of coupling layers. In contrast, the proposed AttnFlow (shown in (c)) further introduces invertible attention mechanisms that can be introduced at any flow positions to learn long-term correlations.

Autoregressive models [12, 47, 48] and flow-based generative models [10, 11, 24] optimize the exact log-likelihood of real data. Despite autoregressive models' better performance on density estimation benchmarks, its sequential property results in non-trivial parallelization. In contrast, the flow-based generative models are conceptually attractive due to tractable log-likelihood, exact latent-variable inference, and parallelizability of both training and synthesis. Notably, they allow exact inference of the actual data log-likelihood via normalizing flow. As shown in Fig.1(a), the normalizing flow model transforms a simple distribution into a complex one by applying a sequence of invertible transformation functions, which leads to an excellent mechanism of simultaneous exact log-likelihood optimization and latent-variable inference. However, due to efficiency constraints in their network designs, most models require several flow layers to approximate non-linear long-range data dependencies to get globally coherent samples. To overcome this drawback *i.e.*, modeling dependencies efficiently over normalizing flows is the key, and presently one of the most sought-after problems [17, 35].

To efficiently model data dependencies in the flow-based generative models, one may opt to combine multi-scale au-

toregressive priors [35]. By comparison, exploiting attention mechanisms has emerged as a remarkable way to model such dependencies in deep neural networks. It imitates human brain actions of selectively concentrating on a few relevant information while ignoring less correlated ones. Traditional self-attention mechanisms like [49, 52, 59] exhibit a good balance between the ability to model range dependencies and the computational and statistical efficiency. In general, the self-attention modules measure the response at a point as a weighted sum of the features at all points, where the attention weights are computed at a small computational cost. Although [17] recently applied the conventional attention directly as a dependent component in the coupling layer (Fig.1(b)), it models dependency within a short-range (*i.e.*, one split of each flow feature map) of the coupling layer. To our knowledge, efficient modeling of data dependencies over normalizing flows is understudied. A natural solution is to exploit new attention mechanisms to learn correlations of the feature maps at any positions of the flow-based models. However, it is generally non-trivial to achieve that goal of exploiting new attention modules as independent flow layers. Concretely, such attentions should maintain the invertibility with tractable Jacobian determinants in the flows.

In this paper, we propose invertible attentions for flow (AttnFlow) models to reliably and efficiently model network data dependencies that can be introduced at any positions of the flow-based models (along the entire flow feature maps, see Fig.1(c)). The key idea is to exploit a masked attention learning scheme to allow for intertible attention learning for normalizing flow based generative models. In addition, the proposed masked attention scheme facilitates tractable Jacobian determinants and hence can be integrated seamlessly into any generative flow models. Particularly, we exploit two different invertible attention mechanisms to encode the various types of correlations respectively on the flow feature maps. The two proposed attention mechanisms are ***(i)*** *invertible map-based (iMap) attention* that directly models the importance of each position in the attention dimension of the flow feature maps, ***(ii)*** *invertible transformer-based (iTrans) attention* that explicitly models the second-order interactions among distant positions in the attention dimension. Since the proposed two invertible attention modules explicitly model the dependencies of flow feature maps, it further enhances a flow-based model's efficiency to represent the deep network dependencies. To show the superiority of our approach, we evaluated the introduced attention models in the context of both unconditional and conditional normalizing flow-based generative models for multiple image synthesis tasks.

2. Related Work

Generative Flows. Early flow-based generative models like [10, 11, 25] are introduced for exact inference of real data log-likelihood. They are generally constructed by a sequence of invertible transformations to map a base distribution to a complex one.

Lately, several *unconditional generative flow* models have emerged that extends the early flow models to multi-scale architectures with split couplings that allow for efficient inference and sampling [4, 17, 24, 35]. For instance, [24] introduces invertible 1×1 convolutions to encode non-linearity in the data distribution for the unconditional setup. [18] introduces more general $d \times d$ invertible convolutions to enlarge the receptive field. [4] exploits residual blocks of flow layers (*i.e.*, a flexible family of transformations) where only Lipschitz conditions are used for enforcing invertibility. [17] improves the coupling layer with variational dequantization, continuous mixture cumulative distribution function, and self-attention. The self-attention is applied directly to the intrinsic neural function of the coupling layer. Because of the nature of the affine coupling layer, the attention is not required to be invertible. Besides, this direct attention application merely learns the dependencies within one of the two splits of channel-wise flow dimensions, and thus its receptive field is greatly limited. In contrast, our introduced attentions are independent flow layers that are invertible and can learn more general and better range dependencies across different splits of flow feature maps[1]. In other words, [17] models within-split dependencies while ours learns cross-split correlations, and hence both are complementary for each other. More recently, [35] models channel-wise dependencies through multi-scale autoregressive priors. The introduced dependency modeling is limited on latent space, and hence it can be complementary to our exploited attentions on intermediate flow dimensions.

Likewise, various *conditional flow models* have appeared aiming at conditional image synthesis [32,33,39,43,44]. For instance, [44] exploits two invertible networks for source and target and a relation network that maps the latent spaces to each other. In this way, conditioning information can be leveraged at the appropriate hierarchy level and hence, can overcome the restriction of using raw images as input. Similarly, [39] exploits a parallel sequence of invertible mappings in which a source flow guides the target flow at every step. [43] introduces conditioning networks that allow all operations in the target-domain flow conditioned on the source-domain information. For better conditioning, [33] exploits conditional affine coupling layers that accept the source domain's feature maps extracted by one external neural network as the conditions. To our knowledge, these conditional flow models rarely learn suitable range dependencies in deep normalizing flow networks.

[1]For the similar purpose, a concurrent work [58] also introduces invertible attentions. The major difference is that [58] employs Lipschitz constraints over the modules for the invertibility, which is similar to the technique presented in [4]. However, the Lipschitz constraints are generally hard to satisfy, leading to inferior results using [58] for invertible models.

