Kernel-convoluted Deep Neural Networks with Data Augmentation

Minjin Kim, [1] **Young-geun Kim,** [1] **Dongha Kim,** [1] **Yongdai Kim,** [2] **Myunghee Cho Paik** [1]
[1] Department of Statistics, Seoul National University
[2] School of Data Science, Seoul National University
kmj1404@snu.ac.kr, cib009@snu.ac.kr, dongha0718@hanmail.net, ydkim0903@gmail.com, myungheechopaik@snu.ac.kr

Abstract

The Mixup method, which uses linearly interpolated data, has emerged as an effective data augmentation tool to improve generalization performance and the robustness to adversarial examples. The motivation is to curtail undesirable oscillations by its implicit model constraint to behave linearly at in-between observed data points and promote smoothness. In this work, we formally investigate this premise, propose a way to impose smoothness constraints explicitly, and extend it to incorporate implicit model constraints. First, we derive a new function class composed of kernel-convoluted models (KCM) where the smoothness constraint is directly imposed by locally averaging the original functions with a kernel function. Second, we propose to incorporate the Mixup method into KCM to expand the domains of smoothness. In both cases of the KCM and the KCM adapted with the Mixup, we provide risk analysis, respectively, under mild conditions on kernel functions. As a result, we show that the upper bound of the excess risk over a new function class is not slower than that of the excess risk over the original function class. Using CIFAR-10 and CIFAR-100 datasets, our experiments demonstrate that the KCM with the Mixup outperforms the Mixup method in terms of generalization and robustness to adversarial examples.

Introduction

Deep neural networks have brought an outstanding performance in various fields such as computer vision (Krizhevsky, Sutskever, and Hinton 2012), speech recognition (Graves, Mohamed, and Hinton 2013), and reinforcement learning (Silver et al. 2016). To train deep neural networks, we solve the empirical risk minimization (ERM) problem given data, but this solution could lead to a small training error, but a large test error, known as overfitting. This means that the deep neural networks could memorize a training sample, have poor generalization ability, and lack robustness against adversarial attacks.

Among the many techniques for regularization to reduce overfitting, data augmentation has been widely used to improve generalization performance in machine learning. In particular, in image classification, various data augmentation methods such as horizontal reflections and rotations have been commonly applied (Krizhevsky, Sutskever, and Hinton 2012; Simard et al. 1998). Recently, sample-mixed augmentation, called Mixup, has emerged as an effective tool (Zhang et al. 2018). The Mixup method generates virtual samples based on linear interpolations between random pairs of inputs and their corresponding labels. Trained deep models using the Mixup-generated samples have demonstrated superb performances in supervised learning (Zhang et al. 2018; Liang et al. 2018), unsupervised learning (Beckham et al. 2019; Xie et al. 2019), and semi-supervised learning (Berthelot et al. 2019; Verma et al. 2019b). The authors of the Mixup method (Zhang et al. 2018) conjecture a poor generalization and lack of the robustness to adversarial examples may be due to unnecessary oscillations at in-between observed data points. The method is designed to encourage the model to behave linearly between training samples to curtail undesirable oscillations when predicting outside the training examples. This motivational claim is cogent, but has not been scrutinized.

In this paper, we formally investigate this premise, propose a way to explicitly impose constraints, and extend it to incorporate with implicit constraints. First, we propose a function class where constraints are explicitly imposed and provide a formal risk analysis. Second, we propose to incorporate the Mixup method in the proposed function class and provide corresponding risk analysis.

In the first part, our strategy starts by paying attention to the role of data augmentation in placing model constraints. For example, data augmentation techniques such as horizontal reflections and rotations encourage training to find invariant functions to corresponding transformations. Recent works have attempted to explicitly impose these constraints through careful construction of models (Cohen and Welling 2016; Tai, Bailis, and Valiant 2019; van der Wilk et al. 2018). Using interpolated data instead of original samples, the Mixup method encourages the models to implicitly satisfy the linearity constraint and promote smoothness. In contrast to data augmentation for rotations or flipping, there has not been an effort to explicitly impose constraints by constructing models. We fill this gap and build models that explicitly bring desirable constraints of smoothness. We introduce a new function class composed of *kernel-convoluted models* (KCM), a derived function class given original functions. In the KCM, the smoothness constraint is directly im-

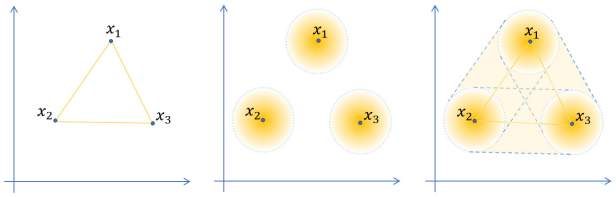

Figure 1: An illustration of regions where the Mixup method (left), KCM (middle), and KCM with the Mixup method (right) impose smoothness when there are three data points.

posed by locally averaging the original functions with a kernel function via convolution. To derive tangible theoretical results, we focus on product kernel functions but allow various kernels. We provide the upper bound of the excess risk in relation to the complexity of the original function class where the original function is a linear function or a deep neural network.

