Predictive Power of Nearest Neighbors Algorithm under Random Perturbation

Yue Xing
Purdue University

Qifan Song
Purdue University

Guang Cheng
Purdue University

Abstract

This work investigates the predictive performance of the classical k Nearest Neighbors (k-NN) algorithm when the testing data are corrupted by random perturbation. The impact of corruption level on the asymptotic regret is carefully characterized and we reveal a phase-transition phenomenon that, when the corruption level of the random perturbation ω is below a critical order (i.e., small-ω regime), the asymptotic regret remains the same; when it is beyond that order (i.e., large-ω regime), the asymptotic regret deteriorates polynomially. More importantly, the regret of k-NN classifier heuristically matches the rate of minimax regret for randomly perturbed testing data, thus implies the strong robustness of k-NN against random perturbation on testing data. We show that the classical k-NN can achieve no worse predictive performance than the NN classifiers trained via the popular noise-injection strategy. Our numerical experiment also illustrates that combining k-NN component with modern learning algorithms will inherit the strong robustness of k-NN. As a technical by-product, we prove that under different model assumptions, the pre-processed 1-NN proposed in Xue and Kpotufe (2017) will achieve a sub-optimal rate when the data dimension $d > 4$ even if k is chosen optimally in the pre-processing step.

1 INTRODUCTION

While modern machine learning achieves a great deal of success via over-parametrized neural network, much of the success is in relatively restricted domains with limited structural variation or few system constraints. Those algorithms would be quite fragile in broader real-world scenarios, especially when the testing data are contaminated. For example, in image classification, when the input data are slightly altered due to a minor optical sensor system malfunction, a deep neural network may yield a different classification result (Goodfellow et al., 2014). Besides efforts on generating adversarial samples (Papernot et al., 2016a,b; Grosse et al., 2017), or ensuring the adversarial robustness of machine learning algorithms (Kurakin et al., 2016; Sinha et al., 2018; Madry et al., 2017), another strand of research focuses on theoretical investigation on how the data corruption affects the algorithm performance (Wang et al., 2017; Yang et al., 2019; Fawzi et al., 2016, 2018).

This work revisits the traditional k Nearest Neighbors (k-NN) algorithm and investigates the robustness of k-NN from a theoretical perspective. In the literature, beyond the nonparametric convergence analysis of k-NN and its variants (Samworth, 2012; Chaudhuri and Dasgupta, 2014; Xue and Kpotufe, 2017; Cannings et al., 2020; Sun et al., 2016; Mao et al., 2018; Duan et al., 2020; Balsubramani et al., 2019; LeJeune et al., 2019; Efremenko et al., 2020), recently there are abundant results on the robustness of k-NN. For example, Cannings et al. (2018); Reeve and Kaban (2019b,a) considered the case where labels for training data are contaminated, and studied the overall excess risk of the trained classifier; Wang et al. (2018); Yang et al. (2020b) considered the case where testing data are contaminated, and studied the *local* testing robustness, i.e., when testing data belong to a certain subset of support, rather than the whole support. In contrast to these existing works, the presented paper aims to address a different question: how the *overall* regret of k-NN classifier, which is trained by uncontaminated training data, is affected when the testing features are corrupted by random perturbation?

Our main theoretical result (derived in the framework of Samworth (2012)) characterizes the asymptotic re-

Proceedings of the 24$^{\text{th}}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s).

gret for randomly perturbed testing data (with an explicit form of multiplicative constant) of k-NN with respect to the choice of k and the level of testing data corruption. There are several interesting implications. First, there exists a critical contamination level, (a) below which the asymptotic order of regret is not affected; (b) above which the asymptotic order of regret deteriorates polynomially. Second, although the regret of k-NN deteriorates polynomially with respect to the corruption level, it achieves the best possible accuracy for testing randomly perturbed data (under a fine-tuned choice of k). Hence k-NN classifier is rate-minimax for both clean data testing task (Audibert and Tsybakov, 2007; Samworth, 2012; Cannings et al., 2020) and randomly perturbed data testing task.

A popular strategy to robustify the learning algorithm is to inject the same random noises into training data, such that the training and testing data are homogeneous. However, our theoretical analysis reveals that the vanilla k-NN achieves the same predictive performance (i.e., the same asymptotic regret) of the k-NN classifier trained via noise-injection method in the beginning stage of the polynomial deterioration regime.

The above regret analysis results imply that k-NN possesses native robustness against random perturbed adversarial samples. It can serve as a useful component for modern adversarial training algorithms and thus deserves more attention from the modern learning community. For instance, we evaluate the robustness of deep k-NN (Papernot and McDaniel, 2018) and show that deep k-NN has similar robustness of k-NN against random perturbation.

As a by-product, our developed theory may also be used to evaluate the asymptotic performance of variants of k-NN algorithms. For example, Xue and Kpotufe (2017) applied 1NN to pre-processed data (which is relabelled by k-NN) to achieve the same accuracy as k-NN. Interestingly, this algorithm can be translated into the classical k-NN algorithm under a type of perturbed samples to which our theory naturally applies. In particular, we prove that the above algorithm, under our model assumption framework, only obtains a sub-optimal rate (worse than k-NN) of regret when $d > 4$.

2 EFFECT OF RANDOM PERTURBATION

In this section, we will introduce the model setup and present our main theorems which characterize the asymptotic regret for perturbed testing samples.

2.1 Model Setup

Denote $P(Y=1|X=x)$ as $\eta(x)$, and its k-NN estimator as $\widehat{\eta}_{k,n}(x)$, an average of k nearest neighbors among n training samples, i.e.

$$\widehat{\eta}_{k,n}(x)=\frac{1}{k}\sum_{i\in N_{n,k}(x)} y_i,$$

where $\{(x_i,y_i)\}_{i=1,\dots,n}$ are i.i.d. samples and $N_{n,k}(x)$ represents the index set of the k nearest neighbors of x in the n samples. The corresponding Bayes classifier and k-NN classifier is defined as $g(x)=1\{\eta(x)>1/2\}$ and $\widehat{g}_{n,k}(x)=1\{\widehat{\eta}_{k,n}(x)>1/2\}$, respectively.

