Diverse Conditional Image Generation by Stochastic Regression with Latent Drop-Out Codes

Yang He, Bernt Schiele, and Mario Fritz

Max Planck Institute for Informatics,
Saarland Informatics Campus, Saarbrücken, Germany

Abstract. Recent advances in Deep Learning and probabilistic modeling have led to strong improvements in generative models for images. On the one hand, Generative Adversarial Networks (GANs) have contributed a highly effective adversarial learning procedure, but still suffer from stability issues. On the other hand, Conditional Variational Auto-Encoders (CVAE) models provide a sound way of conditional modeling but suffer from mode-mixing issues. Therefore, recent work has turned back to simple and stable regression models that are effective at generation but give up on the sampling mechanism and the latent code representation. We propose a novel and efficient stochastic regression approach with latent drop-out codes that combines the merits of both lines of research. In addition, a new training objective increases coverage of the training distribution leading to improvements over the state of the art in terms of accuracy as well as diversity.

Keywords: Image generation, Improving diversity, One-to-many mapping, Nonparametrics

1 Introduction

Many computer vision and graphics problems can be viewed as a conditional generation problem. For example, we can imagine the appearance of a human face when we only see the shape or the keypoints of the face. Typically, this generation process is not deterministic, as the conditioning information (e.g. keypoints) is insufficient to single out a particular face. Despite the diverse scope of applications for such models, these learning problems remain highly challenging, as an efficient sampling process is required that results in accurate and diverse samples that closely mimic the true conditional distribution.

In particular, *Generative Adversarial Networks (GANs)* [1] have contributed in recent years to the state-of-the-art in generative models of such high dimensional output spaces – as we are dealing with in the case of images. These methods allow for sampling by a random generated latent code and the adversarial training leads to highly accurate and realistic samples. The vanilla version of these models lacks the capability for conditional sampling and these methods

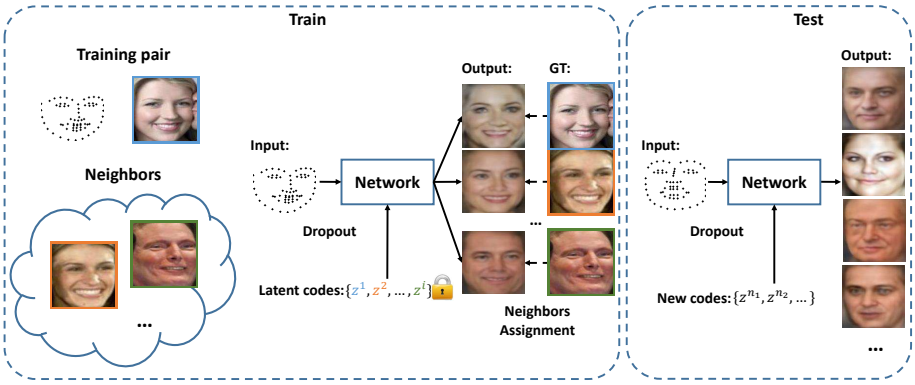

Fig. 1. The pipeline of the proposed method. We present a stochastic regression with latent dropout codes for image generation, which are fixed during training. At test time, we are able to generate more examples by providing newly sampled codes. Besides, Diversity is further improved by sampling many neighbors considering the condition input when the data distribution is dense enough. With those neighbors, we directly learn the one-to-many mapping by assigning sampled neighbors to different network branches.

are known to be notoriously difficult to train and often do not reproduce the full diversity of the training data.

Conditional Variational Autoencoders (CVAE) [2] have been introduced to model a latent code in dependence of the input and by a probabilistic formulation have shown an increased diversity in the generated samples. These models tend to be more stable to train, but still suffer from mode-mixing issues – limiting the success when conditioning data is weak. A series of *Conditional GAN (CGAN)* models [3–5] have been proposed that combine ideas of GANs and CVAE (bicycleGAN [4], pix2pix [5]) and thereby achieve some of the increased diversity of CVAE, but yet suffer from some stability problems of GANs.

In order to address the stability issues, several models have been recently presented [6] that are based on the idea of *Multiple Choice Learning (MCL)*. In essence, this re-phrases the conditional generation problem, as a regression task with a fixed number of output samples. This greatly improves the stability of the learning, but no additional samples can be drawn and there is no latent code that represents the samples.

We present a novel solution to the conditional image generation task that is stable to train, has a latent code representation, can be sampled from and results in accurate and diverse samples. We achieve this in a stochastic regression formulation where dropout patterns are conditioned on latent codes. A new training objective increases coverage of the training distribution – resulting in accurate and diverse samples at test time. Our experimental results on two datasets show improvements in efficiency, accuracy and diversity.

2 Related Work

Image generation with CNNs. The most prominent approach in image generation recently is generative adversarial networks (GANs) [1] based method. In the original work of Goodfellow *et al.*, it is used to generate digital number from random noise. Besides, GANs are extended into conditional models [3], that generate examples conditional on the input given by users. Conditional GAN has been applied in many interesting tasks, like text-to-image synthesis [7,8], image generation from a normal map [9], inpainting [10], or image super resolution [11]. Particularly, Isola *et al.* [5] regarded a set of image generation tasks under the image-to-image translation framework and formulated the translation task as the optimization of combined regression and adversarial loss, and finally achieved remarkable results. Zhu *et al.* [4] modeled the distribution of latent variables in the image-to-image translation framework [5] for generating multimodal images.