In the second part of our paper, we incorporate the Mixup into the KCM and conduct a corresponding risk analysis. As opposed to the previous endeavors that replaced data augmentation by explicitly modeling invariance to a certain transformation, we blend the two to expand the domain of smoothness. While the KCM imposes smoothness over local regions around observed data defined by kernels as depicted by the yellow circles in the second subplot of Figure 1, the Mixup method does over edges between two observed data as vertexes, as depicted by the yellow line in the first subplot of Figure 1. Therefore, combining the two may expand the domains of smoothness, as depicted by the yellow triangular region in the third subplot of Figure 1. Using the two-moon dataset (Pedregosa et al. 2011), we also show that the Mixup method promotes smoothness and the KCM renders the decision boundary visibly smoother than that of the Mixup method in Figure 2. For the minimizer of the risk of the KCM with the Mixup, we provide risk analysis and show that the upper bound of the excess risk can be expressed in terms of the complexity of the function class composed of KCM and the smoothing parameter of the KCM, and the size of perturbation of the Mixup. Our main contributions are summarized as follows.

- We propose a new model called a kernel-convoluted model (KCM) where the smoothness constraint is directly imposed by locally averaging the original functions with a kernel function, and provide a risk analysis under some conditions for kernels.
- We propose to incorporate the Mixup method into the KCM to expand the domains of smoothness. We provide corresponding risk analysis and show that the upper bound of the excess risk is $O(n^{-1/2})$ when the original function is a deep neural network and the perturbation order of the Mixup is faster than $n^{-1/2}$ where n is a sample size (Theorem 3).
- Using CIFAR-10 and CIFAR-100 datasets, we demonstrate that the KCM with the Mixup outperforms the Mixup method in terms of generalization and robustness to adversarial examples.

The rest of the paper is organized as follows: first, we review work related to the Mixup and efforts to build models that explicitly impose invariant constraints handled by data augmentation; then, we introduce a new function where smoothness is explicitly imposed and show its excess risk bound. Afterward, we propose incorporating the Mixup into the proposed model and conduct the corresponding risk analysis; finally, we present experimental results and outline conclusions. We also provide assumptions, some theoretical results, and proofs in Appendix 2 in the Supplementary material. The source-code for conducting our experiments of binary classification on the two-moon dataset and CIFAR-10 (cat vs. dog) and multi-class classification on CIFAR-10 is available at

https://github.com/MJ1021/kcm-code.

Related Works and Preliminaries

Since the Mixup has shown its effectiveness, many researchers have proposed variations and modifications. One kind of modification is to interpolate in the hidden space representations (DeVries and Taylor 2017; Guo, Mao, and Zhang 2019; Verma et al. 2019a). Another variation is to choose more than a pair (Guo, Mao, and Zhang 2019). Tokozume, Ushiku, and Harada (2018) consider alternative interpolation schemes involving labels. Moreover, Liang et al. (2018) use the spatial information to generate a new synthetic sample by stitching the space domain of two images with different proportions of the area of a synthetic image.

There have been efforts to formalize data augmentation. One approach is to establish a theoretical framework for understanding data augmentation. For example, Dao et al. (2019) provide a general model of augmentation as a Markov process in which augmentation is performed via a random sequence of transformations. Other attempts include seeking an alternative yet formal way to replace the role of data augmentation. For example, one of the roles of data augmentation is to impose constraints of invariance on certain transformations. Some authors have proposed to formally construct a model to enforce desired transformation invariance constraints. In some images, rotation, flipping, and rescaling do not change labels. Such transformation invariances of a domain are often encouraged by data augmentation. Group invariance convolutional neural networks (Cohen and Welling 2016; Dieleman, De Fauw, and Kavukcuoglu 2016; Marcos et al. 2017; Worrall et al. 2017) enforces rotation or translation invariance. Equivariant transformer models (Tai, Bailis, and Valiant 2019) offer flexible types of invariance with the use of specially-derived canonical coordinate systems. van der Wilk et al. (2018) demonstrate the explicit imposition of model constraint for the Gaussian process using the marginal likelihood criterion. In their works, the models are constructed in such a way that the predictions are invariant to transformations. Furthermore, van der Wilk et al. (2018) argue the advantage of placing constraints directly through the model because the ob-

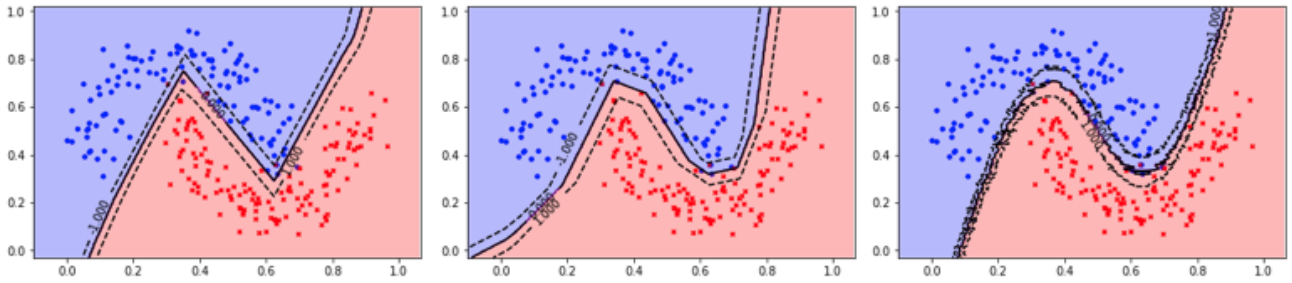

Figure 2: Visualization of the contours of classifiers learned with ERM (left), Mixup (middle), and KCM (right). Data points from classes -1 and 1 of the two-moon dataset are indicated by blue circles and red crosses, respectively. Decision boundaries are represented as solid lines and contours of levels -1 and 1 are represented by dashed lines. The best configuration of α for Mixup and h for KCM is presented in Table 3 in the Supplementary material is presented.

jective function correctly reflects the reduced complexity of constrained models. Recently, Wu et al. (2020) analyze the effect of the label-preserving transformations via the ridge estimator in an over-parametrized linear regression.

