SCIENCE CHINA
Information Sciences

• **RESEARCH PAPER** •

July 2016, Vol. 59 072104:1–072104:12
doi: 10.1007/s11432-015-5470-z

Dropout Rademacher complexity of deep neural networks

Wei GAO[1,2] & Zhi-Hua ZHOU[1,2]*

[1]*National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing* 210023, *China;*
[2]*Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing University, Nanjing* 210023, *China*

Received October 17, 2015; accepted December 8, 2015; published online June 16, 2016

Abstract Great successes of deep neural networks have been witnessed in various real applications. Many algorithmic and implementation techniques have been developed; however, theoretical understanding of many aspects of deep neural networks is far from clear. A particular interesting issue is the usefulness of dropout, which was motivated from the intuition of preventing complex co-adaptation of feature detectors. In this paper, we study the Rademacher complexity of different types of dropouts, and our theoretical results disclose that for shallow neural networks (with one or none hidden layer) dropout is able to reduce the Rademacher complexity in polynomial, whereas for deep neural networks it can amazingly lead to an exponential reduction.

Keywords artificial intelligence, machine learning, deep learning, dropout, Rademacher complexity

Citation Gao W, Zhou Z-H. Dropout Rademacher complexity of deep neural networks. Sci China Inf Sci, 2016, 59(7): 072104, doi: 10.1007/s11432-015-5470-z

1 Introduction

Deep neural networks [1] has become a hot wave during the past few years, and great successes have been achieved in various real applications, such as object recognition [2–4], speech recognition [5–7], video analysis [8, 9], etc. Many effective algorithmic and implementation techniques [10] have been developed; however, theoretical understanding of many aspects of deep neural networks is far from clear.

It is well known that deep neural networks are complicated models with rich representations. For really deep networks, there may be millions or even billions of parameters, and thus, there are high risks of overfitting even with large-scale training data. Indeed, controlling the overfitting risk is a long-standing topic in the research of neural networks, and various techniques have been developed, such as weight elimination [11], early stopping [12], Bayesian control [13], etc.

Dropout is among the key ingredients of the success of deep neural networks. The main idea is to randomly omit some units, either hidden ones or input ones corresponding to different input features; this is executed with certain probability in the forward propagation of training phase, and the weights related to the remaining units are updated in back propagation. This technique is evidently related to overfitting control, though it was proposed with the intuition of preventing complex co-adaptations

* Corresponding author (email: zhouzh@lamda.nju.edu.cn)

by encouraging independent contributions from different features during training phase [14]. Extensive empirical studies [14–16] verified that dropout is able to improve the performance and reduce ovefitting risk. However, theoretical understanding of dropout is far from clear.

In this paper, we study the influence on Rademacher complexity by three types of dropouts, i.e., dropout of units [14], dropout of weights [16] and dropout of both. Our theoretical results disclose that for shallow neural networks with none or one hidden layer, dropout is able to reduce the Rademacher complexity in polynomial, whereas for deep neural networks it is able to reach an exponential reduction of Rademacher complexity.

Related work.

There are several designs of dropout, such as the fast dropout [17] and adaptive dropout [18], whereas the most fundamental dropouts are the dropout of units (hidden units, or input units corresponding to input features) [14, 19] and the dropout of weights [16].

The average and regularizing properties of dropout have been studied in [20]. Wager et al. [21] showed that dropout is first-order regular equivalent to an L_2 regularizer applied after scaling the features by an estimate of the inverse diagonal Fisher information matrix. The generalization bound of dropout has been analyzed in [16, 22]. McAllester [22] presented PAC-Bayesian bounds, whereas Wan et al. [16] derived Rademacher generalization bounds. Both their results show that the reduction of complexity brought by dropout is $\mathrm{O}(\rho)$, where ρ is the probability of keeping an element in dropout.

In contrast to previous studies [16, 22], we present better generalization bounds and disclose that dropout is able to reduce the Rademacher complexity exponentially, i.e., $\mathrm{O}(\rho^{k+1})$ or $\mathrm{O}(\rho^{(k+1)/2})$ for different types of dropouts, where k is the number of hidden layers within neural networks.

Extensive work [23, 24, and reference therein] studied the complexity of neural network based on VC-dimension, covering number, fat-shatter dimension, etc., and it was usually shown that these complexities are polynomial in the total number of units and weights. Note that the polynomial complexities are still very high for deep neural networks that may have million or even billions of parameters. Moreover, it is worth noting that these complexities measure the function space in the worst case, and cannot distinguish situations with/without dropouts. In contrast, we show that Radermacher complexity is proper to study the influence of dropouts, and we prove that the complexities of neural network can be bounded by the L_1 or L_2-norm of weights, irrelevant to the number of units and weights.

This paper is organized as follows: Section 2 introduces some preliminaries. Section 3 presents general Rademacher generalization bounds for dropout. Section 4 analyzes the usefulness of different types of dropouts on shallow as well as deep neural networks. Section 5 concludes.

2 Preliminaries

Let $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y}$ be the input and output space, respectively, where $\mathcal{Y} \subset \mathbb{R}$ for regression and $\mathcal{Y} = \{+1, -1\}$ for binary classification. Throughout this paper, we restrict our attention on regression and binary classification, and it is easy to make similar analysis for multi-class tasks. Let $\mathcal{D}$ be an unknown (underlying) distribution over $\mathcal{X} \times \mathcal{Y}$.

Let $\mathcal{W}$ be the weight space for neural network, and denote $f(\boldsymbol{w}, \boldsymbol{x})$ the general output of a neural network with respect to input $\boldsymbol{x} \in \mathcal{X}$ and weight $\boldsymbol{w} \in \mathcal{W}$. Here f depends on the structure of neural network. During training neural network, dropout randomly omits hidden units, input units corresponding to input features, and connected weights with certain probability; therefore, it is necessary to introduce another space

$$\mathcal{R} = \{\boldsymbol{r} = (r_1, r_2, \ldots, r_s)\colon r_i \in \{0, 1\}\},$$

where s depends on different neural networks and different types of dropouts, and $r_i = 0$ implies dropping out some hidden unit, input unit and weight. Here each r_i is drawn independently and identically from a Bernoulli distribution with parameter ρ, denoted by $\mathrm{Bern}(\rho)$. Further, we denote $f(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{r})$ the dropout output of a neural network, and write

$$\mathcal{F}_{\mathcal{W}} = \{f(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{r}) : \boldsymbol{w} \in \mathcal{W}\}$$

as the function space for dropout. Here we just present a general output $f(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{r})$ for dropout, and detailed expressions will be given for specific neural network in Section 4.

An objective function (or loss function) ℓ is introduced to measure the performance of output of neural network. For example, least square loss and cross entropy are used for regression and binary classification, respectively. We define the expected risk for dropout as

$$R(\boldsymbol{w}) = E_{\boldsymbol{r},(\boldsymbol{x},y)}[\ell(f(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{r}), y)].$$

The goal is to find a $\boldsymbol{w}^* \in \mathcal{W}$ so as to minimize the expected risk, i.e., $\boldsymbol{w}^* \in \arg\min_{\boldsymbol{w}\in\mathcal{W}} R(\boldsymbol{w})$. Notice that the distribution $\mathcal{D}$ is unknown, but it is demonstrated by a training sample

$$S_n = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\},$$

which are drawn i.i.d. from distribution $\mathcal{D}$. Given sample S_n and $RS_n = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_n\}$, we define the empirical risk for dropout as

$$\hat{R}(\boldsymbol{w}, S_n, RS_n) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(\boldsymbol{w}, \boldsymbol{x}_i, \boldsymbol{r}_i), y_i).$$

In this paper, we try to study on generalization bounds for dropouts, i.e., the gap between $R(\boldsymbol{w})$ and $\hat{R}(\boldsymbol{w}, S_n, RS_n)$. Rademacher complexity has always been an efficient measure for function space [25, 26]. For function space $\mathcal{H}$, the classical Rademacher complexity is defined by

$$\hat{\mathfrak{R}}_n(\mathcal{H}) = E\left[\sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} \epsilon_i h(\boldsymbol{x}_i)\right], \tag{1}$$

where $\epsilon_1, \ldots, \epsilon_n$ are independent random variables uniformly chosen from $\{+1, -1\}$, and they are referred as Rademacher variables. Rademacher complexity has been used to develop data-dependent generalization bounds in diverse learning tasks [27–29].