Perceptual loss [12, 13] is another successful tool in image generation. Dosovitskiy *et al.* [13] combined the GAN and minimization of the activation differences in a fixed network during training to generate natural images. Except comparing the activation difference of a perceptual network, Johnson *et al.* [12] also computed the perceptual differences in content and style between images for image style transfer and super resolution. Recently, Chen *et al.* [6] proposed cascaded refinement networks (CRN) and formulated the image generation task as a regression problem in perceptual similarities. CRN gradually generates images from low resolution feature maps to high resolution ones. Therefore, CRN achieved great success in the high resolution seamless street scene synthesis from semantic labels.

Diverse image synthesis & Nonparametrics. Multiple Choice Learning (MCL) [14] produces multiple structured predictions from different branches. In each iteration, only the branch with lowest loss is updated. Chen *et al.* [6] also generated multiple branches at the last layer of the network and applied the MCL framework on CRN to generate diverse examples. Li *et al.* [15] proposed a diversity loss function measuring the visual content difference of generated images in a training batch. By maximizing the visual content differences, multiple textures with same style but difference structures can be obtained with a feed-forward network.

Alternatively, nonparametric methods [16–20] generate a new group of images with a query image and retrieve meaningful visual content from another image database. Among them, making use of neighbors is close to our approach. Liu *et al.* [16] proposed a two stage approach for face hallucination where overall structure is learned with global constraints, and local constraints emit local details of faces. Hays *et al.* [17] presented a scene completion system that match a query image in a 2 million image set. In particular, PixelNN was proposed by Bansal *et al.* [20], which is a two-step example-based image generation pipeline. They first leveraged a CNN to generate a coarse image, and then search patch level nearest neighbors with perceptual similarity [12] from the training data for replacing. By sampling different nearest neighbors, PixelNN can produce multiple diverse examples.

Differences to aforementioned works. Our method focuses on designing an efficient network and efficiently generating multiple examples. Different to [6,14], we enforce the one-to-many mapping to multiple branches instead of only updating one branch. In terms of network architectures, our model is able to emit more examples than the number of branches by sampling a new latent code for our dropout during test (see section 3.1). Although the recent PixelNN [20] is able to synthesize images with multiple outputs as well, a nearest neighbor search procedure is required during test, making the method less suitable for fast image synthesis. In comparison, our approach directly produces large number of images.

3 Method

Despite the recent progress at the intersection of Generative Adversarial Networks and Conditional Variational Autoencoders, improving stability and increasing the generated diversity are topics of ongoing research. Recent work has shown strong results for conditional image generation in a regression framework that greatly improves stability, but comes with the caveat of no latent code representation, no sampling mechanism and limited diversity in the output [6]. We seek a model that produces accurate and diverse samples, which is stable to train and provides a sampling mechanism with a latent code representation.
Model: We propose an image generation system that is based on stochastic regression with latent drop-out codes as shown in Fig. 1. While we are using stable codes to train regression formulation, we are not limiting ourselves to a fixed number of samples due to a fixed number of branches. We rather generate an arbitrary number of branches via dropout patterns derived from a random vector z. In turn, z is characteristic for each sample and acts as a latent code representation.
Training: We sample a set of latent codes – each corresponding to a branch with different dropout pattern that generates one sample. We minimize our new "Neighbors enhanced loss function" which increases the coverage of the training set by those generated samples.
Test: Our model can use both the above training latent codes and newly generated latent codes to produce an arbitrary number of new images. As each images is associated with a latent code, this also allows for additional manipulations like interpolation.

3.1 Stochastic Regression with Latent Drop-out Codes

Formally, given an input X, our stochastic regression model produces multiple outputs $\{Y^i\}$ with different dropout patterns controlled by a set of latent codes $\{z^i\}$ which are drawn from a random distribution. Fig. 2 shows our stochastic model, where $f_1(\cdot)$ and $f_2(\cdot)$ are composed of several convolution, nonlinear, pooling or up-sampling operations. Particularly, $d(\cdot)$ is a function transferring latent codes into binary dropout patterns for selecting features in our network.

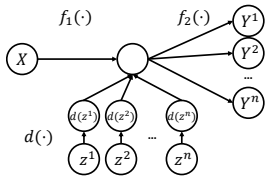

Fig. 2. Latent codes based stochastic multiple branch model.

Given a latent code z^i, our model can be formulated as

$$Y^i = f_2(f_1(X), d(z^i)). \tag{1}$$

Hence, f_1 is shared among all branches and samples, while f_2 depends on the randomized dropout pattern. During training, we fix the latent codes z^i and perform regression on the training set. Yet, in addition to using the training latent codes, we can also use a new code to generate more examples at test time.

Traditional dropout [21] randomly selects activations of a feature map for all channels and locations. This, however, does not lead to separated branches in a network. Therefore, we propose *channel-wise dropout* that, different to traditional dropout, selects the same features for all the locations. Different channels usually encode different kinds of visual cues like color, parts or objects as revealed in the research of understanding neural networks [22, 23]. Therefore, our model selects different visual cues for generating multiple outputs with diverse visual properties.

