ON THE BENEFITS OF DEFINING VICINAL DISTRIBUTIONS IN LATENT SPACE

Puneet Mangla , Vedant Singh, Shreyas Havaldar & Vineeth N Balasubramanian
Department of Computer Science
Indian Institute of Technology, Hyderabad
Sangareddy, 502285, India
{cs17btech11029, cs17btech11047, cs18btech11042, vineethnb}@iith.ac.in

ABSTRACT

The vicinal risk minimization (VRM) principle is an empirical risk minimization (ERM) variant that replaces Dirac masses with vicinal functions. Mixup Training (MT), a popular choice of vicinal distribution, improves generalization performance of models by introducing globally linear behavior in between training examples. Apart from generalization, recent works have shown that mixup trained models are relatively robust to input perturbations/corruptions and at same time are calibrated better than their non-mixup counterparts. In this work, we investigate the benefits of defining these vicinal distributions like mixup in latent space of generative models rather than in input space itself. We propose a new approach - *VarMixup (Variational Mixup)* - to better sample mixup images by using the latent manifold underlying the data. Our empirical studies on CIFAR-10, CIFAR-100 and Tiny-ImageNet demonstrates that models trained by performing mixup in the latent manifold learned by VAEs are inherently more robust to various input corruptions and are significantly better calibrated than vanilla mixup.

1 INTRODUCTION

In most successful applications, Deep Neural Networks are trained to minimize the average error over the training dataset known as the Empirical Risk Minimization (ERM) principle (Vapnik, 1998). However, various empirical and theoretical studies have shown that minimizing Empirical Risk over training datasets in over-parameterized settings leads to their memorization and thus poor generalization on examples just outside the training distribution. To mitigate this problem of memorization in over-parameterized neural networks, Vicinal Risk Minimization (VRM) was proposed which essentially chooses to train networks on similar but different examples to the training data. This technique more popularly known as data augmentation (Simard et al., 2012), requires one to define a vicinity or neighbourhood around each training example (eg. in terms of brightness, contrast, imperceptible noise, etc.). Once defined, more examples can be sampled from their vicinity to enlarge the support of training distribution.

One of the popular choices to create the vicinal distribution is Mixup. Mixup Training (MT) (Zhang et al., 2017b) has emerged as a popular technique to train models for better generalisation in the last couple of years. Recent works have also shown that the idea of Mixup and Mixup training can be leveraged during inference (Pang* et al., 2020) and in many existing techniques like data augmentation (Hendrycks* et al., 2020), adversarial training (Lamb et al., 2019), etc. to improve the robustness of models to various input perturbations (Szegedy et al., 2013; Yuan et al., 2017) and corruptions (Hendrycks* et al., 2020). Other efforts on Mixup (Thulasidasan et al., 2019) have shown that Mixup-trained networks are significantly better calibrated and less prone to over-confident predictions on out-of-distribution than ones trained in the regular fashion. We mention other recent efforts in area of mixup and distinguish ourselves from them in the Appendix A.1.

Although still in its early phase, the above efforts (Zhang et al., 2017b; Verma et al., 2019; Pang* et al., 2020; Thulasidasan et al., 2019) also indicate a trend to viewing Mixup from perspectives of robustness and calibration. In this work, we take another step in this direction and propose a new vicinal distribution/sampling technique called *VarMixup (Variational Mixup)* to sample better

Mixup images during training to induce robustness as well as improve predictive uncertainty of models. In particular, we hypothesize that the latent unfolded manifold underlying the data (through a generative model, a Variational Autoencoder in our case) is linear by construction (manifolds unfold the locally linear structure of a high-dimensional data space), and hence more suitable for the defining vicinal distributions involving linear interpolations, such as Mixup. Importantly, we show that this choice of the distribution for Mixup plays an important role towards robustness and predictive uncertainty (Section 3).

2 Background on Vicinal Risk Minimization

We denote a neural network as $F_w : \mathbb{R}^{c \times h \times w} \rightarrow \mathbb{R}^k$, with weight parameters w. F_w takes an image $x \in \mathbb{R}^{c \times h \times w}$ and outputs logits, $F_w^i(x)$ for each class $i \in \{1...k\}$. Without loss of generality, we assume the classification task with $\mathcal{L}$ as the standard cross-entropy loss function. p_{actual} denotes the training data distribution, and the optimal weight parameter w^* is obtained by training the network using standard empirical risk minimization (Vapnik, 1998), i.e. $w^* = \arg\min_w \mathbb{E}_{(\mathbf{x},y) \sim p_{actual}} \; [\; \mathcal{L}\; (F_w(\mathbf{x}), y)\;]$, where y is the true label associated with input x.