For notational simplicity, we denote $[n] = \{1, 2, \ldots, n\}$ for integer $n > 0$. The inner product between $\boldsymbol{w} = (w_1, \ldots, w_d)$ and $\boldsymbol{x} = (x_1, \ldots, x_d)$ is given by $\langle \boldsymbol{w}, \boldsymbol{x}\rangle = \sum_{i=1}^{d} w_i x_i$, and write $\|\boldsymbol{w}\| = \|\boldsymbol{w}\|_2 = \sqrt{\langle \boldsymbol{w}, \boldsymbol{w}\rangle}$ and $\|\boldsymbol{w}\|_1 = \sum_{i=1}^{d} |w_i|$. Further, the entrywise product (also called Schur product or Hadamard product) is defined as $\boldsymbol{w} \odot \boldsymbol{x} = (w_1x_1, \ldots, w_dx_d)$.

3 General Rademacher generalization bounds

In conventional studies, the generalization performance is mostly affected by training sample [30], and the standard Rademacher complexity is defined on training sample only (as shown in Eq. (1)). For dropout, however, the generalization performance is not only relevant to training sample, but also dropout randomization; thus, we generalize the Rademacher complexity as follows:

Definition 1. For spaces $\mathcal{Z}$ and $\mathcal{R}$, let $\mathcal{H}\colon \mathcal{Z} \times \mathcal{R} \to \mathbb{R}$ be a real-valued function space. For $S_n = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\}$ and $RS_n = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_n\}$, the empirical Rademacher complexity of $\mathcal{H}$ is defined to be

$$\hat{\mathfrak{R}}_n(\mathcal{H}, S_n, RS_n) = E_\epsilon\left[\sup_{h\in\mathcal{H}} \left(\frac{1}{n}\sum_{i=1}^{n} \epsilon_i h(\boldsymbol{z}_i, \boldsymbol{r}_i)\right)\right],$$

where $\epsilon = (\epsilon_1, \ldots, \epsilon_n)$ are Rademacher variables. Further, we define the Rademacher complexity of $\mathcal{H}$ as

$$\mathfrak{R}_n(\mathcal{H}) = E_{S_n, RS_n}[\hat{\mathfrak{R}}_n(\mathcal{H}, S_n, RS_n)].$$

Based on this definition, it is easy to get a useful lemma as follows.

Lemma 1. For function space $\mathcal{H}$, define $\mathrm{absconv}(\mathcal{H}) = \{\sum \alpha_i h_i\colon h_i \in \mathcal{H} \text{ and } \sum |\alpha_i| = 1\}$. Then, we have

$$\hat{\mathfrak{R}}_n(\mathcal{H}, S_n, RS_n) = \hat{\mathfrak{R}}_n(\mathrm{absconv}(\mathcal{H}), S_n, RS_n).$$

Given a set $\mathcal{W}$, we denote composite function space for dropout as

$$\ell \circ \mathcal{F}_{\mathcal{W}} := \{((\boldsymbol{x}, y), \boldsymbol{r}) \to \ell(f(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{r}), y), \boldsymbol{w} \in \mathcal{W}\}.$$

Based on the generalized Rademacher complexity, we present the general Rademacher generalization bounds for dropout as follows.

Theorem 1. Let $S_n = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$ be a sample chosen i.i.d. according to distribution $\mathcal{D}$, and let $RS_n = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_n\}$ be random variable sample for dropout. If the loss function ℓ is bounded by $B > 0$, for every $\delta > 0$ and $\boldsymbol{w} \in \mathcal{W}$, the following holds with probability at least $1 - \delta$,

$$R(\boldsymbol{w}) \leqslant \hat{R}(\boldsymbol{w}, S_n, RS_n) + 2\mathfrak{R}_n(\ell \circ \mathcal{F}_{\mathcal{W}}) + B\sqrt{\ln(2/\delta)/n}, \tag{2}$$

$$R(\boldsymbol{w}) \leqslant \hat{R}(\boldsymbol{w}, S_n, RS_n) + 2\hat{\mathfrak{R}}_n(\ell \circ \mathcal{F}_{\mathcal{W}}, S_n, RS_n) + 3B\sqrt{\ln(2/\delta)/n}. \tag{3}$$

Proof. The proof is motivated from the techniques in [25]. For every $\boldsymbol{w} \in \mathcal{W}$, it is easy to observe

$$R(\boldsymbol{w}) \leqslant \hat{R}(\boldsymbol{w}, S_n, RS_n) + \sup_{\boldsymbol{w}}[R(\boldsymbol{w}) - \hat{R}(\boldsymbol{w}, S_n, RS_n)],$$

and we further denote

$$\Phi(S_n, RS_n) = \sup_{\boldsymbol{w}}[R(\boldsymbol{w}) - \hat{R}(\boldsymbol{w}, S_n, RS_n)] = \sup_{\boldsymbol{w}}\left[R(\boldsymbol{w}) - \frac{1}{n}\sum_{i=1}^{n}\ell(f(\boldsymbol{w}, \boldsymbol{x}_i, \boldsymbol{r}_i), y_i)\right].$$

Let $S_n^{i,(\boldsymbol{x}_i', y_i')} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_i', y_i'), \ldots, (\boldsymbol{x}_n, y_n)\}$ be the sample whose i-th example $(\boldsymbol{x}_i, y_i)$ in S_n is replaced by $(\boldsymbol{x}_i', y_i')$, and $RS_n^{i,\boldsymbol{r}_i'} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_i', \ldots, \boldsymbol{r}_n\}$ be the random variable vector with i-th variable $\boldsymbol{r}_i$ replaced by $\boldsymbol{r}_i'$. For bounded loss $|\ell| < B$, we have

$$|\Phi(S_n, RS_n) - \Phi(S_n, RS_n^{i,\boldsymbol{r}_i'})| \leqslant B/m \text{ and } |\Phi(S_n, RS_n) - \Phi(S_n^{i,(\boldsymbol{x}_i', y_i')}, RS_n)| \leqslant B/m.$$

Based on McDiarmid's inequality [31], it holds that with probability at least $1 - \delta$,

$$\Phi(S_n, RS_n) \leqslant E_{S_n, RS_n}[\Phi(S_n, RS_n)] + B\sqrt{\ln(2/\delta)/n}.$$

Define a ghost sample $\tilde{S}_n = \{(\tilde{\boldsymbol{x}}_1, \tilde{y}_1), \ldots, (\tilde{\boldsymbol{x}}_n, \tilde{y}_n)\}$ and a ghost random variable vector $\widetilde{RS}_n = \{\tilde{\boldsymbol{r}}_1, \ldots, \tilde{\boldsymbol{r}}_1\}$. By using the fact

$$\Phi(S_n, RS_n) = \sup_{\boldsymbol{w}}[E_{\tilde{S}_n, \widetilde{RS}_n}[\hat{R}(\boldsymbol{w}, \tilde{S}_n, \widetilde{RS}_n) - \hat{R}(\boldsymbol{w}, S_n, RS_n)]],$$

we have

$$\begin{aligned} E_{S_n, RS_n}[\Phi(S_n, RS_n)] &\leqslant E\left[\sup_{\boldsymbol{w}}\left[\hat{R}(\boldsymbol{w}, \tilde{S}_n, \widetilde{RS}_n) - \hat{R}(\boldsymbol{w}, S_n, RS_n)\right]\right] \\ &= E\left[\sup_{\boldsymbol{w}}\left[\frac{\sum_{i=1}^{n}\ell(f(\boldsymbol{w}, \tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{r}}_i), \tilde{y}_i) - \ell(f(\boldsymbol{w}, \boldsymbol{x}_i, \boldsymbol{r}_i), y_i)}{n}\right]\right] \\ &\leqslant 2E\left[\sup_{\boldsymbol{w}}\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\ell(f(\boldsymbol{w}, \boldsymbol{x}_i, \boldsymbol{r}_i), y_i)\right] = 2\mathfrak{R}_n(\ell \circ \mathcal{F}_{\mathcal{W}}), \end{aligned}$$

which completes the proofs of Eq. (2). Again, we apply McDiarmid's inequality to $\hat{\mathfrak{R}}_n(\mathcal{W}, S_n, RS_n)$, and we have

$$\mathfrak{R}_n(\ell \circ \mathcal{F}_{\mathcal{W}}) \leqslant \hat{\mathfrak{R}}_n(\ell \circ \mathcal{F}_{\mathcal{W}}, S_n, RS_n) + B\sqrt{\ln(2/\delta)/n},$$

which completes the proof of Eq. (3).

