Supplementary Information: Learning Classwise Untangled Continuums for Conditional Normalizing Flows

Victor Enescu and Hichem Sahbi

Sorbonne University, CNRS, LIP6, F-75005, Paris, France
victor.enescu@lip6.fr

1 ImageNet extra results

Reminder: We evaluated the performance of our NF based augmentations on ImageNet1k dataset [3], by fine-tuning very powerful, pretrained transformers from the timm [15] library. We selected the most performant transformer with less than 100M parameters by referring ourselves to this result page from timm. It is an EVA02 [5,4] transformer originally ranked 9 (in the list), with 87.12M parameters, that was pretrained on ImageNet21k [12], and then fine-tuned on ImageNet. It takes images of size 448×448 pixels, and is called *"eva02_base_patch14_448.mim_in22k_ft_in22k_in1k"* in timm. ImagetNet Real [1] labels were used for testing, as they fix labeling mistakes present in the original testing set.

Extra results: In Table 1 we compare our NF augmentations (NFA) against the baseline, and standard augmentations (SA) commonly used in state-of-the-art attention based models [10]. Those standard augmentations include RandAugment [2] and RandomErasing [16], and are implemented using timm. It can be noted that highest accuracies are reached using GLEM augmentations, which achieve a top-1 (resp. top-5) accuracy of 91.229% (resp. 98.854%). Concerning training details, we use SGD with a global gradient clipping at norm 1. The models are trained for a total of 3 epochs, with a learning rate equal to $1.0 \cdot 10^{-5}$, annealed every epoch using SGDR [11] with default parameters from Pytorch. This results in learning rates respectively equal to $[1.0 \cdot 10^{-5}, 0.75 \cdot 10^{-5}, 0.25 \cdot 10^{-5}]$ at each epoch. For all experiments, the dataset is duplicated 3 times with augmented images.

Table 1: Comparison of our NF augmentations (NFA): with CE and GLEM, against the Baseline and Standard Augmentations (SA) when fine-tuning the EVA02 transformers on ImageNet.

Top1 Acc				Top5 Acc			
Baseline	SA	CE	GLEM	Baseline	SA	CE	GLEM
90.896	91.191	91.182	**91.229**	98.802	98.836	98.845	**98.854**

1.1 Augmented Images on ImageNet

In this section, we show augmented images using GLEM on ImageNet. The Image on the left is the original one, and the 3 images on the right are augmented by randomly disrupting the coordinates of the original images in the latent space, along the principal variation modes of the underlying classes.

Fig. 1: Augmentation on class 22.

Fig. 2: Augmentation on class 14.

Fig. 3: Augmentation on class 2.

Fig. 4: Augmentation on class 0.

Fig. 5: Augmentation on class 6.

Fig. 6: Augmentation on class 60.

Fig. 7: Augmentation on class 511.

Fig. 8: Augmentation on class 895.

Fig. 9: Augmentation on class 814.

Fig. 10: Augmentation on class 849.

Fig. 11: Augmentation on class 794.

Fig. 12: Augmentation on class 523.

1.2 Interpolated Images on ImageNet

In this section, we show selected interpolations between images from identical or different classes. We use the rescaled interpolation from [7].

Fig. 13: Interpolation on class 1.

Fig. 14: Interpolation on class 3.

Fig. 15: Interpolation on class 500.

Fig. 16: Interpolation on class 510.

Fig. 17: Interpolation on class 946.

Fig. 18: Interpolation on class 637.

Fig. 19: Interpolation on class 327.

Fig. 20: Interpolation between random classes.

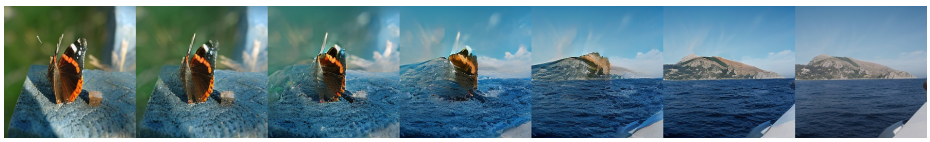


Fig. 21: Interpolation between random classes.