Layer	Function	Layer	Function
Actnorm	$\forall i,j : \mathbf{y}_{i,j} = \mathbf{s} \odot \mathbf{x}_{i,j} + \mathbf{b}$	Invertible 1×1 Convolution	$\forall i,j : \mathbf{y}_{i,j} = \mathbf{W}\mathbf{x}_{i,j}$
Affine Coupling	$\mathbf{x}_a, \mathbf{x}_b = \text{SPLIT}(\mathbf{x})$ $(\log \mathbf{s}, \mathbf{t}) = \text{NN}(\mathbf{x}_b)$ $\mathbf{y}_a = \exp(\log \mathbf{s}) \odot \mathbf{x}_a + \mathbf{t}$ $\mathbf{y} = (\mathbf{y}_a, \mathbf{x}_b)$	Mixture Affine Coupling	$\mathbf{x}_a, \mathbf{x}_b = \text{SPLIT}(\mathbf{x})$ $(\log \mathbf{s}, \mathbf{t}, \pi, \mu, \log \hat{\mathbf{s}}) = \text{NN}(\mathbf{x}_b)$ $\mathbf{y}_a = \sigma^{-1}(f(\mathbf{x}_a, \pi, \mu, \log \hat{\mathbf{s}})) \odot \exp(\log \mathbf{s}) + \mathbf{t}$ $\mathbf{y} = (\mathbf{y}_a, \mathbf{x}_b)$
Conditional Affine Coupling	$\mathbf{x}_a, \mathbf{x}_b = \text{SPLIT}(\mathbf{x})$ $(\log \mathbf{s}, \mathbf{t}) = \text{NN}(\mathbf{x}_b, \mathbf{c})$ $\mathbf{y}_a = \exp(\log \mathbf{s}) \odot \mathbf{x}_a + \mathbf{t}$ $\mathbf{y} = (\mathbf{y}_a, \mathbf{x}_b)$	Conditional Affine Injector	$(\log \mathbf{s}, \mathbf{t}) = \text{NN}(\mathbf{c})$ $\mathbf{y} = \exp(\log \mathbf{s}) \odot \mathbf{x} + \mathbf{t}$

Table 1. STEPOFFLOW layers for either unconditional [35] or conditional [33] flow models used as our backbones. Here $\mathbf{x}, \mathbf{c}, \mathbf{y}$ indicate input, condition and output respectively. SPLIT, NN denote the split operation and the regular neural flow network operations. $\mathbf{x}_a, \mathbf{x}_b$ are the two splits, and $\log \mathbf{s}, \mathbf{t}, \pi, \mu, \log \hat{\mathbf{s}}$ are transformation parameters for $\mathbf{x}_a$ produced by the network function NN acting on $\mathbf{x}_b$. For mixture affine coupling, $f(\mathbf{x}_a, \pi, \mu, \log \hat{\mathbf{s}}) := \sum_i \pi_i \sigma((\mathbf{x}_a - \mu_i) \odot \exp(-\log \hat{\mathbf{s}}_\mathbf{i}))$, and $\sigma(\cdot)$ indicates the sigmoid function [17].

Attention Models. To address the problem of missing global information in convolutional operations, attention mechanisms have emerged. They can better model deep network layer interactions [1, 14, 30, 49, 51, 52, 57, 59]. In particular, self-attention calculates the response at a position in a sequence by attending to all positions within the same sequence allowing for long-range interactions without an increase in the number of parameters. For instance, [37,52,54] introduce map-based attention to improve the performance of convolutional networks on image recognition, where spatial attention maps are learned to scale the features given by convolutional layers. [49] integrates the scaled dot-product attention with its multi-head versions to construct the state-of-the-art attention (i.e., *transformer*), which has become a de-facto standard for natural language processing tasks. [14, 30] achieve the state-of-the-art in a broad range of vision tasks by further applying the vanilla transformer to sequences of image patches. [5, 6, 20, 22, 34, 59] exploit conventional attentions or transformer-based attentions in the context of other generative models like GANs to capture long-range dependencies for better image generation. Despite such remarkable progress, attention models have rarely been explored for flow-based generative models, where each neural operation is constrained to preserve tractability of the inverse and Jacobian determinant computation. To fill the gap, our proposed invertible attentions provide valuable solutions that enable such regular attentions, e.g, *map-based attention* and *transformer-based attention*, to work well in the context of generative flows.

3. Overview and Background

This paper introduces two invertible attention mechanisms to better model the network's depth dependencies for unconditional and conditional flow-based generative models[2]. Our modeling is capable of producing more efficient flow models. Below we provide an overview of the unconditional and conditional generative flow models, followed by an outline of the proposed attention mechanisms.

Unconditional flow: In this setup, the generative flows aim at learning invertible transformations (*i.e.*, f_θ, g_θ, where $\mathbf{z} = f_\theta(\mathbf{x}) = g_\theta^{-1}(\mathbf{x})$ with model parameters θ) between a simple distribution $\mathbf{z} \sim p_\theta(\mathbf{z})$ and a complex one $\mathbf{x} \sim p_\theta(\mathbf{x})$. The function f_θ (and, likewise, g_θ) are parameterized by an invertible neural network, consisting of a sequence of L invertible functions f_{θ_i}. Hence, the network model is typically called as a (normalizing) flow: $f_\theta = f_{\theta_1} \circ f_{\theta_2} \circ \ldots f_{\theta_L}$, mapping the simple distribution density on the latent variable $\mathbf{z}$ to the complex distribution density on the data $\mathbf{x}$:

$$\mathbf{x} \overset{f_{\theta_1}}{\longleftrightarrow} \mathbf{h}_1 \overset{f_{\theta_2}}{\longleftrightarrow} \mathbf{h}_2 \cdots \overset{f_{\theta_L}}{\longleftrightarrow} \mathbf{z}. \tag{1}$$

Given the log-likelihood of $p_\theta(\mathbf{z})$, the change of variables formula enables us to compute the log-likelihood of the data $\mathbf{x}$ under the transformation f_θ:

$$\begin{aligned} \log p_\theta(\mathbf{x}) &= \log p_\theta(\mathbf{z}) + \log|\det(\partial \mathbf{z}/\partial \mathbf{x})| \\ &= \log p_\theta(f_\theta(\mathbf{x})) + \sum_{i=1}^{L} \log |\det(\partial \mathbf{h}_i/\partial \mathbf{h}_{i-1})|, \end{aligned} \tag{2}$$

where $\partial \mathbf{h}_i/\partial \mathbf{h}_{i-1}$ is the Jacobian of the invertible transformation f_{θ_i} moving from $\mathbf{h}_{i-1}$ to $\mathbf{h}_i$ with $\mathbf{h}_0 \equiv \mathbf{x}$. The scalar value $\log |\det J_{\theta_i}|$ is the log-determinant of the Jacobian matrix[3]. The likelihood of $p_\theta(\mathbf{z})$ is commonly modeled as Gaussian likelihood, *e.g.*, $p(\mathbf{z}) = \mathcal{N}(\mathbf{z} \,|\, \mu, \sigma)$. The exact likelihood computation allows us to train the network by minimizing the negative log-likelihood (NLL) loss.

