Supplementary Material:
Adversarial Vertex Mixup: Toward Better Adversarially Robust Generalization

A. Proofs

Theorem 1. *For the variance parameters σ_r and σ_s, let $\sigma_r = \nu\sigma_s$ where $\nu \in [0,1]$. Then, the upper bound on the standard classification error of f_{n,σ_s} and the upper bound on the ℓ_∞^ϵ-robust classification error of f_{n,σ_r} are equal with probability at least $\left(1-2\exp\left(-\frac{d}{8(\sigma_s^2+1)}\right)\right)\cdot\left(1-2\exp\left(-\frac{d}{8(\sigma_r^2+1)}\right)\right)$ if*

$$\epsilon \leq \frac{(2\sqrt{n}-1)(1-\nu)}{2\sqrt{n}+4\sigma_s}. \tag{1}$$

Proof. We recall the following theorems due to Schmidt *et al*. [8]:

Theorem 18. (Schmidt *et al.*) *Let $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n) \in \mathbb{R}^d \times \{\pm 1\}$ be drawn i.i.d. from a $(\theta^\star, \sigma)$-Gaussian model with $\|\theta^\star\|_2 = \sqrt{d}$. Let the weight vector $\boldsymbol{w} \in \mathbb{R}^d$ be the unit vector in the direction of $\bar{\boldsymbol{z}} = \frac{1}{n}\sum_{i=1}^n y_i \boldsymbol{x}_i$. Then, with probability at least $1-2\exp\left(-\frac{d}{8(\sigma^2+1)}\right)$, the linear classifier $f_{\boldsymbol{w}}$ has classification error at most*

$$\exp\left\{\left(-\frac{(2\sqrt{n}-1)^2 d}{2(2\sqrt{n}+4\sigma)^2\sigma^2}\right)\right\}. \tag{2}$$

Theorem 21. (Schmidt *et al.*) *Let $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n) \in \mathbb{R}^d \times \{\pm 1\}$ be drawn i.i.d. from a $(\theta^\star, \sigma)$-Gaussian model with $\|\theta^\star\|_2 = \sqrt{d}$. Let the weight vector $\boldsymbol{w} \in \mathbb{R}^d$ be the unit vector in the direction of $\bar{\boldsymbol{z}} = \frac{1}{n}\sum_{i=1}^n y_i \boldsymbol{x}_i$. Then, with probability at least $1-2\exp\left(-\frac{d}{8(\sigma^2+1)}\right)$, the linear classifier $f_{\boldsymbol{w}}$ has ℓ_∞^ϵ-robust classification error at most β if*

$$\epsilon \leq \frac{2\sqrt{n}-1}{2\sqrt{n}+4\sigma} - \frac{\sigma\sqrt{2\log 1/\beta}}{\sqrt{d}}. \tag{3}$$

When the upper bound on the standard classification error of f_{n,σ_s} and the upper bound on the ℓ_∞^ϵ-robust classification error of f_{n,σ_r} are equal, we can find the condition of ϵ by substituting (2) into β in (3). Then, the right-hand side of (3) can be written as

$$\frac{2\sqrt{n}-1}{2\sqrt{n}+4\sigma_r} - \frac{\sigma_r}{\sigma_s}\cdot\frac{2\sqrt{n}-1}{2\sqrt{n}+4\sigma_s}. \tag{4}$$

Since $\sigma_r = \nu\sigma_s$ where $\nu \in [0,1]$, it satisfies

$$\frac{2\sqrt{n}-1}{2\sqrt{n}+4\sigma_r} - \frac{\sigma_r}{\sigma_s}\cdot\frac{2\sqrt{n}-1}{2\sqrt{n}+4\sigma_s} \geq \frac{2\sqrt{n}-1}{2\sqrt{n}+4\sigma_s} - \frac{\sigma_r}{\sigma_s}\cdot\frac{2\sqrt{n}-1}{2\sqrt{n}+4\sigma_s} = \frac{(2\sqrt{n}-1)(1-\nu)}{2\sqrt{n}+4\sigma_s}. \tag{5}$$