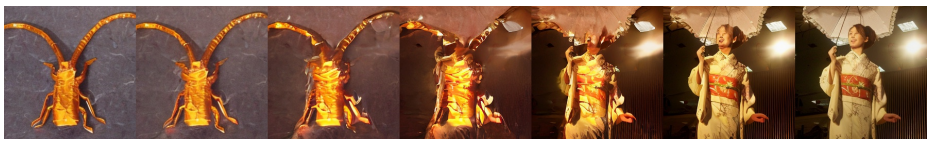


Fig. 22: Interpolation between random classes.

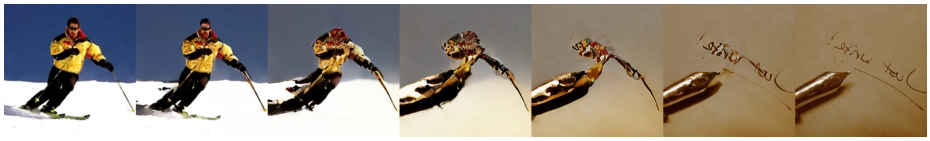


Fig. 23: Interpolation between random classes.

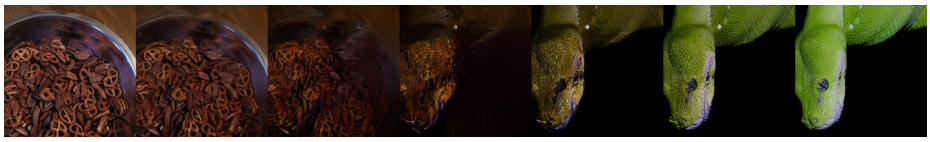


Fig. 24: Interpolation between random classes.

Fig. 25: Interpolation between random classes.

Fig. 26: Interpolation between random classes.

Fig. 27: Interpolation between random classes.

1.3 Sampled Images on ImageNet

In this section, we show class conditional image samples from ImageNet, that were filtered with a classifier.

Fig. 28: Sample from class 980 (Volcano).

Fig. 29: Samples on class 947 (Mushroom).

Fig. 30: Samples on class 109 (Brain coral).

Fig. 31: Samples on class 414 (Backpack).

2 Proposition 1 and its proof

Equations are numbered starting from those in the main paper.

Proposition 1. *Let $P_{\mathbf{X}}(.|\mathbf{y}_1)$, $P_{\mathbf{X}}(.|\mathbf{y}_2)$ be two probability distributions defined on a compact $\mathcal{X} \subseteq \mathcal{B}^d$. The optimization of Eq.3 leads to*

$$\mathbb{E}\big[(P_{\mathbf{Z}}(\mathbf{Z}|\mathbf{y}_1) - P_{\mathbf{Z}}(\mathbf{Z}|\mathbf{y}_2))^2\big] \leq \mathbf{B}, \tag{7}$$

with

$$\mathbf{B} = \mathbb{E}\big[(P_{\mathbf{X}}(\mathbf{X}|\mathbf{y}_1) - P_{\mathbf{X}}(\mathbf{X}|\mathbf{y}_2))^2\big] \times \frac{\exp(1)}{\det(\Sigma_{\mathbf{y}^*})}, \tag{8}$$

being $\mathbf{y}^* = \arg\min_{\mathbf{y}\in\mathcal{Y}} \det(\Sigma_{\mathbf{y}})$ and $\mathcal{B}^d$ a zero centered unit ball enclosing data[1] in $\mathcal{X}$.

Details of the proof are given below. More importantly, the bound in Eq. 8 shows that when optimizing Eq. 3 only, condition (i) is achieved (see line 54 in the main paper) but at the detriment of (ii), (see line 56 in the main paper); i.e., different classes may result into highly confounded gaussians, and this makes their label conditioning erroneous.