The main benefit of dropout lies in the sharp reduction on Rademacher complexities of $\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}})$ as will been shown in Section 4; on the other hand, extensive experiments show that dropout decreases the empirical risk $\hat{R}(\boldsymbol{w}, S_n, RS_n)$ [14–16] because dropout intuitively prevents complex co-adaptations by encouraging independent contributions from different features during training phase [14]. This paper tries to present theoretical analysis on the the former, and leave the latter to future work.

To efficiently estimate $\mathfrak{R}_n(\ell \circ \mathcal{F}_{\mathcal{W}})$, we introduce a concentration as follows.

Lemma 2 ([32]). Let $\mathcal{H}$ be a bounded real-valued function space from some space $\mathcal{Z}$ and $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n \in \mathcal{Z}$. Let $\phi\colon \mathbb{R} \to \mathbb{R}$ be Lipschitz with constant L and $\phi(0) = 0$. Then, we have

$$E_\epsilon \sup_{h\in\mathcal{H}} \frac{1}{n} \sum_{i\in[n]} \epsilon_i \phi(h(\boldsymbol{z}_i)) \leqslant L E_\epsilon \sup_{h\in\mathcal{H}} \frac{1}{n} \sum_{i\in[n]} \epsilon_i h(\boldsymbol{z}_i).$$

Based on this lemma, we have the following lemma.

Lemma 3. If $\ell(\cdot,\cdot)$ is Lipschitz with the first argument and constant L, we have

$$\mathfrak{R}_n(\ell \circ \mathcal{F}_\mathcal{W}) \leqslant L\mathfrak{R}_n(\mathcal{F}_\mathcal{W}).$$

Proof. We first write $\ell'(\cdot,\cdot) = \ell(\cdot,\cdot) - \ell(0,\cdot)$, and it is easy to get $\mathfrak{R}_n(\ell \circ \mathcal{F}_\mathcal{W}) = \mathfrak{R}_n(\ell' \circ \mathcal{F}_\mathcal{W})$. This lemma holds by applying Lemma 2 to ℓ'.

For classification, we always use the entropy loss as the loss function in neural network as follows:

$$\ell(f(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r}), y) = y \ln(y/\phi(f(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r}))) + (1-y)\ln((1-y)/(1-\phi(f(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})))),$$

where $\phi(t) = 1/(1+e^{-t})$. It is easy to find that $\partial\ell(f(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r}),y)/\partial f(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r}) \in [-1,1]$, and thus $\ell(\cdot,\cdot)$ is a Lipschitz function with the first argument.

For regression, we always use the square loss as the loss function in neural network as follows:

$$\ell(f(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r}), y) = (y - f(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r}))^2.$$

For bounded $f(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})$ and y, it is easy to find that $\ell(\cdot,\cdot)$ is a Lipschitz function with the first argument.

Based on Lemma 3, it is easy to estimate $\mathfrak{R}_n(\ell\circ\mathcal{F}_\mathcal{W})$ from $\mathfrak{R}_n(\mathcal{F}_\mathcal{W})$; therefore, we will focus on how to estimate $\mathfrak{R}_n(\mathcal{F}_\mathcal{W})$ for different types of dropouts and different neural networks in the subsequent section.

Finally, we introduce a useful lemma as follows.

Lemma 4. Let $\boldsymbol{r}_1 = (r_{11}, \ldots, r_{1d})$ and $\boldsymbol{r}_2 = (r_{21}, \ldots, r_{2d})$ be two random variable vectors, and each element in $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ is drawn i.i.d. from distribution Bern(ρ). For $\boldsymbol{x} \in \mathcal{X}$, we have

$$E_{\boldsymbol{r}_1}[\langle \boldsymbol{x}\odot\boldsymbol{r}_1, \boldsymbol{x}\odot\boldsymbol{r}_1\rangle] = \rho\langle\boldsymbol{x},\boldsymbol{x}\rangle, \tag{4}$$

$$E_{\boldsymbol{r}_1,\boldsymbol{r}_2}[\langle \boldsymbol{x}\odot\boldsymbol{r}_1\odot\boldsymbol{r}_2, \boldsymbol{x}\odot\boldsymbol{r}_1\odot\boldsymbol{r}_2\rangle] = \rho^2\langle\boldsymbol{x},\boldsymbol{x}\rangle. \tag{5}$$

Further, let $\boldsymbol{r} = (r_1, \ldots, r_k)$ be k random variables drawn i.i.d. from distribution Bern(ρ). We have

$$\mathop{E}_{\boldsymbol{r}_1,\boldsymbol{r}} \left\langle \boldsymbol{x}\odot\boldsymbol{r}_1\prod_{i=1}^k r_i, \boldsymbol{x}\odot\boldsymbol{r}_1\prod_{i=1}^k r_i \right\rangle = \rho^{k+1}\langle\boldsymbol{x},\boldsymbol{x}\rangle, \tag{6}$$

$$\mathop{E}_{\boldsymbol{r},\boldsymbol{r}_1,\boldsymbol{r}_2} \left\langle \boldsymbol{x}\odot\boldsymbol{r}_1\odot\boldsymbol{r}_2\prod_{i=1}^k r_i, \boldsymbol{x}\odot\boldsymbol{r}_1\odot\boldsymbol{r}_2\prod_{i=1}^k r_i \right\rangle = \rho^{k+2}\langle\boldsymbol{x},\boldsymbol{x}\rangle. \tag{7}$$

Proof. Let $\boldsymbol{x} = (x_1, \ldots, x_d)$. From the definitions of inner product and entrywise product, Eq. (4) holds from

$$E_{\boldsymbol{r}_1}[\langle \boldsymbol{x}\odot\boldsymbol{r}_1, \boldsymbol{x}\odot\boldsymbol{r}_1\rangle] = E_{\boldsymbol{r}_1}\left[\sum_{j=1}^d x_j x_j r_{1j}^2\right] = \rho\sum_{j=1}^d x_j x_j,$$

where we use the fact $E_{r_{1j}}[r_{1j}^2] = \rho$ since r_{1j} is drawn i.i.d. from distribution Bern(ρ). In a similar manner, Eqs. (5)–(7) hold from $E_{r_{1j},r_{2j}}[r_{1j}^2 r_{2j}^2] = \rho^2$, $E_{\boldsymbol{r},r_{1j}}[r_{1j}\prod_{i=1}^k (r_i)^2] = \rho^{1+k}$ and $E_{\boldsymbol{r},r_{1j},r_{2j}}[r_{1j}^2 r_{2j}^2 \prod_{i=1}^k (r_i)^2] = \rho^{2+k}$, respectively. This lemma follows as desired.

4 Dropouts on different types of network

We study the two types of most fundamental dropouts: dropout units [14] and dropout weights [16]. In addition, we also study dropout both units and weights. For $\rho \in [0,1]$, these types of dropouts are defined as

• Type I ($\mathrm{Drp}^{(\mathrm{I})}$): randomly drop out each unit including input unit (corresponding to input feature) with probability $1-\rho$.

• Type II ($\mathrm{Drp}^{(\mathrm{II})}$): randomly drop out each weight with probability $1-\rho$.

• Type III ($\mathrm{Drp}^{(\mathrm{III})}$): randomly drop out each weight and unit including input unit (corresponding to input feature) with probability $1-\rho$.