□

Corollary 1. *For the variance parameters σ_r and σ_s, let $\sigma_r = \nu\sigma_s$ where $\nu \in [0,1]$. Let the upper bound on the standard classification error of f_{n,σ_s} and the upper bound on the ℓ_∞^ϵ-robust classification error of f_{n,σ_r} be equal. Then, as σ_r decreases, the upper bound of ϵ increases in proportion to π_{n,σ_s}, which is given by*

$$\pi_{n,\sigma_s} = \frac{2\sqrt{n}-1}{\sigma_s(2\sqrt{n}+4\sigma_s)}. \tag{6}$$

Proof. Let ϵ' be the upper bound on ϵ in Theorem 1. Then, the gradient of ϵ' with respect to σ_r is

$$\frac{\partial \epsilon'}{\partial \sigma_r} = \frac{\partial \epsilon'}{\partial \nu}\frac{\partial \nu}{\partial \sigma_r} = -\frac{2\sqrt{n}-1}{2\sqrt{n}+4\sigma_s} \cdot \frac{1}{\sigma_s} \tag{7}$$

Therefore, when σ_r decreases, the upper bound of ϵ increases in proportion to $-\frac{\partial \epsilon'}{\partial \sigma_r}$, which is π_{n,σ_s}. □

Theorem 2. *Let $\boldsymbol{w}_B \in \mathbb{R}^{d-c}$ be the weight vector for the sufficient non-robust features of $\Psi_{sample,c}$. Let Z_{sc} be a set of strictly convex functions. Then, when the classifier $f_{Z_{sc}}$ is trained on $\Psi_{sample,c}$, the $\boldsymbol{w}_B^\star$ which minimizes the variance of $\boldsymbol{w}^T\boldsymbol{x}$ with respect to $\Psi_{sample,c}$ is*

$$\boldsymbol{w}_B^\star = \vec{0}. \tag{8}$$

Proof. Let $\Sigma \in \mathbb{R}^{(d+1)\times(d+1)}$ be the diagonal matrix, where $\Sigma = diag(\sigma_1^2, \cdots, \sigma_{d+1}^2)$ and σ_i^2 is the variance of each x_i with $i \in \{1, \cdots, d+1\}$. Then, the optimization problem is

$$\min_{\boldsymbol{w}} \ \boldsymbol{w}^T\Sigma\boldsymbol{w} \qquad s.t.\ \boldsymbol{w} \in \mathbb{R}_+^{d+1}, \|\boldsymbol{w}\|_1 = 1. \tag{9}$$

As the objective function over the feasible set is strictly convex, one optimal weight $\boldsymbol{w}^\star$ exists at the most. In addition, we can prove that the optimal weight values for features having the same distribution are equal by using Jensen's Inequality. Thus, we obtain

$$w_i^\star = w_j^\star,\ \forall i,j \in \{1,\cdots,c+1\},\ or\ \forall i,j \in \{c+2,\cdots,d+1\}. \tag{10}$$

We now let $w_A^\star$ be the optimal weight value of x_i, where $\forall i \in \{1,\cdots,c+1\}$, and we let $w_B^\star$ be the optimal weight value of x_j, where $\forall j \in \{c+2,\cdots,d+1\}$. Let $\sigma_z^\star$ be the optimal (minimal) variance of $\boldsymbol{w}^T\boldsymbol{x}$. Then, we have $\sigma_z^{\star 2} = (c+1){w_A^\star}^2\sigma_A^2 + (d-c){w_B^\star}^2\sigma_B^2$. Thus, $\sigma_z^{\star 2}$ is the same as the solution of the optimization problem

$$\min_{w_B} \ (c+1)w_A^2\sigma_A^2 + (d-c)w_B^2\sigma_B^2 \qquad s.t.\ (c+1)w_A + (d-c)w_B = 1. \tag{11}$$

