Dropout Training for Support Vector Machines

Ning Chen Jun Zhu Jianfei Chen Bo Zhang
State Key Lab of Intelligent Tech. & Systems; Tsinghua National TNList Lab;
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
{ningchen@mail, dcszj@mail, chenjf10@mails, dcszb@mail}.tsinghua.edu.cn

Abstract

Dropout and other feature noising schemes have shown promising results in controlling over-fitting by artificially corrupting the training data. Though extensive theoretical and empirical studies have been performed for generalized linear models, little work has been done for support vector machines (SVMs), one of the most successful approaches for supervised learning. This paper presents dropout training for linear SVMs. To deal with the intractable expectation of the non-smooth hinge loss under corrupting distributions, we develop an iteratively re-weighted least square (IRLS) algorithm by exploring data augmentation techniques. Our algorithm iteratively minimizes the expectation of a re-weighted least square problem, where the re-weights have closed-form solutions. The similar ideas are applied to develop a new IRLS algorithm for the expected logistic loss under corrupting distributions. Our algorithms offer insights on the connection and difference between the hinge loss and logistic loss in dropout training. Empirical results on several real datasets demonstrate the effectiveness of dropout training on significantly boosting the classification accuracy of linear SVMs.

Introduction

Artificial feature noising augments the finite training data with an infinite number of corrupted versions, by corrupting the given training examples with a fixed noise distribution. Among the many noising schemes, dropout training (Hinton et al. 2012) is an effective way to control over-fitting by randomly omitting subsets of features at each iteration of a training procedure. By formulating the feature noising methods as minimizing the expectation of some loss functions under the corrupting distributions, recent work has provided theoretical understandings of such schemes from the perspective of adaptive regularization (Wager, Wang, and Liang 2013); and has shown promising empirical results in various applications, including document classification (van der Maaten et al. 2013; Wager, Wang, and Liang 2013), named entity recognition (Wang et al. 2013), and image classification (Wang and Manning 2013).

Regarding the loss functions, though much work has been done on the quadratic loss, logistic loss, or the log-loss induced from a generalized linear model (GLM) (van der Maaten et al. 2013; Wager, Wang, and Liang 2013; Wang et al. 2013), little work has been done on the margin-based hinge loss underlying the very successful support vector machines (SVMs) (Vapnik 1995). One technical challenge is that the non-smoothness of the hinge loss makes it hard to compute or even approximate its expectation under a given corrupting distribution. Existing methods are not directly applicable, therefore calling for new solutions. This paper attempts to address this challenge and fill up the gap by extending dropout training as well as other feature noising schemes to support vector machines.

Previous efforts on learning SVMs with feature noising have been devoted to either explicit corruption or an adversarial worst-case analysis. For example, virtual support vector machines (Burges and Scholkopf 1997) explicitly augment the training data, which are usually support vectors from previous learning iterations for computational efficiency, with additional examples that are corrupted through some invariant transformation models. A standard SVM is then learned on the corrupted data. Though simple and effective, such an approach lacks elegance and the computational cost of processing the additional corrupted examples could be prohibitive for many applications. The other work (Globerson and Roweis 2006; Dekel and Shamir 2008; Teo et al. 2008) adopts an adversarial worst-case analysis to improve the robustness of SVMs against feature deletion in testing data. Though rigorous in theory, a worst-case scenario is unlikely to be encountered in practice. Moreover, the worst-case analysis usually results in solving a complex and computationally demanding problem.

In this paper, we show that it is efficient to train linear SVM predictors on an infinite amount of corrupted copies of the training data by marginalizing out the corruption distributions, an average-case analysis. We concentrate on dropout training, but the results are directly applicable to other noising models, such as Gaussian, Poisson and Laplace (van der Maaten et al. 2013). For all these noising schemes, the resulting expected hinge loss can be upper-bounded by a variational objective by introducing auxiliary variables, which follow a generalized inverse Gaussian distribution. We then develop an iteratively re-weighted least square (IRLS) algorithm to minimize the variational bounds. At each iteration, our algorithm minimizes the expectation of a re-weighted quadratic loss under the given corrupting distribution, where the re-weights are computed in a simple closed form. We further apply the similar ideas to develop a new IRLS algorithm for the dropout training of logistic regression, which extends the well-known IRLS algorithm for standard logistic regression (Hastie, Tibshirani, and Friedman 2009). Our IRLS algorithms shed light on the connection and difference between the hinge loss and logistic loss in the context of dropout training, complementing to the previous analysis (Rosasco et al. 2004; Globerson et al. 2007) in the supervised learning settings. Finally, empirical results on classification and a challenging "nightmare at test time" scenario (Globerson and Roweis 2006) demonstrate the effectiveness of our approaches, in comparison with various strong competitors.

Preliminaries

We setup the problem in question and review the learning with marginalized corrupted features.

Regularized loss minimization

Consider the binary classification, where each training example is a pair $(\mathbf{x}, y)$ with $\mathbf{x} \in \mathbb{R}^D$ being an input feature vector and $y \in \{+1, -1\}$ being a binary label. Given a set of training data $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$, supervised learning aims to find a function $f \in \mathcal{F}$ that maps each input to a label. To find the optimal candidate, it commonly solves a regularized loss minimization problem

$$\min_{f \in \mathcal{F}} \Omega(f) + 2c \cdot \mathcal{R}(\mathcal{D}; f), \tag{1}$$

where $\mathcal{R}(\mathcal{D}; f)$ is the risk of applying f to the training data; $\Omega(f)$ is a regularization term to control over-fitting; and c is a non-negative regularization parameter.