Conditional flow: In this setting, the invertible network f_θ maps the input data-condition pair $(\mathbf{x}, \mathbf{c})$ to a latent variable $\mathbf{z} = f_\theta(\mathbf{x}; \mathbf{c})$. Here, the data $\mathbf{x}$ is reconstructed from the latent encoding $\mathbf{z}$ conditioning on $\mathbf{c}$ as $\mathbf{x} = f_\theta^{-1}(\mathbf{z}; \mathbf{c})$. The log-likelihood of the data $\mathbf{x}$ is computed as

$$\log p_\theta(\mathbf{x}|\mathbf{c}) = \log p_\theta(f_\theta(\mathbf{x}; \mathbf{c})) + \sum_{i=1}^{L} \log |\det(\partial \mathbf{h}_i/\partial \mathbf{h}_{i-1})|, \tag{3}$$

[2]Our paper focuses on studying invertible flows that allow for both efficient exact inference and sampling.

[3]Flow-based generative models choose transformations whose Jacobian is a triangular matrix for tractable computation of log-det.

where, $\mathbf{h}_i = f_{\theta_i}(\mathbf{x}; \mathbf{c})$. For both the unconditional and conditional flow models, the design of flow layers generally respects the protocol that computing the inverse and Jacobian determinant of the involved transformations f_{θ_i} should be tractable. In this paper, we mainly use [35] and [33] as our backbones for unconditional and conditional flow models respectively. In these backbones, the flow network is organized into L flow-levels, each operating at a resolution containing K number of flow-steps. In general, each flow-level f_{θ_i} is composed of SQUEEZE, STEPOFFLOW, and SPLIT operations. SQUEEZE trades off spatial resolution for channel dimension. STEPOFFLOW is commonly a series of affine coupling layers, invertible 1×1 convolutions and normalization layers. SPLIT divides an intermediate layer h_i into two halves, one of which is transformed and the other of which is left unchanged. Table (1) summarizes the functions of the main layers of STEPOFFLOW.

To explicitly model the long-range dependencies for efficient flow models, we study two types of invertible attention mechanisms: *(i) invertible map-based (iMap) attention*: It aims at learning a weighting factor for each position in the attention dimension and scales the flow feature maps with the learned attention weights. The attention models the importance of each position in the attention dimension of flow feature maps explicitly. *(ii) invertible transformer-based (iTrans) attention*: It computes the representation response at a position as a weighted sum of features of all the positions along the attention dimension. The attention weights are computed by scaled dot-product between features of all the positions. Compared to the iMap attention, it explicitly models the second-order dependencies among the distant positions along the attention dimension.

4. Proposed Attention Flow

The proposed attention flow (AttnFlow) aims at inserting invertible map-based (iMap) or transformer-based (iTrans) attention flow layers to conventional flow-based generative models (see Fig.(1) (c)), so that the attention learning can enhance their representation learning efficiency. Like conventional attention mechanisms, an attention operation accepts a feature map $\mathbf{h}_{\text{in}}$ of shape (H, W, C_{in}) as input, and outputs an attended featured map $\mathbf{h}_{\text{out}}$ of shape (H, W, C_{out}) with a transformation $\mathbf{h}_{\text{out}} = G(\mathbf{h}_{\text{in}})$. In practice, attention learning consists of three steps: *(i)* reshaping the input feature map $\mathbf{h}_{\text{in}}$, *(ii)* computing the attention weights $\mathbf{W}_{\text{attn}}$, and *(iii)* applying the learned attention weights to output $\mathbf{h}_{\text{out}}$. To integrate the introduced attention modules into generative flows, we must ensure the attention transformation G preserves the tractability of inverse and Jacobian determinant computation. Hence, we introduce a checkerboard masking scheme for globally permuted binary patterns (*i.e.*, two-split generation $\boldsymbol{x}_1, \boldsymbol{x}_2$) of the entire flow feature maps. Inspired by the existing split techniques [11, 24], we proposed a spatial-channel checkerboard masking scheme. As illustrated in Fig.(2) (c), the permutation is performed on the whole space and channel dimensions of the input feature maps. Compared to the existing spatial checkerboard masking [11] (Fig.(2) (a)) and channel-wise masking [11, 24] (Fig.(2) (b)) that are directly applied to generate binary patterns on the space and channel domains, the introduced spatial-channel checkerboard masking (Fig.(2) (c)) can produce more globally permuted binary patterns. As our methods learn attentions across the splits, the more permuted and staggered binary patterns allow for more complete long-range interactions.

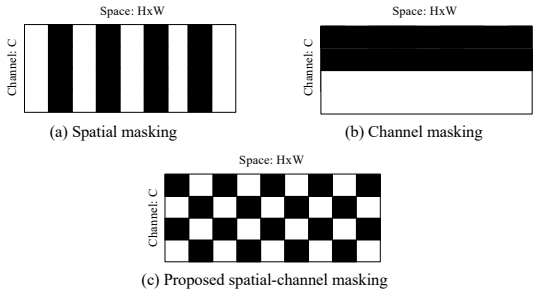


Figure 2. (a) Spatial checkerboard masking [11], (b) channel-wise masking [11, 24], and (c) proposed spatial-channel checkerboard masking, for the binary pattern generation on the space and channel dimensions.

The nature of the introduced global masking strategy better ensures the involved attentions can be invertible directly. Furthermore, it enables an attention transfer from one split $\boldsymbol{x_1}$ to the other split $\boldsymbol{x_2}$, which encourages interaction between the two splits along one attention dimension such as spatial dimension and channel dimension. As illustrated in Fig.(3) (a)-(c), the overall masked flow attention operations can be roughly formulated as

$$\boldsymbol{y}_1 = \boldsymbol{x}_1 \odot \boldsymbol{s}, \tag{4a}$$

$$\boldsymbol{y}_1 = \boldsymbol{x}_1, \tag{4b}$$

$$\boldsymbol{y}_2 = \boldsymbol{x}_2 \odot f(\boldsymbol{x}_1), \tag{4c}$$

where Eq.(4a) and Eq.(4b) are for iMap and iTrans respectively, and Eq.(4c) is for both. $\odot$ represents the element-wise/matrix multiplication for the proposed iMap/iTrans, and $f(\boldsymbol{x}_1)$ indicates the attention weight computation for iMap/iTrans[4]. As shown in Fig.(3) (b)-(d), our approach computes the inverse propagation directly as follows:

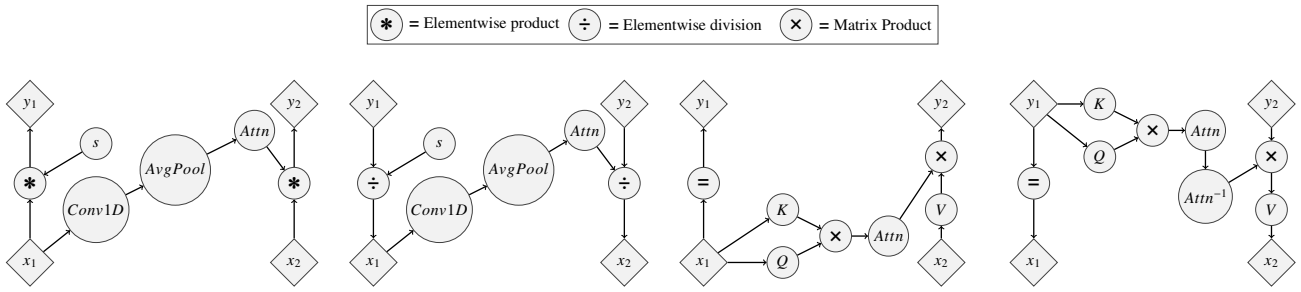
$$\boldsymbol{x}_1 = \boldsymbol{y}_1 \oslash \boldsymbol{s}, \tag{5a}$$

$$\boldsymbol{x}_1 = \boldsymbol{y}_1, \tag{5b}$$

$$\boldsymbol{x}_2 = \boldsymbol{y}_2 \oslash f(\boldsymbol{x}_1), \tag{5c}$$

where Eq.(5a) and Eq.(5b) are for iMap and iTrans respectively, and Eq.(5c) is for both. $\oslash$ indicates the element-

[4]The new masking computation and the attention-oriented transformation make our attention operations distinct from exiting coupling layers [11, 24] and its associated attentions like the one in [17].



(a) MapAttn Forward Propagation (b) MapAttn Inverse Propagation (c) TransAttn Forward Propagation (d) TransAttn Inverse Propagation

Figure 3. Computational graph of forward and inverse propagation of the proposed Map-based (a)(b) and Transformer-based (c)(d) attention mechanisms. Due to the simple nature of the introduced split-based strategy, the involved attentions are both easily invertible and possesses a tractable Jacobian determinant. In (a,b), s is a learnable scale parameter, and the average pooling is performed along channels (/spaces) for spatial (/channel) attention learning. In (c,d), K, Q, V are the three basic elements for the Transformer-based attention. They are computed by regular 1×1 2D convolutions within the masking scheme, which allows for invertible operations.

wise/matrix division for iMap/iTrans, and $f(\boldsymbol{x}_1)$ denotes the computation of attention weights. Below we provide the details of the two introduced invertible attentions, and the corresponding Jacobian determinant computation.

iMap Attention. Following [52] that invents regular map-based attention (i.e., diagonal attention), we exploit an invertible map-based attention to scale the feature map with the learned attention weights that encodes the importance of individual flow dimensions along the attention dimension. The main difference is that we apply attention weights calculated on one split $\boldsymbol{x}_1$ to the other split $\boldsymbol{x}_2$, due to the invertiable design in Eq.(4) and Eq.(5). Concretely, we apply a sequence of analogous functions from [52] to realize iMap over the spatial domain of flow feature maps. Mathematically, the attention weights can be calculated as

$$\mathbf{W}_{\text{imap}} = G_5\Big((1-\mathbf{M})G_4\big(G_3(G_2(G_1(\mathbf{h}_{\text{in}})))\big) + \mathbf{M}\mathbf{b}\Big), \tag{6}$$

where $\mathbf{M}$ is the proposed checkerboard mask (Fig.(2) (c)), $\mathbf{b}$ is a learnable variable, $G_1(\mathbf{h}_{\text{in}}) = \mathbf{M}\mathbf{h}_{\text{in}}$, G_2 is a 1D convolutional layer with kernel size as 1, which reduces the dimension of the feature response of each channel from C_{in} to C', and outputs a feature map of shape $(H \times W, C')$. Without loss of generality, G_3 applies average pooling[5] to each channel dimensions and outputs an $(H \times W)$-dim vector for spatial attention learning. The operator G_4 is to reorganize the $(H \times W)$ attention weights into a $(H \times W) \times (H \times W)$ matrix, where the attention weights of shape $(H \times W)$ are placed on the diagonal of the matrix. The derived attention weight matrix $\mathbf{W}_{\text{imap}}$ is a diagonal matrix. The function G_5 corresponds to standard activation functions such as softmax and sigmoid. Finally, we apply the attention weight matrix $\mathbf{W}_{\text{imap}}$ to the input feature map through matrix multiplication to obtain the attended feature map $\mathbf{h}_{\text{out}} = \mathbf{W}_{\text{imap}}\mathbf{h}_{\text{in}}$. The forward and inverse propagation of *AttenFlow-iMap* module are illustrated in Fig.(3) (a)-(b). The Jacobian determinant of the introduced iMap transformation is computed as follows:

$$\det\Big(\frac{\partial \mathbf{h}_{\text{in}}}{\partial \mathbf{h}_{\text{out}}}\Big) = \det(W_{\text{imap}}) = \left(\prod_{M_{j,:}=\mathbf{1}} G_5(\mathbf{b_j})\right) * \big(G_5(G^{'}(\mathbf{h}_{\text{in}}))\big)^{C_{\text{in}}/2}, \tag{7}$$

where $\boldsymbol{M}$ is the enforced mask, C_{in} is the channel number of h_{in}, G_5 indicates the corresponding activation function, $G^{'}(\mathbf{h}_{\text{in}}) = G_3(G_2(G_1(\mathbf{h}_{\text{in}})))$, G_1, G_2, G_3 are masking, 1D convolution, and average pooling respectively.

iTrans Attention. The conventional transformer-based attention was proposed in [49]. The success of this type of attention mechanism mainly stems from the effective learning of second-order correlations among involved feature maps and the exploitation of three different representations for attention learning. The attention function is expressed as mapping a query $\mathbf{q}_{\text{in}}$ and a set of key-value $(\mathbf{k}_{\text{in}}, \mathbf{v}_{\text{in}})$ pairs to an output $\mathbf{h}_{\text{out}}$. The query and the key are employed to learn the second-order attention weights through a scaled dot-product computation, which is further applied to the input value for the final attended output.