Proof 1. Using Eq.2 (line 186 in the main paper), one may write

$$\mathbb{E}\big[(P_{\mathbf{X}}(\mathbf{X}|\mathbf{y}_1) - P_{\mathbf{X}}(\mathbf{X}|\mathbf{y}_2))^2\big] = \mathbb{E}\big[(P_{\mathbf{X}}(f(\mathbf{X})|\mathbf{y}_1) - P_{\mathbf{X}}(f(\mathbf{X})|\mathbf{y}_2))^2 . \det(\mathbf{J}_{f(\mathbf{X})})^2\big], \tag{9}$$

here the expectation is w.r.t. the marginal distribution of $\mathbf{X}$. The class of widely used flows[2] can be written using quasi-linear mapping as $\mathbf{Z} = f(\mathbf{X}) = \mathbf{W}_{\mathbf{X}}\mathbf{X} + \mathbf{c}_{\mathbf{X}}$, with $(\mathbf{Z}|\mathbf{Y}) \sim \mathcal{N}(\mu_{\mathbf{Y}}, \Sigma_{\mathbf{Y}})$. With this quasi-linear form, $\det(\mathbf{J}_{f(\mathbf{X})})) = \det(\mathbf{W}_{\mathbf{X}})$, and $\exists \mathbf{x}_0 \in \mathcal{X}$ s.t.

$$\frac{\mathbb{E}\big[(P_{\mathbf{Z}}(\mathbf{Z}|\mathbf{y}_1) - P_{\mathbf{Z}}(\mathbf{Z}|\mathbf{y}_2))^2\big]}{\mathbb{E}\big[(P_{\mathbf{X}}(\mathbf{X}|\mathbf{y}_1) - P_{\mathbf{X}}(\mathbf{X}|\mathbf{y}_2))^2\big]} \leq \frac{1}{\det(\mathbf{W}_{\mathbf{x}_0})^2}, \tag{10}$$

by plugging $f(\mathbf{X})$ in Eq.3 (line 195 in the main paper), $\mathcal{L}_{NF}$ becomes

$$\mathbb{E}\left[\frac{1}{2}(\mathbf{W}_{\mathbf{X}}\mathbf{X}+\mathbf{c}_{\mathbf{X}}-\mu_{\mathbf{Y}})'\Sigma_{\mathbf{Y}}^{-1}(\mathbf{W}_{\mathbf{X}}\mathbf{X}+\mathbf{c}_{\mathbf{X}}-\mu_{\mathbf{Y}})+\frac{1}{2}\log(\det(\Sigma_{\mathbf{Y}}))-\log|\det(\mathbf{W}_{\mathbf{X}})|\right]. \tag{11}$$

For $\mathbf{X} \simeq \mathbf{x}_0$, the stationary solution of $\mathcal{L}_{NF}$ w.r.t. $\mathbf{W}_{\mathbf{x}_0}$ leads to

$$\Sigma_{\mathbf{y}_0}^{-1}(\mathbf{c}_{\mathbf{x}_0} + \mathbf{W}_{\mathbf{x}_0}\mathbf{x}_0 - \mu_{\mathbf{y}_0})\mathbf{x}_0' - \frac{\mathbf{sign}(\det(\mathbf{W}_{\mathbf{x}_0}))}{|\det(\mathbf{W}_{\mathbf{x}_0})|}.\mathbf{adj}(\mathbf{W}_{\mathbf{x}_0})' = 0 \tag{12}$$

[1] This is easily obtainable by rescaling the data in $\mathcal{X}$.