Consider a feature map $I \in R^{C\times H\times W}$ having C channels with size of $H \times W$, the feed-forward operation of channel-wise dropout for the c-th channel at location (h, w) of the i-th latent code can be described as

$$\begin{aligned} &z_c^i \sim U(0,1), \\ &d(z_c^i) = \begin{cases} 1, & \text{if } z_c^i > r \\ 0, & \text{otherwise} \end{cases} \\ &O(c,h,w) = \frac{I(c,h,w) \times d(z_c^i)}{1-r}, \end{aligned} \tag{2}$$

where z_c^i is a scalar for the latent code of c-th channel, $r \in (0, 1)$ is the dropout ratio, and $O \in R^{C\times H\times W}$ is the output of our dropout.

Interestingly, we are able to perform interpolation between two generation images in the test as shown in Fig. 4, and the interpolated output can be described as

$$Y_{ij}(a) = f_2(f_1(X), d(a \cdot z^i + (1-a) \cdot z^j)). \tag{3}$$

3.2 Neighbors Enhanced Loss Function

In order to improve diversity, we propose a new *Neighbors enhanced loss function* that samples neighbors with respect to a condition input as shown in

Fig. 1. We leverage the sampled neighbors to encourage diversity during training. We update multiple branches for a network by assigning sampled neighbors to different branches. We first describe a simpler loss function based on the best neighbor (similar to MCL [14]) and then our Neighbors enhanced loss function for increased diversity.

Best neighbor loss. With our stochastic model formulated in Eq. 1, the scheme generates n hypotheses $\{f_2(f_1(X), d(z^i))|i = 1, 2, ..., n\}$ from the same input X. A simple version would only update the best branch which has the smallest loss. Formally, such a loss function for a batch $\{(X_m, Y_m)|m = 1, 2, ..., M\}$ with size M is defined as

$$L = \sum_{m=1}^{M} \min_i l(f_i(X_m), Y_m), \tag{4}$$

where $l(\cdot)$ is a $L1$-based perceptual loss in this paper, defined in Eq. 6, and $f_i(X_m) = f_2(f_1(X_m), d(z^i))$ for short.

Neighbors enhanced loss. Given a training pair (X^0, Y^0), we first sample several data pairs $\{(X^i, Y^i)|i = 1, ..., N\}$ satisfying the inputs $\{X^i\}$ are close enough to X^0, i.e., $\{dis(X^i, X^0) < \theta|i = 1, ..., N\}$. We directly approximate the conditional distribution $P(Y|X^0)$ at X^0 by $\{(X^0, Y^i)|i = 0, 1, .., N\}$.

With data pairs $\{(X^0, Y^i)|i = 0, 1, .., N\}$ and n network output hypotheses $\{f_i(X^0)|i = 1, ..., n\}$. We design a neighbors assignment procedure to give a sampled images to one of branches as a ground truth. We assign those sampled neighbors iteratively. Ideally, we hope to assign every samples Y^i to its best hypothesis $f_{best}(X^0)$ where the loss $l(f_{best}(X^0), Y^i)$ is smaller than any other hypothesis. However, there might be more than one sample assigned to the same hypothesis with this condition. To address this issue, we design several assigning rules. First, Y^0 is supposed to assign to its best branch, as (X^0, Y^0) is a well-aligned data pair while others are approximations. Second, because each branch is able to has only one ground truth, we assign the sample with smallest loss if there are more than one sample for the same hypothesis. The matching is proceeded iteratively until all the sampled neighbors are assigned, and output a matching set $S_0 = \{(f_i(X^0), Y^j)\}$.

After applying neighbors assignment, the neural network is optimized with standard back propagation [24]. We formulate the neighbors enhanced loss function on a batch $\{(X_m, Y_m)|m = 1, 2, ..., M\}$ with the neighbors $\{Y_m^j\}$ as

$$L = \sum_{m=1}^{M} \sum_{(f_i(X_m), Y_m^j) \in S_m} l(f_i(X_m), Y_m^j), \tag{5}$$

where S_m is the matching set for m-th example in a batch. Particularly, we utilize $L1$-base perceptual loss [12, 13] for optimization in this paper, that is

$$l(X, Y) = \sum_i \lambda_i |\Phi_i(X) - \Phi_i(Y)|, \tag{6}$$

where λ_i is the loss weights and Φ_i is the i-th representation form a network Φ.

3.3 Architectures and Parameter-Sharing

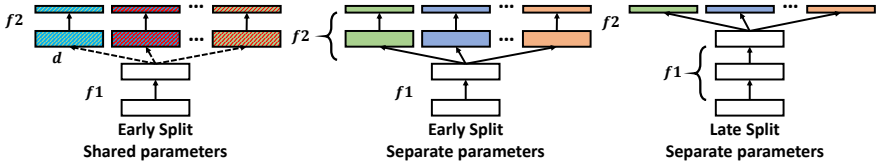

Fig. 3. The illustration of comparison between different network architectures to produce multiple outputs. Solid lines are deterministic modules and dash lines are stochastic modules.