For linear models, the function f is simply parameterized as $f(\mathbf{x}; \mathbf{w}, b) = \mathbf{w}^\top \mathbf{x} + b$, where $\mathbf{w}$ is the weight vector and b is an offset. We will denote $\boldsymbol{\theta} := \{\mathbf{w}, b\}$ for clarity. Then, the regularization can be any Euclidean norms[1], e.g., the ℓ_2-norm, $\Omega(\mathbf{w}) = \|\mathbf{w}\|_2^2$, or the ℓ_1-norm, $\Omega(\mathbf{w}) = \|\mathbf{w}\|_1$. For the loss functions, the most relevant measure is the training error, $\sum_{n=1}^N \delta(f(\mathbf{x}_n; \boldsymbol{\theta}) \neq y_n)$, which however is not easy to optimize. A convex surrogate loss is used instead, which normally upper bounds the training error. Two popular examples are the hinge loss and logistic loss[2]:

$$\mathcal{R}_h(\mathcal{D}; \boldsymbol{\theta}) = \sum_{n=1}^N \max\left(0, \ell - y_n f(\mathbf{x}_n; \boldsymbol{\theta})\right),$$

$$\mathcal{R}_l(\mathcal{D}; \boldsymbol{\theta}) = \sum_{n=1}^N \left(-\log p(y_n | \mathbf{x}_n, \boldsymbol{\theta})\right),$$

where $\ell (> 0)$ is the cost of making a wrong prediction, and $p(y_n|\mathbf{x}_n, \boldsymbol{\theta}) := 1/(1 + \exp(-y_n f(\mathbf{x}_n; \boldsymbol{\theta})))$ is the logistic likelihood. Other losses include the quadratic loss, $\sum_{n=1}^N (f(\mathbf{x}_n; \boldsymbol{\theta}) - y_n)^2$, and the exponential loss, $\sum_{n=1}^N \exp(-y_n f(\mathbf{x}_n; \boldsymbol{\theta}))$, whose feature noising analyses are relatively simpler (van der Maaten et al. 2013).

[1]It is a common practice to not regularize the offset.

[2]The natural logarithm is not an upper bound of the training error. We can simply change the base without affecting learning.

Learning with marginalized corruption

Let $\tilde{\mathbf{x}}$ be the corrupted version of the input features $\mathbf{x}$. Consider the commonly used independent corrupting model:

$$p(\tilde{\mathbf{x}}|\mathbf{x}) = \prod_{d=1}^D p(\tilde{x}_d | x_d; \eta_d),$$

where each individual distribution is a member of the exponential family, with the natural parameter η_d. Another common assumption is that the corrupting distribution is unbiased, that is, $\mathbb{E}_p[\tilde{\mathbf{x}}|\mathbf{x}] = \mathbf{x}$, where we use $\mathbb{E}_p[\cdot] := \mathbb{E}_{p(\tilde{\mathbf{x}}|\mathbf{x})}[\cdot]$ to denote the expectation taken over the corrupting distribution $p(\tilde{\mathbf{x}}|\mathbf{x})$. Such examples include the unbiased blankout (or dropout) noise, Gaussian noise, Laplace noise, and Poisson noise (Vincent et al. 2008; van der Maaten et al. 2013).

For the *explicit corruption* in (Burges and Scholkopf 1997), each example $(\mathbf{x}_n, y_n)$ is corrupted M times from the corrupting model $p(\tilde{\mathbf{x}}_n|\mathbf{x}_n)$, resulting in the corrupted examples $(\tilde{\mathbf{x}}_{nm}, y_n)$, $m \in [M]$. This procedure generates a new corrupted data set $\tilde{\mathcal{D}}$ with a larger size of NM. The generated dataset can be trained by minimizing the average loss function over M corrupted data points:

$$\mathcal{L}(\tilde{\mathcal{D}}; \boldsymbol{\theta}) = \sum_{n=1}^N \frac{1}{M} \sum_{m=1}^M \mathcal{R}(\tilde{\mathbf{x}}_{nm}, y_n; \boldsymbol{\theta}), \tag{2}$$

where $\mathcal{R}(\mathbf{x}, y; \boldsymbol{\theta})$ is the loss function of the model incurred on the training example $(\mathbf{x}, y)$. As $\mathcal{L}(\tilde{\mathcal{D}}; \boldsymbol{\theta})$ scales linearly with the number of corrupted observations, this approach may suffer from high computational costs.

Dropout training adopts the strategy of *implicit corruption*, which learns the model with marginalized corrupted features by minimizing the expectation of a loss function under the corrupting distribution

$$\mathcal{L}(\mathcal{D}; \boldsymbol{\theta}) = \sum_{n=1}^N \mathbb{E}_p[\mathcal{R}(\tilde{\mathbf{x}}_n, y_n; \boldsymbol{\theta})]. \tag{3}$$

The objective can be seen as a limit case of (2) when $M \to \infty$, by the law of large numbers. Such an expectation scheme has been widely adopted in previous work (Wager, Wang, and Liang 2013; van der Maaten et al. 2013; Wang et al. 2013; Wang and Manning 2013).

The choice of the loss function $\mathcal{R}$ in (3) can make a significant difference, in terms of computation cost and prediction accuracy. Previous work on feature noising has covered the quadratic loss, exponential loss, logistic loss, and the loss induced from generalized linear models (GLM). For the quadratic loss and exponential loss, the expectation in Eq. (3) can be computed analytically, thereby leading to simple gradient descent algorithms (van der Maaten et al. 2013). However, it does not have a closed form to compute the expectation for the logistic loss or the GLM loss. Previous analysis has resorted to approximation methods, such as using the second-order Taylor expansion (Wager, Wang, and Liang 2013) or an upper bound by applying Jensen's inequality (van der Maaten et al. 2013), both of which lead to effective algorithms in practice. In contrast, little work has been done on the hinge loss, for which the expectation under corrupting distributions cannot be analytically computed either, therefore calling for new algorithms.

Learning SVMs with Corrupting Noise

We now present a simple iteratively re-weighted least square (IRLS) algorithm to learn SVMs with the expected hinge loss under corrupting distributions. Our method consists of a variational upper bound of the expected loss and a simple algorithm that iteratively minimizes an expectation of a re-weighted quadratic loss. We also apply the similar ideas to develop a simple IRLS algorithm for minimizing the expected logistic loss, thereby allowing for a systematical comparison of the hinge loss with the logistic and quadratic losses in the context of feature noising.

