Adaptive Dropout Rates for Learning with Corrupted Features

Jingwei Zhuo, Jun Zhu, Bo Zhang
Dept. of Comp. Sci. & Tech., State Key Lab of Intell. Tech. & Sys., TNList Lab,
Center for Bio-Inspired Computing Research, Tsinghua University, Beijing, 100084, China
Jiangsu Collaborative Innovation Center for Language Ability, Xuzhou, 221009, China
zhuojw10@mails.tsinghua.edu.cn, dcszj@tsinghua.edu.cn, dcszb@tsinghua.edu.cn

Abstract

Feature noising is an effective mechanism on reducing the risk of overfitting. To avoid an explosive searching space, existing work typically assumes that all features share a single noise level, which is often cross-validated. In this paper, we present a Bayesian feature noising model that flexibly allows for dimension-specific or group-specific noise levels, and we derive a learning algorithm that adaptively updates these noise levels. Our adaptive rule is simple and interpretable, by drawing a direct connection to the fitness of each individual feature or feature group. Empirical results on various datasets demonstrate the effectiveness on avoiding extensive tuning and sometimes improving the performance due to its flexibility.

1 Introduction

Feature noising is effective on protecting machine learning methods from overfitting and increasing the invariance. The early work on explicit corruption includes virtual support vector machines [Burges and Scholkopf, 1997], which explicitly augment the training data through some invariant transformation models. Though effective, such an approach lacks elegance and may suffer from the increased computational cost on processing the artificially corrupted samples. The work on marginalized corrupted features (MCF) [Maaten *et al.*, 2013; Chen *et al.*, 2014] takes a strategy of implicit corruption, where an infinite number of corrupted samples are considered via an expectation operator with respect to some noising model. This strategy has proven effective on introducing adaptive regularization [Bishop, 1995; Wager *et al.*, 2013]. Among many noising models, the dropout noise is particularly interesting because of its simplicity and flexibility for modeling the feature-wise difference between training data and test data (e.g., some features exist in training data but are absent in test data), especially in document classification and image processing [Globerson and Roweis, 2006; Bovik, 2010]. It has been used to improve the generalization of deep networks [Hinton *et al.*, 2012] and to learn semantic features in denoising auto-encoders [Vincent *et al.*, 2008].

However, previous work has primarily focused on the fixed noise distribution with some noise level parameters, and often made the assumption that all features share a single dropout noise level to avoid an explosive search space. Such simplification may ignore the feature-specific details. For example, some features might be more discriminative than others. Furthermore, it has been empirically shown that tuning the noise level influences a lot on the performance. Using cross-validation to search for a good noise level is often time-consuming. The recent work [Geras and Sutton, 2014] presents a scheduled noising scheme to learn multiple-levels of representations, but still assuming a single noise level shared by all feature dimensions. Little effort has been spent on adaptively learning the dropout levels, except [Ba and Frey, 2013] which builds a binary belief network that is stochastically overlaid on a deep neural network (DNN) to selectively regularize each hidden unit of the DNN.

In this paper, we revisit the MCF framework and propose a Bayesian feature noising model for adaptively choosing the dropout noise levels. Our method flexibly assigns each feature with its own noise level and infers all the noise levels from the training data. In the cases where features form non-trivial groups [Meier *et al.*, 2008; Lee *et al.*, 2010], we adapt our method to learn group-wise noise levels, where the features within the same group share a common noise level. By optimizing the expected loss function under the Bayesian noising model with variational methods, we build a new algorithm that iteratively updates the noise level parameters and learns the model parameters under the MCF framework with a given noising model. Our adaptive update rule for dropout noise is simple and interpretable; it has a direct connection to the fitness of each single feature or a feature group. Extensive empirical studies on various datasets demonstrate that our algorithm can adaptively learn the noise levels, without sacrificing the prediction performance. In fact, it often achieves better prediction performance due to the flexibility on capturing the differences among features or feature groups.

2 Preliminaries

We consider the binary classification problem. Let $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ be the given training data, where $\mathbf{X} \in \mathbb{R}^{n\times D}$ is the data matrix and $\mathbf{y} \in \{+1, -1\}^n$ is the response vector. The learning problem is commonly formulated as a regularized empirical risk minimization (ERM) framework:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, R(\mathbf{y}; \mathbf{X}, \boldsymbol{\theta}) + c \cdot \Omega(\boldsymbol{\theta}), \tag{1}$$

where $\Omega(\boldsymbol{\theta})$ is a regularization term and c is the corresponding hyper-parameter. Let $\ell(y; \mathbf{x}, \boldsymbol{\theta})$ be the loss function on data $\mathbf{x}$ when the true label is y. $R(\mathbf{y}; \mathbf{X}, \boldsymbol{\theta}) = \sum_{i=1}^{n} \ell(y_i; \mathbf{x}_i, \boldsymbol{\theta})$ is defined as the empirical risk.