Given the data distribution p_{actual}, a neural network F_w and loss function $\mathcal{L}$, the *expected risk* (average of loss function over p_{actual}) is given by $R(F_w) = \int \mathcal{L}(F_w(x), y) \cdot dp_{actual}(x, y)$. In practice, the true distribution p_{actual} is unknown, and is approximated by the training dataset $D = \{(x_i, y_i)\}_{i=1}^N$, which represents the *empirical distribution*: $p_\delta(x, y) = \frac{1}{N} \cdot \sum_{i=1}^N \delta(x = x_i, y = y_i)$. Here, $\delta(x = x_i, y = y_i)$ is the Dirac delta function centered at (x_i, y_i). Using p_δ as an estimate to p_{actual}, we define *expected empirical risk* as $R_\delta(F_w) = \frac{1}{N} \cdot \sum_{i=1}^N \mathcal{L}(F_w(x_i), y_i)$

Minimizing $R_\delta(F_w)$ to find optimal F_{w^*} is typically termed *Empirical Risk Minimization (ERM)* (Vapnik, 1998). However overparametrized neural networks can suffer from memorizing, leading to undesirable behavior of network outside the training distribution, p_δ (Zhang et al., 2017a; Szegedy et al., 2013). Addressing this concern, (Vapnik, 1995) and (Chapelle et al., 2001) proposed *Vicinal Risk Minimization (VRM)*, where p_{actual} is approximated by a vicinal distribution p_v, given by $p_v(x, y) = \frac{1}{N} \cdot \sum_{i=1}^N v(x, y|x_i, y_i)$, where v is the *vicinal distribution* that calculates the probability of a data point (x, y) in the vicinity of other samples (x_i, y_i).

Thus, using p_v to approximate p_{actual}, *expected vicinal risk* is given by $R_v(F_w) = \frac{1}{N} \cdot \sum_{i=1}^N g(F_w, \mathcal{L}, x_i, y_i)$

where $g(F_w, \mathcal{L}, x_i, y_i) = \int \mathcal{L}(F_w(x), y) \cdot dv(x, y|x_i, y_i)$. The superiority of VRM over ERM has been theoretically as well as empirically verified by many recent works (Ni & Rockett, 2015; Cao & Rockett, 2015; Hai-Yan & Hua, 2010; Zhang et al., 2018).

Popular examples of vicinal distributions include: (i) *Gaussian Vicinal distribution*: Here, $v_{gaussian}(x, y|x_i, y_i) = \mathcal{N}(x - x_i, \sigma^2) \cdot \delta(y = y_i)$, which is equivalent to augmenting the training samples with Gaussian noise; and (ii) *Mixup Vicinal distribution* : Here $v_{mixup}(x, y|x_i, y_i) = \frac{1}{n} \cdot \sum_{j=1}^N \mathbb{E}_\lambda[\delta(x = \lambda \cdot x_i + (1 - \lambda) \cdot x_j, y = \lambda \cdot y_i + (1 - \lambda) \cdot y_j)]$, where $\lambda \sim \beta(\eta, \eta)$ and $\eta > 0$.

3 Our Approach: VarMixup (Variational Mixup)

In this work, we build on the recent success of using Mixup as a vicinal distribution by proposing the use of the latent spaces learned by a generative deep neural network model. The use of generative models such as Variational Autoencoders (VAEs) (Kingma & Welling, 2013) to capture the latent space from which a distribution is generated provides us an unfolded manifold (the low-dimensional latent space), where the linearity in between training examples is more readily observed. Defining vicinal distributions by using neighbors on this latent manifold, which is more linear in the low-dimensional space, learned by generative models provides us more effective linear interpolations than the ones in input space. We hence leverage such an approach to capture the induced global linearity in between examples, and define Mixup vicinal distributions on this latent surface.

To capture the latent manifold of the training data through a generative model, we opt for a Variational Autoencoder (VAE). VAE (Kingma & Welling, 2013) is an autoencoder which is trained using Variational Inference, which serves as an implicit regularizer to ensure that the obtained latent space allows us to generate new data from the same distribution as training data.