A variational bound with data augmentation

Let $\zeta_n := \ell - y_n(\mathbf{w}^\top \tilde{\mathbf{x}}_n)$.[3] Then, the expected hinge loss can be written as

$$\mathcal{R}_h(\mathcal{D};\boldsymbol{\theta}) = \sum_{n=1}^{N} \mathbb{E}_p[\max(0, \zeta_n)], \tag{4}$$

Since we do not have a closed-form of the expectation of the max function, minimizing the expected loss (4) is intractable. Here, we derive a variational upper bound based on a data augmentation formulation of the expected hinge loss. Let $\phi(y_n|\tilde{\mathbf{x}}_n, \boldsymbol{\theta}) = \exp\{-2c\max(0, \zeta_n)\}$ be the pseudo-likelihood of the response variable for sample n. Then we have

$$\mathcal{R}_h(\mathcal{D};\boldsymbol{\theta}) = -\frac{1}{2c}\sum_n \mathbb{E}_p[\log\phi(y_n|\tilde{\mathbf{x}}_n, \boldsymbol{\theta})]. \tag{5}$$

Using the ideas of data augmentation (Polson and Scott 2011; Zhu et al. 2014), the pseudo-likelihood can be expressed as

$$\phi(y_n|\tilde{\mathbf{x}}_n, \boldsymbol{\theta}) = \int_0^\infty \frac{1}{\sqrt{2\pi\lambda_n}} \exp\left\{-\frac{(\lambda_n + c\zeta_n)^2}{2\lambda_n}\right\} d\lambda_n, \tag{6}$$

where $\lambda_n,\ n \in [N]$, is the augmented variable. Using (6) and Jensen's inequality, we can derive a variational upper bound $\mathcal{L}$ of the expected hinge loss as

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, q(\boldsymbol{\lambda})) = \sum_{n=1}^{N} \Big\{ &- H(\lambda_n) + \frac{1}{2}\mathbb{E}_q[\log\lambda_n] \\ &+ \mathbb{E}_q\Big[\frac{1}{2\lambda_n}\mathbb{E}_p(\lambda_n + c\zeta_n)^2\Big]\Big\} + \text{constant},\end{aligned} \tag{7}$$

where $H(\lambda_n)$ is the entropy of the variational distribution $q(\lambda_n)$; $q(\boldsymbol{\lambda}) := \prod_n q(\lambda_n)$ is joint distribution; and we have defined $\mathbb{E}_q[\cdot] := \mathbb{E}_{q(\boldsymbol{\lambda})}[\cdot]$ to denote the expectation taken over a variational distribution q. Now, our variational optimization problem is

$$\min_{\boldsymbol{\theta}, q(\boldsymbol{\lambda})\in\mathcal{P}} \|\mathbf{w}\|_2^2 + \mathcal{L}(\boldsymbol{\theta}, q(\boldsymbol{\lambda})), \tag{8}$$

where $\mathcal{P}$ is the simplex space of normalized distributions. We should note that when there is no feature noise (i.e., $\tilde{\mathbf{x}} = \mathbf{x}$), the bound is tight and we are learning the standard SVM classifier. Please see Appendix A for the derivation. We will empirically compare with SVM in experiments.

[3]We treat the offset b implicitly by augmenting $\mathbf{x}_n$ and $\tilde{\mathbf{x}}_n$ with one dimension of deterministic 1. More details will be given in the algorithm.

Iteratively Re-weighted Least Square Algorithm

In the upper bound, we note that when the variational distribution $q(\boldsymbol{\lambda})$ is given, the term $\mathbb{E}_p[(\lambda_n + c\zeta_n)^2]$ is an expectation of a quadratic loss, which can be analytically computed. We leverage such a nice property and develop a coordinate descent algorithm to solve problem (8). Our algorithm iteratively solves the following two steps, analogous to the common two-step procedure of a variational EM algorithm.

For $q(\boldsymbol{\lambda})$ (i.e., E-step): infer the variational distribution $q(\boldsymbol{\lambda})$. Specifically, optimize $\mathcal{L}$ over $q(\boldsymbol{\lambda})$, we get:

$$\begin{aligned} q(\lambda_n) &\propto \frac{1}{\sqrt{\lambda_n}}\exp\left\{-\frac{1}{2}\left(\lambda_n + \frac{c^2\mathbb{E}_p[\zeta_n^2]}{\lambda_n}\right)\right\} \\ &\sim \mathcal{GIG}\left(\lambda_n; \frac{1}{2}, 1, c^2\mathbb{E}_p[\zeta_n^2]\right), \end{aligned} \tag{9}$$

where the second-order expectation is

$$\begin{aligned}\mathbb{E}_p[\zeta_n^2] = \mathbf{w}^\top(&\mathbb{E}_p[\tilde{\mathbf{x}}_n]\mathbb{E}_p[\tilde{\mathbf{x}}_n]^\top + V_p[\tilde{\mathbf{x}}_n])\mathbf{w} \\ &-2\ell y_n\mathbf{w}^\top\mathbb{E}_p[\tilde{\mathbf{x}}_n] + \ell^2;\end{aligned} \tag{10}$$

and $V_p[\tilde{\mathbf{x}}_n]$ is a $D\times D$ diagonal matrix with the dth diagonal element being the variance of $\tilde{x}_{nd}$, under the corrupting distribution $p(\tilde{\mathbf{x}}_n|\mathbf{x}_n)$. We have denoted $\mathcal{GIG}(x;p,a,b) \propto x^{p-1}\exp(-\frac{1}{2}(\frac{b}{x} + ax))$ as a generalized inverse Gaussian distribution. Thus, λ_n^{-1} follows an inverse Gaussian distribution

$$q(\lambda_n^{-1}|\tilde{\mathbf{x}}_n, \boldsymbol{\theta}) \sim \mathcal{IG}\left(\lambda_n^{-1}; \frac{1}{c\sqrt{\mathbb{E}[\zeta_n^2]}}, 1\right) \tag{11}$$

For $\boldsymbol{\theta} := \mathbf{w}$ (i.e., M-step): removing irrelevant terms, this step involves minimizing the following objective:

$$\mathcal{L}_{[\boldsymbol{\theta}]} = \|\mathbf{w}\|_2^2 + \sum_{n=1}^{N}\mathbb{E}_p\left[c\zeta_n + \frac{c^2}{2}\gamma_n\zeta_n^2\right], \tag{12}$$

where $\gamma_n := \mathbb{E}_q[\lambda_n^{-1}]$. We observe that this substep is equivalent to minimizing the expectation of a re-weighted quadratic loss, as summarized in Lemma 1, whose proof is deferred to Appendix B, for brevity.