As for the Mixup in the classification problem, few studies have constructed a model to enforce desired constraints. This paper fills this gap. We present the models with explicit constraints that the Mixup attempts to attain and propose a class of models that achieves flexible types of smoothness.

Kernel-convoluted Neural Networks and its Excess Risk Bound

Problem Setup

Before we describe the Mixup and study its constraints, we introduce the problem setup and notations. We restrict our attention to a binary classification problem although the proposed method can be easily extended to a multi-class classification. Let $\mathcal{X} \subseteq \mathbb{R}^d$ be an input space and $\mathcal{Y}$ be an output space. Assume that $\mathcal{Y} = \{-1, 1\}$. For a given real-valued function $f : \mathcal{X} \to \mathbb{R}$, we denote the classifier by $C^*(\mathbf{x}; f)$ where $C^*(\mathbf{x}; f) = \text{sign}\big(f(\mathbf{x})\big)$ for $\mathbf{x} \in \mathcal{X}$.

Other notations are as follows. For a real-valued function g, we denote L_g by a Lipschitz constant of g. The p-norm of $\mathbf{x}$ is defined by $\|\mathbf{x}\|_p = \big(|x_1|^p + \cdots + |x_d|^p\big)^{1/p}$ where $\mathbf{x} = (x_1, \cdots, x_d)^T$ and $p \geq 1$. For a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times m}$, we denote the spectral norm of $\boldsymbol{A}$ by $\|\boldsymbol{A}\|_2$. Given two positive real values a, b, we write $a \lesssim b$ if $a \leq cb$ for some generic constant $c > 0$. For two sequences $\{a_n\}$ and $\{b_n\}$, we write $a_n = O(b_n)$ if there exists a positive real number M and a natural number n_0 such that $|a_n| \leq Mb_n$ for all $n \geq n_0$. We let $[n] = \{1, \ldots, n\}$ for $n \in \mathbb{N}$.

To elicit implicit model constraints of the Mixup method, we briefly describe its steps (Zhang et al. 2018). We denote the training data as $S = \{(\mathbf{x}_i, y_i) : i \in [n]\}$, which is a set of random samples drawn from the unknown joint distribution $\mathbb{P}_{\text{data}}$. For a randomly selected pair from the training data S, $(\mathbf{x}_i, y_i)$ and $(\mathbf{x}_j, y_j)$, consider linear interpolations,

$$\tilde{\mathbf{x}}_i = (1-\lambda)\mathbf{x}_i + \lambda\mathbf{x}_j \text{ and } \tilde{y}_i = (1-\lambda)y_i + \lambda y_j,$$

where $\lambda \in [0, 1]$ is the interpolation weight. The Mixup method uses $\{(\tilde{\mathbf{x}}_i, \tilde{y}_i) : i \in [n]\}$ instead of originally observed data. By plugging in $(\tilde{\mathbf{x}}_i, \tilde{y}_i)$, the trained model is forced to satisfy $f(\tilde{\mathbf{x}}_i) \approx (1-\lambda)f(\mathbf{x}_i) + \lambda f(\mathbf{x}_j)$. Thus, the Mixup method implicitly imposes the linear constraint, which induces smoothing out the model to increase generalization power and robustness to adversarial examples.

Kernel-convoluted Models (KCM)

In this section, we define kernel-convoluted models (KCM). Let $\mathcal{F} = \{f \mid f : \mathcal{X} \to \mathbb{R},\ f \text{ is a measurable function}\}$.

Definition 1 (Kernel-convoluted models) *Assume that $\mathcal{X}$ is convex. For a real-valued function f and a measure defined on Borel σ-algebra Σ of subsets of $\mathcal{X}$, K^*, we call f^{K^*} kernel-convoluted models where*

$$f^{K^*}(\mathbf{x}) = K^* * f(\mathbf{x}) = \int f(\mathbf{x} - \mathbf{u}) dK^*(\mathbf{u}) \text{ for } \mathbf{x} \in \mathcal{X}. \quad (1)$$

The resulting function, f^{K^*}, is a weighted average of f at different input locations. If K^* is absolute continuous with respect to the Lebesgue measure μ, we denote the corresponding density by K. Then, (1) reduces to

$$f^{K^*}(\mathbf{x}) = \int f(\mathbf{x}–\boldsymbol{u}) K(\boldsymbol{u}) d\boldsymbol{u} = \int K(\mathbf{x}–\boldsymbol{u}) f(\boldsymbol{u}) d\boldsymbol{u}. \quad (2)$$

A convenient kernel to handle multivariate case is a product kernel function defined by

$$K(\mathbf{u}) = \prod_{j=1}^{d} k_h(u_j) = h^{-d} \prod_{j=1}^{d} k\Big(\frac{u_j}{h}\Big),$$

where k is a univariate kernel function, $k_h(u) = h^{-1}k\big(u/h\big)$, and $h > 0$ is a bandwidth. When k is the Gaussian probability density function, the corresponding product kernel is called a d-dimensional Gaussian kernel.