To introduce the transformer-based attention to flow models, as shown in Fig.(3)(c), we apply two invertible 1×1 2D convolutions to the input feature maps to obtain a query-key pair $(\mathbf{q}_{\text{in}}, \mathbf{k}_{\text{in}})$, and use the input feature maps to play the role of the value $\mathbf{v}_{\text{in}}$. The attention is applied between patches of the input following [14]. In particular, the whole input is split into N patches and the iTrans attention is applied to the image patches. The primary goal is to capture the inter-patch interaction with the attention weights. In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix $\mathbf{Q}$. The keys and values are also packed together into matrices $\mathbf{K}$ and $\mathbf{V}$. The mapping process is formulated as follows:

$$\mathbf{h}_{\text{out}} = \mathbf{W}_{\text{itrans}}\mathbf{V} = G_4\Big(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\Big)\mathbf{V}, \tag{8}$$

[5] Performing average pooling over the spatial domain learns channel attention, provided the spatial resolutions for train and validation are same.

where $\mathbf{Q} = G_2(G_1(\mathbf{h}_{\text{in}}))$, $\mathbf{K} = G_3(G_1(\mathbf{h}_{\text{in}}))$, $\mathbf{V} = G_1(\mathbf{h}_{\text{in}}) = \mathbf{M}\mathbf{h}_{\text{in}}$, $\mathbf{M}$ is the suggested checkerboard mask (Fig.(2) (c)), G_2, G_3 correspond to two regular 1×1 2D convolutions, G_4 corresponds to the activation function. G_2, G_3 are computed within the introduced masking, which allows for invertible operations. In general, dot-product values often get large to influence the final negative log-likelihood scales. Hence, inspired by [49], we apply d to scale the dot-product values. To achieve a general scale, we made d learnable. In addition, we follow the vanilla transformer [49] to exploit multiple branches of scaled dot-products (Eq.(8)) for the multi-head attention. Fig.(3)(d) show the inverse propagation of *AttnFlow-iTrans*, which can be computed in a straightforward manner.

The Jacobian of the iTrans transformation, with the attended feature map being $\boldsymbol{h}_{\text{out}} = \boldsymbol{W}_{\text{itrans}}\boldsymbol{M}\boldsymbol{h}_{\text{in}}$, is a lower block triangular matrix, with the attention weights $\mathbf{W}_{\text{itrans}}$ forming the (repeated) block diagonal entries. As the determinant of a lower block triangular matrix is simply the product of determinants of the matrices along the diagonal, the Jacobian determinant of iTrans can be computed as

$$\det(\frac{\partial \mathbf{h}_{\text{in}}}{\partial \mathbf{h}_{\text{out}}}) = (\det(\mathbf{W}_{\text{itrans}}))^{P/2} = (\det(G_4(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}})))^{P/2}, \tag{9}$$

where $G_4, \boldsymbol{Q}, \boldsymbol{K}, d$ are defined around Eq.(8), and P is the patch size, *i.e.*, feature dimension within each patch.

5. Experimental Evaluation

We evaluated the proposed unconditional and conditional attention flow (AttnFlow, cAttnFlow)[6] models for image generation, image super-resolution and general image translation tasks respectively[7]. Besides, we present more experimental details and evaluations in the suppl. material.

Image Generation. We use two datsets *i.e.*, MNIST [28] and CIFAR10 [27] for unconditional image generation.

1) AttnFlow Setup: The proposed AttnFlows can be applied to any off-the-shelf unconditional generative flows. For the image generation task, we utilize the architecture of mARFlow[8] [35] as the backbone of AttnFlows, where our proposed iMap and iTrans attention flow layers can be inserted. Each level of mARFlow sequentially stacks an actnorm layer, an invertible 1×1 convolution layer, and a coupling layer. Under the mARFlow backbone, we inserted our proposed attention modules (either iMap or iTrans) into one of the following four positions: *(i)* Before actnorm (pos-1), *(ii)* after actnorm (pos-2), *(iii)* after invertible convolution (pos-3), and *(iv)* after coupling (pos-4). To study AttnFlows' efficiency, we evaluate their various setups on the numbers of flow-levels, flow-steps, and channels. We use sigmoid for the activation function, and empirically set the patch number as $N = 4$ for AttnFlow-iTrans.

Method	Levels	Steps	Channels	Parameters (MB)	bits/dim (↓)
Glow	3	32	512	–	1.05
Residual Flow	3	16	–	–	0.97
mARFlow	3	4	96	46.01	0.56 (0.88*)
AttnFlow-iMap	3	4	96	46.03	0.43
AttnFlow-iTrans	3	4	96	46.25	0.44
AttnFlow-iMap	3	2	96	23.78	0.41
AttnFlow-iTrans	3	2	96	23.89	0.42
AttnFlow-iMap	3	2	48	8.94	**0.39**
AttnFlow-iTrans	3	2	48	9.05	<u>0.40</u>

Table 2. Evaluation of sample quality on MNIST. $*$ indicates the result reported in the mARFlow paper [35]. As MNIST is a small dataset and very complex model is not at all required, the performance gets decreased when our model's complexity increases. (**Bold**: best, <u>Underline</u>: second best)

2) Competing Methods: We compare four state-of-the-art unconditional generative flows, i.e., Glow [24], Flow++ [17], Residual Flow [4] and mARFlow [35]. Our AttnFlows' architecture is based on mARFlow with coupling layers closest to Glow. By comparison, Flow++ does not include the SPLIT operation, and uses a different uniform dequantization. Hence, the comparison with Glow and mARFlow serves as a better ablation to measure the effectiveness and efficiency of AttnFlows. Besides, we compare the concurrent work [58], with its two variants (iResNet-iDP, iResNet-iCon), which applies Lipschitz constraints to dot-product and concatenation attentions under the specific flow framework (iResNet) [4]. For a reference, we also compare one representative GAN model, *i.e.*, DCGAN [40].

3) Comparison: Table (2) and Table (3) summarize the quantitative results of our AttnFlows and the competing methods on MNIST and CIFAR10. For evaluation, we use per-pixel log-likelihood metric in bits/dims. Further, we use three more standard metrics, *i.e.*, Fréchet Inception Distance (FID) [16], inception scores [41] and Kernel Inception Distance (KID) [2], to measure the generated sample quality on CIFAR10. From the results, we can see that both of our AttnFlow-iMap and AttnFlow-iTrans clearly outperform the backbone mARFlow with similar model complexities (*i.e.*, the same level and step numbers), and our AttnFlows can achieve better results than the other state-of-the-art flow models[9]. Furthermore, our lighter model (with a smaller number of steps or a smaller channel) typically achieve comparable performances (or even better results) compared to those heavier mARFlow models. In particular, the proposed method achieves remarkable improvement (i.e., 0.17 bits/dim) over mARFlow with about $5\times$ smaller parameter size and $2\times$ less steps/channels (Table

[6] AttnFlow's official code: https://github.com/rheasukthanker/AttnFlow

[7] Following [33, 35, 43], we evaluate the proposed method and all the competing methods with one single run on the employed datasets.