Lemma 1. *Given $q(\boldsymbol{\lambda})$, the M-step minimizes the re-weighted quadratic loss (with the ℓ_2-norm regularizer):*

$$\|\mathbf{w}\|_2^2 + \frac{c^2}{2}\sum_n \gamma_n\mathbb{E}_p\left[(\mathbf{w}^\top\tilde{\mathbf{x}}_n - y_n^h)^2\right], \tag{13}$$

where $y_n^h = (\ell + \frac{1}{c\gamma_n})y_n$ is the re-weighted label, and the re-weights are computed in closed-form:

$$\gamma_n := \mathbb{E}_q[\lambda_n^{-1}] = \frac{1}{c\sqrt{\mathbb{E}_p[\zeta_n^2]}}. \tag{14}$$

For low-dimensional data, we can solve for the closed form solutions by doing matrix inversion. Specifically, optimizing $\mathcal{L}_{[\boldsymbol{\theta}]}$ over $\mathbf{w}$, we get[4]:

$$\mathbf{w} = \left(\frac{2}{c^2}I + \sum_{n=1}^{N}\gamma_n(\mathbb{E}_p[\tilde{\mathbf{x}}_n\tilde{\mathbf{x}}_n^\top])\right)^{-1}\left(\sum_{n=1}^{N}\gamma_n y_n^h\mathbb{E}_p[\tilde{\mathbf{x}}_n]\right),$$

[4]To consider offset, we simply augment $\mathbf{x}$ and $\tilde{\mathbf{x}}$ with an additional unit of 1. The variance $V_p[\tilde{\mathbf{x}}_n]$ is augmented accordingly. The identity matrix I is augmented by adding one zero row and one zero column.

where $\mathbb{E}_p[\tilde{\mathbf{x}}_n\tilde{\mathbf{x}}_n^\top] = \mathbb{E}_p[\tilde{\mathbf{x}}_n]\mathbb{E}_p[\tilde{\mathbf{x}}_n]^\top + V_p[\tilde{\mathbf{x}}_n]$. However, if the data are in a high-dimensional space, e.g., text documents, the above matrix inversion will be computationally expensive. In such cases, we can use numerical methods, e.g., the quasi-Newton method.

To summarize, our algorithm iteratively minimizes the expectation of a simple re-weighted quadratic loss under the given corrupting distribution, where the re-weights γ_n are computed in an analytic form. Therefore, it is an extension of the classical *iteratively re-weighted least square* (IRLS) algorithm (Hastie, Tibshirani, and Friedman 2009) for dropout training. We also observe that if we fix γ_n at $\frac{1}{c}$ and set $\ell = 0$, we are minimizing the quadratic loss under the corrupting distribution, as studied in (van der Maaten et al. 2013). We will empirically show that our iterative algorithm for the expected hinge-loss will consistently improve over the standard quadratic loss by adaptively updating γ_n. Finally, as we assume that the corrupting distribution is unbiased, i.e., $\mathbb{E}_p[\tilde{\mathbf{x}}|\mathbf{x}] = \mathbf{x}$, we only need to compute the variance of the corrupting distribution, which is easy for all the existing exponential family distributions. An overview of the variance of the commonly used corrupting distributions can be found in (van der Maaten et al. 2013).

An IRLS algorithm for the logistic Loss

We now extend the above ideas to develop a new IRLS algorithm for the logistic-loss, which also minimizes the expectation of a re-weighted quadratic loss under the corrupting distribution and computes the re-weights analytically.

Let $\omega_n := \mathbf{w}^\top\tilde{\mathbf{x}}_n$. Then the expected logistic loss under a corrupting distribution is

$$\mathcal{R}_l(\mathcal{D};\mathbf{w}) = -\sum_{n=1}^{N}\mathbb{E}_p\left[\log\left(\frac{e^{y_n\omega_n}}{1+e^{y_n\omega_n}}\right)\right]. \quad (15)$$

Again since the expectation cannot be computed in closed-form, we derive a variational bound as a surrogate. Specifically, let $\psi(y_n|\tilde{\mathbf{x}}_n,\mathbf{w}) = p^c(y_n|\tilde{\mathbf{x}}_n,\mathbf{w}) = \frac{e^{cy_n\omega_n}}{(1+e^{y_n\omega_n})^c}$ be the pseudo-likelihood of the response variable for sample n. We have $\mathcal{R}_l(\mathcal{D};\mathbf{w}) = -\frac{1}{c}\sum_n\mathbb{E}_p[\log\psi(y_n|\tilde{\mathbf{x}}_n,\mathbf{w})]$. Using the recent work of data augmentation (Polson, Scott, and Windle 2012; Chen et al. 2013), the pseudo-likelihood can be expressed as

$$\psi(y_n|\tilde{\mathbf{x}}_n,\mathbf{w}) = \frac{1}{2^c}e^{\kappa_n\omega_n}\int_0^\infty e^{-\frac{\lambda_n(y_n\omega_n)^2}{2}}p(\lambda_n)d\lambda_n, \quad (16)$$

where $\kappa_n := \frac{c}{2}y_n$ and λ_n is the augmented Polya-gamma variable, $p(\lambda_n) \sim \mathcal{PG}(\lambda_n; c, 0)$. Using (16), we can derive the upper bound of the expected logistic loss:

$$\mathcal{L}'(\mathbf{w},q(\boldsymbol{\lambda})) = \sum_{n=1}^{N}\left\{\frac{1}{2}\mathbb{E}_q[\lambda_n]\mathbb{E}_p[\omega_n^2] - H(\lambda_n) \quad (17)\right.$$
$$\left. -\mathbb{E}_q[\log p(\lambda_n)] - \frac{c}{2}y_n\mathbb{E}_p[\omega_n]\right\} + \text{constant},$$

and get the variational optimization problem

$$\min_{\mathbf{w},q(\boldsymbol{\lambda})\in\mathcal{P}} \|\mathbf{w}\|_2^2 + \mathcal{L}'(\mathbf{w},q(\boldsymbol{\lambda})), \quad (18)$$

Table 1: Comparison of hinge loss and logistic loss under the IRLS algorithmic framework.