With the above definition on kernel function , h represents degree of heterogeneity of kernelizing models. As h shrinks, the kernel-convoluted model converges to the original function: that is,

$$\lim_{h\to 0} \int f(\mathbf{x} - \boldsymbol{u}) K(\boldsymbol{u}) d\boldsymbol{u} = f(\mathbf{x}).$$

Generalization Error Bounds via the Empirical Rademacher Complexity of KCM

We restrict our attention to the case where K^* is absolutely continuous with respect to the Lebesgue measure μ as (2) and suppress superscript * for brevity. We denote $\mathcal{F}^K = \{f^K \mid f \in \mathcal{F}\}$ where K is the corresponding density function. Assume that K is the product kernel. For a given real-valued kernel-convoluted function f^K defined on $\mathcal{X}$, we consider a new classifier $C(\mathbf{x}; f^K)$ as $C(\mathbf{x}; f^K) = \operatorname{sign}\big(f^K(\mathbf{x})\big)$ for $\mathbf{x} \in \mathcal{X}$. The classification risk of f^K is defined by

$$\begin{aligned} R(\mathbb{P}_{\text{data}}, f^K) &= \mathbb{E}_{(\mathbf{x},y)\sim\mathbb{P}_{\text{data}}}\left[I\big(C(\mathbf{x}; f^K) \neq y\big)\right] \\ &= \mathbb{E}_{(\mathbf{x},y)\sim\mathbb{P}_{\text{data}}}\left[\ell\big(yf^K(\mathbf{x})\big)\right], \end{aligned}$$

where $I(\cdot)$ is an indicator function and $\ell(z) = 1$ if $z \leq 0$ and is 0 otherwise. While the 0-1 loss can be used for binary classification problem, directly minimizing the corresponding empirical risk is NP-hard due to the non-convexity of the ℓ (Hoffgen, Simon, and Vanhorn 1995). To resolve this issue, many authors have been considered a suitable surrogate loss that has advantageous properties (Bartlett, Jordan, and McAuliffe 2006; Kim, Ohn, and Kim 2018; Mohri, Rostamizadeh, and Talwalkar 2018; Yin, Kannan, and Bartlett 2019). Denote a surrogate loss function $\phi : \mathbb{R} \to [0, B]$. To avoid confusion, we add ϕ to a sub-index in the above definition of the risk and call it as ϕ-risk: that is, for a given surrogate loss function ϕ, we define

$$\begin{aligned} R_\phi(\mathbb{P}_{\text{data}}, f^K) &= \mathbb{E}_{(\mathbf{x},y)\sim\mathbb{P}_{\text{data}}}\phi\big(yf^K(\mathbf{x})\big), \\ R_\phi(\mathbb{P}_{\text{data}}, f^*_{\phi,\text{conv}}) &= \inf_{f^K\in\mathcal{F}^K} R_\phi(\mathbb{P}_{\text{data}}, f^K), \\ R_\phi(\mathbb{P}_{\text{data}}, f^*_\phi) &= \inf_{f\in\mathcal{F}} R_\phi(\mathbb{P}_{\text{data}}, f). \end{aligned}$$

To estimate the ϕ-risk we replace $\mathbb{P}_{\text{data}}$ by empirical distribution, $\mathbb{P}_n$ and the surrogate empirical risk based on the training data S can be expressed as $R_\phi(\mathbb{P}_n, f^K) = \frac{1}{n}\sum_{i=1}^n \phi\big(y_i f^K(\mathbf{x}_i)\big)$.

We use the empirical Rademacher complexity of $\mathcal{G}$ given S_z (Bartlett and Mendelson 2002), defined as

$$\hat{\mathcal{R}}_{S_z}(\mathcal{G}) = \mathbb{E}_{\boldsymbol{\epsilon}}\Big[\sup_{g\in\mathcal{G}} \frac{1}{n}\sum_{i=1}^n \epsilon_i g(z_i)\Big]$$

to bound the generalization error where $\mathcal{G}$ is a family of functions mapping from $\mathcal{Z}$ to $[a, b]$, $S_z = \{z_i : i \in [n]\}$ is a fixed sample of size n, and $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)^T$ and $\epsilon_1, \ldots, \epsilon_n$ are random variables with $\mathbb{P}(\epsilon_i = 1) = \mathbb{P}(\epsilon_i = -1) = 1/2$ (Mohri, Rostamizadeh, and Talwalkar 2018). Let

$$\hat{f}_{\phi,\text{conv}} = \operatorname*{arg\,min}_{f^K\in\mathcal{F}^K} R_\phi(\mathbb{P}_n, f^K).$$

Now, we present the excess risk of $\hat{f}_{\phi,\text{conv}}$ relative to $R_\phi(\mathbb{P}_{\text{data}}, f^*_{\phi,\text{conv}})$, and to $R_\phi(\mathbb{P}_{\text{data}}, f^*_\phi)$, respectively. To avoid confusion, we call the former the excess KCM risk, and the latter, the excess risk. For the excess KCM risk, we can extend the results on the risks over $\mathcal{F}$ via Rademacher complexity (Bartlett and Mendelson 2002; Mohri, Rostamizadeh, and Talwalkar 2018), and show that the upper bound of the excess KCM risk is bounded as long as the Rademacher complexity of $\mathcal{F}^K$ is well-controlled. Details are in Corollary 1 in the Supplementary material.

Theorem 1 *(Excess risk bound) Assume that the Lipschitz continuity on ϕ and f stated in (A1) and (A2) in the Supplementary material as well as (A3) stated in below hold. For any $\eta > 0$, with probability at least $(1-\eta)$, we have*

$$\begin{aligned} &R_\phi(\mathbb{P}_{\text{data}}, \hat{f}_{\phi,\text{conv}}) - R_\phi(\mathbb{P}_{\text{data}}, f^*_\phi) \\ &\qquad \leq 4L_\phi\hat{\mathcal{R}}_S(\mathcal{F}^K) + 6B\sqrt{\frac{\log\frac{4}{\eta}}{2n}} + O(h). \end{aligned}$$

The first term of the upper bound is due to the Ledoux-Talagrand contraction inequality (Mohri, Rostamizadeh, and Talwalkar 2018), and is a function of h through $\mathcal{F}^K$. Theorem 1 states that the excess risk bound depends on the empirical Rademacher complexity, the degree of heterogeneity of kernelized models (h), the sample size (n), and the certainty in the success of the whole procedure (η). Therefore, a wide bandwidth can be a dominating term of the upper bound and leads to a slower convergence. The next section shows when f is a deep neural network, certain choices of kernels can keep the empirical Ramemacher complexity of the new function class the same as that of the original function class asymptotically.