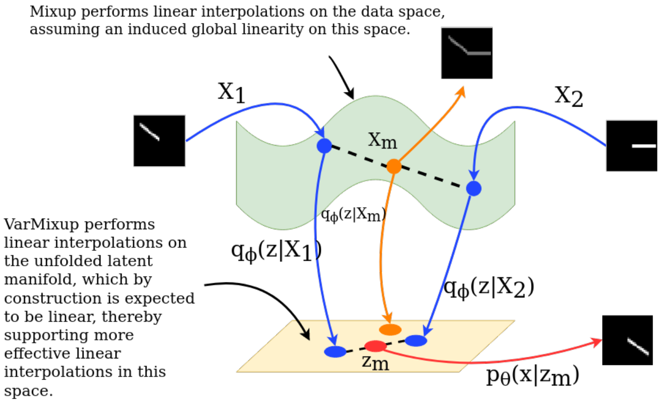
Mixup performs linear interpolations on the data space, assuming an induced global linearity on this space.
X1
X2
Xm
qϕ(z|Xm)
qϕ(z|X1)
qϕ(z|X2)
VarMixup performs linear interpolations on the unfolded latent manifold, which by construction is expected to be linear, thereby supporting more effective linear interpolations in this space.
zm
pθ(x|zm)

Figure 1: Illustration of conceptual idea behind VarMixup. We interpolate on the unfolded manifold, as defined by a generative model (VAE, in our case).

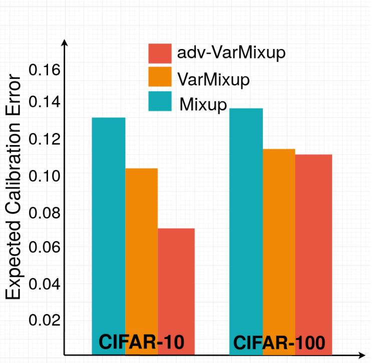

Figure 2: Expected Calibration Error (ECE) (Guo et al., 2017) of Mixup, VarMixup and adv-VarMixup trained models.

We denote the encoding and decoding distribution of VAE as $q_\phi(z|x)$ and $p_\theta(x|z)$ respectively, parametrized by ϕ and θ respectively. Given $p(z)$ as the desired prior distribution for encoding, the general VAE objective is given by the loss function:

$$\mathcal{L}_{VAE} = -\gamma \cdot D(q_\phi(z)\|p(z)) + \mathbb{E}_{x\sim p_{actual}}\mathbb{E}_{z\sim q_\phi(z|x)}[\log(p_\theta(x|z))] \tag{1}$$

Here, D is any strict divergence, meaning that $D(q\|p) \geq 0$ and $D(q\|p) = 0$ if and only if $q = p$, and $\gamma > 0$ is a scaling coefficient. The second term in the objective acts as a image reconstruction loss and $q_\phi(z) = \mathbb{E}_{x\sim p_{actual}}[q_\phi(z|x)]$. The original VAE (Kingma & Welling, 2013) uses KL-divergence in Eqn 1. However, using KL-divergence in Eqn 1 has some shortcomings, as pointed out in (Chen et al., 2016; Sønderby et al., 2016b; Zhao et al., 2017; Sønderby et al., 2016a). KL-divergence encourages the encoding $q_\phi(z|x)$ to be a random sample from $p(z)$ for each x, making them uninformative about the input. Also, it is not strong enough a regularizer compared to the reconstruction loss and tends to overfit data, consequently, learning a $q_\phi(z|x)$ that has high variance. Both the aforementioned shortcomings can affect the encoding distribution by making them uninformative of inputs with high variance. Since we use VAEs to better capture a linear latent manifold and subsequently define interpolations there, a bad latent distribution can affect our method significantly. We hence use a variant *Maximum Mean Discrepancy VAE* (MMD-VAE)(Zhao et al., 2017) which uses a MMD Loss (Gretton et al., 2007) instead of KL-divergence, and hence optimizes the following objective:

$$\mathcal{L}_{MMD-VAE} = \gamma \cdot MMD(q_\phi(z)\|p(z)) + \mathbb{E}_{x\sim p_{actual}}\mathbb{E}_{z\sim q_\phi(z|x)}[\log(p_\theta(x|z))] \tag{2}$$