We assume that a full-connected neural network has k hidden layers, and the ith hidden layer has m_i hidden units. The general output for this neural network is given by

$$f(\boldsymbol{w},\boldsymbol{x})=\langle \boldsymbol{w}_1^{[k]},\Psi_k\rangle \text{ with } \Psi_i=(\sigma(\langle \boldsymbol{w}_1^{[i-1]},\Psi_{i-1}\rangle),\ldots,\sigma(\langle \boldsymbol{w}_{m_i}^{[i-1]},\Psi_{i-1}\rangle)) \text{ for } i\in[k]$$

and $\Psi_0=\boldsymbol{x}$, where $\boldsymbol{w}=(\boldsymbol{w}_1^{[k]},\boldsymbol{w}_1^{[k-1]},\ldots,\boldsymbol{w}_{m_k}^{[k-1]},\ldots,\boldsymbol{w}_1^{[0]},\ldots,\boldsymbol{w}_{m_1}^{[0]})$ in which each $\boldsymbol{w}_i^{[j]}$ has the same size as Ψ_j and σ is an activation function.

Throughout this work, we assume that activation function σ is Lipschitz with constant L and $\sigma(0)=0$, and many commonly used activation functions satisfy such assumptions, e.g., tanh, center sigmoid, relu [33], etc.

Formally, three types of dropouts for the full-connected network are defined as

• The output for $\mathrm{Drp}^{(\mathrm{I})}$ (first type) is given by

$$\begin{aligned}&f^{(\mathrm{I})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})=\langle \boldsymbol{w}_1^{[k]},\Psi_k\odot \boldsymbol{r}^{[k]}\rangle \text{ with}\\&\Psi_i=(\sigma(\langle \boldsymbol{w}_1^{[i-1]},\Psi_{i-1}\odot \boldsymbol{r}^{[i-1]}\rangle),\ldots,\sigma(\langle \boldsymbol{w}_{m_i}^{[i-1]},\Psi_{i-1}\odot \boldsymbol{r}^{[i-1]}\rangle)),\end{aligned}\tag{8}$$

for $i\in[k]$ and $\Psi_0=\boldsymbol{x}$. Here $\boldsymbol{r}=(\boldsymbol{r}^{[k]},\ldots,\boldsymbol{r}^{[1]},\boldsymbol{r}^{[0]})$, each $\boldsymbol{r}^{[i]}$ has the same size with Ψ_i and each element in $\boldsymbol{r}^{[i]}$ is drawn i.i.d. from $\mathrm{Bern}(\rho)$.

• The output for $\mathrm{Drp}^{(\mathrm{II})}$ (second type) is given by

$$\begin{aligned}&f^{(\mathrm{II})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})=\langle \boldsymbol{w}_1^{[k]}\odot \boldsymbol{r}_1^{[k]},\Psi_k\rangle \text{ with}\\&\Psi_i=(\sigma(\langle \boldsymbol{w}_1^{[i-1]}\odot \boldsymbol{r}_1^{[i-1]},\Psi_{i-1}\rangle),\ldots,\sigma(\langle \boldsymbol{w}_{m_i}^{[i-1]}\odot \boldsymbol{r}_{m_i}^{[i-1]},\Psi_{i-1}\rangle)),\end{aligned}\tag{9}$$

for $i\in[k]$ and $\Psi_0=\boldsymbol{x}$. Here $\boldsymbol{r}=\{\boldsymbol{r}_1^{[k]},\boldsymbol{r}_1^{[k-1]},\ldots,\boldsymbol{r}_{m_k}^{[k-1]},\ldots,\boldsymbol{r}_1^{[0]},\ldots,\boldsymbol{r}_{m_1}^{[0]}\}$, and for $0\leqslant j\leqslant k$, $\boldsymbol{r}_i^{[j]}$ has the same size with Ψ_j, and each element in $\boldsymbol{r}_i^{[j]}$ is drawn i.i.d. from $\mathrm{Bern}(\rho)$.

• The output for $\mathrm{Drp}^{(\mathrm{III})}$ (third type) is given by

$$\begin{aligned}&f^{(\mathrm{III})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})=\langle \boldsymbol{w}_1^{[k]}\odot \boldsymbol{r}_1^{[k]},\Psi_k\odot \boldsymbol{r}_2^{[k]}\rangle \text{ with}\\&\Psi_i=(\sigma(\langle \boldsymbol{w}_1^{[i-1]}\odot \boldsymbol{r}_1^{[i-1]},\Psi_{i-1}\odot \boldsymbol{r}_{m_i+1}^{[i-1]}\rangle),\ldots,\sigma(\langle \boldsymbol{w}_{m_i}^{[i-1]}\odot \boldsymbol{r}_{m_i}^{[i-1]},\Psi_{i-1}\odot \boldsymbol{r}_{m_i+1}^{[i-1]}\rangle)),\end{aligned}\tag{10}$$

for $i\in[k]$ and $\Psi_0=\boldsymbol{x}$. Here $\boldsymbol{r}=(\boldsymbol{r}_1^{[k]},\boldsymbol{r}_2^{[k]},\ldots,\boldsymbol{r}_1^{[0]},\ldots,\boldsymbol{r}_{m_1+1}^{[0]})$, and for $0\leqslant j\leqslant k$, $\boldsymbol{r}_j^{[i]}$ has the same size with Ψ_j, and each element in $\boldsymbol{r}_j^{[i]}$ is drawn i.i.d. from $\mathrm{Bern}(\rho)$.

Given a set $\mathcal{W}$, we denote

$$\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})}=\{f^{(\mathrm{I})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})\colon \boldsymbol{w}\in\mathcal{W}\},\tag{11}$$

$$\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}=\{f^{(\mathrm{II})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})\colon \boldsymbol{w}\in\mathcal{W}\},\tag{12}$$

$$\mathcal{F}_{\mathcal{W}}^{(\mathrm{III})}=\{f^{(\mathrm{III})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})\colon \boldsymbol{w}\in\mathcal{W}\},\tag{13}$$

where $f^{(\mathrm{I})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})$, $f^{(\mathrm{II})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})$ and $f^{(\mathrm{III})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})$ are defined in Eqs. (8)–(10).

We will focus on full-connected neural networks, both shallow ones (with none or one hidden layer) and deep ones (with more hidden layers).

4.1 Shallow network without hidden layer

We first consider the shallow network without hidden layer, and therefore, the output is a linear function, i.e., $f(\boldsymbol{w},\boldsymbol{x})=\langle \boldsymbol{w},\boldsymbol{x}\rangle$. Further, the outputs for $\mathrm{Drp}^{(\mathrm{I})}$, $\mathrm{Drp}^{(\mathrm{II})}$ and $\mathrm{Drp}^{(\mathrm{III})}$ are given, respectively, by

$$f^{(\mathrm{I})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})=\langle \boldsymbol{w},\boldsymbol{x}\odot \boldsymbol{r}\rangle,$$

$$f^{(\mathrm{II})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})=\langle\boldsymbol{w}\odot\boldsymbol{r},\boldsymbol{x}\rangle,$$
$$f^{(\mathrm{III})}(\boldsymbol{w},\boldsymbol{x},(\boldsymbol{r}_1,\boldsymbol{r}_2))=\langle\boldsymbol{w}\odot\boldsymbol{r}_1,\boldsymbol{x}\odot\boldsymbol{r}_2\rangle,$$

where $\boldsymbol{r}$, $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are of size d, and each element in $\boldsymbol{r}$, $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ is drawn i.i.d. from Bern(ρ). The following theorem shows the Rademecher complexity for three types of dropouts:

Theorem 2. Let $\mathcal{W}=\{\boldsymbol{w}\colon\|\boldsymbol{w}\|<B_1\}$, $\mathcal{X}=\{\boldsymbol{x}\colon\|\boldsymbol{x}\|\leqslant B_2\}$, and $\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})}$, $\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}$ and $\mathcal{F}_{\mathcal{W}}^{(\mathrm{III})}$ are defined in Eqs. (11)–(13). Then, we have

$$\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(1)})=\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(2)})\leqslant B_1B_2\sqrt{\rho/n}\quad\text{and}\quad\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(3)})\leqslant B_1B_2\rho/\sqrt{n}.$$

If we do not drop out any weights and input units (corresponding to input features), i.e., $\rho=1$, the above theorem gives a similar estimation for the Rademacher complexity of linear function space as stated in [34, Theorem 3]. Also, these complexities are independent to feature dimension, and thus can be applied to high-dimensional data. In addition, such result has independent interests in missing feature problems.