Our regression model applies channel-wise dropout to split the network into a multiple branches architecture. Because we apply our channel-wise dropout to select different feature maps for multiple branches, we do not have to learn separate parameters for different branches, which will not increase any model size when we increase the number of network branches. Secondly, we embed channel-wise dropout earlier instead of end of a network, which we call "early split" strategy. "Early split" is able to evolute information in a network earlier on a higher dimension, which benefits the generation of diverse examples than the "late split" as shown in Fig. 3.

When removing our channel-wise dropout, we learn separate parameters for different branches, and thus our full model degenerates to the "early split with separate parameters" setting. In that case we can only generate a fixed number of outputs and need more parameters to represent a model. Comparing to the "late split" setting, used in previous work [6], it still has the merit that already intermediate layers contribute to generate different samples, which we show experimentally to be important to generate more diverse images.

Furthermore, our neighbors enhanced loss function can be applied to all architectures in Fig. 3. Also, CRN [6] is a "degernated" case of our approach with deterministic late split network architecture and no neighbors considered.

3.4 Discussion

Comparison to Multiple Choice Learning (MCL). Our approach generalizes the MCL scheme [14] with sampled neighbors and stochastic regression. MCL is a kind of ensemble learning that produce multiple outputs of high quality. In MCL [14], only one branch gets gradients to update the model. It is therefore not as efficient to learn a model due to limited parameter sharing, especially in the case of a large number of network branches to produce highly diverse examples. It also lacks the latent code representation and sampling capability.

4 Experimental Results

We compare our approach for conditional image generation to state of the art methods [4–6, 20] on established datasets for generating human faces from facial landmarks and animal heads from normal maps. We systematically compare three different architecture choices in Fig. 3 under different loss functions (CRN [6], "Separate" model and latent codes based "Shared" model) along two dimensions (accuracy and diversity). We implement the proposed network using the *Caffe* framework [25], and the source code is available at https://github.com/SSAW14/Image_Generation_with_Latent_Code.

4.1 Datasets and Implementation Details

Oxford-IIIT Pet [26]: It contains 3,868 images of cats and dogs with bounding boxes of animal heads. In our experiments, we use 3,000 images to train a model and 686 images to test the model, which follows previous work [20]. We first use bounding boxes to crop the animals' heads and resize them into 96 $\times$ 96 pixels. Finally, we utilize PixelNet [27] to estimate the normals.
LFW [28]: We utilize the deep funneling aligned LFW dataset, and apply the peopleDevTrain/peopleDevTest split containing 4,038 and 1,711 images to train and evaluate the performance. For each image, we first employ the MTCNN [29] face detection model to extract faces. Next, we employ the TCDCN [30] extracting 68 facial landmarks for each face, and use the heat map of key points as the input of the network. For all the faces, we resize the bounding box regions into 128$\times$128 pixels, and thus we generate 128$\times$128 color faces from the input with size 128 $\times$ 128 $\times$ 68.
Implementation details. We implement our channel-wise dropout and networks with the Caffe [25] deep learning framework. We set dropout ratio as $r = 0.5$ for all the channel-wise dropout in our experiments. For all models, we apply Adam [31] to optimize our models and use the "poly" learning rate policy that current learning rate is $lr_{init} \times (1 - \frac{iter}{iter_{max}})^{power}$. And we set power as 0.9 and initial learning rate lr_{init} as 1×10^{-4}. We set max iterations number as 110,000 and 90,000 for animal head generation and face generation tasks, respectively. To have a fair comparison of separate parameters networks and shared parameters networks, we split feature representation or introduce channel-wise dropout at the same location. For CRN and our models, we generate 96$\times$ 96 images for head animal generation task, and generate 128$\times$ 128 images for face generation We will release our source code at the time of publication.

4.2 Animal Head Generation from Normal Map

In this experiment, we test three kinds of architectures: For CRN and "Separate" model, we learn a set of models with 2, 4, 8, 20, 30, 40, 50, 72 branches. For our "Shared" models, we learn 4, 8, 20, 72 branches to test.

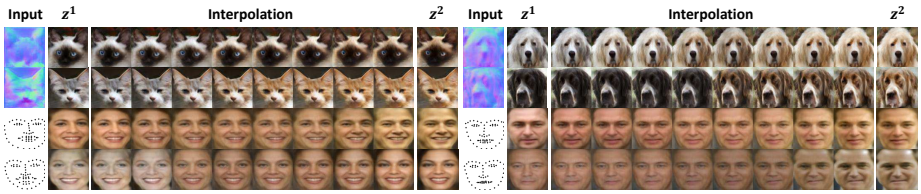

Fig. 4. Interpolation results from the proposed architecture using channel-wise dropout. Images below z^i correspond to the respective latent code.

Quantitative results and analysis. In terms of accuracy, we apply root mean square error (RMSE), SSIM [32] and FSIM [33] as the measurements for appearance similarity. Besides, we also evaluate the consistency of predicted normals from generated images and real images with 6 evaluation criteria following [20, 34]. We report the performance against strong competing methods pix2pix [5], BicycleGAN [4], PixelNN [20] and CRN [6] on the accuracy of normal prediction with generated images. For all those methods [4–6, 20], we choose the best example among all 72 generated heads and compare results in Table 1. We also report the performance of our final "Shared" models with different branches. We can observe that our model with 72 branches achieves best performance . And our 4 branches model also achieves better results than pix2pix [5], BicycleGAN [4] and PixelNN [20], and only 6% worse than CRN [6].