Define ω as the level of perturbation. For any intended testing data x, we only observe its randomly perturbed version: $\widetilde{x}\sim \text{Unif}(\partial B(x,\omega))$, that is, $\widetilde{x}$ is uniformly distributed over $\partial B(x,\omega)$, the boundary of an Euclidean ball $B(x,\omega)$.

In this case, we define the "perturbed" regret as

$$\text{Regret}(k,n,\omega)=P(Y\neq \widehat{g}_{n,k}(\widetilde{X}))-P(Y\neq g(X)),$$

and $\text{Regret}(n,\omega)=\min_{k=1,\dots,n}\text{Regret}(k,n,\omega)$. Note that the k-NN classifier $\widehat{g}_{n,k}$ is trained by uncontaminated training samples. When $\omega=0$, the above definition reduces to the traditional regret that is used in statistical classification literature.

Regret analysis is common to evaluate the classification performance as in Chaudhuri and Dasgupta (2014); Samworth (2012); Sun et al. (2016); Belkin et al. (2018). If $\eta(x)\neq 0$ or 1, the mis-classification rate $P(\widehat{g}(X)\neq Y|X=x)$ is always bounded away from zero for any estimator $\widehat{g}$. Therefore, instead of analyzing mis-classification rate, people use regret to evaluate the performance gap between any estimator $\widehat{g}$ and the optimal Bayes classifier.

The following assumptions are imposed on X and the underlying η, to facilitate our theoretical analysis.

A.1 X is a random variable on a compact d-dimensional manifold $\mathcal{X}$ with boundary $\partial\mathcal{X}$. Density function of X is twice-continuously differentiable, finite and bounded away from 0.

A.2 The set $\mathcal{S}=\{x|\eta(x)=1/2\}$ is non-empty. There exists an open subset U_0 in $\mathbb{R}^d$ which contains $\mathcal{S}$ such that, for an open set containing $\mathcal{X}$ (defined as U), η is continuous on $U\backslash U_0$.

A.3 There exists some constant $c_x>0$ such that when $|\eta(x)-1/2|\leq c_x$, η has bounded fourth-order derivative; when $\eta(x)=1/2$, $\dot{\eta}(x)\neq 0$, where $\dot{\eta}$ is the gradient of η in x. Also the derivative of $\eta(x)$ within restriction on the boundary of support is non-zero.

Assumptions A.1 ensures that for any $x \in \mathcal{X}$, all its k nearest neighbors are close to x with high probability. This is due to the fact that if the density at a point x is positive and finite, its distance to its kth nearest will be of $O_p((k/n)^{1/d}) = o_p(1)$. Assumption A.2 ensures the existence of x in $\{x \in \mathcal{X} | \ \eta(x) = 1/2\}$ and $\eta(x)$ is continuous in other regions of $\mathcal{X}$. Assumption A.3 on $\eta(x)$ is slightly stronger than that imposed in Samworth (2012) due to the consideration of testing data contamination. Specifically, the additional smoothness on $\eta(x)$ imposed in Assumption A.3 guarantees that some higher-order terms in the Taylor expansion of $\mathbb{E}\{\widehat{\eta}_{k,n}(\widetilde{x}) - \eta(x)\}$ are negligible.

2.2 Asymptotic Regret and Phase Transition Phenomenon

We are now ready to conduct regret analysis for k-NN in the presence of randomly perturbed testing samples. For any $x \in \mathcal{X}$, define $t_{k,n}(x)$ as

$$\mathbb{E}\big(\|X_i - x\|_2^2 \,\big|\, X_i \text{ is in the } k \text{ nearest neighbors of } x\big).$$

Therefore, $t(x)$ represents the expected squared distance from x to any of its k nearest neighbors. Let's define $t = \max_x t(x)$, and denote $\bar{f}(x, y)$ and $\bar{f}(x)$ as the joint density of (x, y) and marginal density of x respectively. Let $f_1(x) := \bar{f}(x, 0)$, $f_2(x) := \bar{f}(x, 1)$, and $\Psi(x) := f_1(x) - f_2(x)$.

We first characterize the asymptotic perturbed regret.

Theorem 1. *Define $\epsilon_{k,n,\omega} = \max(\log k/\sqrt{k}, t_{k,n} + \omega)$. Under [A.1] to [A.3] in Appendix A, if testing data is randomly perturbed, then it follows that*

$$\begin{aligned} &\textit{Regret}(k,n,\omega)\\ =&\underbrace{\frac{1}{2}\int_{\mathcal{S}} \frac{\|\dot{\Psi}(x_0)\|}{\|\dot{\eta}(x_0)\|^2} \left(b(x_0)t_{k,n}(x_0)\right)^2 d\,Vol^{d-1}(x_0)}_{Bias}\\ &+\underbrace{\frac{1}{2}\int_{\mathcal{S}} \frac{\omega^2}{d}\|\dot{\Psi}(x_0)\| d\,Vol^{d-1}(x_0)}_{Corruption} \qquad (1)\\ &+\underbrace{\frac{1}{2}\int_{\mathcal{S}} \frac{1}{4k}\frac{\|\dot{\Psi}(x_0)\|}{\|\dot{\eta}(x_0)\|^2} d\,Vol^{d-1}(x_0)}_{Variance} + \textit{Remainder}, \end{aligned}$$

where Remainder=$O(\epsilon_{k,n,\omega}^3)$ as $k, n \to \infty$. The term $b(\cdot)$ relies on the true $\eta(x)$ and the distribution of X, and does not change with respect to k and n:

$$b(x) \;=\; \frac{1}{\bar{f}(x)d}\left\{\sum_{j=1}^{d}[\dot{\eta}_j(x)\dot{\bar{f}}_j(x) + \ddot{\eta}_{j,j}(x)\bar{f}(x)/2]\right\}.$$

Here $\dot{\eta}$, $\ddot{\eta}$, and $\dot{\bar{f}}$ represent the gradient, Hessian of η, and the gradient of $\bar{f}$ respectively. The subscript j denotes the j'th element of $\dot{\eta}$ or $\dot{\bar{f}}$, and the subscript j,j denotes the (j,j)'th element of $\ddot{\eta}$.