Proof. From $\langle\boldsymbol{w}\odot\boldsymbol{r},\boldsymbol{x}\rangle=\langle\boldsymbol{w},\boldsymbol{x}\odot\boldsymbol{r}\rangle$, it is easy to prove $\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})})=\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})})$. For $S_n=\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n\}$ and $RS_n=\{\boldsymbol{r}_1,\ldots,\boldsymbol{r}_n\}$, we have

$$\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})},S_n,RS_n)=\frac{1}{n}E_\epsilon\sup_{\boldsymbol{w}\in\mathcal{W}}\sum_{i=1}^n\epsilon_i\langle\boldsymbol{w},\boldsymbol{x}_i\odot\boldsymbol{r}_i\rangle,$$

where $\epsilon=(\epsilon_1,\ldots,\epsilon_n)$ are rademacher variables, and this yields that

$$\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})},S_n,RS_n)=\frac{1}{n}E_\epsilon\sup_{\boldsymbol{w}\in\mathcal{W}}\left\langle\boldsymbol{w},\sum_{i=1}^n\epsilon_i\boldsymbol{x}_i\odot\boldsymbol{r}_i\right\rangle.$$

By using the Cauchy-Schwartz inequality $\langle a,b\rangle\leqslant\|a\|\|b\|$ and $\|\boldsymbol{w}\|\leqslant B_1$, we have

$$\begin{aligned}\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})},S_n,RS_n)&\leqslant\frac{B_1}{n}E_\epsilon\left\|\sum_{i=1}^n\epsilon_i\boldsymbol{x}_i\odot\boldsymbol{r}_i\right\|\\&=\frac{B_1}{n}E_\epsilon\left(\sum_{i=1}^n\sum_{j=1}^n\epsilon_i\epsilon_j\langle\boldsymbol{x}_i\odot\boldsymbol{r}_i,\boldsymbol{x}_j\odot\boldsymbol{r}_j\rangle\right)^{1/2}\\&\leqslant\frac{B_1}{n}\left(\sum_{i=1}^n\sum_{j=1}^nE_{\epsilon_i,\epsilon_j}\epsilon_i\epsilon_j\langle\boldsymbol{x}_i\odot\boldsymbol{r}_i,\boldsymbol{x}_j\odot\boldsymbol{r}_j\rangle\right)^{1/2},\end{aligned}$$

where the last inequality holds from Jensen's inequality. Since $E_{\epsilon_i,\epsilon_j}\epsilon_i\epsilon_j=0$ for $i\neq j$ and $E_{\epsilon_i}\epsilon_i\epsilon_i=1$ for rademacher variables, we have

$$\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})},S_n,RS_n)\leqslant\frac{B_1}{n}\left(\sum_{i=1}^n\langle\boldsymbol{x}_i\odot\boldsymbol{r}_i,\boldsymbol{x}_i\odot\boldsymbol{r}_i\rangle\right)^{1/2}.\tag{14}$$

Based on the above inequality, it holds that

$$\begin{aligned}\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})})&=E_{S_n,RS_n}[\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})},S_n,RS_n)]\\&\leqslant\frac{B_1}{n}E_{S_n,RS_n}\left(\sum_{i=1}^n\langle\boldsymbol{x}_i\odot\boldsymbol{r}_i,\boldsymbol{x}_i\odot\boldsymbol{r}_i\rangle\right)^{1/2}\\&\leqslant\frac{B_1}{n}E_{S_n}\left(\sum_{i=1}^nE_{\boldsymbol{r}_i}\langle\boldsymbol{x}_i\odot\boldsymbol{r}_i,\boldsymbol{x}_i\odot\boldsymbol{r}_i\rangle\right)^{1/2},\end{aligned}$$

where the second inequality holds from Jensen's inequality. Finally, we have $\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})})\leqslant B_1B_2\sqrt{\rho/n}$ from Eq. (4) and $\|\boldsymbol{x}_i\|\leqslant B_2$.

In a similar manner, we have $\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{III})})\leqslant B_1B_2\rho/\sqrt{n}$ from $\langle\boldsymbol{w}\odot\boldsymbol{r}_1,\boldsymbol{x}\odot\boldsymbol{r}_2\rangle=\langle\boldsymbol{w},\boldsymbol{x}\odot\boldsymbol{r}_1\odot\boldsymbol{r}_2\rangle$ and Eq. (5). This theorem follows as desired.

4.2 Shallow network with one hidden layer

We consider the shallow network with only one hidden layer, and assume that the hidden layer has m hidden units. The output for such network is given by

$$f(\boldsymbol{w},\boldsymbol{x})=\langle \boldsymbol{w}^{[1]},\Psi(\boldsymbol{w}_1^{[0]},\ldots,\boldsymbol{w}_m^{[0]},\boldsymbol{x})\rangle$$

with

$$\Psi(\boldsymbol{w}_1^{[0]},\ldots,\boldsymbol{w}_m^{[0]},\boldsymbol{x})=(\sigma(\langle\boldsymbol{w}_1^{[0]},\boldsymbol{x}\rangle),\ldots,\sigma(\langle\boldsymbol{w}_m^{[0]},\boldsymbol{x}\rangle)), \tag{15}$$

where $\boldsymbol{w}=(\boldsymbol{w}^{[1]},\boldsymbol{w}_1^{[0]},\ldots,\boldsymbol{w}_m^{[0]})$, and $\boldsymbol{w}^{[1]}$ and $\boldsymbol{w}_i^{[0]}$ ($i\in[m]$) are of size m and d, respectively.

From Eqs. (8)–(10), the outputs for $\mathrm{Drp}^{(\mathrm{I})}$, $\mathrm{Drp}^{(\mathrm{II})}$ and $\mathrm{Drp}^{(\mathrm{III})}$ are given, respectively, by

$$f^{(\mathrm{I})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})=\langle\boldsymbol{w}^{[1]},\boldsymbol{r}_1^{[1]}\odot\Psi(\boldsymbol{w}_1^{[0]},\ldots,\boldsymbol{w}_m^{[0]},\boldsymbol{x}\odot\boldsymbol{r}_1^{[0]})\rangle,$$
$$f^{(\mathrm{II})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})=\langle\boldsymbol{w}^{[1]}\odot\boldsymbol{r}_1^{[1]},\Psi(\boldsymbol{w}_1^{[0]}\odot\boldsymbol{r}_1^{[0]},\ldots,\boldsymbol{w}_m^{[0]}\odot\boldsymbol{r}_m^{[0]},\boldsymbol{x})\rangle$$

and

$$f^{(\mathrm{III})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})=\langle\boldsymbol{w}^{[1]}\odot\boldsymbol{r}_1^{[1]},\boldsymbol{r}_2^{[1]}\odot\Psi(\boldsymbol{w}_1^{[0]}\odot\boldsymbol{r}_1^{[0]},\ldots,\boldsymbol{w}_m^{[0]}\odot\boldsymbol{r}_1^{[0]},\boldsymbol{x}\odot\boldsymbol{r}_{m+1}^{[0]})\rangle,$$

where Ψ is defined in Eq. (15). Here $\boldsymbol{r}_i^{[1]}$ and $\boldsymbol{r}_j^{[0]}$ are of size m and d, respectively, and each element in $\boldsymbol{r}_i^{[1]}$ and $\boldsymbol{r}_j^{[0]}$ is drawn i.i.d. from Bern(ρ). The following theorem shows the Rademecher complexity for three types of dropouts.