Next, we provide comparison plots for different architectures including CRN, our "Separate" model and our "Shared" model with channel-wise dropout as shown in Fig. 5. In this figure, we show the best performance for all 9 evaluation metrics used in our experiments. For our "Shared" models, we use both the training latent codes and newly sampled codes to generate 100 outputs. Differently colored curves show the performance of sampling different number of examples. First, we observe that the performance of the best example gradually increases for all the architectures when the number of branches is increased. Second, we can see that the performance of our "Shared" model is better than other two when the number of branches is large, which means our models generate better results with a large number of branches even without sampling more outputs.

Table 1. Comparison of predicted normals of best generated images.

Method	Mean	Median	RMSE	11.25°	22.5°	30°
pix2pix [5]	13.2	11.4	15.7	49.2	87.1	95.3
BicycleGAN [4]	21.6	19.3	24.9	24.3	60.2	77.5
PixelNN [20]	13.8	11.9	16.6	46.9	84.9	94.1
CRN [6]	11.8	10.3	13.9	56.3	91.4	97.6
Ours-4	12.4	10.9	14.5	52.9	90.0	97.0
Ours-8	12.4	10.8	14.4	53.6	90.2	97.1
Ours-20	12.0	10.5	14.1	55.2	91.1	97.6
Ours-72	**11.7**	**10.2**	**13.7**	**56.7**	**91.9**	**97.8**

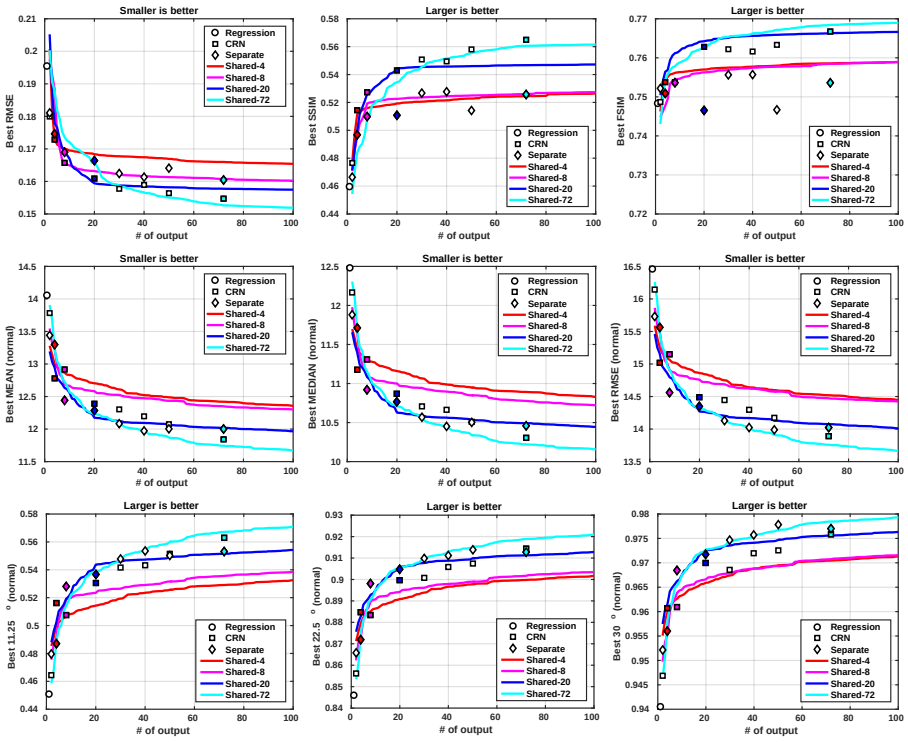

Fig. 5. Evaluation of best example on Oxford-IIIT Pet dataset. The first row draws the evaluation plot for appearance similarity. The bottom two rows show the evaluation plots for the consistency of predicted normals between generated images and real images. "Regression" model produces one output image. For CRN and "Separate" models, we learned eight models (2, 4, 8, 20, 30, 40, 50, 72 branches) to test them. For "Shared" model, we learned four models(4, 8, 20, 72 branches) and samples totally 100 outputs during test time. Best viewed in color.

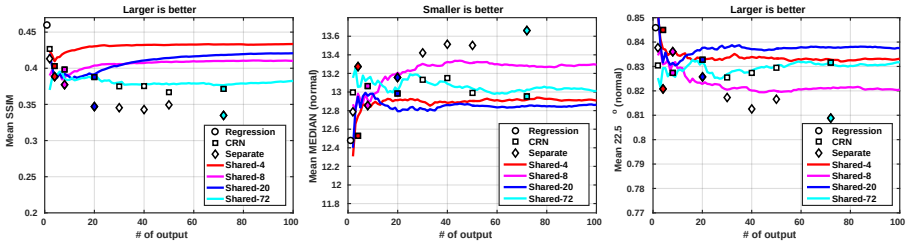

Fig. 6. Evaluation of average performance of all the generated images on Oxford-IIIT Pet dataset.