[2] including linear mapping, affine and additive coupling layers as well as their composition.

since

$$\frac{\mathbf{sign}(\det(\mathbf{W}_{\mathbf{x}_0}))}{|\det(\mathbf{W}_{\mathbf{x}_0})|}.\mathbf{adj}(\mathbf{W}_{\mathbf{x}_0}) = \mathbf{W}_{\mathbf{x}_0}^{-1}, \tag{13}$$

and using Eq. 12

$$\mathbf{W}_{\mathbf{x}_0}(\mathbf{x}_0\mathbf{x}_0' + \mathbf{W}_{\mathbf{x}_0}^{-1}(\mathbf{c}_{\mathbf{x}_0} - \mu_{\mathbf{y}_0})\mathbf{x}_0')\mathbf{W}_{\mathbf{x}_0}' = \Sigma_{\mathbf{y}_0}, \tag{14}$$

being $\mathbf{x}_0\mathbf{x}_0'$ the autocorrelation matrix of $\mathbf{x}_0$. Let $I \in \mathbb{R}^{d\times d}$ be the identity matrix, since

$$\det((\mathbf{x}_0\mathbf{x}_0' + \mathbf{W}_{\mathbf{x}_0}^{-1}(\mathbf{c}_{\mathbf{x}_0} - \mu_{\mathbf{y}_0})\mathbf{x}_0') \leq \det(\mathbf{x}_0\mathbf{x}_0' + I), \tag{15}$$

Eq. 14 leads to

$$\begin{aligned}
\frac{1}{\det(\mathbf{W}_{\mathbf{x}_0})^2} &\leq \det(I + \mathbf{x}_0\mathbf{x}_0')\det(\Sigma_{\mathbf{y}_0})^{-1} \\
&\leq \left(\frac{\mathbf{tr}(\mathbf{x}_0\mathbf{x}_0' + I)}{d}\right)^d \det(\Sigma_{\mathbf{y}_0})^{-1} \\
&= \left(\frac{\mathbf{tr}(\mathbf{x}_0\mathbf{x}_0') + \mathbf{tr}(I))}{d}\right)^d \det(\Sigma_{\mathbf{y}_0})^{-1} \\
&= \left(\frac{\|\mathbf{x}_0\|_2^2 + d)}{d}\right)^d \det(\Sigma_{\mathbf{y}_0})^{-1} \\
&\leq \left(\frac{1}{d} + 1\right)^d \det(\Sigma_{\mathbf{y}_0})^{-1} \\
&\leq \lim_{d\to\infty}\left(\frac{1}{d} + 1\right)^d \det(\Sigma_{\mathbf{y}_0})^{-1} \\
&\leq \exp(1)\det(\Sigma_{\mathbf{y}^*})^{-1},
\end{aligned}$$

which also results from $\mathbf{x}_0 \in \mathcal{B}^d$. By plugging this upper bound in Eq. 10, we complete the proof.

■

3 Implementation Details

3.1 NFs

This section gives details on the training of the NFs. All experiments are run on Pytorch2, with V100 GPUs that have 16GB or 32GB RAM.

Table 2: NF models are built using L levels, and each one contains F step of flows. Each step of flow is made of N_i residual blocks (RBs) [6] with 128 hidden channels, where i denotes the index of the level L. The optimizer used to trained the NF is Adamax [9] with a learning rate of 10^{-2}. This learning rate is linearly warmed up for the first 1000 iterations, and is also divided by a factor of 2 at several intervals. For CIFAR datasets, those intervals are every 100 epochs starting from epoch 300 (included). In all cases, the learning rate reaches a minimum of $2.5 \cdot 10^{-3}$, however if the learning rate is smaller than $1.25 \cdot 10^{-3}$, it is capped at $1.25 \cdot 10^{-3}$. On CIFAR datasets, the last 100 epochs are run with Stochastic Weight Averaging [8]. The gaussian hyperparameters (μ, Σ) are updated once per epoch using Adam [9] optimizer, with linear and geometric schedulers. The weight λ for the $\mathcal{L}_{KLD}$ loss is also updated using linear and geometric schedulers. After a certain number of epochs, the D_{NF} and D_G datasets are merged, meaning the training of the gaussians is stopped. Only the NF model is trained once the datasets are merged.