Our result (1) decomposes the asymptotic regret into squared bias term, data corruption effect term, variance term as well as a remainder term. The first three terms are of order $O((k/n)^{4/d})$, $O(\omega^2)$ and $O(1/k)$ respectively, and the remainder term is technically derived from high order Taylor expansion and Berry-Essen theorem. When k is within a reasonable range, the remainder term is negligible compared with the rest three main terms. When $\omega = 0$, (1) reduces to the bias-variance decomposition observed in the nonparametric regression literature (e.g., Kandasamy and Yu, 2016).

Based on Theorem 1, through changing ω, we have the following observations:

Phase Transition Phenomenon Theorem 1 reveals a phase transition phenomenon for the regret w.r.t. the level of testing data contamination.

1. When $\omega^2 \preccurlyeq (1/k \wedge t_{k,n}^2)$ [1], the asymptotic regret is barely affected by the testing data corruption: $\text{Regret}(k,n,\omega)/\text{Regret}(k,n,0) \to 1$;
2. When $\omega^2 = \Theta(1/k \vee t_{k,n}^2)$, the regret is of the same order as $\text{Regret}(k,n,0)$ but with a different multiplicative constant depending on $\bar{f}$ and η;
3. When $\omega^2 \succcurlyeq (1/k \vee t_{k,n}^2)$, $\text{Regret}(k,n,\omega) = \Theta(\omega^2)$ and $\text{Regret}(k,n,\omega) \succcurlyeq \text{Regret}(k,n,0)$.

Impact on Regret(n,ω) and the choice of k The value k plays an important role in the k-NN algorithm. It is essential to understand how the intensity level ω affects the optimal value of k and the corresponding optimal $\text{Regret}(n,\omega)$. Theorem 1 implies that, if $\omega \preccurlyeq n^{-2/(d+4)}$, $\text{Regret}(n,\omega) = \Theta(n^{-4/(d+4)})$; if $\omega \succcurlyeq n^{-2/(d+4)}$, $\text{Regret}(n,\omega) = \Theta(\omega^2)$. In other words, $\text{Regret}(n,\omega) = \Theta(\omega^2 \vee n^{-4/(d+4)})$. The above rate can be achieved if we choose $k = \Theta(n^{4/(4+d)})$ when $\omega \preccurlyeq n^{-4/3(4+d)}$ and $1/\omega^2 \preccurlyeq k \preccurlyeq n\omega^{d/2}$ when $\omega \succcurlyeq n^{-4/3(4+d)}$.

Distribution of $\widetilde{x}$ In Theorem 1, we assume $\widetilde{x}$ randomly distributed on a $\mathcal{L}_2$ sphere uniformly. As will be shown in the sketch of proof, we only utilize the distributional information of $\widetilde{x}$ at the last step of derivation. An illustration and example of relaxing the distributional condition of $\widetilde{x}$ is postponed to the Appendix D.

[1] To prevent the conflict of definitions of ω, we use $\preccurlyeq$ and $\succcurlyeq$ to replace $o(.)$ and $\omega(.)$ in O/Ω notation. Moreover, for $a(n) \preccurlyeq b(n) \preccurlyeq 1$, we mean that $b(n)/1 < n^{-\varepsilon_1}$ and $a(n)/b(n) < n^{-\varepsilon_2}$ for some $\varepsilon_1, \varepsilon_2 > 0$ when $n \to \infty$.

Effect of Metric of Noise Note that $\widetilde{x}$ can be defined on $\mathcal{L}_p$ ball / sphere for different $p \geq 1$. As showed by Theorem 1, the effect of ω (i.e., the corruption term in (1)) is irrelevant to t and $1/k$. As a result, Theorem 1 generalizes to $\mathcal{L}_p$ perturbation by replacing ω^2/d in (1) with $\mathbb{E}(\varepsilon_p^\top \dot{\eta}(x_0))^2/\|\dot{\eta}(x_0)\|^2$, where $\varepsilon_p = \tilde{x} - x$ is the random variable uniformly distributed in a $\mathcal{L}_p$ ball/ sphere.

Minimax Rate To assess the rate of perturbed regret of k-NN, we conduct the following minimax study to obtain the best worst-case performance among all possible estimators.

Theorem 2 (Informal Statement for Minimax Rate). *If the distribution of (X, Y) satisfies*

1. *η is α-Holder smoothness for all x;*
2. *$P(|\eta(X) - 1/2| < t) \leq Bt^\beta$ for some $\beta > 0$,*

together with some other general assumptions, the minimax rate of perturbed Regret is

$$\Theta\left(\omega^{\alpha(\beta+1)} \vee n^{-\alpha(\beta+1)/(2\alpha+d)}\right).$$

Formal assumptions and results for Theorem 2 are postponed in Section F in Appendix. From Theorem 2, the rate of regret is dominated by the larger one between the random perturbation effect ($\omega^{\alpha(\beta+1)}$) and the minimax rate for clean data ($n^{-\alpha(\beta+1)/(2\alpha+d)}$). Similar as in Samworth (2012), our regret result matches the minimax rate of Theorem 2 by taking $\alpha = 2$ and $\beta = 1$.