Finally, we observe that the red, pink, blue and cyan lines always become better.

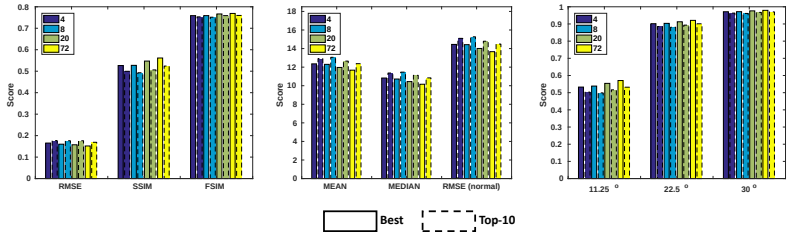

Fig. 7. Comparison of best example and top-10 examples on Oxford-IIIT Pet dataset. For RMSE, MEAN and MEDIAN, smaller is better. For other measurements, larger is better.

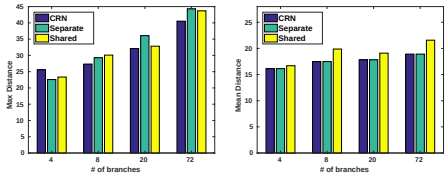

Fig. 8. Evaluation of diversity on Oxford-IIIT Pet dataset. It reports the max distance and mean distance to the average of generated images.

Fig. 9. Channel-wise dropout vs. dropout. In each block, top row shows the results from the network with channel-wise dropout, and bottom row shows the results applying dropout.

This means our latent codes based regression models generate better examples with new codes, even though they have never been used during training.

To further demonstrate the overall quality of samples drawn by our models, we also analyze three plots of average performance in Fig. 6 on the appearance similarity and the consistency of normal prediction. We observe that all of our "Shared" models maintain their performance comparing to the case of using training latent codes. Besides, we also report the performance of top-10 examples comparing to the best one in Fig. 7. This figure shows that the average performance of top-10 examples is very similar to the best number, which is another evidence for the effective sampling of our approach.

Diversity analysis. To evaluate the diversity quantitatively, we compute the max distance and mean distance of all the generated images to their center, and present the comparison in the Fig. 8. The results show that an early split improves diversity over the late split, which is used in CRN [6]. Besides, this figure also shows that our "Shared" model performs comparably to our "Separate" model, even it has smaller model size and is able to generate an arbitrary number of examples at test time. Beyond sampling, we also show in Fig. 4 interpolation results using Eq. 3. The smooth transition between the samples corresponding to z_1 and z_2 gives evidence that the latent codes indeed serves as a meaningful representation.

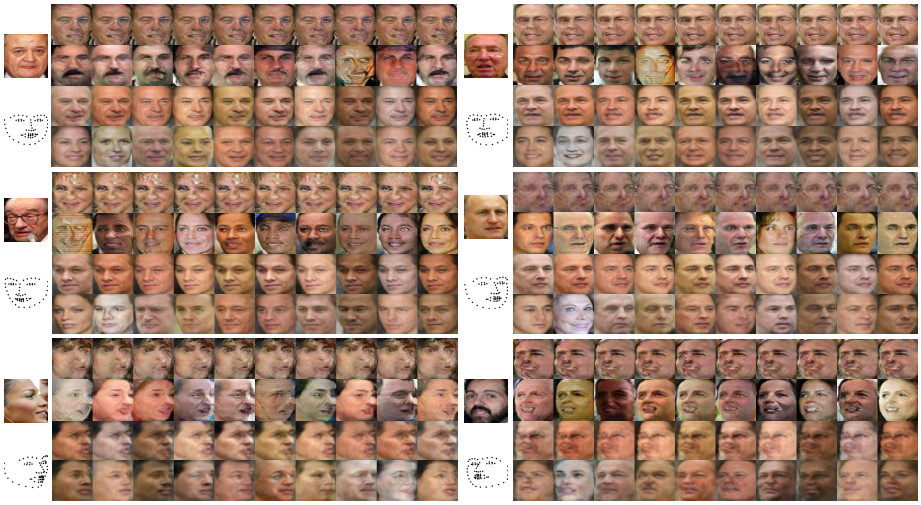

Fig. 10. The example and comparison with competing methods [4–6] on the task of face generation from landmarks. Best viewed in color.

Channel-wise dropout vs. classic dropout. Traditional dropout [21] can be also used in our architecture. However, it cannot select or reject a whole feature channel like our channel-wise dropout. We provide a comparison between our channel-wise dropout and traditional dropout in the proposed model. Fig. 9 shows some visualization results, which clearly demonstrate that channel-wise dropout successfully selects different features and then generates animals with different color, while dropout generate more similar animal heads.

4.3 Face Generation from Facial Landmarks

In this task, we generate human faces from 68 facial landmarks. For both models, we generate 10 output faces from the networks. Except evaluating on the architectures, we also test our neighbor assignment strategy for improving diversity. After acquiring neighbors, we use kmeans on HSV color space to cluster the neighbors. For each cluster, we randomly select an example as the sampled neighbor. We use $L1$ distance between the coordinates of two landmarks.