A MMD-VAE doesn't suffer from the aforementioned shortcomings (Zhao et al., 2017), as it maximizes mutual information between x and z by matching the distribution over encodings $q_\phi(z)$ with prior $p(z)$ only in expectation, rather than for every input. We hence train an MMD-VAE to characterize the training distribution more effectively. We now define a Mixup vicinal distribution in the latent space of the trained VAE as $v_{VarMixup}(z, y|x_i, y_i) =$

$$\frac{1}{n} \cdot \sum_{j=1}^{N} \mathbb{E}_\lambda[\delta(z = \lambda \cdot \mathbb{E}_z[q_\phi(z|x_i)] + (1-\lambda) \cdot \mathbb{E}_z[q_\phi(z|x_j)], y = \lambda \cdot y_i + (1-\lambda) \cdot y_j)]$$

where $\lambda \sim \beta(\eta, \eta)$ and $\eta > 0$. Using the above vicinal distribution, $v_{VarMixup}$ and the MMD-VAE decoder, $p_\theta(x|z)$, we construct VarMixup samples as:

$$x' = \mathbb{E}_x[p_\theta(x|\lambda \cdot \mathbb{E}_z[q_\phi(z|x_i)] + (1-\lambda) \cdot \mathbb{E}_z[q_\phi(z|x_j)])] \text{ and } y' \quad = \lambda \cdot y_i + (1-\lambda) \cdot y_j$$

From another perspective, one could view our new sampling technique as performing Manifold Mixup (Verma et al., 2019), however over the latent space of an MMD-VAE (instead of the neural network feature space) and using it for sample reconstruction. We compare against Manifold Mixup in our results to show the improved performance of the learned generative latent space in our VarMixup. Figure 1 illustrates the conceptual idea behind VarMixup.

Table 1: Robustness to common input corruptions on CIFAR-10-C, CIFAR-100-C and Tiny-Imagenet-C (Hendrycks & Dietterich, 2019) datasets. Best results in **bold** and second best underlined. Clean accuracy is reported in parentheses using gray colour. Standard deviations are reported over 10 trials.

Method	CIFAR-10-C	CIFAR-100-C	Tiny-Imagenet-C
AT (Madry et al., 2018)	73.12 ± 0.31 (85.58 ± 0.14)	45.09 ± 0.31 (60.28 ± 0.13)	15.74 ± 0.36 (22.33 ± 0.16)
TRADES (Madry et al., 2018)	75.46 ± 0.21 (88.11 ± 0.43)	45.98 ± 0.41 (63.3 ± 0.32)	16.20 ± 0.23 (26.12 ± 0.38)
IAT (Lamb et al., 2019)	81.05 ± 0.42 (89.7 ± 0.33)	50.71 ± 0.25 (62.7 ± 0.21)	18.69 ± 0.45 (18.08 ± 0.34)
ERM	69.29 ± 0.21 (94.5 ± 0.14)	47.3 ± 0.32 (64.5 ± 0.10)	17.34 ± 0.27 (49.96 ± 0.12)
Mixup	74.74 ± 0.34 (95.5 ± 0.35)	52.13 ± 0.43 (76.8 ± 0.41)	21.55 ± 0.37 (53.83 ± 0.17)
Mixup-R	74.27 ± 0.22 (89.88 ± 0.11)	43.54 ± 0.15 (62.24 ± 0.21)	21.34 ± 0.32 (53.5 ± 0.28)
Manifold-Mixup	72.54 ± 0.14 (95.2 ± 0.18)	41.42 ± 0.23 (75.3 ± 0.48)	-
VarMixup	**82.57** ± **0.42** (93.91 ± 0.45)	52.57 ± 0.39 (73.2 ± 0.44)	24.87 ± 0.32 (50.98 ± 0.11)
adv-VarMixup	82.12 ± 0.46 (92.19 ± 0.32)	**54.0** ± **0.41** (72.13 ± 0.34)	**25.36** ± **0.21** (50.58 ± 0.23)

4 Experiments and Results

We now present our experimental studies and results using our method, VarMixup where we focus explicitly on the usefulness of our approach on out-of-distribution test data and addressing predictive uncertainty. We also perform several ablation studies in Appendix A.3 to analyze our approach.

Implementation Details: It has been shown (Ilyas et al., 2019) that adversarial training removes irrelevant biases (e.g. texture biases) in their hidden representations, thus making them more informative. We hence hypothesize that the considered VAE, if trained adversarially, will have more informative latent encoding than its regular equivalent. This would hence help improve the vicinal distributions like VarMixup. Empirically, we validate this hypothesis in our subsequent experiments and use prefix *adv*- (eg: *adv*-VarMixup) to distinguish them from their regular variants. All models are trained using the Resnet-34 (He et al., 2016) backbone across all datasets.