Model	Bits	L	F	RBs Per Level	Epochs E	Epochs M	Batch Size	a; b; c;	Gaussian Adam and Scheduler	λ_{KLD} and scheduler	GPUs / Hours	Params (M)
CIFAR10 Ablations	5	3	8	[8, 4, 2]	500	100	512	2.0; 0.2; 0.1;	lr: [1e-1, 1e-1, 1e-2, 1e-3] epochs: [1, 10, 11, 100] scheduler: geometric	λ: [50, 25] epochs: [1, 100] scheduler: linear	8 - 80h	42.9
CIFAR10 NF Acc	5	4	8	[8, 4, 2, 1]	500	100	512	2.0; 0.2; 0.1;	lr: [1e-1, 1e-1, 1e-2, 1e-3] epochs: [1, 10, 11, 100] scheduler: geometric	λ : [50, 25] epochs: [1, 100] scheduler: linear	8 - 96h	49.0
CIFAR10 CNN Acc	8	4	16	[8, 4, 2, 1]	700	100	512	2.0; 0.2; 0.1;	lr: [1e-1, 1e-1, 1e-2, 1e-3] epochs: [1, 10, 11, 100] scheduler: geometric	λ: [50, 25] epochs: [1, 100] scheduler: linear	8 - 280h	97.9
CIFAR100 Ablations	5	3	8	[8, 4, 2]	500	100	512	0.5; 0.2; 0.05;	lr: [1e-1, 1e-1, 1e-2, 1e-3] epochs: [1, 10, 11, 100] scheduler: geometric	λ: [10, 1e-1, 1e-10] epochs: [1, 50, 100] scheduler: geometric	8 - 88h	43.4
CIFAR100 NF Acc	5	4	8	[16, 8, 4, 2]	500	100	512	0.5; 0.2; 0.05;	lr: [1e-1, 1e-1, 1e-2, 1e-3] epochs: [1, 10, 11, 100] scheduler: geometric	λ : [15, 1e-1, 1e-10] epochs: [1, 50, 100] scheduler: geometric	8 - 160h	87.0
CIFAR100 CNN Acc	8	4	16	[8, 4, 2, 1]	700	100	512	2.0; 0.2; 0.05;	lr_μ: [1e-1, 1e-1, 1e-2, 1e-3] epochs_μ: [1, 10, 11, 100] lr_Σ: [1e-1, 1e-1, 1e-2] epochs_Σ: [1, 10, 100] scheduler: geometric	λ_μ: [10, 1e-1, 1e-10] epochs_μ: [1, 50, 100] λ_Σ: [10, 10, 1e-1] epochs_Σ: [1, 50, 100] scheduler: geometric	8 - 280h	98.5
ImageNet	VAE latent space [13]	3	8	[8, 4, 2]	90	25	1024	0.5; 1.0; 0.05;	lr_μ: [1e-1, 1e-1, 1e-2] epochs_μ: [1, 10, 25] lr_Σ: [1e-1, 1e-1, 1e-2] epochs_Σ: [1, 10, 25] scheduler: geometric	λ_μ: [1, 1] epochs_μ: [1, 25] λ_Σ: [1, 1] epochs_Σ: [1, 25] scheduler: geometric	16 - 768h	49.0

Random initialization Concerning the gaussian hyperparameters that were randomly initialized in Table 7 of the paper, the means are randomly fixed following a uniform distribution in [-1, 1], and diagonal entries of the covariances are randomly initialized following a uniform distribution in $[\frac{a+c}{10}, a+c]$ to ensure the covariance matrix remains well conditioned.

Sampling Times

Table 3: Processing time in seconds, required by the NF (with 3 levels) to sample as many images as in the original datasets.

Dataset	Time in seconds	Number of Images
CIFAR10	100	50k
CIFAR100	134	50k

3.2 CNNs

The CNN used for classification is a ResNet18 [6]. The optimizer used is SGD with a learning rate of 1e−3, a momentum of 9e−1 and a weight decay of 5e−4. One Cycle Scheduler is used [14], with a maximum learning rate of 1e−1 that is updated every iteration. The ResNet18 model is also adapted for small datasets, by replacing the first 7×7 convolution with a 3×3 convolution, and replacing MaxPooling layer with Identity function. Table 4 shows the number of training epochs and batch size on every dataset.