The Empirical Rademacher Complexity of $\mathcal{F}^K$

In this section, we study the bounds of $\hat{\mathcal{R}}_S(\mathcal{F}^K)$ shown in Theorem 1 and its relationship with $\hat{\mathcal{R}}_S(\mathcal{F})$. To delve into the effect of the convolution, we first consider the case when f is a linear function with l_p norm constraints. A linear function bases on a rich class of hypotheses used in several learning algorithms such as ridge regression (Hoerl and Kennard 1970) and support vector machines (Cortes and Vapnik 1995). Moreover, the l_p norm constraints have been widely considered to constrain the family of linear predictors by many authors until recently (Awasthi, Frank, and Mohri 2020). We show that a symmetric univariate kernel function allows us to avoid polynomial dimension dependence even if $p > 1$ and keep the same Rademacher complexity as that of the original class. Details are given in Theorem 2 in the Supplementary material.

Now, we characterize the KCM when the original function f is a deep neural network. For deep neural networks with L hidden layers, we denote a sequence of matrices by $\mathcal{W} = \{\boldsymbol{W}_1, \ldots, \boldsymbol{W}_{L+1}\}$, where $\boldsymbol{W}_s \in \mathbb{R}^{d_s\times d_{s-1}}$ for $s \in [L+1]$. Note that $d_0 = d$ and $d_{L+1} = 1$. Let

$$f_{\mathcal{W}}(\mathbf{x}) = \boldsymbol{W}_{L+1}\rho\Big(\boldsymbol{W}_L\rho\big(\cdots\rho(\boldsymbol{W}_1\mathbf{x})\big)\Big),$$

where $[\rho(\mathbf{x})]_j = \rho(x_j)$ and ρ is an activation function. We use the rectifier linear unit (ReLU) activation function $\rho(t) = \max\{0, t\}$ for $t \in \mathbb{R}$. Let $d_{\max} = \max\{d_0, d_1, \ldots, d_{L+1}\}$. The loss function can be written as $\phi\big(yf_{\mathcal{W}}(\mathbf{x})\big)$ where $\phi : \mathbb{R} \to [0, B]$ is L_ϕ-Lipschitz. Define the function class for deep neural networks with spectrally-normalized constraints as follows:

$$\mathcal{F}_{\text{DNN},\|\cdot\|_2} = \{\mathbf{x} \mapsto f_{\mathcal{W}}(\mathbf{x}) \mid \|\boldsymbol{W}_s\|_2 \leq r_s, s \in [L+1]\}.$$

We state the assumptions for the following theorem.

(A3) For some constant $c_K^1 > 0$, $\int K_{\mathrm{p}}(\boldsymbol{u})\|\boldsymbol{u}\|_2 d\boldsymbol{u} < c_K^1 < \infty$ where $K_{\mathrm{p}}(\boldsymbol{t}) = \prod_{j=1}^d k(t_j)$.

(A4) For any $\mathbf{x} \in \mathcal{X}$, $\|\mathbf{x}\|_2 \leq B_x$ for some constant $B_x > 0$.

Under (A4), Li et al. (2018) show that $\hat{\mathcal{R}}_S\big(\mathcal{F}_{\mathrm{DNN},\|\cdot\|_2}\big) \lesssim G$, where

$$G = \frac{B_x \prod_{s=1}^{L+1} r_s}{\sqrt{n}} \sqrt{d_w \log\left(\frac{(L+1)\sqrt{n} \max_{1\leq s\leq L+1}\{r_s m_s\}}{\sqrt{d_w} \min_{1\leq s\leq L+1} r_s}\right)},$$

$d_w = d_0 \times d_1 + \cdots + d_L \times d_{L+1}$, $\mathrm{rank}(\boldsymbol{A})$ is the rank of a matrix $\boldsymbol{A}$, and $m_s = \sqrt{\mathrm{rank}(\boldsymbol{W}_s)}$. Let $\mathcal{F}^K_{\mathrm{DNN},\|\cdot\|_2} = \{\mathbf{x} \mapsto K * f_{\mathcal{W}}(\mathbf{x}) \mid f_{\mathcal{W}} \in \mathcal{F}_{\mathrm{DNN},\|\cdot\|_2}\}$. The following theorem shows the empirical Rademacher complexity of $\mathcal{F}^K_{\mathrm{DNN},\|\cdot\|_2}$ given S.

Theorem 2 *Assume that K is a symmetric density function. Under (A3) and (A4), we have*

$$\hat{\mathcal{R}}_S\big(\mathcal{F}^K_{\mathrm{DNN},\|\cdot\|_2}\big) \lesssim \frac{B_*}{B_x} G,$$

where $B_* = 3hc_K^1 + B_x$.

We have analogous results with the linear case in that choosing symmetric kernel functions: the complexity of the new kernel-convoluted function class given S will be asymptotically equivalent to that of the original one.