Remark 1 (Adversarial Data Corruption). *So far, we focus on the case of random perturbation. As a by-product, we analyze the effect of some special non-random data corruption. The detailed results and discussions are postponed to Section C in the appendix due to the page limit. In general, k-NN is more robust to random perturbed data corruption than adversarial (defined formally in the appendix) corruption. However, our rigorous analysis shows that the regret under adversarial data corruption is of the same order as in the case of random perturbation but with a larger multiplicative constant when $\omega \succcurlyeq n^{-2/(d+4)}$. Some literature studied improving the adversarial robustness of NN-type algorithms, e.g., Wang et al. (2018); Yang et al. (2020b,a). However, there is no improvement in the convergence rate of Regret.*

2.3 Comparison with Noise-Injected k-NN

Iterative adversarial training algorithms (e.g., Sinha et al., 2018) usually consist of (1) attacking the training data based on the current model and (2) updating the model parameter based on attacked training data. A similar idea to enhance the robustness of k-NN is to inject random perturbation noise into the training data so that training and testing data share the same distribution, i.e., we train k-NN classifier using the randomly perturbed training data set. Comparing the traditional k-NN methods with this noise-injection k-NN, we find no performance lost for the former even when the corruption level is in the early stage of the polynomial deterioration regime.

Denote $\widetilde{g}(\widetilde{x}) := P(Y = 1|\widetilde{x} \text{ is observed})$ as the Bayes estimator and $\widehat{g}_n'$ as the estimator trained using randomly perturbed training data. Let both estimators $\widehat{g}_n$ and $\widehat{g}_n'$ adopt their best choices of k respectively. Then we have

Theorem 3. *Under the same conditions as Theorem 1, when $0 < \omega^3 \preccurlyeq n^{-4/(d+4)}$,*

$$\frac{P(Y \neq \widehat{g}_n(\widetilde{x})) - P(Y \neq \widetilde{g}(\widetilde{x}))}{P(Y \neq \widehat{g}_n'(\widetilde{x})) - P(Y \neq \widetilde{g}(\widetilde{x}))} \to 1. \tag{2}$$

Although it is intuitive to consider perturbing training data such that they match the distribution of the corrupted testing data, result (2) implies that the estimators $\widehat{g}_n$ and $\widehat{g}_n'$ asymptotically share the same predictive performance for randomly perturbed testing data, and the native robustness of k-NN is as strong as if it were adversarially trained. Note that this result holds when ω is small. Combined with our result in Theorem 1, within the range $n^{-2/(d+4)} \preccurlyeq \omega \preccurlyeq n^{4/3(d+4)}$, the regret deteriorates polynomially due to the testing data corruption and can not be improved by noise-injection adversarial training at all. One heuristic explanation is that such an injected perturbation may introduce additional noise to the estimation procedure and change some underlying properties (e.g., smoothness), and consequently, this strategy of perturbing training data does not necessarily help to achieve smaller regret, especially when ω is small.

2.4 Implications to other machine learning algorithms

Section 2.2 and 2.3 imply that k-NN is a robust algorithm against random perturbation in testing data, and injecting noise in training data does not improve the robustness when the corruption level is small. This strength of k-NN reveals the potential of a better robustness when combining k-NN with other modern fancy learning algorithms (e.g., Papernot and McDaniel, 2018; Plötz and Roth, 2018; Bahri et al., 2020).

An instance of a combination of k-NN and other algorithms is deep k-NN (Papernot and McDaniel, 2018). To implement a deep k-NN, one can first train a deep neural network, then use the output of its layers as features to determine the distance of the training samples to the testing sample. In Papernot and McDaniel

Algorithm 1 Data Pre-processing

```
Input: data $(x_1, y_1)$,..., $(x_n, y_n)$, number of neighbors $k$.
for $i = 1$ to $n$ do
    Find the $k$ nearest neighbors of $x_i$ in $x_1$,...,$x_n$, excluding $x_i$ itself. Denote the index set of these $k$ neighbors as $N_i$.
    Estimate a label for $x_i$ as
        $\widehat{\eta}(x_i) = \frac{1}{k} \sum_{j \in N_i} y_j,$
        $\widehat{y}_i = 1_{\{\widehat{\eta}(x_i) > 1/2\}}.$
end for
Output: $(x_1, \widehat{y}_1)$,...,$(x_n, \widehat{y}_n)$.
```

(2018), deep k-NN aims to improve the confidence and robustness of deep neural networks, which coincides with our idea that k-NN is robust. Our numerical experiments will show that deep k-NN inherits the robustness of k-NN. More specifically, our simulation shows that injecting noise does not improve predictive performance, which indicates that deep k-NN is already robust enough to resist small random perturbations.

3 APPLICATION TO VARIANTS OF NN ALGORITHM

Our theoretical analysis can be adapted to other NN-type algorithms: pre-processed 1NN (Xue and Kpotufe, 2017) and distributed-NN (Qiao et al., 2019). We prove that the regret of the former is sub-optimal for some class of distributions and explain why the regret of the latter converges in the optimal rate, both in the aspect of the random perturbation viewpoint.

3.1 Pre-processed 1NN

In some literature (Xue and Kpotufe, 2017; Wang et al., 2017), the algorithms run 1NN to make prediction using pre-processed data instead of running k-NN using raw data. The pre-processing step (or called de-noising step) is reviewed in Algorithm 1. Specifically, we firstly run k-NN to predict labels for the training data set, then replace the original labels with the predict labels $\widehat{y}_i$'s. In this way, applying 1NN on data $(x_1, \widehat{y}_1)$,...,$(x_n, \widehat{y}_n)$ can achieve good accuracy while the computational cost is smaller than k-NN.

This in fact can be treated as an application of random perturbation of testing data in k-NN, in the sense that this classifier can be equivalently represented as k-NN under corrupted testing sample:

$$\widehat{g}_{\mathrm{1NN}}(x) = \widehat{g}_{n,k}(\widetilde{x}),$$

where $\widehat{g}_{\mathrm{1NN}}$ is the pre-processed 1NN classifier, and $\widetilde{x}$ is the corrupted observation of x, which is the nearest neighbor of x. Although $\widetilde{x}$ is not exactly induced by random perturbation, it can be viewed as randomly perturbed x with level of contamination $\omega = \Theta(n^{-1/d})$, which is the order for the expected length from x to its nearest neighbor.