Dataset	Epochs	Batch Size
CIFAR10	30	128
CIFAR100	50	128

Table 4: ResNet18 training details on CIFAR10 and CIFAR100. Concerning CIFAR10 and CIFAR100, the model is trained on 8 GPUs with a batch size of 16 per GPU (for a total batch size of 128).

3.3 Metrics

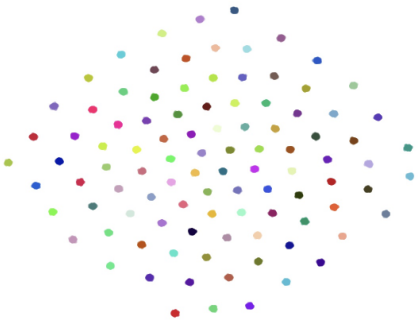
In order to calculate the FID and the Coverage metric, feature vectors (penultimate last linear layer from CNN) are commonly used. In our case, we use the feature vector from a ResNet18 architecture (which has a dimension of 512), which was trained with 5 bit images on CIFAR100 dataset. It should be noted that the produced score **cannot directly be compared with results from other papers**, because we are using 5 bit images, and because the use of our pretrained ResNet18 on CIFAR100, and not a pretrained Inception on ImageNet.

4 Extra Visualizations

4.1 Gaussian hyperparameters ablation

(a) w/o KLD, w/o Reparam

(b) w/ KLD w/o Reparam



(c) w/ KLD w/ Reparam

Fig. 32: The figures show the TSNE on CIFAR100, with ablations on the KLD and the reparametrization. It can be noted that when neither the KLD nor the reparametrization are used (see 32a), the datapoints are completely mixed. When the KLD is added (see 32b), some clusters are starting to get formed. Finally, when both the reparametrization and the KLD are used (see 32c) all the clusters are distinctly separated.

4.2 Sampling at different Temperatures

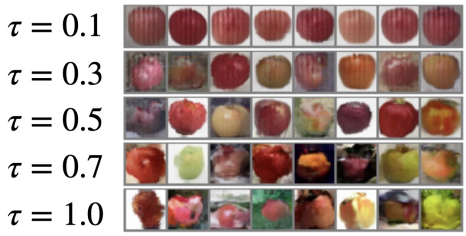


Fig. 33: Apple images sampled (from CIFAR100 dataset) with temperatures $\tau \in [1.0, 0.7, 0.5, 0.3, 0.1]$

4.3 Interpolations

Fig. 34: Interpolations produced with GLEM model on CIFAR10 dataset.

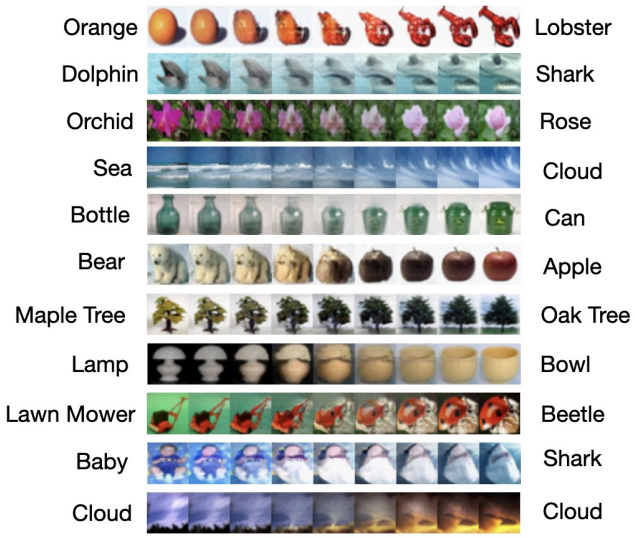


Fig. 35: Interpolations produced with GLEM model on CIFAR100 dataset.