Theorem 3. Let $\mathcal{W}=\{(\boldsymbol{w}^{[1]},\boldsymbol{w}_1^{[0]},\ldots,\boldsymbol{w}_m^{[0]})\colon\|\boldsymbol{w}^{[1]}\|_1\leqslant B_1,\|\boldsymbol{w}_i^{[0]}\|\leqslant B_0\}$, $\mathcal{X}=\{\boldsymbol{x}\in\mathbb{R}^d\colon\|\boldsymbol{x}\|\leqslant\hat{B}\}$ and $\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})}$, $\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}$, $\mathcal{F}_{\mathcal{W}}^{(\mathrm{III})}$ are defined in Eqs. (11)–(13). Suppose that the activation σ is Lipschitz with constant L and $\sigma(0)=0$. Then, we have

$$\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})})\leqslant\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})})\leqslant LB_1B_0\hat{B}\rho/\sqrt{n}\quad\text{and}\quad\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{III})})\leqslant LB_1B_0\hat{B}\rho^2/\sqrt{n}.$$

Proof. We first have

$$\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})})\leqslant\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})})$$

from $f^{(\mathrm{II})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r})=f^{(\mathrm{I})}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{r}')$ by selecting $\boldsymbol{r}_1^{[0]}=\cdots=\boldsymbol{r}_m^{[0]}=\boldsymbol{r}'^{[0]}$ and $\boldsymbol{r}^{[1]}=\boldsymbol{r}'^{[1]}$. In the following, we will estimate $\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})})$.

Given $S_n=\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n\}$ and $RS_n=\{\boldsymbol{r}_1,\ldots,\boldsymbol{r}_n\}$, it holds that

$$\begin{aligned}\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})},S_n,RS_n)&=\frac{1}{n}E_\epsilon\left[\sup_{\boldsymbol{w}}\left\langle\boldsymbol{w}^{[1]},\sum_{i=1}^n\epsilon_i\boldsymbol{r}_i^{[1]}\odot\Delta_i\right\rangle\right]\\&\leqslant B_1E_\epsilon\left[\sup_{\boldsymbol{w}}\left\langle\frac{\boldsymbol{w}^{[1]}}{\|\boldsymbol{w}^{[1]}\|_1},\frac{1}{n}\sum_{i=1}^n\epsilon_i\boldsymbol{r}_i^{[1]}\odot\Delta_i\right\rangle\right],\end{aligned}$$

where $\Delta_i=\Psi(\boldsymbol{w}_1^{[0]}\odot\boldsymbol{r}_{i1}^{[0]},\ldots,\boldsymbol{w}_m^{[0]}\odot\boldsymbol{r}_{im}^{[0]},\boldsymbol{x}_i)$ and Ψ is defined by Eq. (15) and the inequality holds from $\|\boldsymbol{w}^{[1]}\|_1\leqslant B$. From Lemma 1, we have

$$\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})},S_n,RS_n)\leqslant\frac{B_1}{n}E_\epsilon\sup_{\boldsymbol{w}_1^{[0]}}\sum_{i=1}^n\epsilon_ir_{i1}^{[1]}\sigma(\langle\boldsymbol{w}_1^{[0]}\odot\boldsymbol{r}_{i1}^{[0]},\boldsymbol{x}_i\rangle). \tag{16}$$

From $\sigma(0)=0$ and $r_{i1}^{[1]}\in\{0,1\}$, we have $r_{i1}^{[1]}\sigma(t)=\sigma(r_{i1}^{[1]}t)$. Since $\sigma(\cdot)$ is Lipschitz with constant L, Lemma 2 gives

$$E_\epsilon\sup_{\boldsymbol{w}_1^{[0]}}\sum_{i=1}^n\epsilon_ir_{i1}^{[1]}\sigma(\langle\boldsymbol{w}_1^{[0]}\odot\boldsymbol{r}_{i1}^{[0]},\boldsymbol{x}_i\rangle)\leqslant LE_\epsilon\sup_{\boldsymbol{w}_1^{[0]}}\sum_{i=1}^n\epsilon_i\langle\boldsymbol{w}_1^{[0]}\odot\boldsymbol{r}_{ij}^{[0]},r_{i1}^{[1]}\boldsymbol{x}_i\rangle.$$

Similarly to the proof of Eq. (14), we have

$$E_\epsilon\sup_{\boldsymbol{w}_1^{[0]}}\sum_{i=1}^n\epsilon_i\langle\boldsymbol{w}_1^{[0]},r_{i1}^{[1]}\boldsymbol{x}_i\odot\boldsymbol{r}_{i1}^{[0]}\rangle=B_0\left(\sum_{i=1}^n\langle r_{i1}^{[1]}\boldsymbol{x}_i\odot\boldsymbol{r}_{i1}^{[0]},r_{i1}^{[1]}\boldsymbol{x}_i\odot\boldsymbol{r}_{i1}^{[0]}\rangle\right)^{\frac{1}{2}}.$$

Combining with the previous analysis, we have

$$\begin{aligned}\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}) &= E[\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}, S_n, RS_n)] \\ &\leqslant \frac{LB_1B_0}{n} E_{S_n,RS_n}\Big(\sum_{i=1}^n \langle r_{i1}^{[1]}\boldsymbol{x}_i \odot \boldsymbol{r}_{i1}^{[0]}, r_{i1}^{[1]}\boldsymbol{x}_i \odot \boldsymbol{r}_{i1}^{[0]}\rangle\Big)^{\frac{1}{2}} \\ &\leqslant \frac{LB_1B_0}{n} E_{S_n}\sum_{i=1}^n E_{RS_n}\langle r_{i1}^{[1]}\boldsymbol{x}_i \odot \boldsymbol{r}_i^{[0]}, r_{i1}^{[1]}\boldsymbol{x}_i \odot \boldsymbol{r}_i^{[0]}\rangle \leqslant LB_1B_0\hat{B}\rho/\sqrt{n},\end{aligned}$$

where the last inequality holds from Eq. (6) and $\|\boldsymbol{x}_i\| \leqslant \hat{B}$.

In a similar way, we can prove $\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{III})}) \leqslant B_1B_0\hat{B}\rho^2/\sqrt{n}$ by combining with Eqs. (6) and (7), and $\langle \boldsymbol{w} \odot \boldsymbol{r}_1, \boldsymbol{x} \odot \boldsymbol{r}_2\rangle = \langle \boldsymbol{w}, \boldsymbol{x} \odot \boldsymbol{r}_1 \odot \boldsymbol{r}_2\rangle$. This completes the proof.

4.3 Deep network with k hidden layers

Now we consider the neural network with k ($k \geqslant 1$) hidden layers, and the ith layer has m_i hidden units ($i \in [k]$). The output for this neural network is given by

$$\begin{aligned}&f(\boldsymbol{w}, \boldsymbol{x}) = \langle \boldsymbol{w}_1^{[k]}, \Psi_k\rangle \text{ with } \Psi_0 = \boldsymbol{x}, \text{ and for } i \in [k], \\ &\Psi_i = (\sigma(\langle \boldsymbol{w}_1^{[i-1]}, \Psi_{i-1}\rangle), \ldots, \sigma(\langle \boldsymbol{w}_{m_i}^{[i-1]}, \Psi_{i-1}\rangle)),\end{aligned}$$

and three types of dropouts $\mathrm{Drp}^{(\mathrm{I})}$, $\mathrm{Drp}^{(\mathrm{II})}$ and $\mathrm{Drp}^{(\mathrm{III})}$ are defined by Eqs. (8)–(10). The following theorem shows the Rademecher complexity for three types of dropouts.

Theorem 4. Let $\mathcal{W} = \{(\boldsymbol{w}_1^{[k]}, \boldsymbol{w}_1^{[k-1]}, \ldots, \boldsymbol{w}_{m_k}^{[k-1]}, \ldots, \boldsymbol{w}_1^{[0]}, \ldots, \boldsymbol{w}_{m_2}^{[0]})\colon \|\boldsymbol{w}_i^{[0]}\| \leqslant B_0, \|\boldsymbol{w}_i^{[j]}\|_1 \leqslant B_j \text{ for } j \geqslant 1\}$, $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^d\colon \|\boldsymbol{x}\| \leqslant \hat{B}\}$, and $\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})}$, $\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}$, and $\mathcal{F}_{\mathcal{W}}^{(\mathrm{III})}$ are defined in Eqs. (11)–(13). Suppose that the activation σ is Lipschitz with constant L and $\sigma(0) = 0$. Then, we have

$$\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})}) \leqslant \mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}) \leqslant \frac{\rho^{(k+1)/2}}{\sqrt{n}} L^k\hat{B}\prod_{j=0}^k B_j \quad \text{and} \quad \mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{III})}) \leqslant \frac{\rho^{(k+1)}}{\sqrt{n}} L^k\hat{B}\prod_{j=0}^k B_j.$$

Here the Lipschitz constant L is dependent on the activation function, e.g., $L \leqslant 1$ if we choose the center sigmoid and relu [33] as activation function, and this follows $L\rho < 1$ for $\rho < 1$, which could improve the generalization bounds.