From this point of view, Theorem 1 can be applied to derive the regret of the pre-processed 1NN algorithm, whose rate of convergence turns out to be slower than the optimal rate $\Theta(n^{-4/(d+4)})$ of k-NN when the data dimension d is relatively high, say $d > 4$.

Theorem 4. *Under the same conditions as Theorem 1, the regret of pre-processed 1NN under un-corrupted testing data is*

$$\begin{aligned} \mathit{Regret}_{\mathit{1NN}}(k,n) = & \frac{1}{2}\int_{\mathcal{S}} \frac{\|\dot{\Psi}(x_0)\|}{\|\dot{\eta}(x_0)\|^2}\left(b(x_0)t(x_0)\right)^2 d\,\mathit{Vol}^{d-1}(x_0) \\ & + \frac{1}{2}\int_{\mathcal{S}} \frac{1}{4k}\frac{\|\dot{\Psi}(x_0)\|}{\|\dot{\eta}(x_0)\|^2} d\,\mathit{Vol}^{d-1}(x_0) \\ & + \mathit{Corruption} + \mathit{Remainder}, \end{aligned}$$

where

$$\mathit{Corruption} = \Theta(n^{-2/d}),\ \mathit{Remainder} = o(n^{-2/d})$$

when both $1/k$ and $(k/n)^{4/d}$ are of $O(n^{-1/d})$, and $k = O(n^{6/d})$. As a result, pre-processed 1NN is sub-optimal when $d > 4$ (compared with optimal rate $n^{-4/(d+4)}$).

The result in Theorem 4 reveals a sub-optimal rate for the pre-processed 1NN under our Assumption A.1-A.3 (in Appendix A), in contrast to the optimal rate claimed by Xue and Kpotufe (2017) under different assumptions.

3.2 Distributed-NN

The computational complexity of k-NN is huge if n is large, therefore we consider a distributed NN algorithm: we randomly partition the original data into s equal-size parts, then given x, for each machine, the k/s nearest neighbors of x are selected and calculate $\widehat{\eta}_j(x)$ for $j = 1, ..., s$, finally we average $\widehat{\eta}_1(x)$,...,$\widehat{\eta}_s(x)$ to obtain $\widehat{\eta}(x)$. The algorithm is shown in Algorithm 2 as in Qiao et al. (2019).

Distributed-NN is practically different from k-NN in a single machine since the k selected neighbors aggregated from s subsets of data are not necessarily the same k nearest neighbors selected in a single machine. Therefore, an additional assumption $k/s \to \infty$ is imposed to ensure that the neighborhood set selected by

Algorithm 2 Distributed-NN

```
  **Input:** data $(x_1, y_1)$,..., $(x_n, y_n)$, number of neighbors $k$, number of slaves $s$, a point $x$ for prediction.
  Randomly divide the whole data set into $s$ parts, with index sets $S_1$,...,$S_s$.
  **for** $i = 1$ **to** $s$ **do**
    Find the $k/s$ nearest neighbors of $x$ in $\{x_j \mid j \in S_i\}$. Denote the index set of these $k/s$ neighbors as $N_i$.
    Estimate $\widehat{\eta}_i(x) = \frac{1}{k/s} \sum_{j \in N_i} y_j$.
  **end for**
  Estimate the label of $x$ as
    $\widehat{\eta}(x) = \frac{1}{s} \sum_{i=1}^{s} \widehat{\eta}_i(x)$,
    $\widehat{y} = 1_{\{\widehat{\eta}(x) > 1/2\}}$.
  **Output:** $(x, \widehat{y})$.
```

distributed NN behaves similarly to the neighborhood set selected by single machine k-NN, in the sense that $E\|X_i - x\|^2$, where X_i belongs to the distributed NN neighborhood set, is of the same order of $t(x)$. Therefore, based on Theorem 1, we obtain that the order of regret of Distributed-NN is, in fact, of the same order as k-NN. Formally, we have the following corollary:

Corollary 5. *Under the same conditions as Theorem 1, when the multiplicative constants in (1) are not zero, if the number of machines $s \preccurlyeq k$, then*

$$Regret_{\mathrm{DNN}}(k, n) = \Theta(Regret_{\mathrm{kNN}}(k, n)).$$

where $Regret_{\mathrm{DNN}}$ and $Regret_{\mathrm{kNN}}$ denote the (clean testing data) regret of distributed NN and k-NN algorithms.

4 NUMERICAL EXPERIMENTS

In Section 4.1, we evaluate the empirical performance of k-NN algorithm for randomly perturbed testing data, where we compare the k-NN classifiers trained by raw un-corrupted training data and trained by noise injected training data (i.e., $\widehat{g}_n$ versus $\widehat{g}'_n$ defined in Section 2.3). In Section 4.2, we use deep k-NN as an example to explore how k-NN helps to improve the robustness of modern learning algorithms. In Section 4.3, we conduct experiments to compare k-NN with pre-processed 1NN for un-corrupted testing data. These numerical experiments are intended to show: (i) k-NN has a similar testing performance as if trained by noise injected training data when ω is small, which validates our Theorem 3; (ii) deep k-NN inherits the strong robustness of k-NN; and (iii) for un-corrupted testing data, the regret of pre-processed 1NN is worse than that of k-NN if $d > 4$, which validates Theorem 4.

For all figures in numerical experiments, we provide the detailed mean and standard deviation information in Appendix B.