Kernel-convoluted Models with Mixup

Generalization Error Bound of KCM with Mixup

In this section, we adapt the Mixup method to the KCM. A motivation of such adaptation is to broaden the domain of smoothness as depicted in Figure 1. We conduct risk analysis of the proposed function class when the Mixup method is incorporated. The proposed estimator is

$$\hat{f}_{\phi,\mathrm{prop}} = \operatorname*{arg\,min}_{f^K \in \mathcal{F}^K} R_{\phi,\mathrm{prop}}(\mathbb{P}_n, f^K),$$

where $R_{\phi,\mathrm{prop}}(\mathbb{P}_n, f^K) = \frac{1}{n}\sum_{i=1}^n \phi\big(\tilde{y}_i f^K(\tilde{\mathbf{x}}_i)\big)$. In this paper, we consider the case where the interpolation weight depends on n, and is denoted by λ_n. We show in Theorem 3 in the Supplementary material that the excess KCM risk with Mixup of $\hat{f}_{\phi,\mathrm{prop}}$ defined by

$$R_\phi(\mathbb{P}_{\mathrm{data}}, \hat{f}_{\phi,\mathrm{prop}}) - R_\phi(\mathbb{P}_{\mathrm{data}}, f^*_{\phi,\mathrm{conv}})$$

can be bounded by $\hat{\mathcal{R}}_S(\mathcal{F}^K)$, the term involving λ_n and the bandwidth h.

Theorem 3 *(Excess risk with Mixup) Let $\mathcal{X}$ be a compact set. Under (A1)-(A6) in the Supplementary material for any $\eta > 0$, with probability at least $(1-\eta)$ we have:*

$$R_\phi(\mathbb{P}_{\mathrm{data}}, \hat{f}_{\phi,\mathrm{prop}}) - R_\phi(\mathbb{P}_{\mathrm{data}}, f^*_\phi) \leq 4L_\phi \hat{\mathcal{R}}_S(\mathcal{F}^K) + 6B\sqrt{\frac{\log\frac{4}{\eta}}{2n}} + O(\lambda_n) + O(h).$$

Algorithm 1 Training Mixup with KCM

Input: Training data $S = \{(\mathbf{x}_i, y_i)\}_{i\in[n]}$, kernel density function K, parameter of Beta distribution α, size of mini-batch n_B, sample size for Monte Carlo approximation N
Parameter: Parameter $\boldsymbol{\theta}$ in a given model $f_{\boldsymbol{\theta}}$
Output: Updated parameter $\hat{\boldsymbol{\theta}}$
Initialize parameters $\boldsymbol{\theta}$ in the model $f_{\boldsymbol{\theta}}$
for $t = 1$ **to** $\mathrm{iter}_{\max}$ **do**
  Randomization of λ: $\lambda_j \sim \mathrm{Beta}(\alpha, \alpha)$ for $j \in [n_B]$ and put $\lambda = \min\{(1-\lambda), \lambda\}$
  Mixup: for randomly selected pairs $(\mathbf{x}_j, y_j)$ and $(\mathbf{x}_{j'}, y_{j'})$, construct $\tilde{S}_B = \{(\tilde{\mathbf{x}}_j, \tilde{y}_j)\}_{j\in[n_B]}$, where

$$\tilde{\mathbf{x}}_j = (1-\lambda_j)\mathbf{x}_j + \lambda_j \mathbf{x}_{j'}$$
$$\tilde{y}_j = (1-\lambda_j) y_j + \lambda_j y_{j'}.$$

  Realization of $\mathbf{u}$: $\mathbf{u}_{i'} \sim K(\cdot)$ for $i' \in [N]$
  for $j = 1$ **to** n_B **do**
    Averaging $\bar{f}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_j) = \frac{1}{N}\sum_{i'} f_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_j - \mathbf{u}_{i'})$
  end for
  Update $\boldsymbol{\theta}$ **by descending**: $1/n_B \sum_j \phi\big(\tilde{y}_j \bar{f}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_j)\big)$
end for

Theorem 3 shows that the excess risk with Mixup of $\hat{f}_{\phi,\mathrm{prop}}$ can be bounded by the terms from the upper bound of the excess KCM risk with Mixup and the size of the bandwidth. This implies that even with choosing a symmetric kernel, the dependence on h still remains. When perturbation of the Mixup and smoothing range of KCM are smaller than $O(n^{-1/2})$, the upper bound would be asymptotically equivalent to that of the original class.

Algorithm

The algorithm for the proposed method is provided in Algorithm 1. In implementing the KCM, the kernel-convoluted function can be approximated by Monte Carlo approximation, $f^K(\mathbf{x}) \approx \hat{f}^K(\mathbf{x}) = N^{-1}\sum_{i'=1}^N f(\mathbf{x} - \mathbf{u}_{i'})$ where $\mathbf{u}_{i'}$ are random samples from a kernel density function K for $i' \in [N]$. We bring attention to the fact that the case of N being 1 gives an unbiased estimator of the integral. In our experiments on CIFAR-10 and CIFAR-100, we find that the proposed method with N=1 or 5 produces better test accuracies than the Mixup method as described in the next section.