Qualitative comparison. We show generated faces in Figure 10. For each of the six blocks, the left two images show the facial landmarks and the corresponding image. Next to them results of pix2pix [5], BicycleGAN [4], CRN [6] and our "Shared" model are shown row by row. Top two blocks show "normal" case, middle blocks show landmarks of immediate difficulty and bottom blocks show the hard examples. In the top two blocks, we observe that our model is not only able to generate faces with different skin, but also to generate faces from different identity or gender, or with different local details. While the baseline

Table 2. Accuracy comparison of face image generation on LFW dataset. We report the average as well as the best performance (Mean / Best) on four measurements.

Method	Neighbors	RMSE	SSIM	FSIM	Landmarks
pix2pix [5]		9.952/9.649	0.696/0.708	0.749/0.755	1.913/1.645
BicycleGAN [4]		9.948/6.856	0.621/0.715	0.725/0.758	1.704/2.197
CRN [6]		9.107/5.414	**0.705**/**0.767**	**0.760**/**0.779**	**1.697**/1.452
Our CRN	✓	**8.612**/5.309	0.687/0.763	0.754/0.777	1.863/1.440
Ours (Separate)		8.871/5.528	0.704/0.761	0.755/**0.779**	1.718/**1.308**
Ours (Separate)	✓	9.034/5.768	0.678/0.757	0.748/0.774	1.915/1.442
Ours (Shared)		8.908/**5.269**	0.701/0.762	0.757/**0.779**	1.728/1.365
Ours (Shared)	✓	8.956/5.679	0.686/0.764	0.753/**0.779**	1.864/1.453

CRN [6] just generates faces with very similar structure, even different colors of skin are also covered. Second, comparing to pix2pix [4] and BicycleGAN [4], which try to apply GAN to generate realistic faces, we also generate faces with better quality. Although BicycleGAN can generate many visually realistic faces, it also generates some very poor results due to its unstable training. Finally, observing the last row, we even generate plausible faces in hard cases.

Accuracy analysis. We evaluate our method and compare to pix2pix [5] and CRN [6]. For pix2pix [5] and BicycleGAN [4], we run the code provided by authors with default settings except increase the training epochs from 200 to 300. For CRN, we use the same number of feature maps and branches to ours.

For accuracy, we apply root RMSE, SSIM [32] and FSIM [33] to evaluate the appearance similarity between generated faces and ground truth faces. Besides, we also run the landmark detector [30] on the generated faces and compare the accuracy of detected landmarks and ground truth landmarks. The accuracy of landmarks is measured by the sum of distance of each predicted points to ground truth normalized by the width of the images. We report the best performance as well as average performance among 10 outputs as summarized in Table 2.

From Table 2, we can observe that our models and CRN achieves better performance than pix2pix and BicycleGAN in all four metrics on best performance and mean performance, benefiting from the cascaded refinement network architecture and the effectiveness of perceptual loss. Comparing to CRN, the overall performance of our "Shared" and "Separate" models are comparable in four metrics. Even the quantitative performance decreases a little in SSIM and landmarks accuracy after applying neighbors enhanced loss function, the decrease is just a little and we found that our approach can generate better faces visually as shown in Fig. 10.

Diversity analysis. To quantify the measurement of diversity for multiple output from the same landmarks, we compute standard deviations of different levels of face representation from [35] and an identity embedding for face recognition from [36]. Table 3 lists the scores for competing approaches. Clearly, our "Separate" and "Shared" models achieves better diversities than CRN [6]. Besides, we also observe that the standard deviations get consistent and significant improvements for CRN, "Separate" and "Shared" models, when we apply our neighbor

Table 3. Standard deviation of the convolutional features from [35] and the identity embedding from [36]. For all the values, larger is better.

Method	Neighbors	pool1	pool2	pool3	pool4	fc5	identity
pix2pix [5]		0.134	0.359	0.364	0.193	0.370	0.660
BicycleGAN [4]		**0.428**	**0.817**	**0.716**	0.350	0. 696	**2.055**
CRN [6]		0.198	0.337	0.350	0.186	0.389	1.098
Our CRN	✓	0.272	0.549	0.591	0.322	0.669	1.299
Ours (Separate)		0.225	0.491	0.551	0.302	0.610	1.235
Ours (Separate)	✓	0.276	0.596	0.640	**0.353**	**0.733**	1.427
Ours (Shared)		0.210	0.422	0.471	0.259	0.523	1.198
Ours (Shared)	✓	0.310	0.591	0.617	0.322	0.670	1.370

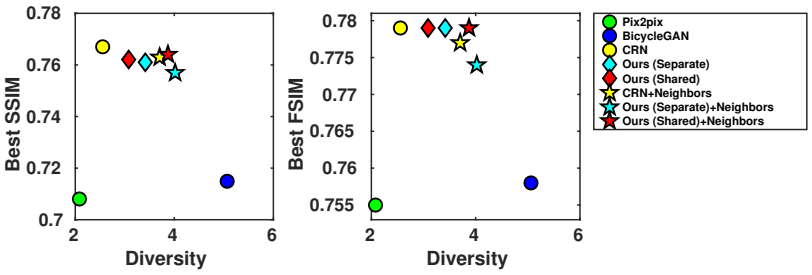

Fig. 11. Accuracy-Diversity plots on LFW dataset. The diversity score is the sum of stand deviation of all the representations in Table 3.