This theorem shows that dropout can lead to an exponential reduction of the Rademacher complexity with respect to the number of hidden layers within neural network. If we do not drop out any weights and units (including hidden units, or input units corresponding to input features), i.e., $\rho = 1$, the above theorem improves the results in [35, Lemma 26]. The Rademacher complexities are dependent on the norms of weights, but irrelevant to the number of units and weights in the network, as well as the dimension of input datasets.

Previous empirical studies showed that dropout tends to be resistant to overfitting in many real applications [7, 15, 16, 19]. This theoretical result discloses the reason that dropout makes an exponential reduction of the Rademacher complexity, i.e., dropout can reduce the model complexity so as to avoid overfitting.

Finally, this theorem is also consistent with the empirical experience of training deep network1), e.g., multi-layer network with multi-layers of dropout performs not worse than that of single layer of dropout. An interesting open problem is to study the convergence rate of learning algorithms such as SGD for multi-layer dropout network.

Proof. We first have

$$\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{I})}) \leqslant \mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})})$$

from $f^{(\mathrm{II})}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{r}) = f^{(\mathrm{I})}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{r}')$ by selecting $\boldsymbol{r}_1^{[i]} = \cdots = \boldsymbol{r}_{m_{i+1}}^{[i]} = \boldsymbol{r}'^{[i]}$ for $0 \leqslant i \leqslant k-1$ and $\boldsymbol{r}_1^{[k]} = \boldsymbol{r}'^{[k]}_1$.

1) This is devoted to one reviewer.

For any $S_n = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ and $RS_n = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_n\}$, we will prove that

$$\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}, S_n, RS_n) \leqslant \frac{L^k}{n} E_\epsilon\left[\sup_{\boldsymbol{w}_1^{[0]}} \sum_{i=1}^n \epsilon_i \Big\langle \boldsymbol{w}_1^{[0]} \odot \boldsymbol{r}_{i,1}^{[0]}, \boldsymbol{x}_i \prod_{s=1}^k r_{i,j_{s+1},j_s}^{[s]} \Big\rangle\right] \prod_{j=1}^k B_j \tag{17}$$

by induction on k, i.e., the number of layers in neural network, where $j_{k+1} = 1$. It is easy to find the above holds for $k = 1$ from Eq. (16). Assume that Eq. (17) holds for neural network of $k-1$ layers ($k \geqslant 2$), and in the following we will prove for neural network of k layers.

For $\|\boldsymbol{w}_1^{[k]}\|_1 \leqslant B_k$, we have

$$\begin{aligned}\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}, S_n, RS_n) &= \frac{1}{n} E_\epsilon \sup_{\boldsymbol{w}} \sum_{i=1}^n \epsilon_i \langle \boldsymbol{w}_1^{[k]} \odot \boldsymbol{r}_{i1}^{[k]}, \Psi_{ik}\rangle \\ &= \frac{1}{n} E_\epsilon \sup_{\boldsymbol{w}} \left\langle \boldsymbol{w}_1^{[k]}, \sum_{i=1}^n \epsilon_i \Psi_{ik} \odot \boldsymbol{r}_{i1}^{[k]} \right\rangle \leqslant \frac{B_k}{n} E_\epsilon \sup_{\boldsymbol{w}} \left\langle \frac{\boldsymbol{w}_1^{[k]}}{\|\boldsymbol{w}_1^{[k]}\|_1}, \sum_{i=1}^n \epsilon_i \Psi_{ik} \odot \boldsymbol{r}_{i1}^{[k]} \right\rangle,\end{aligned}$$

where $\Psi_{ij} = (\sigma(\langle \boldsymbol{w}_1^{[j-1]} \odot \boldsymbol{r}_{i1}^{[j-1]}, \Psi_{i,j-1}\rangle), \ldots, \sigma(\langle \boldsymbol{w}_{m_j}^{[j-1]} \odot \boldsymbol{r}_{i,m_j}^{[j-1]}, \Psi_{i,j-1}\rangle))$ for $j \in [k]$ and $\Psi_{i0} = \boldsymbol{x}_i$. From Lemma 1, we have

$$E_\epsilon \sup_{\boldsymbol{w}} \left\langle \frac{\boldsymbol{w}_1^{[k]}}{\|\boldsymbol{w}_1^{[k]}\|_1}, \sum_{i=1}^n \epsilon_i \Psi_{ik} \odot \boldsymbol{r}_{i1}^{[k]} \right\rangle \leqslant E_\epsilon \sup_{\boldsymbol{w}} \sum_{i=1}^n \epsilon_i r_{i,1,1}^{[k]} \times \sigma(\langle \boldsymbol{w}_1^{[k-1]} \odot \boldsymbol{r}_{i,1}^{[k-1]}, \Psi_{i,k-1}\rangle),$$

which yields that

$$\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}, S_n, RS_n) \leqslant \frac{B_k}{n} E_\epsilon \sup_{\boldsymbol{w}} \sum_{i=1}^n \epsilon_i r_{i,1,1}^{[k]} \times \sigma(\langle \boldsymbol{w}_1^{[k-1]} \odot \boldsymbol{r}_{i,1}^{[k-1]}, \Psi_{i,k-1}\rangle).$$

Since $\sigma(0) = 0$ and σ is Lipschitz with constant L, Lemma 2 gives

$$\hat{\mathfrak{R}}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}, S_n, RS_n) \leqslant \frac{LB_k}{n} E_\epsilon \sup_{\boldsymbol{w}} \sum_{i=1}^n \epsilon_i r_{i,1,1}^{[k]} \langle \boldsymbol{w}_1^{[k-1]} \odot \boldsymbol{r}_{i,1}^{[k-1]}, \Psi_{i,k-1}\rangle. \tag{18}$$

By using $r_{i,1,1}^{[k]}\sigma(t) = \sigma(r_{i,1,1}^{[k]} t)$, the term

$$\frac{1}{n} E_\epsilon \sup_{\boldsymbol{w}} \sum_{i=1}^n \epsilon_i r_{i,1,1}^{[k]} \langle \boldsymbol{w}_1^{[k-1]} \odot \boldsymbol{r}_{i,1}^{[k-1]}, \Psi_{i,k-1}\rangle$$

can be viewed as the empirical Rademacher for another $k-1$ layers neural network with respect to sample $S_n' = \{\boldsymbol{x}_1 r_{i,1,1}^{[k]}, \ldots, \boldsymbol{x}_n r_{i,1,1}^{[k]}\}$ and $RS_n' = (\boldsymbol{r}_1^{[k-1]}, \boldsymbol{r}_1^{[k-2]}, \ldots, \boldsymbol{r}_{m_{k-2}}^{[k-2]}, \boldsymbol{r}_1^{[0]}, \ldots, \boldsymbol{r}_{m_1}^{[0]})$. Therefore, by our assumption that Eq. (17) holds for any $k-1$ layers neural network, we have

$$\begin{aligned}&\frac{1}{n} E_\epsilon \sup_{\boldsymbol{w}} \sum_{i=1}^n \epsilon_i r_{i,1,1}^{[k]} \langle \boldsymbol{w}_1^{[k-1]} \odot \boldsymbol{r}_{i,1}^{[k-1]}, \Psi_{i,k-1}\rangle \\ &\quad\leqslant \frac{L^{k-1}\prod_{i=1}^{k-1} B_i}{n} E_\epsilon \sum_{i=1}^n \epsilon_i \left\langle \boldsymbol{w}_1^{[0]} \odot \boldsymbol{r}_{i,1}^{[0]}, \boldsymbol{x}_i \prod_{s=1}^{k-1} r_{i,j_{s+1},j_s}^{[s]} \right\rangle,\end{aligned}$$

which proves that Eq. (17) holds for k layers neural network by combining with Eq. (18).