Experiment

We conduct experiments using three datasets, the two-moon dataset (Pedregosa et al. 2011), CIFAR-10, and CIFAR-100 (Krizhevsky and Hinton 2009) for both binary classification and multi-class classification. The results for binary classification are summarized in Tables 3 and 4 in Appendix 1 in the Supplementary material. For multi-class classification, we extend our binary classification setup by employing the cross-entropy loss function. Using the datasets, we compare (i) ERM, (ii) Mixup, (iii) KCM with various degrees of h and N and (iv) Mixup with KCM and compare the effect

Dataset	Learning rule	Test accuracy
CIFAR-10	ERM	95.01
	KCM $(0.01, 1)$	95.06
	MIXUP	96.11
	MIXUP + KCM $(0.01, 5)$	**96.39**
CIFAR-100	ERM	74.44
	KCM $(0.01, 1)$	75.34
	MIXUP	78.47
	MIXUP + KCM $(0.01, 1)$	**79.12**

Table 1: (Test accuracy) The median test accuracies of the last 10 epochs. For KCM, the configuration pair (h, N) represents the combination of the bandwidth h and the sample size for Monte Carlo approximation N. For MIXUP, we set $\alpha = 1$. The full results are summarized in Table 5 in the Supplementary material.

of implicit constraint, explicit constraint through the proposed KCM, and combined constraint through the Mixup with KCM. We denote the Mixup method as 'MIXUP' and the Mixup with KCM as 'MIXUP+KCM'. We also conduct experiments for robustness to adversarial examples to show that MIXUP+KCM can significantly improve the robustness of neural networks compared to MIXUP. We use the d-dimensional Gaussian kernel. We provide implementation details and results in Appendix 1 in the Supplementary material.

CIFAR-10 and CIFAR-100

The CIFAR-10 dataset consists of 60000 RGB images in 10 classes, with 6000 images per class. The CIFAR-100 dataset is similar to CIFAR-10, except it has 100 classes containing 600 images each. Both datasets have 50000 training images and 10000 test images.

To make a direct comparison with the original Mixup using CIFAR-10/100, we adopt the experimental configuration in the Mixup paper (Zhang et al. 2018) and use the author's official code[1]. As for MIXUP+KCM, we add code for the local averaging part (the fourth line from the bottom in Algorithm 1). We note that there are training and test phases, and the performance of the methods is measured by the median of test accuracies of the last 10 epochs as the Mixup paper considered. We use ResNet-34, which is one of the architectures from the official code.

Table 1 reports the median test accuracies of the last 10 epochs with the best configuration for KCM and MIXUP+KCM. The detailed results are summarized in Table 5 in Supplementary material. In particular, in CIFAR-10, with an appropriate choice of (h, N), KCM/MIXUP+KCM (resp.) outperforms ERM/MIXUP (resp.). However, in CIFAR-100, consideration of KCM leads to better performance both ERM and MIXUP for all configurations.

[1]https://github.com/facebookresearch/mixup-cifar10

Robustness to Adversarial Examples

Neural networks trained using ERM are vulnerable to visually imperceptible but thoroughly chosen adversarial perturbations, which lead to deterioration of the performance of the model (Szegedy et al. 2014). Data augmentation also has been commonly used to increase model robustness (Goodfellow, Shlens, and Szegedy 2015; Zhang et al. 2018; Rajput et al. 2019). For example, Rajput et al. (2019) analyzes the effect of data augmentation via the lens of margin to provably improve robustness, especially the required size of the augmented data set for ensuring a positive margin.

To examine the robustness to adversarial examples and make a direct comparison to the Mixup method, we follow the experimental setting as the author employed in the original Mixup paper (Zhang et al. 2018). We use ResNet-34 model: two of them trained via ERM on CIFAR-10/100, the third trained using KCM, the fourth trained using the Mixup, and the fifth trained using the Mixup with KCM. We administer white/black box attacks generated by FGSM and I-FGSM (Goodfellow, Shlens, and Szegedy 2015). For every pixel, the maximum perturbation levels are 0.031 and 0.03 for CIFAR-10 and CIFAR-100, respectively. The number of iterations for I-FGSM is 10. The other setup is the same as that of the Mixup paper.

The results are summarized in Table 2. For all types of attacks, Top-1 test accuracies of KCM+MIXUP were higher than those of MIXUP with margins ranging from 0.24% to 4.25% in CIFAR-10, and with margins ranging from 0.09% to 2.18% in CIFAR-100.

Discussion and Conclusions

While the Mixup method reduces unnecessary oscillations or promotes smoothness by implicit linear constraints via data augmentation, we propose a new smoothed network, KCM. To expand the domain of smoothness, we adapt KCM to the Mixup method. We provide upper bounds of excess risk for the KCM and its adapted version with the Mixup. The results offer insights on how the degree of heterogeneity of kernelizing models and the size of perturbation in the Mixup play a role in risk analysis. The experimental results demonstrate that the Mixup method helps to smooth the decision boundary and improves accuracy, but the proposed

Dataset	Attacks	Learning rule	FGSM	I-FGSM
CIFAR-10	White-box	MIXUP	75.63	51.82
		MIXUP + KCM	**78.24**	**56.07**
	Black-box	MIXUP	85.31	88.87
		MIXUP + KCM	**85.79**	**89.11**
CIFAR-100	White-box	MIXUP	38.79	22.99
		MIXUP + KCM	**39.65**	**25.17**
	Black-box	MIXUP	63.62	67.97
		MIXUP + KCM	**63.79**	**68.06**

Table 2: (Robustness) The test accuracies of each method based on Top-1 accuracy. For KCM, we choose the configuration pair (h, N) that shows the best performance in Table 1.

method with explicit model constraints attains smoothness more effectively. Experiments on CIFAR-10 and CIFAR-100 show that the proposed method can improve performance and increase robustness to adversarial examples with proper hyperparameters.

Acknowledgments

Minjin Kim, Young-geun Kim, and Myunghee Cho Paik were supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2020R1A2C1A01011950). We are grateful to all the reviewers for their thoughtful comments and suggestions.