	Parameter ℓ	Parameter c	Update γ_n	Update y_n
Hinge	ℓ	c	Eq. (14)	y_n^h
Logistic	–	c	Eq. (22)	y_n^l

where $q(\boldsymbol{\lambda})$ is the variational distribution

We solve the variational problem with a coordinate descent algorithm as follows:

For $q(\boldsymbol{\lambda})$ (i.e., E-step): optimizing $\mathcal{L}'$ over $q(\boldsymbol{\lambda})$, we have:

$$q(\lambda_n) \propto \exp\left(-\frac{1}{2}\lambda_n\mathbb{E}_p[\omega_n^2]\right)p(\lambda_n|c,0)$$
$$\sim \mathcal{PG}\left(\lambda_n; c, \sqrt{\mathbb{E}_p[\omega_n^2]}\right) \quad (19)$$

a Polya-Gamma distribution (Polson, Scott, and Windle 2012), where $\mathbb{E}_p[\omega_n^2] = \mathbf{w}^\top(\mathbb{E}_p[\tilde{\mathbf{x}}_n]\mathbb{E}_p[\tilde{\mathbf{x}}_n]^\top + V_p[\tilde{\mathbf{x}}_n])\mathbf{w}$.

For $\mathbf{w}$ (i.e., M-step): removing irrelevant terms, this step minimizes the objective

$$\mathcal{L}'_{[\mathbf{w}]} = \|\mathbf{w}\|_2^2 + \sum_{n=1}^{N}\frac{1}{2}\mathbb{E}_q[\lambda_n]\mathbb{E}_p[\omega_n^2] - \frac{c}{2}y_n\mathbb{E}_p[\omega_n]. \quad (20)$$

We then have the optimal solution[5]:

$$\mathbf{w} = \left(I + \frac{1}{2}\sum_{n=1}^{N}\mathbb{E}_q[\lambda_n]\mathbb{E}_p[\tilde{\mathbf{x}}_n\tilde{\mathbf{x}}_n^\top]\right)^{-1}\left(\frac{c}{4}\sum_{n=1}^{N}y_n\mathbb{E}_p[\tilde{\mathbf{x}}_n]\right).$$

This is actually equivalent to minimizing the expectation of a re-weighted quadratic loss, as in Lemma 2. The proof is similar to that of Lemma 1 and the expectation of a Polya-Gamma distribution follows (Polson, Scott, and Windle 2012).

Lemma 2. *Given $q(\boldsymbol{\lambda})$, the M-step minimizes the re-weighted quadratic loss (with the ℓ_2-norm regularizer)*

$$\|\mathbf{w}\|_2^2 + \frac{c}{2}\sum_n\gamma_n^l\mathbb{E}_p[(\mathbf{w}^\top\tilde{\mathbf{x}}_n - y_n^l)^2], \quad (21)$$

where $y_n^l = \frac{c}{2\gamma_n}y_n$ is the re-weighted label, and $\gamma_n^l = \frac{\gamma_n}{c}$ with

$$\gamma_n := \mathbb{E}_q[\lambda_n] = \frac{c}{2\sqrt{\mathbb{E}_p[\omega_n^2]}} \times \frac{e^{\sqrt{\mathbb{E}_p[\omega_n^2]}}-1}{1+e^{\sqrt{\mathbb{E}_p[\omega_n^2]}}}. \quad (22)$$

It can be observed that if we fix $\gamma_n = \frac{c}{2}$, the IRLS algorithm reduces to minimizing the expected quadratic loss under the corrupting distribution. This is similar as in the case with SVMs, where if we set $\ell = 0$ and fix $\gamma_n = \frac{1}{c}$, the IRLS algorithm for SVMs essentially minimizes the expected quadratic loss under the corrupting distribution. Furthermore, by sharing a similar iterative structure, our IRLS algorithms shed light on the similarity and difference between the hinge loss and the logistic loss, as summarized in Table 1. Specifically, both losses can be minimized via iteratively minimizing the expectation of a re-weighted quadratic loss, while they differ in the update rules of the weights γ_n and the labels y_n at each iteration.

[5]The offset can be similarly incorporated as in the hinge loss.

Experiments

We now present empirical results on both classification and the challenging "nightmare at test time" scenario (Globerson and Roweis 2006) to demonstrate the effectiveness of the dropout training algorithm for SVMs, denoted by Dropout-SVM, and the new IRLS algorithm for the dropout training of the logistic loss, denoted by Dropout-Logistic. We consider the unbiased dropout (or blankout) noise model[6], that is, $p(\tilde{\mathbf{x}} = 0) = q$ and $p(\tilde{\mathbf{x}} = \frac{1}{1-q}\mathbf{x}) = 1 - q$, where $q \in [0, 1)$ is a pre-specified corruption level. The variance of this model for each dimension d is $V_p[\tilde{x}_d] = \frac{q}{1-q}x_d^2$.

Binary classification

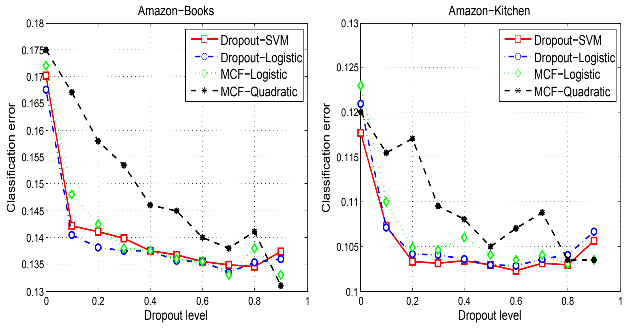
We first evaluate Dropout-SVM and Dropout-Logistic on binary classification tasks. We use the public Amazon book review and kitchen review datasets (Blitzer, Dredze, and Pereira 2007), which consist of the text reviews about books and kitchen, respectively. In both datasets, each document is represented as a 20,000 dimensional bag-of-words feature. The binary classification task is to distinguish whether a review content is positive or negative. Following the previous settings, we choose 2,000 documents for training and approximately 4,000 for testing.