Baseline Models: We compare our method, VarMixup, against Vanilla ERM, Vanilla Mixup (Zhang et al., 2017b), Manifold Mixup (Verma et al., 2019), l_∞ PGD/TRADES adversarial training (Madry et al., 2018; Zhang et al., 2019) and l_∞ Interpolated adversarial training (Lamb et al., 2019). These choice of baselines include non-VRM variants, mixup variants and state-of-the-art adversarial techniques. We also compare against a variant of mixup, which we call Mixup-R where mixup training is done on MMD-VAE's reconstructed image space rather than actual image space. Please refer to Appendix A.2 for more details on training of these baselines.

Generalization Performance and Robustness to Out-of-Distribution shifts We first evaluate the trained models on their robustness to various common input corruptions (Hendrycks & Dietterich, 2019), along with their generalization performance on "clean data" (test data without corruptions). We evaluate their robustness on the newer CIFAR-10-C, CIFAR-100-C and Tiny-Imagenet-C datasets (Hendrycks & Dietterich, 2019). These datasets contain images, corrupted with 15 different distortions at 5 severity levels. We report the mean classification accuracy over all distortions on aforementioned corrupted datasets in Table 1. The results show that our method - VarMixup/*adv*-VarMixup achieves superior performance by a margin of $\sim 2-4\%$ consistently across the datasets. The slight drop in the clean accuracy of VarMixup models (shown in parentheses in Table 1) is due to the tradeoff between robustness and clean accuracy, which is a common trend observed in robustness literature.

Calibration: A recent study (Thulasidasan et al., 2019) showed that DNNs trained with Mixup are significantly better calibrated than DNNs trained in a regular fashion. Calibration (Guo et al., 2017) measures how good softmax scores are as indicators of the actual likelihood of a correct prediction. We measure the *Expected Calibration Error (ECE)* (Thulasidasan et al., 2019; Guo et al., 2017) of the proposed method, following (Thulasidasan et al., 2019). Figure 2 shows the calibration error on CIFAR-10 and CIFAR-100 datasets using Mixup, VarMixup and adv-VarMixup. The figure illustrates that our VarMixup models are also better calibrated than regular Mixup.

5 Conclusions

In this work, we proposed a Mixup-based vicinal distribution, VarMixup, which performs linear interpolation on an unfolded latent manifold where linearity in between training examples is likely to be preserved by construction. We show that VarMixup trained models are more robust to common input corruptions and are better calibrated. Our work highlights the efficacy of defining vicinal distributions by using neighbors on unfolded latent manifold rather than data manifold and we believe that our work can open a discussion around this notion of robustness and choice of vicinal distributions on generative latent spaces.

Acknowledgements

This work has been partly supported by the funding received from DST, Govt of India, through the IMPRINT program (IMP/2019/000250). We also acknowledge IIT-Hyderabad and JICA for provision of GPU servers for the work.

References

Christopher Beckham, Sina Honari, Vikas Verma, Alex M Lamb, Farnoosh Ghadiri, R Devon Hjelm, Yoshua Bengio, and Chris Pal. On adversarial mixup resynthesis. In *Advances in Neural Information Processing Systems 32*, pp. 4346–4357. Curran Associates, Inc., 2019.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems 32*, pp. 5049–5059. Curran Associates, Inc., 2019.

Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018.

Yilong Cao and Peter I. Rockett. The use of vicinal-risk minimization for training decision trees. *Appl. Soft Comput.*, 31(C), June 2015. ISSN 1568-4946.

Olivier Chapelle, Jason Weston, Léon Bottou, and Vladimir Vapnik. Vicinal risk minimization. In T. K. Leen, T. G. Dietterich, and V. Tresp (eds.), *Advances in Neural Information Processing Systems 13*, pp. 416–422. MIT Press, 2001.

Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. *CoRR*, abs/1611.02731, 2016.

Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J. Smola. A kernel method for the two-sample-problem. In B. Schölkopf, J. C. Platt, and T. Hoffman (eds.), *Advances in Neural Information Processing Systems 19*, pp. 513–520. MIT Press, 2007.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. *CoRR*, abs/1706.04599, 2017.