References

Awasthi, P.; Frank, N.; and Mohri, M. 2020. On the Rademacher Complexity of Linear Hypothesis Sets. *arXiv preprint arXiv:2007.11045* .

Bartlett, P. L.; Jordan, M. I.; and McAuliffe, J. D. 2006. Convexity, classification, and risk bounds. *Journal of the American Statistical Association* 101(473): 138–156.

Bartlett, P. L.; and Mendelson, S. 2002. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research* 3(Nov): 463–482.

Beckham, C.; Honari, S.; Verma, V.; Lamb, A. M.; Ghadiri, F.; Hjelm, R. D.; Bengio, Y.; and Pal, C. 2019. On Adversarial Mixup Resynthesis. In *Advances in Neural Information Processing Systems*, 4348–4359.

Berthelot, D.; Carlini, N.; Goodfellow, I.; Papernot, N.; Oliver, A.; and Raffel, C. A. 2019. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, 5050–5060.

Cohen, T.; and Welling, M. 2016. Group equivariant convolutional networks. In *International conference on machine learning*, 2990–2999.

Cortes, C.; and Vapnik, V. 1995. Support-vector networks. *Machine learning* 20(3): 273–297.

Dao, T.; Gu, A.; Ratner, A. J.; Smith, V.; De Sa, C.; and Ré, C. 2019. A kernel theory of modern data augmentation. *Proceedings of machine learning research* 97: 1528.

DeVries, T.; and Taylor, G. W. 2017. Dataset augmentation in feature space. *arXiv preprint arXiv:1702.05538* .

Dieleman, S.; De Fauw, J.; and Kavukcuoglu, K. 2016. Exploiting cyclic symmetry in convolutional neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48*, 1889–1898. JMLR. org.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. *International Conference on Learning Representations* .

Graves, A.; Mohamed, A.-r.; and Hinton, G. 2013. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, 6645–6649. IEEE.

Guo, H.; Mao, Y.; and Zhang, R. 2019. Mixup as locally linear out-of-manifold regularization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3714–3722.

Hoerl, A. E.; and Kennard, R. W. 1970. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* 12(1): 55–67.

Hoffgen, K.-U.; Simon, H.-U.; and Vanhorn, K. S. 1995. Robust trainability of single neurons. *Journal of Computer and System Sciences* 50(1): 114–125.

Kim, Y.; Ohn, I.; and Kim, D. 2018. Fast convergence rates of deep neural networks for classification. *arXiv preprint arXiv:1812.03599* .

Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. Technical report, University of Toronto.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Li, X.; Lu, J.; Wang, Z.; Haupt, J.; and Zhao, T. 2018. On tighter generalization bound for deep neural networks: Cnns, resnets, and beyond. *arXiv preprint arXiv:1806.05159* .

Liang, D.; Yang, F.; Zhang, T.; and Yang, P. 2018. Understanding mixup training methods. *IEEE Access* 6: 58774–58783.

Marcos, D.; Volpi, M.; Komodakis, N.; and Tuia, D. 2017. Rotation equivariant vector field networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 5048–5057.

Mohri, M.; Rostamizadeh, A.; and Talwalkar, A. 2018. *Foundations of machine learning*. MIT press, second edition.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12: 2825–2830.

Rajput, S.; Feng, Z.; Charles, Z.; Loh, P.-L.; and Papailiopoulos, D. 2019. Does data augmentation lead to positive margin? *International Conference on Machine Learning* .

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature* 529(7587): 484.

Simard, P. Y.; LeCun, Y. A.; Denker, J. S.; and Victorri, B. 1998. Transformation invariance in pattern recognition—tangent distance and tangent propagation. In *Neural networks: tricks of the trade*, 239–274. Springer.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2014. Intriguing properties of neural networks. *International Conference on Learning Representations* .

Tai, K. S.; Bailis, P.; and Valiant, G. 2019. Equivariant Transformer Networks. In *International Conference on Machine Learning*.

Tokozume, Y.; Ushiku, Y.; and Harada, T. 2018. Between-class learning for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5486–5494.

van der Wilk, M.; Bauer, M.; John, S.; and Hensman, J. 2018. Learning invariances using the marginal likelihood. In *Advances in Neural Information Processing Systems*, 9938–9948.

Verma, V.; Lamb, A.; Beckham, C.; Najafi, A.; Mitliagkas, I.; Lopez-Paz, D.; and Bengio, Y. 2019a. Manifold Mixup: Better Representations by Interpolating Hidden States. In *Proceedings of the 36th International Conference on Machine Learning*, 6438–6447. PMLR.

Verma, V.; Lamb, A.; Kannala, J.; Bengio, Y.; and Lopez-Paz, D. 2019b. Interpolation Consistency Training for Semi-supervised Learning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 3635–3641. AAAI Press.

Worrall, D. E.; Garbin, S. J.; Turmukhambetov, D.; and Brostow, G. J. 2017. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5028–5037.

Wu, S.; Zhang, H. R.; Valiant, G.; and Ré, C. 2020. On the Generalization Effects of Linear Transformations in Data Augmentation. *International Conference on Machine Learning* .

Xie, Q.; Dai, Z.; Hovy, E.; Luong, M.-T.; and Le, Q. V. 2019. Unsupervised data augmentation. *arXiv preprint arXiv:1904.12848* .

Yin, D.; Kannan, R.; and Bartlett, P. 2019. Rademacher Complexity for Adversarially Robust Generalization. In *International Conference on Machine Learning*.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2018. mixup: Beyond Empirical Risk Minimization. *International Conference on Learning Representations* .