[8] mARFlow's official code: https://github.com/visinf/mar-scf/

[9] Flow++ [17] merely reported its performance on CIFAR10. After transferring its implementation from CIFAR10 to MNIST, its bits/dim is 0.66 that is also clear worse than ours (0.39).

(a) Flow++ (3.08 bits/dim) (b) mARFlow (3.24 bits/dim, 41.9 FID) (c) *AttnFlow-iMap* (3.216 bits/dim, 33.6 FID) (d) *AttnFlow-iTrans* (3.217 bits/dim, 33.8 FID)

Figure 4. Comparison of samples from the proposed models (AttnFlow-iMap, AttnFlow-iTrans) with state-of-the-art models on CIFAR10.

Method	Level	Step	Channel	Parameter (MB)	bits/dim (↓)	FID (↓)	Incep (↑)	KID (↓)
DCGAN	–	–	–	–	–	37.1	6.4	–
Glow	3	32	512	–	3.35	46.9	–	–
Flow++	3	–	96	–	3.29	46.9	–	–
Residual Flow	3	16	–	–	3.28	46.3	5.2	–
iResNet-iDP	–	–	–	–	3.65	–	–	–
iResNet-iCon	–	–	–	–	3.39	–	–	–
mARFlow	3	4	96	46.01	3.27 (3.254*)	(40.5*)	(5.8*)	(0.033*)
mARFlow	3	4	256	252.77	3.24 (3.222*)	41.9 (33.9*)	5.7 (6.5*)	(0.026*)
AttnFlow-iMap	3	4	96	46.03	3.247	40.5	6.0	0.031
AttnFlow-iTrans	3	4	96	46.25	3.248	40.2	5.9	0.032
AttnFlow-iMap	3	4	256	252.79	**3.216**	**33.6**	6.6	**0.025**
AttnFlow-iTrans	3	4	256	253.01	3.217	33.8	**6.7**	**0.025**

Table 3. Evaluation of sample quality on CIFAR10. Note that $*$ indicates the results for the ICML workshop version of mARFlow [35]. (**Bold**: best, Underline: second best)

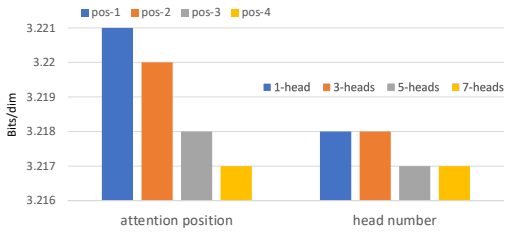


Figure 5. Ablation studies of the proposed attention on different positions in the flow layers (pos-1: before actnorm, pos-2: after actnorm,pos-3: after permutation, pos-4: after coupling layer) and different number of attention heads (1 head, 3 heads, 5 heads, 7 heads) for AttnFlow-iTrans on CIFAR10.

(2)). For CIFAR10, our models (Channel=256) get visibly better FIDs, Incepts and KIDs over mARFlow (Table (3)). The visual comparison in Fig.(4) shows that the proposed models have much clear visual quality compared to the competing methods. Despite the intuitive superiority of the proposed spatial-channel masking against the existing ones [11, 24] (Fig.(2)), we evaluate these maskings and the random binary masking with our AttnFlow-iMap on MNIST. The bits/dim are 0.99 (Spatial), 0.75 (Channel), 0.50 (Random), 0.39 (Ours), showing ours' clear advantage.

Ablation Study. As shown in Fig.(5), we perform the ablation test of the proposed attention models in the following two settings: *(i)* different attention positions, and *(ii)* different head numbers. We observe that inserting the attention layers in the position after the permutation layer (pos-3) and after the coupling layer (pos-4) are the most favourable. On the other hand, the use of more than 5 attention heads for AttnFlow-iTrans does not provide a clear improvement.

Image Super-Resolution. We follow [33] to use CelebA dataset split for image super-resolution (SR) task [31].

Method	Levels	Steps	Parameters (MB)	SSIM (↑)	PSNR (↑)	LR-PSNR (↑)	LPIPS (↓)
Bicubic	–	–	–	0.63	23.15	35.19	0.58
ESRGAN	–	–	–	0.63	22.88	34.04	**0.12**
SRFlow	1	1	6.622	0.67	**25.57**	44.20	0.23
SRFlow	2	8	13.25	0.73	25.47	38.94	0.17
cAttnFlow-iMap	1	1	6.623	0.71	25.50	**44.75**	0.19
cAttnFlow-iTrans	1	1	6.630	0.73	25.50	44.23	0.18
cAttnFlow-iMap	2	8	13.30	**0.74**	25.38	41.88	0.17
cAttnFlow-iTrans	2	8	13.93	0.73	25.24	42.49	0.16

Table 4. Results for $8\times$ SR on CelebA. We report average SSIM, PSNR, LR-PSNR and LPIPS scores for SRFlow and ours at different temperatures (0.1-0.9). (**Bold**: best, Underline: second best)

1) cAttnFlow Setup: Our conditional AttnFlow (cAttnFlow) is based on the architecture of the SRFlow model [33][10]. The flow network is organized into $L = 4$ flow-levels, each of which operate a specific resolution of $H/2^l \times W/2^l$ ($H \times W, L$ indicate the resolution of HR images and the l-th flow-level respectively). Each flow-level is composed of K flow-steps. Each flow-step stacks four different layers: *(i)* Acnorm, *(ii)* 1×1 invertible convolution, *(iii)* affine injector, and *(iv)* conditional affine layers. Similar to image generation, we insert our proposed attentions after the existing flow layers in each level of SRFlow.

2) Competing Methods: Following SRFlow, we compare our results with bicubic and other recent SR methods, which includes ESRGAN [53] and SRFlow [33]. As SRFlow is our cAttnFlow's backbone, comparison against it helps us realize the improvement using our introduced attentions.