The constraint results in $w_A = (1-(d-c)w_B)/(c+1)$, which enable us to rewrite the optimization problem as

$$\min_{w_B} \ \frac{(1-(d-c)w_B)^2}{c+1}\sigma_A^2 + (d-c)w_B^2\sigma_B^2. \tag{12}$$

Then, we can find the optimal $w_B^\star$ as follows:

$$\frac{d}{dw_B}\left(\frac{(1-(d-c)w_B)^2}{c+1}\sigma_A^2 + (d-c)w_B^2\sigma_B^2\right) = 2w_B\left(\frac{(d-c)^2\sigma_A^2}{c+1} + (d-c)\sigma_B^2\right) - 2\left(\frac{(d-c)\sigma_A^2}{c+1}\right) = 0. \tag{13}$$

Since $0 < \sigma_A \ll \sigma_B$, we have the optimal $\boldsymbol{w}_B^\star$

$$\boldsymbol{w}_B^\star = \frac{\sigma_A^2}{(d-c)\sigma_A^2 + (c+1)\sigma_B^2} \cdot \vec{1} \approx \vec{0}. \tag{14}$$

as desired.

□

Theorem 3. *Let $\boldsymbol{w}_B \in \mathbb{R}^{d-c}$ be the weight vector for the sufficient non-robust features of $\Psi_{sample,c}$. Let Z_{sc} be a set of strictly convex functions. Then, when the classifier $f_{Z_{sc}}$ is trained on $\Psi_{sample,c}$, the $\boldsymbol{w}_B^\star$ which minimizes the variance of $\boldsymbol{w}^T\boldsymbol{x}$ with respect to Ψ_{true} is*

$$\boldsymbol{w}_B^\star = \frac{c}{cd+2c+1} \cdot \vec{1}. \tag{15}$$

Proof. As with Theorem 2, let σ_z be the variance of $\boldsymbol{w}^T\boldsymbol{x}$. Because we train the classifier $f_{Z_{sc}}$ on $\Psi_{sample,c}$ but minimize the variance of $\boldsymbol{w}^T\boldsymbol{x}$ with respect to Ψ_{true}, we have $\sigma_z^{\star 2} = {w_A^\star}^2\sigma_A^2 + c{w_A^\star}^2\sigma_B^2 + (d-c){w_B^\star}^2\sigma_B^2$. Thus, $\sigma_z^{\star 2}$ is the same as the solution of the optimization problem

$$\min_{w_B} \ w_A^2\sigma_A^2 + cw_A^2\sigma_B^2 + (d-c)w_B^2\sigma_B^2 \qquad s.t.\ (c+1)w_A + (d-c)w_B = 1. \tag{16}$$

The constraint results in $w_A = (1-(d-c)w_B)/(c+1)$, which enable us to rewrite the objective function as

$$\frac{(1-(d-c)w_B)^2}{(c+1)^2}(\sigma_A^2 + c\sigma_B^2) + (d-c)w_B^2\sigma_B^2. \tag{17}$$

Then, we can find the optimal $w_B^\star$ as follows:

$$\begin{aligned}\frac{d}{dw_B}&\left(\frac{(1-(d-c)w_B)^2}{(c+1)^2}(\sigma_A^2 + c\sigma_B^2) + (d-c)w_B^2\sigma_B^2\right)\\ &= 2w_B\left(\frac{(d-c)^2}{(c+1)^2}(\sigma_A^2 + c\sigma_B^2) + (d-c)\sigma_B^2\right) - 2\left(\frac{(d-c)(\sigma_A^2 + c\sigma_B^2)}{(c+1)^2}\right) = \vec{0}.\end{aligned} \tag{18}$$