enhanced loss function. It clearly shows that the effectiveness of assigning sampled neighbors. Particular, recent proposed BicycleGAN model has the best score on $pool_1$, $pool_2$, $pool_3$ and *identity*, but the accuracy of BicycleGAN is not comparable to our generation results. More importantly, diversity and accuracy are often a trade-off. To observe this relationship clearly, we plot a accuracy-diversity scatter plot in Fig. 11, whose top-right corner is best. It shows that our stochastic regression model and neighbors enhanced loss function both contribute to generating more diverse example while maintain a comparable accuracy.

5 Conclusions

In this paper, we have presented a novel image generation approach, which learns to produce multiple diverse examples from a single conditional input. We have tested our method on the tasks of generating human faces from facial landmarks and animal heads from normal maps. Based on a series of ablation studies and comparisons with state-of-the-art image generation frameworks, we demonstrate the effectiveness of enforcing diversity by sampling neighboring examples, and efficiency of our network architectures by introduce channel-wise dropout to randomly select feature maps to generate accurate and diverse images of various styles and structures.

References

1. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NIPS. (2014)
2. Sohn, K., Lee, H., Yan, X.: Learning structured output representation using deep conditional generative models. In: NIPS. (2015)
3. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
4. Zhu, J.Y., Zhang, R., Pathak, D., Darrell, T., Efros, A.A., Wang, O., Shechtman, E.: Toward multimodal image-to-image translation. In: NIPS. (2017)
5. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR. (2017)
6. Chen, Q., Koltun, V.: Photographic image synthesis with cascaded refinement networks. In: ICCV. (2017)
7. Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H.: Generative adversarial text to image synthesis. In: ICML. (2016)
8. Zhang, H., Xu, T., Li, H., Zhang, S., Huang, X., Wang, X., Metaxas, D.: Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In: ICCV. (2017)
9. Wang, X., Gupta, A.: Generative image modeling using style and structure adversarial networks. In: ECCV. (2016)
10. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., Efros, A.A.: f: Feature learning by inpainting. In: CVPR. (2016)
11. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. arXiv preprint arXiv:1609.04802 (2016)
12. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: ECCV. (2016)
13. Dosovitskiy, A., Brox, T.: Generating images with perceptual similarity metrics based on deep networks. In: Advances in Neural Information Processing Systems. (2016) 658–666
14. Guzman-Rivera, A., Batra, D., Kohli, P.: Multiple choice learning: Learning to produce multiple structured outputs. In: NIPS. (2012)
15. Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., Yang, M.H.: Diversified texture synthesis with feed-forward networks. (2017)
16. Liu, C., Shum, H.Y., Freeman, W.T.: Face hallucination: Theory and practice. IJCV (2007)
17. Hays, J., Efros, A.A.: Scene completion using millions of photographs. In: SIGGRAPH. (2007)
18. Efros, A.A., Leung, T.K.: Texture synthesis by non-parametric sampling. In: ICCV. (1999)
19. Efros, A.A., Freeman, W.T.: Image quilting for texture synthesis and transfer. In: SIGGRAPH. (2001)
20. Bansal, A., Sheikh, Y., Ramanan, D.: Pixelnn: Example-based image synthesis. ICLR (2018)
21. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. JMLR (2014)
22. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Object detectors emerge in deep scene cnns. In: ICLR. (2015)

23. Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: Quantifying interpretability of deep visual representations. In: CVPR. (2017)
24. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature (1986)
25. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: ACM Multimedia. (2014)
26. Parkhi, O.M., Vedaldi, A., Zisserman, A., Jawahar, C.: Cats and dogs. In: CVPR. (2012)
27. Bansal, A., Chen, X., Russell, B., Ramanan, A.G., et al.: Pixelnet: Representation of the pixels, by the pixels, and for the pixels. arXiv preprint arXiv:1702.06506 (2017)
28. Learned-Miller, E., Huang, G.B., RoyChowdhury, A., Li, H., Hua, G.: Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Advances in Face Detection and Facial Image Analysis (2016)
29. Zhang, K., Zhang, Z., Li, Z., Qiao, Y.: Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Processing Letters (2016)
30. Zhang, Z., Luo, P., Loy, C.C., Tang, X.: Facial landmark detection by deep multi-task learning. In: ECCV. (2014)
31. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
32. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. TIP (2004)
33. Zhang, L., Zhang, L., Mou, X., Zhang, D.: Fsim: A feature similarity index for image quality assessment. TIP (2011)
34. Wang, X., Fouhey, D., Gupta, A.: Designing deep networks for surface normal estimation. In: CVPR
35. Wen, Y., Zhang, K., Li, Z., Qiao, Y.: A discriminative feature learning approach for deep face recognition. In: ECCV. (2016)
36. Oh, S.J., Fritz, M., Schiele, B.: Adversarial image perturbation for privacy protection–a game theory perspective. In: ICCV. (2017)