3) Comparison: Table (4) reports the comparison of our cAttnFlow against the competing methods in terms of four standard metrics, including SSIM, PSNR, LR-PSNR and LPIPS. The results imply that our model can achieve the best balance among the four used metrics compared to the competing methods. The improvements of our cAttnFlow over the backbone SRFlow are visible at two different level model complexities, showing that our introduced attention can enhance the efficiency of the flow models. The visual comparison in Fig.(6) shows that the outputs from our cAttnFlows are comparable or better than those from the others.

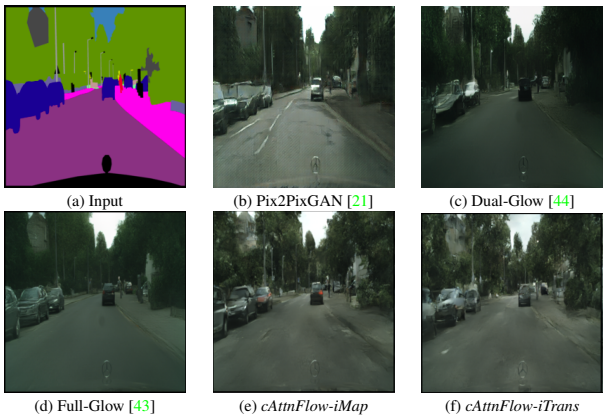
Image Translation. We use Cityscapes [8] to evaluate the proposed cAttnFlows for image translation, where segmentation label images are translated into RGB images.

1) cAttnFlow Setup: Our conditional AttnFlow (cAttnFlow)

[10] SRFlow's official code: https://github.com/andreas128/SRFlow/

Input ESRGAN [53] SRFlow [33] *cAttnFlow-iMap* *cAttnFlow-iTrans*

Figure 6. Super-resolved samples of the proposed cAttnFlows and the state-of-the-art models for $8\times$ face SR on the CelebA dataset.



(a) Input (b) Pix2PixGAN [21] (c) Dual-Glow [44]

(d) Full-Glow [43] (e) *cAttnFlow-iMap* (f) *cAttnFlow-iTrans*

Figure 7. Generated samples of the proposed cAttnFlows and the state-of-the-art models for image translation on the Cityscapes dataset. The competing methods and ours are conditioned on the semantic segmentation labels (a) to synthesize the RGB images with the resolution being of 256×256.

is based on the conditional flow (Full-Glow)[11] [43] model. The normalizing flow network is organized into $L = 2$ flow-levels, each of which operate a specific resolution of $H/2^l \times W/2^l$, where $H \times W$ indicates the resolution of input images and the l-th flow-level respectively. Each flow-level is composed of $K = 8$ flow-steps. Note that our flow model is much smaller than the Full-Glow model that consists of 4 levels and each contains 16 steps (i.e., $L = 4, K = 16$). Each flow-step stacks four different layers: 1) Acnorm, 2) 1×1 invertible convolution, 3) affine injector, and 4) conditional affine layers. As done for image generation, we also insert our proposed flow attentions after the existing flow layers in each level of the Full-Glow model.

2) Competing Methods: Following [43], we compare with the state-of-the-art conditional flow methods, C-Glow [32], Dual-Glow [44], and Full-Glow [43]. We also compare the GAN model (Pix2Pix) [21] for a refrence. As we use Full-Glow as our cAttnFlow's backbone, we will focus on the comparison with it, which can clearly show the improvement using our introduced attention mechanisms.

[11] Full-Glow's official code: https://github.com/MoeinSorkhei/glow2

Method	Levels	Steps	Parameters (MB)	Conditional bits/dim (↓)
C-Glow v.1	–	–	–	2.568
C-Glow v.2	–	–	–	2.363
Dual-Glow	–	–	–	2.585
Full-Glow	4	16	155.33	2.345
cAttnFlow-iMap	2	8	34.68	**2.310**
cAttnFlow-iTrans	2	8	34.70	<u>2.314</u>

Table 5. Quantitative results of the proposed AttnFlow and the state-of-the-art models on the Cityscapes dataset for label → photo image translation. (**Bold**: best, <u>Underline</u>: second best)

3) Comparison: For likelihood-based models, we follow [43] to measure the conditional bits per dimension, $-\log_2 p(\mathbf{x}_b|\mathbf{x}_a)$, as a metric of how well the conditional distribution learned by the model matches the real conditional distribution, when tested on held-out examples. Table (5) summarizes the results of the proposed cAttnFlows and its competitors. The comparison shows that the proposed cAttnFlows can achieve better performances than the state-of-the-art conditional flow models. In particular, compared to the backbone model (Full-Glow), the proposed cAttnFlows achieve better bits/dim with about $5\times$ smaller parameter size and $2\times$ less levels/steps, showing that the proposed attention can highly enhance the efficiency of flow models. The visual comparison in Fig.(7) shows that the synthesized images of our cAttnFlow are more visually pleasing (e.g, owning clearly richer texture details and better illumination) compared to the competing generative flow models, and they look relatively comparable with that produced by Pix2PixGAN [21].

6. Conclusion and Future Work

This paper introduces invertible map-based and transformer-based attentions for both unconditional and conditional generative normalizing flows. The proposed attentions are capable of learning network dependencies efficiently to strengthen the representation power of flow-based generative models. The evaluation on image generation, super-resolution and image translation show clear improvement of our proposed attentions over the used unconditional and conditional flow-based backbones.

As conventional attention mechanisms, one of our models' major limitations lies in its unsatisfactory scaling ability to deeper neural networks such as full SRFlow [33], due to the common attention vanishing problem studied in [13]. As a future work, we will follow [13] to address the problem in the context of deeper invertible flow models.

Acknowledgments. This work was supported in part by the ETH Zürich Fund (OK), an Amazon AWS grant, and an Nvidia GPU grant. Suryansh Kumar's project is supported by "ETH Zürich Foundation 2019-HE-323, 2020-HS-411" for bringing together best academic and industrial research. This work was also supported by the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 grant (MSS21C002). The authors would like to thank Andreas Lugmayr for valuable discussions.