Since $0 < \sigma_A \ll \sigma_B$, we have the optimal $\boldsymbol{w}_B^\star$

$$\boldsymbol{w}_B^\star = \frac{\sigma_A^2 + c\sigma_B^2}{(d-c)(\sigma_A^2 + c\sigma_B^2) + (c+1)^2\sigma_B^2} \cdot \vec{1} \approx \frac{c}{cd + 2c + 1} \cdot \vec{1}. \tag{19}$$

as desired. If we confine our feasible set to

$$w_i = w_j,\ \forall i, j \in \{1, \cdots, c+1\},\ \mathit{or}\ \forall i, j \in \{c+2, \cdots, d+1\}, \tag{20}$$

which does not change the optimal weight $\boldsymbol{w}^\star$ we can easily predict the robust generalization performance in the robust learning procedure using (17). □

Lemma 1. (Tsipras *et al.*) *Minimizing the adversarial empirical risk results in a classifier that assigns 0 weight to non-robust features.*

Proof. We can follow the proof of Tsipras *et al*. [10], *i.e*., let $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w})$ be the loss function of our classifier $f_{\boldsymbol{w}}$. Then, the optimization problem of our model in adversarial training is

$$\min_{\boldsymbol{w}} \mathbb{E}\left[\max_{\|\boldsymbol{\delta}\|_\infty \le \epsilon} \mathcal{L}(\boldsymbol{x} + \boldsymbol{\delta}, y; \boldsymbol{w})\right]. \tag{21}$$

Consider any weight $\boldsymbol{w}$ for which $w_i > 0$ for some $i \geq c+2$. Because the sufficient non-robust features $x_{c+2}, \ldots, x_{d+1}$ are normally distributed random variables with mean ηy, the adversary reverses the sign of the sufficient non-robust features if $\eta < \epsilon$. Then, it assigns negative update values with a high probability, because the sufficient non-robust features are moved such that their signs are opposite to that of the label. Thus, the optimizer constantly updates the weight vector such that the weight values w_i, where $\forall i \geq c+2$, move toward 0. Ultimately, the optimization of the objective function assigns 0 weight to the non-robust features.

□

B. Further Related Work

In this section, we introduce two recent works [4, 7] that utilized Mixup [11] in adversarial defense. Pang *et al*. [7] developed Mixup Inference (MI), an inference principle, by focusing on the globally linear behavior in-between the data manifolds introduced by the mixup training method. MI feeds the linear interpolation between the adversarial example and a sampled clean example into the classifier. The main difference between AVmixup and MI is that MI is an algorithm applied only in the inference step while AVmixup is applied only in the training step.

Lamb *et al*. [4] proposed Interpolated Adversarial Training (IAT), the interpolation based adversarial training. IAT takes two losses in a training step, one is from interpolations between clean examples, and the other is from that between adversarial exampels. AVmixup, on the other hand, takes interpolations between the clean example and its adversarial vertex. Ultimately, IAT improves standard test error while maintaining the robustness, and AVmixup simultaneously improves both the standard accuracy and the adversarial robustness.

C. Dataset

We conducted experiments on the CIFAR10 [3], CIFAR100 [3], SVHN [6], and Tiny Imagenet [2] datasets. As shown in Table 7, the CIFAR10 and SVHN datasets comprise fewer classes and more data per class than other datasets such as the CIFAR100 and Tiny Imagenet datasets. Thus, the classification task of CIFAR100 and Tiny Imagenet are significantly more difficult than the tasks of CIFAR10 and SVHN.

Table 7: Dataset configuration used in the experiments.

Dataset	Train Size	Test Size	Class Size	Image Size
CIFAR10	50,000	10,000	10	(32, 32)
CIFAR100	50,000	10,000	100	(32, 32)
SVHN	73,257	26,032	10	(32, 32)
Tiny Imagenet	100,000	20,000	200	(64, 64)

D. Detailed Experimental Results

We present the detailed results of accuracy against white-box attacks on CIFAR100 in Table 8. AVmixup significantly improves the accuracy for Clean, FGSM, and PGD, but its improvement on CW20 is limited.

Additionally, we also provide the results of accuracy against black-box attacks on CIFAR10 and CIFAR100 in Tables 9 and 10, respectively. The models of each column represent the attack models of the transfer-based black-box attacks, and the models of each row represent the evaluated defense models. It is evident that the AVmixup model is the most robust against black-box attacks from all of the attack models with significant margins.