L. Hai-Yan and J. Hua. Vicinal risk minimization based probability density function estimation algorithm using svm. In *2010 Third International Conference on Information and Computing*, volume 4, pp. 161–164, 2010.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, June 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

Dan Hendrycks*, Norman Mu*, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple method to improve robustness and uncertainty under data shift. In *International Conference on Learning Representations*, 2020.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems 32*, pp. 125–136. Curran Associates, Inc., 2019.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.

Alex Lamb, Vikas Verma, Juho Kannala, and Yoshua Bengio. Interpolated adversarial training: Achieving robust neural networks without sacrificing too much accuracy. AISec'19, 2019.

X. Liu, Y. Zou, L. Kong, Z. Diao, J. Yan, J. Wang, S. Li, P. Jia, and J. You. Data augmentation via latent space interpolation for image classification. In *2018 24th International Conference on Pattern Recognition (ICPR)*, Aug .

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Ji Ni and Peter I. Rockett. Training genetic programming classifiers by vicinal-risk minimization. *Genet. Program. Evolvable Mach.*, 16(1):3–25, 2015.

Tianyu Pang*, Kun Xu*, and Jun Zhu. Mixup inference: Better exploiting mixup to defend adversarial attacks. In *International Conference on Learning Representations*, 2020.

Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Alhussein Fawzi, Soham De, Robert Stanforth, Pushmeet Kohli, et al. Adversarial robustness through local linearization. *arXiv preprint arXiv:1907.02610*, 2019.

Patrice Y. Simard, Yann A. LeCun, John S. Denker, and Bernard Victorri. *Transformation Invariance in Pattern Recognition – Tangent Distance and Tangent Propagation*. Springer Berlin Heidelberg, 2012.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 3745–3753, Red Hook, NY, USA, 2016a. Curran Associates Inc. ISBN 9781510838819.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 3745–3753, Red Hook, NY, USA, 2016b. Curran Associates Inc. ISBN 9781510838819.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *Advances in Neural Information Processing Systems 32*, pp. 13888–13899. Curran Associates, Inc., 2019.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg, 1995. ISBN 0387945598.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 6438–6447, 2019.

Minghao Xu, Jia yu Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. Adversarial domain adaptation with domain mixup. *ArXiv*, abs/1912.01805, 2019.

Xiaoyong Yuan, Pan He, Qile Zhu, Rajendra Rana Bhat, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *CoRR*, abs/1712.07107, 2017.

Chao Zhang, Min-Hsiu Hsieh, and Dacheng Tao. Generalization bounds for vicinal risk minimization principle. *arXiv preprint arXiv:1811.04351*, 2018.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *ICLR*, abs/1611.03530, 2017a.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. *CoRR*, abs/1901.08573, 2019.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017b.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. Infovae: Information maximizing variational autoencoders. *CoRR*, abs/1706.02262, 2017.

A Appendix

A.1 Related work on Mixup

(Zhang et al., 2017b) proposed Mixup, a method to train models on the convex combination of pairs of examples and their labels. In other words, it constructs virtual training examples as: $x' = \lambda \cdot x_i + (1-\lambda) \cdot x_j; y' = \lambda \cdot y_i + (1-\lambda) \cdot y_j$, where x_i, x_j are input vectors; y_i, y_j are one-hot label encodings and λ is a mixup coefficient, usually sampled from a $\beta(\eta, \eta)$ distribution. By doing so, it regularizes the network to behave linearly in between training examples, thus inducing global linearity between them. A recent variant, Manifold Mixup (Verma et al., 2019), exploits interpolations at hidden representations, thereby obtaining neural networks with smoother decision boundaries at different levels of hidden representations. AugMix (Hendrycks* et al., 2020) mixes up multiple augmented images and uses a Jensen-Shannon Divergence consistency loss on them to achieve better robustness to common input corruptions (Hendrycks & Dietterich, 2019). In semi-supervised learning, MixMatch (Berthelot et al., 2019) obtains state-of-the-art results by guessing low-entropy labels for data-augmented unlabeled examples and mixes labeled and unlabeled data using Mixup. It has been shown that apart from better generalization, Mixup also improves the robustness of models to adversarial perturbations as well. To further boost this robustness at inference time, Pang et al. (Pang* et al., 2020) recently proposed a Mixup Inference technique which performs a mixup of input x with a clean sample x_s and passes the corresponding mixup sample $(\lambda \cdot x + (1-\lambda) \cdot x_s)$ into the classifier as the processed input. Other efforts related to Mixup (Thulasidasan et al., 2019) have shown that Mixup-trained networks are better calibrated i.e., the predicted softmax scores are better indicators of the actual likelihood of a correct prediction than DNNs trained in the regular fashion. Additionally, they also observed that mixup-trained DNNs are less prone to over-confident predictions on out-of-distribution and random-noise data. None of these efforts however address Mixup from a generative latent space, which is the focus of this work. Efforts such as (Pang* et al., 2020) and (Thulasidasan et al., 2019), in fact, have inferences that motivate the need to consider a latent Mixup space to address a model's robustness and predictive uncertainty.