4.1 Tackling Random Perturbation

4.1.1 Simulation

The random variable X is of 5 dimension, and each dimension independently follows exponential distribution with mean 0.5. The conditional mean of Y is defined as

$$\eta(x) \quad = \quad \frac{e^{x^\top w}}{e^{x^\top w} + e^{-x^\top w}} \qquad (3)$$

where $w_i = i - d/2$ for $i = 1, ..., d$. For each pair of (k, n), we use $2^6, ..., 2^{11}$ training samples, 10000 testing samples and repeated 50 times to calculate the average regret. In each repetition, 5-fold cross validation was used to obtain $\widetilde{k}$. Then based on Samworth (2012), we adjust the number of neighbors to $\widehat{k} = \widetilde{k}(5/4)^{4/(4+d)}$ since $\widetilde{k}$ is the best k value for $4n/5$ samples instead of n samples. The two classifiers, trained via un-corrupted training data and corrupted training data respectively, used to predict corrupted testing data. From Figure 1, as the number of training samples increases, the regret for both k-NNs gets reduced for $0 < \omega \leq 0.05$ in the same speed. This verifies that these two k-NNs do not differ a lot when ω is small, i.e., Theorem 3. Empirically, the regret of k-NN trained by corrupted training data is worse than the one trained by un-corrupted training data when $\omega \leq 0.5$. On the other hand, when ω is large (such that required condition in Theorem 3 fails), the two k-NNs may perform significantly differently. For example, we tried $\omega = 3$, when sample size $n = 64$, $\log_2$(Regret) is -2.86 using uncontaminated data, and is -3.11 using corrupted training data.

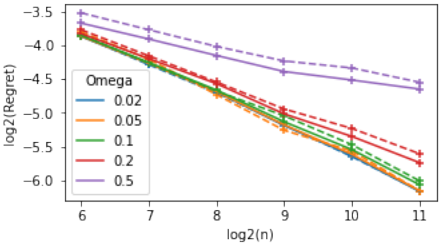

Figure 1: Comparison between k-NN trained by raw training data (solid line) and k-NN trained by noise injected training data (dashed line) in Simulation. When $\omega \leq 0.05$, regret converges in the same speed.

In addition, we compare how the dimension d affects the regret under different corruption level ω. We tried $d = 5, 10, 15, 20, 20, 50, 100$ and $n = 128$. The distribution of x is the same as the previous experiment, and the true model only relates to the first five attributes and follows (3). In this case, the multiplicative constants for bias, variance, and corruption as in Theorem 1 are unchanged among d. The results are summarized in Figure 6. When $\omega = 0$, the regret for $d = 5$ is the smallest. However, when ω increases, the regret for $d = 5$ increases much faster than others. Note that in Figure 2, considering it is unfair to use the same level of corruption to different d's, the x-axis represents $\omega/\sqrt{d}$ instead of ω. For $d = 100$, its regret is the least sensitive to ω. To explain this, as mentioned in Section 2.2, ω has little effect on regret when $\omega \ll n^{-2/(d+4)}$, and $n^{-2/(d+4)}$ in an increasing function in d.

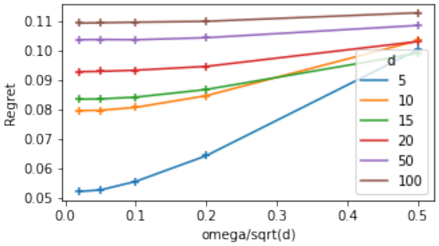

Figure 2: Comparison among regrets using different d. A smaller d indicates a lower regret when $\omega = 0$. When ω gets larger, regret with small d increases rapidly, and can even exceed those associated with large d.

4.1.2 Real Data

We use two real data sets for the comparison of 2 k-NNs: Abalone (Dua and Graff, 2017), HTRU2 (Lyon et al., 2016). For the Abalone data set, the data set contains 4177 samples, and all attributes except for gender are used in this experiment. The classification label is whether an abalone is older than 10.5 years. For HTRU2 data set (Lyon et al., 2016), the data has a size of 17,898 with 8 continuous attributes. For each data set, 25% of the samples are contaminated by random noise and are used as testing data.

As shown in Figure 3, when ω is small, for both data sets, the error rate (misclassification rate) of the two k-NNs do not differ a lot when ω is small. The value of ω over maximum pairwise distance, when $\omega = 3$, is 3/20 for HTRU2 and 1/9 for Abalone.

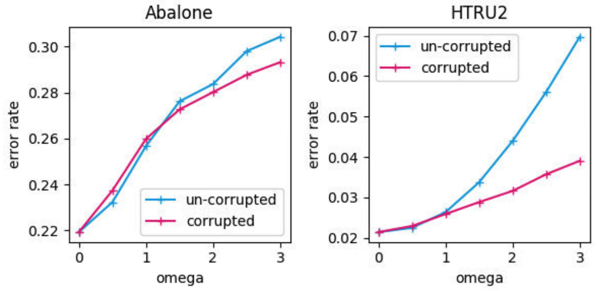

Figure 3: k-NN trained by raw Training Data vs noise injected Training Data. When ω is small, there is no great difference between k-NN using un-corrupted training data or corrupted training data.

4.2 Robustness of Deep k-NN

Inspired by our observation that k-NN is robust to random perturbations and injecting noise to training samples is futile when ω is small, we evaluate the robustness of deep k-NN against random perturbations.

In this experiment, we use (i) a network with two convolution layers and one fully connected layer to predict labels for MNIST, and (ii) a network with four convolution layers and two fully connected layers to predict labels for CIFAR-10 Krizhevsky et al. (2009), using the implementation from Papernot and McDaniel (2018). For each choice of ω, we train a deep k-NN using uncorrupted training (deep k-NN clean) and train another one using randomly perturbed training data with corruption level ω (deep k-NN perturbed) to compare their error rate under randomly perturbed testing data (with level ω), and repeat 10 times to obtain the mean and standard deviation. Some details of the CNN parameters, training configurations, and choice of k are postponed to Appendix B. A short summary can be found in Table 1 and 2. Similar to the results for k-NN, deep k-NN is robust to random perturbation, and there is no need to inject noise in training data when ω is small.