We compare our methods with the methods presented in (van der Maaten et al. 2013) that minimize the quadratic loss with marginalized corrupted features (MCF), denoted by MCF-Quadratic, and that minimize the expected logistic loss, denoted by MCF-Logistic. MCF-Logistic was shown to be the state-of-the-art method for dropout training on these datasets, outperforming a wide range of competitors, including the dropout training of the exponential loss and the various loss functions with a Poisson noise model. As we have discussed, both Dropout-SVM and Dropout-Logistic iteratively minimize the expectation of a re-weighted quadratic loss, with the re-weights updated in closed-form. We include MCF-Quadratic as a baseline to demonstrate the effectiveness of our methods on adaptively tuning the re-weights to get improved results. We implement both Dropout-SVM and Dropout-Logistic using C++, and solve the re-weighted least square problems using L-BFGS methods (Liu and Nocedal 1989), which are very efficient by exploring the sparsity of bag-of-words features when computing gradients[7].

Figure 1 shows classification errors, where the results of MCF-Logistic and MCF-Quadratic are cited from (van der Maaten et al. 2013). We can see that on both datasets, Dropout-SVM and Dropout-Logistic generally outperform MCF-Quadratic except when the dropout level is larger than 0.9. In the meanwhile, the proposed two models give comparable results with (a bit better than on the kitchen dataset) the state-of-art MCF-Logistic which means that dropout training on SVMs is an effective strategy for binary classification. Finally, by noting that Dropout-SVM reduces to the standard SVM when the corruption level q is zero, we can see that dropout training can significantly boost the classification performance for the simple linear SVMs.

[6]Other noising models (e.g., Poisson) were shown to perform worse than the dropout model (van der Maaten et al. 2013). We have similar observations for Dropout-SVM and the new IRLS algorithm for logistic regression.

[7]We don't compare time with MCF methods, whose implementation are in Matlab (http://homepage.tudelft.nl/19j49/mcf/Marginalized_Corrupted_Features.html), slower than ours.



(a) books (b) kitchen

Figure 1: Classification errors on the Amazon datasets.

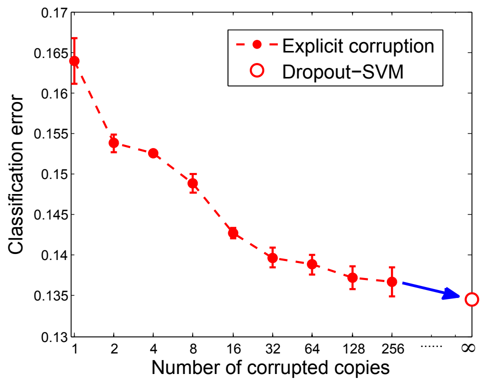


Figure 2: Comparison between Dropout-SVM and the explicit corruption for SVM on the Amazon-books datasets.

Dropout-SVM vs. Explicit corruption

Figure 2 shows the classification errors on the Amazon-books dataset when a SVM classifier is trained using the explicit corruption strategy as in Eq. (2). We change the number of corrupted copies (i.e., M) from 1 to 256. Following the previous setups (van der Maaten et al. 2013), for each value of M we choose the dropout model with q selected by cross-validation. The hyper-parameter of the SVM classifier is also chosen via cross-validation on the training data. We can observe a clear trend that the error decreases when the training set contains more corrupted versions of the original training data, i.e., M gets larger in Eq. (2). It also shows that the best performance is obtained when M approaches infinity, which is equivalent to our Dropout-SVM.

Multi-class classification

We also evaluate our methods on multiclass classification tasks. We choose the CIFAR-10 image categorization dataset[8]. The CIFAR-10 dataset is the subset of the 80 million tiny images (Torralba, Fergus, and Freeman 2008). It consists of 10 classes of 32×32 tiny images. We follow the experimental setup of the previous work (Krizhevsky 2009;

[8]http://www.cs.toronto.edu/∼kriz/cifar.html

Table 2: Classification errors on CIFAR-10 data set.

Model	No Corrupt	Poisson	Dropout
Dropout-SVM	0.322	0.309	0.294
Dropout-Logistic	0.312	0.302	0.293
MCF-Logistic	0.325	0.300	0.294
MCF-Quadratic	0.326	0.291	0.323

van der Maaten et al. 2013) and represent each image as a 8,192 dimensional feature descriptor. We use the same 50,000 images for training and 10,000 for testing. There are various approaches to applying the binary Dropout-SVM and Dropout-Logistic to multiclass classification, including "one-vs-all" and "one-vs-one" strategies. Here we choose "one-vs-all", which has shown effectiveness in many applications (Rifkin and Klautau 2004). The hyper-parameters are selected via cross-validation on the training set.

Table 2 presents the results, where the results of quadratic loss and logistic loss under the MCF learning setting[9] are cited from (van der Maaten et al. 2013). We also report the results using Poisson noise. We can see that all the methods (except for the quadratic loss) can significantly boost the performance by adopting dropout training; meanwhile both Dropout-SVM and Dropout-Logistic are competitive, in fact achieving comparable performance as the state-of-the-art method (i.e., MCF-Logistic) under the dropout training setting. Finally, the Poisson corruption model is slightly worse than the dropout noise, consistent with the previous observations (van der Maaten et al. 2013).

Nightmare at test time

Finally, we evaluate our methods under the "nightmare at test time" (Globerson and Roweis 2006) supervised learning scenario, where some input features that were present when building the classifiers may "die" or be deleted at testing time. In such a scenario, it is crucial to design algorithms that do not assign too much weight to any single feature during testing, no matter how informative it may seem at training. Previous work has conducted the worst-case analysis as well as the learning with marginalized corrupted features. We take this scenario to test the robustness of our dropout training algorithms for both SVM and logistic regression.