Similarly to the proof of Theorem 2, the term in Eq. (17) can be further bounded by

$$\begin{aligned}&E_\epsilon \sup_{\boldsymbol{w}_1^{[0]}} \sum_{i=1}^n \epsilon_i \left\langle \boldsymbol{w}_1^{[0]} \odot \boldsymbol{r}_{i,1}^{[0]}, \boldsymbol{x}_i \prod_{s=1}^k r_{i,j_{s+1},j_s}^{[s]} \right\rangle \\ &\quad\leqslant B_0 \left(\sum_i \langle \boldsymbol{r}_{i,1}^{[0]} \odot \boldsymbol{x}_i \prod_{s=1}^k r_{i,j_{s+1},j_s}^{[s]}, \boldsymbol{r}_{i,1}^{[0]} \odot \boldsymbol{x}_i \prod_{s=1}^k r_{i,j_{s+1},j_s}^{[s]} \rangle \right)^{\frac{1}{2}},\end{aligned}$$

which yields that, from Eq. (6),

$$\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{II})}) \leqslant \frac{1}{\sqrt{n}} L^k \rho^{(k+1)/2} \hat{B} \prod_{i=0}^{k} B_i.$$

In a similar manner, we can prove that

$$\mathfrak{R}_n(\mathcal{F}_{\mathcal{W}}^{(\mathrm{III})}) \leqslant \frac{1}{\sqrt{n}} L^k \rho^{k+1} \hat{B} \prod_{i=0}^{k} B_i,$$

by using Eq. (7), and this completes the proof.

5 Conclusion

Deep neural networks have witnessed great successes in various real applications. Many implementation techniques have been developed, however, theoretical understanding of many aspects of deep neural networks is far from clear. Dropout is an effective strategy to improve the performance as well as reduce the influence of overfitting during training of deep neural network, and it is motivated from the intuition of preventing complex co-adaptation of feature detectors. In this work, we study the Rademacher complexity of different types of dropouts, and our theoretical results disclose that for shallow neural networks (with one or none hidden layer) dropout is able to reduce the Rademacher complexity in polynomial, whereas for deep neural networks it can amazingly lead to an exponential reduction of the Rademacher complexity. An interesting future work is to present tighter generalization bounds for dropouts. In this work, we focused on very fundamental types of dropouts. Analyzing other types of dropouts is another interesting issue for future work, and we believe that the current work sheds a light on the way for the analysis.

Acknowledgements This work was supported by National Basic Research Program of China (Grant No. 2014CB340501), National Natural Science Foundation of China (Grant No. 61333014) and Jiangsu Science Foundation (Grant No. BK20150586).

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks. Science, 2006, 313: 504–507
2 Cireşan D C, Meier U, Gambardella L M, et al. Deep, big, simple neural nets for handwritten digit recognition. Neural Comput, 2010, 22: 3207–3220
3 Cireşan D C, Meier U, Schmidhuber J. Multi-column deep neural networks for image classification. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Providence, 2012. 3642–3649
4 Coates A, Huval B, Wang T, et al. Deep learning with COTS HPC systems. In: Proceedings of the 30th International Conference on Machine Learning, Atlanta, 2013. 1337–1345
5 Dahl G E, Sainath T N, Hinton G E. Improving deep neural networks for lvcsr using rectified linear units and dropout. In: Proceeding of IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, 2013. 8609–8613
6 Dahl G E, Yu D, Deng L, et al. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. IEEE Trans Audio, Speech, Language Process, 2012, 20: 30–42
7 Hinton G E, Deng L, Yu D. Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process Mag, 2012, 29: 82–97
8 Bo L, Lai K, Ren X, et al. Object recognition with hierarchical kernel descriptors. In: Proceeding of IEEE Conference on Computer Vision and Pattern Recognition, Colorado Springs, 2011. 1729–1736
9 Mobahi H, Collobert R, Weston J. Deep learning from temporal coherence in video. In: Proceedings of the 26th International Conference on Machine Learning, Montreal, 2009. 737–744
10 Liu Y J, Liu L, Tong S C. Adaptive neural network tracking design for a class of uncertain nonlinear discrete-time systems with dead-zone. Sci China Inf Sci, 2014, 57: 032206
11 Andreas S W, David E R, Bernardo A H. Generalization by weight-elimination with application to forecasting. In: Advances in Neural Information Processing Systems 3. Cambridge: MIT Press, 1991. 875–882

12 Amari S, Murata N, Muller K, et al. Asymptotic statistical theory of overtraining and cross-validation. IEEE Trans Neural Netw, 1997, 8: 985–996

13 Neal R M. Bayesian learning for neural networks. In: Lecture Notes in Statistics. New York: Springer, 1996

14 Hinton G E, Srivastava N, Krizhevsky A, et al. Improving neural networks by preventing co-adaptation of feature detectors. arXiv:1207.0580, 2012

15 Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25. Cambridge: MIT Press, 2012. 1106–1114

16 Wan L, Zeiler M, Zhang S, et al. Regularization of neural networks using dropconnect. In: Proceedings of the 30th International Conference on Machine Learning, Atlanta, 2013. 1058–1066

17 Wang S I, Manning C D. Fast dropout training. In: Proceedings of the 30th International Conference on Machine Learning, Atlanta, 2013. 118–126

18 Ba J, Frey B. Adaptive dropout for training deep neural networks. In: Advances in Neural Information Processing Systems 26. Cambridge: MIT Press, 2013. 3084–3092

19 Srivastava N, Hinton G, Krizhevsky A. Dropout: a simple way to prevent neural networks from overfitting. J Mach Lear Res, 2014, 15: 1929–1958

20 Baldi P, Sadowski P J. The dropout learning algorithm. Artif Intel, 2014, 210: 78–122

21 Wager S, Wang S, Liang P. Dropout training as adaptive regularization. In: Advances in Neural Information Processing Systems 26. Cambridge: MIT Press, 2013. 351–359

22 McAllester D. A pac-bayesian tutorial with a dropout bound. arXiv:1307.2118, 2013

23 Karpinski M, Macintyre A. Polynomial bounds for VC dimension of sigmoidal and general pfaffian neural networks. J Comput Syst Sci, 1997, 54: 169–176

24 Anthony M, Bartlett P L. Neural Network Learning: Theoretical Foundations. Cambridge: Cambridge University Press, 2009

25 Bartlett P L, Mendelson S. Rademacher and gaussian complexities: risk bounds and structural results. J Mach Lear Res, 2002, 3: 463–482

26 Koltchinskii V, Panchenko D. Empirical margin distributions and bounding the generalization error of combined classifiers. Ann Stat, 2002, 30: 1–50

27 Cortes C, Mohri M, Rostamizadeh A. Generalization bounds for learning kernels. In: Proceedings of the 27th International Conference on Machine Learning, Haifa, 2010. 247–254

28 Maurer A. Bounds for linear multi-task learning. J Mach Lear Res, 2006 7: 117–139

29 Meir R, Zhang T. Generalization error bounds for bayesian mixture algorithms. J Mach Lear Res, 2003, 4: 839–860

30 Zou B, Peng Z M, Xu Z B. The learning performance of support vector machine classification based on Markov sampling. Sci China Inf Sci, 2013, 56: 032110

31 McDiarmid C. On the method of bounded differences. In: Surveys in Combinatorics. Cambridge: Cambridge University Press, 1989. 148–188

32 Ledoux M, Talagrand, M. Probability in Banach Spaces: Isoperimetry and Processes. Berlin: Springer, 2002

33 Nair V, Hinton G E. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning, Haifa, 2010. 807–814

34 Kakade S M, Sridharan K, Tewari A. On the complexity of linear prediction: risk bounds, margin bounds, and regularization. In: Advances in Neural Information Processing Systems 24. Cambridge: MIT Press, 2008. 351–359

35 Bartlett P L. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. IEEE Trans Inf Theory, 1998, 44: 525–536