From a different perspective, Xu et al. (Xu et al., 2019) used domain mixup to improve the generalization ability of models in domain adaptation. Adversarial Mixup Resynthesis (Beckham et al., 2019) attempted mixing latent codes used by autoencoders through an arbitrary mixing mechanism that can recombine codes from different inputs to produce novel examples. This work however has a different objective and focuses on generative models in a GAN-like setting, while our work focuses on robustness and predictive uncertainty. The work by Liu et al. (Liu et al.) may be closest to ours in terms of approach as they use an adversarial autoencoder (AAE) to impose a uniform distribution on the feature representations. However, their work deals with improving generalization performance, while ours looks at robustness and predictive uncertainty, as already stated. Furthermore, we propose a new method, VarMixup, which focuses on directly exploiting the manifold learned by a Variational Autoencoder (VAE) (and do not regularize it unlike previous work) during Mixup and report improved adversarial robustness. We also present useful insights into the working of VarMixup (which is lacking in earlier work including (Liu et al.)), thus making our contributions unique and more complete.

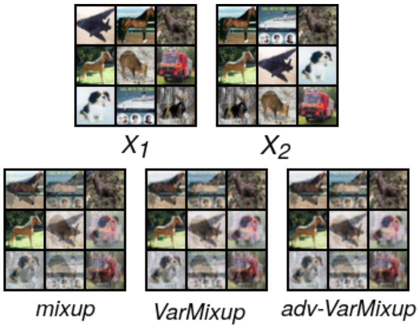

Figure 3: Samples generated by mixup, VarMixup and *adv*-VarMixup on CIFAR-10 (Mixup coefficient $\lambda = 0.5$).

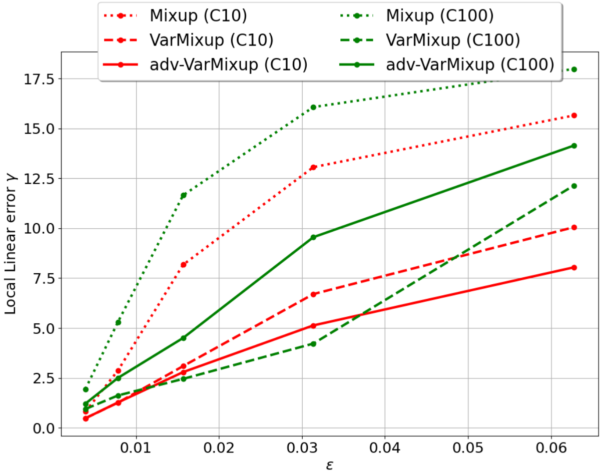

Figure 4: Local linear error of loss landscapes of models trained on CIFAR-10/-100 (denoted as C10 and C100)

A.2 Baseline Training Details

Here, we report the training details of baselines used for comparing out approach in Table 1.

1. *ERM* - Vanilla Empirical Risk Minimization using Adam optimizer ($lr = 1e - 3$) for 100 epochs on all datasets.
2. *Mixup* - Vanilla Mixup training (Zhang et al., 2017b) using Adam optimizer ($lr = 1e - 3$) for 150 epochs on all datasets. Mixup coefficient is sampled from $\beta(1, 1)$.
3. *Mixup-R* - Mixup training on MMD-VAE's reconstructed image space (Zhang et al., 2017b) using Adam optimizer ($lr = 1e - 3$) for 150 epochs on all datasets. Mixup coefficient is sampled from $\beta(1, 1)$.
4. *Manifold Mixup* - Manifold Mixup training (Verma et al., 2019) using Adam optimizer ($lr = 1e - 3$) for 150 epochs on all datasets. Mixup coefficient is sampled from $\beta(2, 2)$.
5. *AT and TRADES* - l_∞ PGD/TRADES adversarial training (Madry et al., 2018; Zhang et al., 2019) with $\epsilon = 8/255$ and step-size $\alpha = 2/255$. Models are trained using Adam optimizer ($lr = 1e - 3$) for 250 epochs on all datasets.
6. *IAT* - l_∞ Interpolated adversarial training (Lamb et al., 2019) with $\epsilon = 8/255$ and step-size $\alpha = 2/255$. Interpolation coefficient is sampled from $\beta(1, 1)$. Models are trained using Adam optimizer ($lr = 1e - 3$) for 350 epochs on all datasets.