4.4 TSNE Visualization

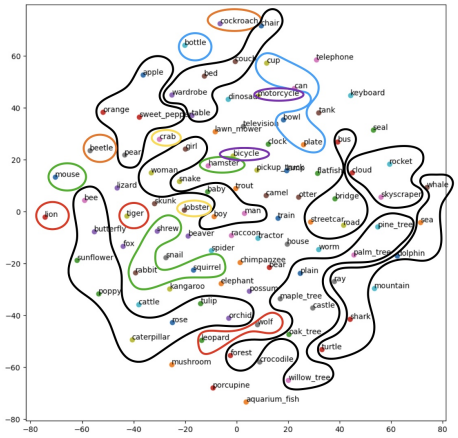


Fig. 36: TSNE produced with GLEM model on CIFAR100 dataset. Clusters delineated in back show some similar classes that were successfully placed together. Clusters delineated in colors show some classes that would have preferably been closer.

4.5 Generated Images

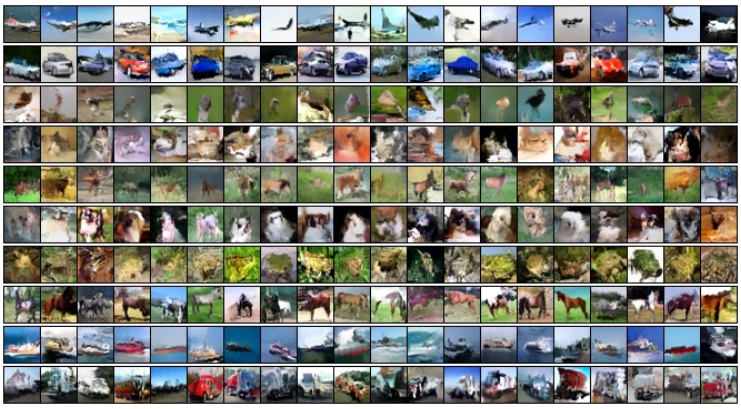


Fig. 37: Images sampled using GLEM Method, on CIFAR10 dataset at temperature $\tau = 1.0$ on a 8 bits model. From top to bottom, the classes are, airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. The mean vector of every gaussian was also rescaled with a temperature of 1.3, as we found it gave more realistic images (but less diverse).

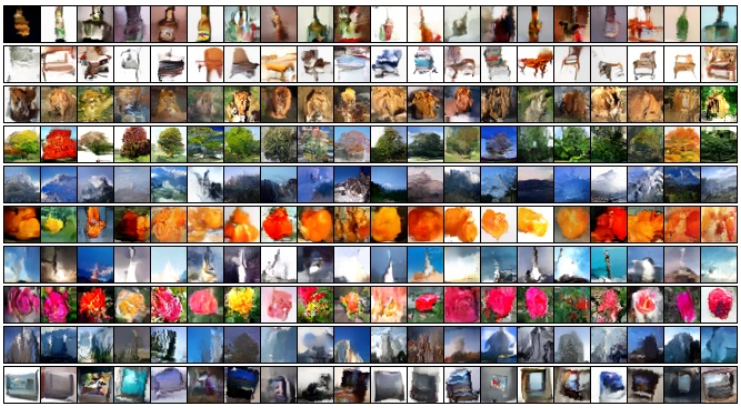
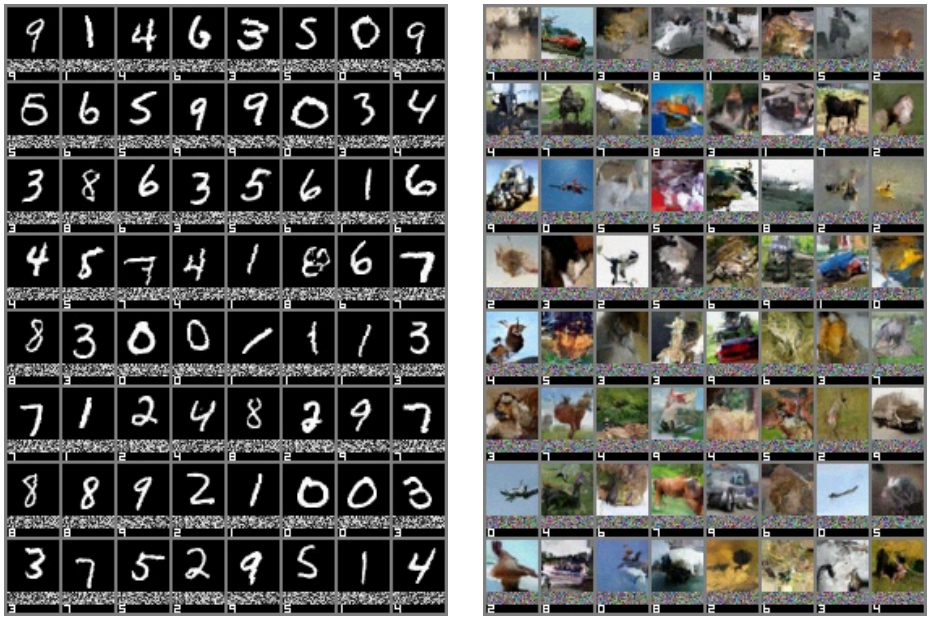