We follow the setup of (van der Maaten et al. 2013). Specifically, we choose the the MNIST dataset, which consists of 60,000 training and 10,000 testing handwritten digital images from 10 categories (i.e., $0, \cdots, 9$). The images are represented by 28×28 pixels which results in the feature dimension of 784. We train the models on the full training set, and evaluate the performance on different versions of test set in which a certain level of the features are randomly dropped out, i.e., set to zero. We compare the performance of our dropout learning algorithms with the state-of-art MCF-predictors that use the logistic loss and quadratic loss. These two models also show the state-of-art performance on the same task to the best of our knowledge. We also compare with FDROP (Globerson and Roweis 2006), which is a state-of-the-art algorithm for the "nightmare at test time" setting

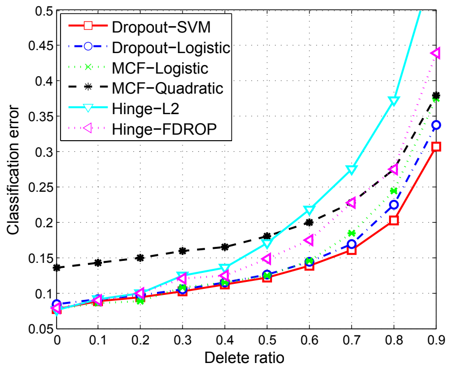


Figure 3: Classification errors of nightmare at test time on MNIST dataset.

that minimizes the hinge loss under an adversarial worst-case analysis. During training, we choose the best models over different dropout levels via cross-validation. For both Dropout-SVM and Dropout-Logistic, we adopt the "one-vs-all" strategy as above for the multiclass classification task.

Figure 3 shows the classification errors of different methods as a function of the random deletion percentage of features at the testing time. Following previous settings, for each deletion percentage, we use a small validation set with the same deletion level to determine the regularization parameters and the dropout level q on the whole training data. From the results, we can see that the proposed Dropout-SVM is consistently more robust than all the other competitors, including the two methods to minimize the expected logistic-loss, especially when the feature deletion percentage is high (e.g., $> 50\%$). Comparing with the standard SVM (i.e., the method Hinge-L2) and the worst-case analysis of hinge loss (i.e., Hinge-FDROP), Dropout-SVM consistently boosts the performance when the deletion ratio is greater than 10%. As expected, Dropout-SVM also significantly outperforms the MCF method with a quadratic loss (i.e., MCF-Quadratic), which is a special case of Dropout-SVM as shown in our theory. Finally, we also note that our iterative algorithm for the logistic-loss works slightly better than the previous algorithm (i.e., MCF-Logistic) when the deletion ratio is larger than 50%.

Conclusions

We present dropout training for SVMs, with an iteratively re-weighted least square (IRLS) algorithm by using data augmentation techniques. Similar ideas are applied to develop a new IRLS algorithm for the dropout training of logistic regression. Our IRLS algorithms provide insights on the connection and difference among various losses in dropout learning settings. Empirical results on various tasks demonstrate the effectiveness of our approaches.

For future work, it is remained open whether the kernel trick can be incorporated in dropout learning. We are also interested in developing more efficient algorithms, e.g., online dropout learning, to deal with even larger datasets, and investigating whether Dropout-SVM can be incorporated into a deep learning architecture or learning with latent structures (Zhu et al. 2014).

[9]The exponential loss was shown to be worse; thus omitted.

Acknowledgments

This work is supported by National Key Project for Basic Research of China (Grant Nos: 2013CB329403, 2012CB316301), National Natural Science Foundation of China (Nos: 61305066, 61322308, 61332007), Tsinghua Self-innovation Project (Grant Nos: 20121088071) and China Postdoctoral Science Foundation Grant (Grant Nos: 2013T60117, 2012M520281).

References

Blitzer, J.; Dredze, M.; and Pereira, F. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Association of Computational Linguistics*.

Burges, C., and Scholkopf, B. 1997. Improving the accuracy and speed of support vector machiens. In *Advances in Neural Information Processing Systems*.

Chen, N.; Zhu, J.; Xia, F.; and Zhang, B. 2013. Generalized relational topic models with data augmentation. In *International Joint Conference on Artificial Intelligence*.

Dekel, O., and Shamir, O. 2008. Learning to classify with missing and corrpted features. In *International Conference on Machine Learning*.

Globerson, A., and Roweis, S. 2006. Nightmare at test time: Robust learning by feature deletion. In *International Conference on Machine Learning*.

Globerson, A.; Koo, T. Y.; Carreras, X.; and Collins, M. 2007. Exponentiated gradient algorithms for log-linear structured prediction. In *ICML*.

Hastie, T.; Tibshirani, R.; and Friedman, J. 2009. *The elements of statistical learning: data mining, inference, and prediction*. Springer.

Hinton, G.; Srivastava, N.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580v1, preprint*.

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

Liu, D. C., and Nocedal, J. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming* (45):503–528.

Polson, N. G., and Scott, S. L. 2011. Data Augmentation for Support Vector Machines. *Bayesian Analysis* 6(1):1–24.

Polson, N. G.; Scott, J. G.; and Windle, J. 2012. Bayesian Inference for Logistic Models using Polya-Gamma Latent Variables. *arXiv:1205.0310v1*.

Rifkin, R., and Klautau, A. 2004. In defense of one-vs-all classification. *Journal of Machine Learning Research* (5):101–141.

Rosasco, L.; Vito, E. D.; Caponnetto, A.; Piana, M.; and Verri, A. 2004. Are loss functions all the same? *Neural Computation* 16(5):1063–1076.

Teo, C.; Globerson, A.; Roweis, S.; and Smola, A. 2008. Convex learning with invariances. In *Advances in Neural Information Processing Systems*.

Torralba, A.; Fergus, R.; and Freeman, W. 2008. A large dataset for non-parametric object and scene recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence* 30(11):1958–1970.

van der Maaten, L.; Chen, M.; Tyree, S.; and Weinberger, K. Q. 2013. Learning with marginalized corrupted features. In *International Conference on Machine Learning*.

Vapnik, V. 1995. *The nature of statistical learning theory*. Springer-Verlag.