ω	Deep k-NN clean	Deep k-NN perturbed
0.4	0.013(0.007)	0.019(0.013)
0.8	0.018(0.009)	0.012(0.011)
1	0.021(0.01)	0.02(0.012)
1.5	0.039(0.042)	0.042(0.026)
2	0.08(0.062)	0.06(0.016)

Table 1: Mean and Standard Deviation of Error Rate of Perturbed Testing Data (MNIST). The error rates have only slight changes when increasing ω. Furthermore, there is little difference between training using clean data or training using perturbed data.

ω	Deep k-NN clean	Deep k-NN perturbed
0.1	0.44(0.036)	0.408(0.033)
0.2	0.43(0.054)	0.448(0.033)
0.3	0.434(0.035)	0.451(0.062)
0.4	0.461(0.03)	0.483(0.058)

Table 2: Mean and Standard Deviation of Error Rate of Perturbed Testing Data (CIFAR-10).

4.3 1NN with Pre-processed Data

4.3.1 Simulation

To observe a clear difference, instead of w_i in (3), we use a model where each dimension of x follows uniform $[0,1]$ distribution, with η in (3), and $w_i = i - d/2 - 0.5$ for $i = 1, ..., d$. for different values of d to compare the performance between k-NN and pre-processed 1NN. X now follows d-dimensional uniform $(0,1)$. From Figure 4, we show that the order of regret of pre-processed 1NN is different from that of k-NN when $d \geq 4$.

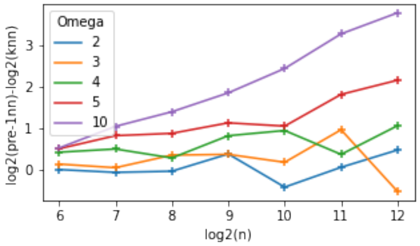

Figure 4: Simulation Comparison between k-NN and pre-processed 1NN, the y axis denotes the $\log_2$(Regre of pre-processed 1NN) $-$ $\log_2$(Regre of kNN). When $d \geq 4$, the regret for pre-processed 1NN is much larger than that of k-NN.

4.3.2 Real Data

We use four data sets: MNIST, Abalone, HTRU2, and Credit Yeh and Lien (2009). For MNIST, this data set contains 70000 samples, and the data dimension is 784. We randomly pick 25% samples as testing data and randomly pick 2^i $(i = 7, 8, ..., 12)$ samples to train k-NN classifier and pre-processed 1NN classifier, where the choices of k for both algorithms are determined by 5-folds cross-validation. We repeated this procedure 50 times to obtain the mean testing error.

As shown in Figure 5, through increasing the number of training samples, the error rate ratio between pre-process 1NN classifier and k-NN classifier is stably above 1 and is around 1.17.

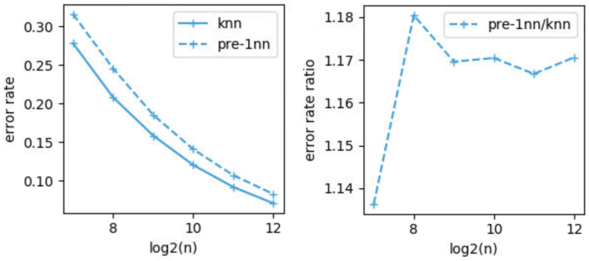

Figure 5: MNIST, k-NN vs pre-processed 1NN. Pre-processed 1NN always has larger error rate than k-NN. The error rate ratio can be regarded as regret ratio for MNIST, and the ratio is always larger than 1.12.

For Abalone data set, we conducted experiment in the same way as MNIST, and observe that the error rate using pre-processed 1NN is always greater than k-NN. As is shown in Figure 6, while the error rate for both pre-processed 1NN and k-NN are decreasing in n, their difference changes little when $n \leq 2^{11}$.

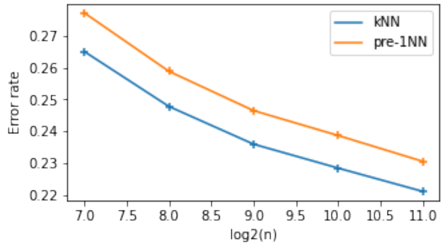

Figure 6: Comparison between kNN and Pre-processed 1NN in Abalone Data Set. The error rate of pre-processed 1NN is always greater than k-NN.

The results for Credit and HTRU2 are in appendix.

5 CONCLUSION AND DISCUSSION

In this work, we conduct asymptotic regret analysis of k-NN classification for randomly perturbed testing data. In particular, a phase transition phenomenon is observed: when the corruption level is below a threshold order, it does not affect the asymptotic regret; when the corruption level is beyond this threshold order, the asymptotic regret grows polynomially. Moreover, k-NN is robust enough, such that when the level of corruption is small, there is no need to perform noise injected training approach. Besides verifying the robustness of k-NN itself, our result implies a potential of combining k-NN with other machine learning algorithms so as to improve their robustness.

Moreover, using the idea of random perturbation, we can further explain why pre-processed 1NN converges in a sub-optimal rate: it can be treated as k-NN with perturbation in testing data while ω, the distance from x to its nearest neighbor, is large when $d > 4$. Our analysis can also be applied to Distributed-NN to verify the optimal rate obtained in Qiao et al. (2019) as well.

An interesting observation from the numerical experiment is that using traditional k-NN leads to an even better performance than the k-NN trained via noise injection method. This observation contradicts to common belief that injecting an attack into a training algorithm to obtain an adversarially robust algorithm (e.g., optimization method in Sinha et al. (2018)). Therefore, it deserves further theoretical investigation to understand how one can indeed benefit from the noise injection strategy.

Acknowledgements

Dr. Songs research activities are partially supported by National Science Foundation DMS-1811812.

References

Audibert, J.-Y. and Tsybakov, A. B. (2007), "Fast learning rates for plug-in classifiers," *The Annals of statistics*, 35, 608–633.

Bahri, D., Jiang, H., and Gupta, M. (2020), "Deep k-NN for Noisy Labels," *arXiv preprint arXiv:2004.12289*.

Balsubramani, A., Dasgupta, S., Moran, S., et al. (2019), "An adaptive nearest neighbor rule for classification," in *Advances in Neural Information Processing Systems*, pp. 7579–7588.