Fig. 38: Images sampled using GLEM Method, on CIFAR100 dataset at temperature $\tau = 1.0$ on a 8 bits model. From top to bottom, the classes are apple, aquarium fish, baby, bear, beaver, bed, bee, beetle, bike, bottle. The mean vector of every gaussian was also rescaled with a temperature of 1.3, as we found it gave more realistic images (but less diverse)

4.6 Label Encoding - CE Method



(a) MNIST **(b)** CIFAR10

Fig. 39: Conditional Encoding (CE) image samples with the NF model on MNIST (left) and CIFAR10 (right). The lowest part of an image shows the index of a class label, which was found and added by looking at the encoding (which is right above it). On MNIST it almost always matches the image content.

The classes for CIFAR10 are

- 0: Airplane
- 1: Car
- 2: Bird
- 3: Cat
- 4: Deer
- 5: Dog
- 6: Frog
- 7: Horse
- 8: Boat
- 9: Truck

References

1. Beyer, L., Hénaff, O.J., Kolesnikov, A., Zhai, X., Oord, A.v.d.: Are we done with imagenet? arXiv preprint arXiv:2006.07159 (2020)
2. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. pp. 702–703 (2020)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
4. Fang, Y., Sun, Q., Wang, X., Huang, T., Wang, X., Cao, Y.: Eva-02: A visual representation for neon genesis. arXiv preprint arXiv:2303.11331 (2023)
5. Fang, Y., Wang, W., Xie, B., Sun, Q., Wu, L., Wang, X., Huang, T., Wang, X., Cao, Y.: Eva: Exploring the limits of masked visual representation learning at scale. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 19358–19369 (2023)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
7. Huang, C.W., Dinh, L., Courville, A.: Augmented normalizing flows: Bridging the gap between generative flows and latent variable models. arXiv preprint arXiv:2002.07101 (2020)
8. Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., Wilson, A.G.: Averaging weights leads to wider optima and better generalization. arXiv preprint arXiv:1803.05407 (2018)
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
10. Liu, Y., Tian, Y., Zhao, Y., Yu, H., Xie, L., Wang, Y., Ye, Q., Liu, Y.: Vmamba: Visual state space model. arXiv preprint arXiv:2401.10166 (2024)
11. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983 (2016)
12. Ridnik, T., Ben-Baruch, E., Noy, A., Zelnik-Manor, L.: Imagenet-21k pretraining for the masses. arXiv preprint arXiv:2104.10972 (2021)
13. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)
14. Smith, L.N., Topin, N.: Super-convergence: Very fast training of neural networks using large learning rates. In: Artificial intelligence and machine learning for multi-domain operations applications. vol. 11006, pp. 369–386. SPIE (2019)
15. Wightman, R.: Pytorch image models. `https://github.com/rwightman/pytorch-image-models` (2019). `https://doi.org/10.5281/zenodo.4414861`
16. Zhong, Z., Zheng, L., Kang, G., Li, S., Yang, Y.: Random erasing data augmentation. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 13001–13008 (2020)