Vincent, P.; Larochelle, H.; Bengio, Y.; and Manzagol, P. A. 2008. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine Learning*.

Wager, S.; Wang, S.; and Liang, P. 2013. Dropout training as adaptive regularization. In *Advances in Neural Information Processing*.

Wang, S., and Manning, C. 2013. Fast dropout training. In *International Conference on Machine Learning*.

Wang, S.; Wang, M.; Wager, S.; Liang, P.; and Manning, C. 2013. Feature noising for log-linear structured prediction. In *Empirical Methods in Natural Language Processing*.

Zhu, J.; Chen, N.; Perkins, H.; and Zhang, B. 2014. Gibbs max-margin topic models with data augmentation. *Journal of Machine Learning Research (JMLR)* 15:1073–1110.

Appendix

Appendix A: Derivation of the Upper Bound

We provide details on deriving the variational bound of the expected hinge loss in (4). To simplify notations, we derive the bound for a single data point. For a data set with N examples, a simple summation will give the final bound. Define $g(\boldsymbol{\theta};\mathbf{x}) := \mathbb{E}_p[\log \phi(y|\tilde{\mathbf{x}},\boldsymbol{\theta})]$. We have

$$\begin{aligned} g(\boldsymbol{\theta};\mathbf{x}) &= \mathbb{E}_p\Big[\log \int \frac{1}{\sqrt{2\pi\lambda}} \exp\Big\{-\frac{(\lambda + c\zeta))^2}{2\lambda}\Big\} d\lambda\Big] \\ &= \mathbb{E}_p\Big[\log \int \frac{q(\lambda)}{q(\lambda)\sqrt{2\pi\lambda}} \exp\Big\{-\frac{(\lambda + c\zeta)^2}{2\lambda}\Big\} d\lambda\Big] \\ &\geq \mathbb{E}_p\Big[\mathbb{E}_{q(\lambda)} \log \frac{1}{q(\lambda)\sqrt{2\pi\lambda}} \exp\Big\{-\frac{(\lambda + c\zeta)^2}{2\lambda}\Big\}\Big] \\ &= \Big\{H(\lambda) - \frac{1}{2}\mathbb{E}_q[\log\lambda] - \mathbb{E}_q\Big[\frac{1}{2\lambda}\mathbb{E}_p(\lambda + c\zeta)^2\Big]\Big\} + c' \end{aligned}$$

where λ is the augmented variable, and c' is a constant. Note that the data augmentation at the first two equalities are exact and does not incur any approximation. The approximation is from the assumption that $q(\lambda)$ is independent of the "corrupted" observations $\tilde{\mathbf{x}}$. If there is no uncertainty in the feature corruption (e.g., the corruption level in the dropout (or blankout) noise is 0), the bound is tight. That is, the optimal solution of q will give the original hinge loss.

Appendix B. Proof of Lemma 1

Proof. Ignore the ℓ_2-norm regularizer, we have the objective of the M-step:

$$\mathcal{L}_{[\mathbf{w}]} = \sum_{n=1}^{N} \mathbb{E}_p\left[c\zeta_n + \frac{c^2}{2}\gamma_n\zeta_n^2\right], \tag{23}$$

where $\gamma_n := \mathbb{E}_q[\lambda_n^{-1}]$. Using the definition of $\zeta_n := \ell - y_n \mathbf{w}^\top \tilde{\mathbf{x}}_n$ and ignoring the constants, we have the simplified objective function (again without the ℓ_2-regularizer):

$$\begin{aligned}
\mathcal{L}_{[\mathbf{w}]} &= \sum_{n=1}^{N} \mathbb{E}_p \left[\frac{c^2}{2} \gamma_n \mathbf{w}^\top \tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n^\top \mathbf{w} - (c + \ell c^2 \gamma_n) y_n \mathbf{w}^\top \tilde{\mathbf{x}}_n \right] \\
&= \frac{c^2}{2} \sum_{n=1}^{N} \gamma_n \mathbb{E}_p \left[\mathbf{w}^\top \tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n^\top \mathbf{w} - 2 y_n^h \mathbf{w}^\top \tilde{\mathbf{x}}_n \right] \\
&= \frac{c^2}{2} \sum_{n=1}^{N} \gamma_n \mathbb{E}_p \left[(\mathbf{w}^\top \tilde{\mathbf{x}}_n - y_n^h)^2 \right], \qquad (24)
\end{aligned}$$

where $y_n^h := (\frac{1}{c\gamma_n} + \ell) y_n$ is the re-weighted label.

We now derive the equations to compute γ_n. Let x be a random variable and $y = f(x)$ is a function of x. Then, we have the transformation rule of probability distributions, $p(x) = p(f(x)) |\frac{df(x)}{dx}|$. For our case, let $x = \lambda_n$, and $f(x) = \frac{1}{\lambda_n}$, we have $q(\lambda_n) = \frac{1}{\lambda_n^2} q(\frac{1}{\lambda_n})$. Then

$$\begin{aligned}
\mathbb{E}_{q(\lambda_n)}[\lambda_n^{-1}] &= \int_0^\infty q(\lambda_n) \frac{1}{\lambda_n} d\lambda_n \\
&= \int_0^\infty q\left(\frac{1}{\lambda_n}\right) \frac{1}{\lambda_n^3} d\lambda_n \\
&= \int_\infty^0 q(\mu_n) \mu_n^3 d\mu_n^{-1} \ \ (\text{define } \mu_n = \frac{1}{\lambda_n}) \\
&= \int_0^\infty q(\mu_n) \mu_n d\mu_n \\
&= \mathbb{E}_{q(\lambda_n^{-1})}[\lambda_n^{-1}]. \qquad (25)
\end{aligned}$$

Since $q(\lambda_n^{-1})$ is an inverse Gaussian distribution as shown in Eq. (11), it is easy to get

$$\mathbb{E}_{q(\lambda_n)}[\lambda_n^{-1}] = \mathbb{E}_{q(\lambda_n^{-1})}[\lambda_n^{-1}] = \frac{1}{c\sqrt{\mathbb{E}[\zeta_n^2]}}. \qquad (26)$$

Combining the above results finishes the proof of Lemma 1. $\square$