Belkin, M., Hsu, D., and Mitra, P. (2018), "Overfitting or perfect fitting? Risk bounds for classification and regression rules that interpolate," *arXiv preprint arXiv:1806.05161*.

Cannings, T. I., Berrett, T. B., Samworth, R. J., et al. (2020), "Local nearest neighbour classification with applications to semi-supervised learning," *Annals of Statistics*, 48, 1789–1814.

Cannings, T. I., Fan, Y., and Samworth, R. J. (2018), "Classification with imperfect training labels," *arXiv preprint arXiv:1805.11505*.

Chaudhuri, K. and Dasgupta, S. (2014), "Rates of convergence for nearest neighbor classification," in *Advances in Neural Information Processing Systems*, pp. 3437–3445.

Dua, D. and Graff, C. (2017), "UCI Machine learning repository," .

Duan, J., Qiao, X., and Cheng, G. (2020), "Statistical Guarantees of Distributed Nearest Neighbor Classification," in *Advances in Neural Information Processing Systems*.

Efremenko, K., Kontorovich, A., and Noivirt, M. (2020), "Fast and Bayes-consistent nearest neighbors," in *International Conference on Artificial Intelligence and Statistics*, pp. 1276–1286.

Fawzi, A., Fawzi, H., and Fawzi, O. (2018), "Adversarial vulnerability for any classifier," in *Advances in Neural Information Processing Systems*, pp. 1178–1187.

Fawzi, A., Moosavi-Dezfooli, S.-M., and Frossard, P. (2016), "Robustness of classifiers: from adversarial to random noise," in *Advances in Neural Information Processing Systems*, pp. 1632–1640.

Goodfellow, I., Shlens, J., and Szegedy, C. (2014), "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*.

Grosse, K., Papernot, N., Manoharan, P., Backes, M., and McDaniel, P. (2017), "Adversarial examples for malware detection," in *European Symposium on Research in Computer Security*, Springer, pp. 62–79.

Kandasamy, K. and Yu, Y. (2016), "Additive approximations in high dimensional nonparametric regression via the SALSA," in *International conference on machine learning*, pp. 69–78.

Krizhevsky, A., Hinton, G., et al. (2009), "Learning multiple layers of features from tiny images," .

Kurakin, A., Goodfellow, I., and Bengio, S. (2016), "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*.

LeJeune, D., Baraniuk, R. G., and Heckel, R. (2019), "Adaptive estimation for approximate k-nearest-neighbor computations," in *International Conference on Artificial Intelligence and Statistics*, pp. 3099–3107.

Lyon, R. J., Stappers, B., Cooper, S., Brooke, J., and Knowles, J. (2016), "Fifty years of pulsar candidate selection: from simple filters to a new principled real-time classification approach," *Monthly Notices of the Royal Astronomical Society*, 459, 1104–1123.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017), "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*.

Mao, C., Hu, B., Chen, L., Moore, P., and Zhang, X. (2018), "Local Distribution in Neighborhood for Classification," *arXiv preprint arXiv:1812.02934*.

Papernot, N. and McDaniel, P. (2018), "Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning," *arXiv preprint arXiv:1803.04765*.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. (2016a), "The limitations of deep learning in adversarial settings," in *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, IEEE, pp. 372–387.

Papernot, N., McDaniel, P., Swami, A., and Harang, R. (2016b), "Crafting adversarial input sequences for recurrent neural networks," in *Military Communications Conference, MILCOM 2016-2016 IEEE*, IEEE, pp. 49–54.

Plötz, T. and Roth, S. (2018), "Neural nearest neighbors networks," in *Advances in Neural Information Processing Systems*, pp. 1087–1098.

Qiao, X., Duan, J., and Cheng, G. (2019), "Rates of Convergence for Large-scale Nearest Neighbor Classification," in *Advances in Neural Information Processing Systems*, pp. 10769–10780.

Reeve, H. W. and Kaban, A. (2019a), "Classification with unknown class conditional label noise on non-compact feature spaces," *arXiv preprint arXiv:1902.05627*.

— (2019b), "Fast rates for a kNN classifier robust to unknown asymmetric label noise," *arXiv preprint arXiv:1906.04542*.

Samworth, R. J. (2012), "Optimal weighted nearest neighbour classifiers," *The Annals of Statistics*, 40, 2733–2763.

Sinha, A., Namkoong, H., and Duchi, J. (2018), "Certifying some distributional robustness with principled adversarial training," .

Sun, W. W., Qiao, X., and Cheng, G. (2016), "Stabilized nearest neighbor classifier and its statistical properties," *Journal of the American Statistical Association*, 111, 1254–1265.

Wang, Y., Jha, S., and Chaudhuri, K. (2017), "Analyzing the robustness of nearest neighbors to adversarial examples," *arXiv preprint arXiv:1706.03922*.

— (2018), "Analyzing the robustness of nearest neighbors to adversarial examples," in *International Conference on Machine Learning*, pp. 5133–5142.

Xue, L. and Kpotufe, S. (2017), "Achieving the time of 1-NN, but the accuracy of k-NN," *arXiv preprint arXiv:1712.02369*.

Yang, Y.-Y., Rashtchian, C., Wang, Y., and Chaudhuri, K. (2019), "Adversarial examples for non-parametric methods: attacks, defenses and large sample limits," *arXiv preprint arXiv:1906.03310*.

— (2020a), "Robustness for non-parametric classification: A generic attack and defense," in *International Conference on Artificial Intelligence and Statistics*, pp. 941–951.

Yang, Y.-Y., Rashtchian, C., Zhang, H., Salakhutdinov, R., and Chaudhuri, K. (2020b), "Adversarial Robustness Through Local Lipschitzness," *arXiv preprint arXiv:2003.02460*.

Yeh, I.-C. and Lien, C.-h. (2009), "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients," *Expert Systems with Applications*, 36, 2473–2480.