Table 8: Accuracy comparisons of our proposed approach AVmixup with PGD [5] and LSλ ($\lambda \in \{0.8, 0.9\}$) [9] against white-box attacks on CIFAR100.

Model	Clean	FGSM	PGD10	PGD20	CW20
Standard	78.57	6.56	0.0	0.0	0.0
PGD	61.29	46.04	25.76	25.17	22.98
LS0.8	62.1	52.33	29.47	28.81	**26.15**
LS0.9	61.77	53.17	27.71	27.13	25.34
AVmixup	**74.81**	**62.76**	**39.98**	**38.49**	23.46

Table 9: Accuracy comparisons against transfer-based black-box attacks on CIFAR10.

	PGD20					CW20				
Model	Standard	PGD	LS0.8	LS0.9	AVmixup	Standard	PGD	LS0.8	LS0.9	AVmixup
PGD	85.6	-	65.70	64.91	72.30	85.77	-	66.32	65.68	72.25
LS0.8	86.03	63.6	-	64.83	72.34	86.02	64.37	-	65.51	72.28
LS0.9	86.4	63.74	65.78	-	72.67	86.59	64.46	66.31	-	72.82
AVmixup	**89.53**	**68.51**	**71.48**	**70.50**	-	**89.65**	**69.40**	**71.72**	**70.97**	-

Table 10: Accuracy comparisons against transfer-based black-box attacks on CIFAR100.

	PGD20					CW20				
Model	Standard	PGD	LS0.8	LS0.9	AVmixup	Standard	PGD	LS0.8	LS0.9	AVmixup
PGD	51.59	-	36.68	36.74	40.88	51.97	-	37.99	38.62	40.15
LS0.8	60.27	41.44	-	37.61	46.27	60.35	42.15	-	38.01	44.6
LS0.9	59.93	40.68	36.86	-	45.73	60.3	41.61	36.97	-	44.42
AVmixup	**67.69**	**44.78**	**44.86**	**45.3**	-	**68.02**	**43.83**	**44.8**	**45.95**	-

E. Empirical Evidence on the Nature of Mixup

We previously mentioned that utilizing virtual data constructed by linear interpolation promotes the tight generalization of features observed in the training steps. In addition, we demonstrated that the accuracy of the combination of AVmixup with a feature scattering-based approach (Feature Scatter) against FGSM on CIFAR100 is slightly higher than that on Clean. In this section, we introduce the experimental results, which support the hypothesis. The following settings are compared in the experiment:

1. Standard: The model that is trained with the original dataset.
2. Mixup: With the original dataset, we apply Mixup [11] for the model.
3. Gvrm: The model that is trained by Vicinal Risk Minimization (VRM) with a Gaussian kernel [1].
4. Noisy mixup: With the Gaussian-noise-added dataset, we apply Mixup [11] for the model.

The original test set and the Gaussian-noise-added-test set are denoted as Clean and Noise, respectively, and we used the same Gaussian distribution $N(0, 8^2)$ for both the training and evaluation procedures.

Based on Table 11, we confirm the strong generalization performance of the mixup algorithm. In other words, the Mixup model and the Noisy mixup model show the highest accuracy on Clean and Noise, respectively.

To understand the property of AVmixup, we should focus on the results of the Noisy mixup model, which shows the worst accuracy on Clean and the best accuracy on Noise. Moreover, the accuracy on Noise is significantly higher than that on Clean by 2.29%p. This implies that the Noisy mixup model tightly generalizes the input distribution, which is slightly changed owing to Gaussian noise during the training. Therefore, AVmixup with the PGD-based approach does not show a high level of robustness against other types of adversarial attacks, and the accuracy of AVmixup with Feature Scatter against FGSM on CIFAR100 can be higher than that on Clean for a high scaling factor ($\gamma = 1.5$).