A.3 Ablation Studies

Analyzing VarMixup samples: Figure 3 shows sample data generated by regular Mixup, VarMixup, and *adv*-VarMixup on two images. Although mixup or VarMixup samples look perceptually similar, they are quite different at a statistical level. We measure the Frechet Inception Distance (FID) (Heusel et al., 2017) and Kernel Inception Distance (Bińkowski et al., 2018) between regular training data and training data generated by mixup/VarMixup/ *adv*-VarMixup. These scores summarize how similar the two groups are in terms of statistics on computer vision features of the raw images calculated using the Inceptionv3 model used for image classification. Lower scores indicate the two groups of images are more similar, or have more similar statistics, with a perfect score being 0.0 indicating that the two groups of images are identical. Figure 5 reports these metrics on CIFAR-10 and CIFAR-100 respectively. The greater FID and KID scores indicate that we are adding off-manifold samples (w.r.t. the manifold characterized by training data) to the training using our approach.

Local linearity on loss landscapes: (Qin et al., 2019) showed that the local linearity of loss landscapes of neural networks is related to model robustness. The more the loss landscapes are linear,

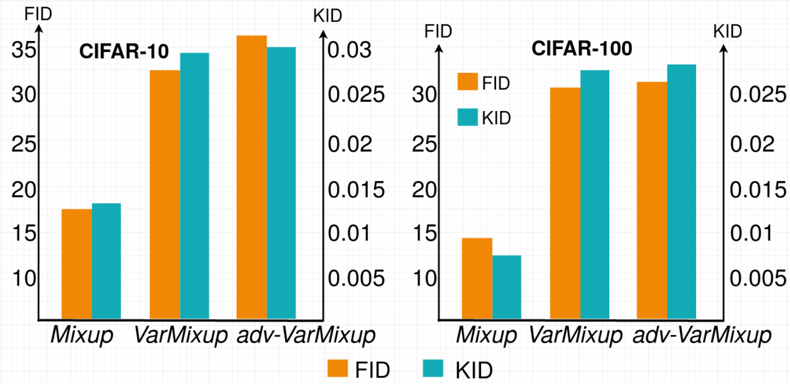

Figure 5: FID and KID scores between training set and mixup/VarMixup generated samples on CIFAR-10 and CIFAR-100

the more the robustness. To further study this observation using our method, we analyze the local linearity of loss landscapes of VarMixup and regular mixup trained models. Qin et al. (Qin et al., 2019) defines local linearity at a data-point x within a neighbourhood $B(\epsilon)$ as:

$$\gamma(\epsilon, x, y) = \max_{\delta \in B(\epsilon)} |\mathcal{L}(F_w(x+\delta), y) - \mathcal{L}(F_w(x), y) - \delta_x^T \mathcal{L}(F_w(x), y)|$$

Figure 4 shows the average local linear error (over test set) with increasing L_∞ max-perturbation ϵ on CIFAR-10 and CIFAR-100 datasets. As can be seen, VarMixup/*adv*-VarMixup makes the local linear error significantly ($\times$ 2) lesser as compared to regular mixup, thus inducing robustness.

Computational Overhead: We compare the computational time of our trained models using VarMixup/*adv*-VarMixup with commonly used adversarial training techniques, AT and TRADES. VarMixup, *adv*-VarMixup, AT and TRADES take around 3, 5, 8.8 and 15 hours respectively for training. The training time of the MMD-VAE was also considered here. While already significantly faster than AT and TRADES, the proposed method will be more scalable and time-efficient, if a VAE trained on a dataset such as ImageNet can be directly used to generate VarMixup samples for other datasets. This is a typical transfer learning setting, and we hence study the performance of training VarMixup models on CIFAR-10 and CIFAR-100 datasets using MMD-VAE trained on the Tiny-Imagenet dataset. Respectively, VarMixup obtained mean corruption accuracy of 82.0 % and 53.25 % on CIFAR-10-C and CIFAR-100-C benchmarks, thus making our approach time-efficient and also scalable.