References

[1] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *ICCV*, 2019. 3

[2] Mikołaj Bińkowski, Dougal J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *ICLR*, 2018. 6

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2018. 1

[4] Ricky TQ Chen, Jens Behrmann, David Duvenaud, and Joern-Henrik Jacobsen. Residual flows for invertible generative modeling. In *NeurIPS*, 2019. 2, 6

[5] Xinyuan Chen, Chang Xu, Xiaokang Yang, and Dacheng Tao. Attention-GAN for object transfiguration in wild images. In *ECCV*, 2018. 3

[6] Yu Cheng, Zhe Gan, Yitong Li, Jingjing Liu, and Jianfeng Gao. Sequential attention GAN for interactive image editing. In *ACMMM*, 2020. 3

[7] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018. 1

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 7

[9] Stefano d'Apolito, Danda Pani Paudel, Zhiwu Huang, Andrés Romero, and Luc Van Gool. GANmut: Learning interpretable conditional space for gamut of emotions. In *CVPR*, 2021. 1

[10] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear independent components estimation. In *ICLR Workshop*, 2015. 1, 2

[11] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *ICLR*, 2017. 1, 2, 4, 7

[12] Justin Domke, Alap Karapurkar, and Yiannis Aloimonos. Who killed the directed model? In *CVPR*, 2008. 1

[13] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. *arXiv preprint arXiv:2103.03404*, 2021. 8

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3, 5

[15] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In *NeurIPS*, 2014. 1

[16] 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 *NeurIPS*, 2017. 6

[17] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *ICML*, 2019. 1, 2, 3, 4, 6

[18] Emiel Hoogeboom, Rianne Van Den Berg, and Max Welling. Emerging convolutions for generative normalizing flows. In *ICML*, 2019. 2

[19] Rui Huang, Shu Zhang, Tianyu Li, and Ran He. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In *ICCV*, 2017. 1

[20] Drew A Hudson and Larry Zitnick. Generative adversarial transformers. In *ICML*, 2021. 3

[21] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 8

[22] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. TransGAN: Two pure transformers can make one strong GAN, and that can scale up. *NeurIPS*, 2021. 3

[23] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1

[24] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, 2018. 1, 2, 4, 6, 7

[25] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *NeurIPS*, 2016. 2

[26] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 1

[27] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical report*, 2009. 6

[28] Yann LeCun, L eon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *PROC. OF THE IEEE*, 1998. 6

[29] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 1

[30] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 3

[31] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 7

[32] You Lu and Bert Huang. Structured output learning with conditional generative flows. In *AAAI*, 2020. 2, 8

[33] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. SRFlow: Learning the super-resolution space with normalizing flow. In *ECCV*, 2020. 2, 3, 4, 6, 7, 8

[34] Shuang Ma, Jianlong Fu, Chang Wen Chen, and Tao Mei. DA-GAN: Instance-level image translation by deep attention generative adversarial networks. In *CVPR*, 2018. 3

[35] Shweta Mahajan, Apratim Bhattacharyya, Mario Fritz, Bernt Schiele, and Stefan Roth. Normalizing flows with multi-scale autoregressive priors. In *CVPR*, 2020. 1, 2, 3, 4, 6, 7

[36] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015. 1

[37] Jongchan Park, Sanghyun Woo, Joon-Young Lee, and In So Kweon. BAM: Bottleneck attention module. In *BMVC*, 2018. 3

[38] Albert Pumarola, Antonio Agudo, Aleix M Martinez, Alberto Sanfeliu, and Francesc Moreno-Noguer. GANimation: Anatomically-aware facial animation from a single image. In *ECCV*, 2018. 1

[39] Albert Pumarola, Stefan Popov, Francesc Moreno-Noguer, and Vittorio Ferrari. C-Flow: Conditional generative flow models for images and 3d point clouds. In *CVPR*, 2020. 2

[40] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 6

[41] Tim Salimans, Ian J Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *NeurIPS*, 2016. 6

[42] Mohamad Shahbazi, Zhiwu Huang, Danda Pani Paudel, Ajad Chhatkuli, and Luc Van Gool. Efficient conditional GAN transfer with knowledge propagation across classes. In *CVPR*, 2021. 1

[43] Moein Sorkhei, Gustav Eje Henter, and Hedvig Kjellström. Full-Glow: Fully conditional glow for more realistic image generation. In *GCPR*, 2021. 2, 6, 8

[44] Haoliang Sun, Ronak Mehta, Hao H Zhou, Zhichun Huang, Sterling C Johnson, Vivek Prabhakaran, and Vikas Singh. Dual-Glow: Conditional flow-based generative model for modality transfer. In *ICCV*, 2019. 2, 8

[45] Yuan Tian, Qin Wang, Zhiwu Huang, Wen Li, Dengxin Dai, Minghao Yang, Jun Wang, and Olga Fink. Off-policy reinforcement learning for efficient and effective GAN architecture search. In *ECCV*, 2020. 1

[46] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. Wasserstein auto-encoders. In *ICLR*, 2018. 1

[47] Aäron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Koray Kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional image generation with PixelCNN decoders. In *NeurIPS*, 2016. 1

[48] Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *ICML*, 2016. 1

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 3, 5, 6

[50] Eason Wang, Henggang Cui, Sai Yalamanchi, Mohana Moorthy, and Nemanja Djuric. Improving movement predictions of traffic actors in bird's-eye view models using GANs and differentiable trajectory rasterization. In *SIGKDD*, 2020. 1

[51] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 3

[52] Xiaofang Wang, Xuehan Xiong, Maxim Neumann, AJ Piergiovanni, Michael S Ryoo, Anelia Angelova, Kris M Kitani, and Wei Hua. Attentionnas: Spatiotemporal attention cell search for video classification. In *ECCV*, 2020. 2, 3, 5

[53] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, and Xiaoou Tang. ESRGAN: Enhanced super-resolution generative adversarial networks. In *ECCV*, 2018. 1, 7, 8

[54] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. CBAM: Convolutional block attention module. In *ECCV*, 2018. 3

[55] Jiqing Wu, Zhiwu Huang, Dinesh Acharya, Wen Li, Janine Thoma, Danda Pani Paudel, and Luc Van Gool. Sliced wasserstein generative models. In *CVPR*, 2019. 1

[56] Jiqing Wu, Zhiwu Huang, Janine Thoma, Dinesh Acharya, and Luc Van Gool. Wasserstein divergence for GANs. In *ECCV*, 2018. 1

[57] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. AttnGAN: Fine-grained text to image generation with attentional generative adversarial networks. In *CVPR*, 2018. 3

[58] Jiajun Zha, Yiran Zhong, Jing Zhang, Liang Zheng, and Richard Hartley. Invertible attention. *arXiv preprint arXiv:2106.09003*, 2021. 2, 6

[59] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *ICML*, 2019. 2, 3

[60] Marc Yanlong Zhang, Zhiwu Huang, Danda Pani Paudel, Janine Thoma, and Luc Van Gool. Weakly paired multi-domain image translation. In *BMVC*, 2020. 1