Table 11: Accuracy (median over 5 runs) comparisons on Clean and Noise.

Model	Clean	Noise
Standard	94.52	85.8
Mixup	**95.93**	88.97
Gvrm	92.77	92.96
Noisy mixup	92.56	**94.85**

F. Ablation Study on CIFAR10

Table 12: Accuracy results of AVmixup with PGD-based approach on CIFAR10. γ is the scaling factor of AVmixup, and λ_1 and λ_2 are label-smoothing factors for the raw input and the adversarial vertex, respectively.

	$\gamma = 1.0$		$\gamma = 2.0$	
λ_1 / λ_2	Clean	PGD10	Clean	PGD10
1.0 / 0.1	**92.88**	7.43	**92.63**	**53.19**
1.0 / 0.5	91.29	42.49	92.04	49.33
0.5 / 0.1	92.59	14.07	91.78	51.43
0.5 / 0.7	90.12	**47.36**	84.93	53.06

Table 13: Accuracy results of AVmixup with feature scattering-based approach on CIFAR10.

	$\gamma = 1.0$				$\gamma = 2.0$			
λ_1 / λ_2	Clean	FGSM	PGD20	CW20	Clean	FGSM	PGD20	CW20
0.1 / 0.5	91.94	80.09	74.43	62.87	90.87	93.42	**72.94**	**70.11**
0.5 / 0.3	92.82	77.81	57.67	55.18	95.41	95.32	60.95	47.32
0.5 / 0.7	92.37	**83.49**	**82.31**	**71.88**	95.2	**95.86**	52.87	47.79
1.0 / 0.5	**93.07**	79.55	53.42	56.72	**95.43**	95.12	47.41	44.14

To analyze the sensitivity of AVmixup to the hyperparameters, we conducted an ablation study, and the results are summarized in Tables 12 and 13. As shown in both tables, the larger the λ_1, the higher is the accuracy on Clean. Furthermore, λ_1

and λ_2 imply the weights on standard accuracy and adversarial robustness, respectively. However, they do not fit perfectly as changes occured by other conditions.

From the comparisons based on γ, it is evident that AVmixup with the PGD-based and feature scattering-based approaches yielded the highest accuracy at $\gamma = 1$ and at $\gamma = 2$, respectively. This is owing to the difference in the construction of adversarial examples between the two approaches. Because the PGD-based approach constructs adversarial examples using a multi-step method, adversarial examples are located in the ℓ_∞-bounded cube. Otherwise, because the feature scattering-based approach constructs adversarial examples using a one-step method, adversarial examples are located on the surface of the ℓ_∞-bounded cube. Therefore, with the PGD-based approach, AVmixup requires a larger scaling factor to cover the ℓ_∞-bounded space. Likewise, with the feature scatter-based approach, AVmixup requires the scaling factor of 1 to cover the ℓ_∞-bounded space.

The sensitivity of AVmixup is largely related to the label-smoothing factors, which tune the input data labels for robust generalization based on our theoretical results. They are also connected to the Lipschitz constant for the classifier, with respect to the training samples in adversarial directions. This complex relationship causes the observed high-sensitivity. This may be interpreted as a fault of AVmixup, but also provides meaningful insight into useful directions for future research.

References

[1] Olivier Chapelle, Jason Weston, Léon Bottou, and Vladimir Vapnik. Vicinal risk minimization. In *Advances in neural information processing systems*, pages 416–422, 2001.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. `https://tiny-imagenet.herokuapp.com/`.

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

[4] Alex Lamb, Vikas Verma, Juho Kannala, and Yoshua Bengio. Interpolated adversarial training: Achieving robust neural networks without sacrificing accuracy. *stat*, 1050:16, 2019.

[5] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

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

[7] Tianyu Pang, Kun Xu, and Jun Zhu. Mixup inference: Better exploiting mixup to defend adversarial attacks. In *International Conference on Learning Representations*, 2019.

[8] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems*, pages 5014–5026, 2018.

[9] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[10] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

[11] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.