When n is small compared to the model complexity, the empirical risk may be a poor estimate to the true expected loss, and minimizing the empirical risk may lead to a model with poor generalization to the data outside the range of training data, which results in overfitting. The MCF framework [Maaten *et al.*, 2013] mitigates overfitting by generating an infinite corrupted dataset implicitly and training the model over it. More specifically, in MCF the label-invariant noise $\boldsymbol{\alpha}_i$ is assigned over the original training data $\mathbf{x}_i$ and transforms it into the corrupted version $\tilde{\mathbf{x}}_i$, under an additive noise or multiplicative noise scheme. The dropout noise is one type of multiplicative noise and corrupts the training data by

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i \circ \boldsymbol{\alpha}_i, \tag{2}$$

where $\boldsymbol{\alpha}_i$ is a binary vector with entry 0 denoting the deletion of a feature and $\circ$ denotes the element-wise product. The vector $\boldsymbol{\alpha}_i$ is generated from some noising model, such as:

$$p(\boldsymbol{\alpha}_i|\boldsymbol{\eta}) = \prod_{d=1}^{D} (1-\eta_d)^{\alpha_{id}} \eta_d^{1-\alpha_{id}}, \tag{3}$$

which leads to the expectations $\mathbb{E}[\alpha_{id}] = 1 - \eta_d$ and $\mathbb{E}[\tilde{\mathbf{x}}_i] = \mathbf{x} \circ \boldsymbol{\eta}$, a biased version of $\mathbf{x}$. Some unbiased feature noising schemes can be constructed as well, e.g., $\tilde{\mathbf{x}}_i = \mathbf{x}_i \circ \boldsymbol{\alpha}_i \circ \frac{1}{1-\boldsymbol{\eta}}$. In this paper we concentrate on the biased dropout noise, but our results can be extended to the unbiased case.

Then, MCF optimizes the corrupted loss function (again under the ERM framework) as follows:

$$R(\mathbf{y}; \mathbf{X}, \boldsymbol{\theta}, \boldsymbol{\eta}) = \sum_{i=1}^{n} \mathbb{E}_{p(\boldsymbol{\alpha}_i|\boldsymbol{\eta})}[\ell(y_i; \tilde{\mathbf{x}}_i, \boldsymbol{\theta})]. \tag{4}$$

Asymptotically, Eq. (4) is the limit case of explicitly generating finite corrupted copies of training data. This implicit corruption strategy makes the model training faster than explicit corruption. Many recent works follow this strategy, by adopting various loss functions, including the quadratic loss, exponential loss, logistic loss [Maaten *et al.*, 2013] and hinge loss [Chen *et al.*, 2014]. In the sequel, we will use the unified representation:

$$\ell(y; \mathbf{x}, \boldsymbol{\theta}) = -\log p(y|\mathbf{x}, \boldsymbol{\theta}),$$

where $p(y|\mathbf{x}, \boldsymbol{\theta})$ is some likelihood function. For example, we have $p(y|\mathbf{x}, \boldsymbol{\theta}) = \frac{1}{1+\exp(-y\boldsymbol{\theta}^\top \mathbf{x})}$ for logistic loss and $p(y|\mathbf{x}, \boldsymbol{\theta}) = \exp(-(\ell - y\boldsymbol{\theta}^{\mathrm{T}}\mathbf{x})_+)$ for hinge loss, where $(x)_+ = \max(0, x)$.

However, the price MCF has to pay is that the computation of the corrupted loss becomes more complex than that of the empirical loss, thereby calling for efficient approximate algorithms, such as using second-order Taylor expansion [Wager *et al.*, 2013], Gaussian approximation [Wang and Manning, 2013] or variational bounds (e.g., the Jensen's bound [Maaten *et al.*, 2013] and data augmentation bound [Chen *et al.*, 2014]). We follow the idea of data augmentation bound

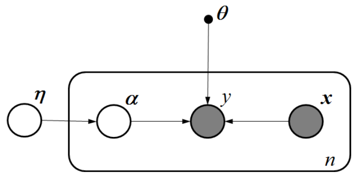


Figure 1: The Bayesian feature noising model.

in this paper. Here we use the hinge loss as an example. The bound for logistic loss is similar and can be found in [Chen *et al.*, 2014]. Let $\zeta_i = \ell - y_i \boldsymbol{\theta}^{\mathrm{T}} \tilde{\mathbf{x}}_i$, and $\phi(y_i|\tilde{\mathbf{x}}_i, \boldsymbol{\theta}) = \exp\{-2\max(0, \zeta_i)\}$ be the pseudo-likelihood of the ith instance[1]. Using the ideas of data augmentation [Polson *et al.*, 2011; Zhu *et al.*, 2014], the pseudo-likelihood can be expressed as

$$\phi(y_i|\tilde{\mathbf{x}}_i, \boldsymbol{\theta}) = \int_0^\infty \frac{1}{\sqrt{2\pi\lambda_i}} \exp\left\{-\frac{(\lambda_i + \zeta_i)^2}{2\lambda_i}\right\} \mathrm{d}\lambda_i, \tag{5}$$

Using Eq. (5) and Jensen's inequality, a variational upper bound of Eq. (4) can be derived as

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}, q(\boldsymbol{\lambda})) = \frac{1}{2} \sum_{i=1}^{n} \Bigg\{ & -H(\lambda_i) + \frac{1}{2}\mathbb{E}_q[\log \lambda_i] \\ & + \mathbb{E}_q\left[\frac{1}{2\lambda_i}\mathbb{E}_{p(\alpha_i|\boldsymbol{\eta})}(\lambda_i + \zeta_i)^2\right] \Bigg\} + \gamma, \end{aligned} \tag{6}$$

where $H(\lambda_i)$ is the entropy of the variational distribution $q(\lambda_i)$, $\mathbb{E}_q[\cdot]$ is the expectation over $q(\boldsymbol{\lambda}) = \prod_i q(\lambda_i)$ and γ is some constant. Then, an EM algorithm can be used to optimize the upper bound iteratively.

However, all the existing MCF methods have treated the dropout levels (i.e., $\boldsymbol{\eta}$) as fixed unknown parameters. Furthermore, to avoid an explosive searching space for a good value of $\boldsymbol{\eta}$, a simplifying assumption is often made that $\boldsymbol{\eta} = \eta \cdot \mathbf{1}$, where $\mathbf{1}$ is the D-dimensional vector with all elements being the unit 1. Then, the single parameter η can be effectively searched via cross-validation.

3 Bayesian feature noising model

We now present a Bayesian feature noising model, which flexibly treats the noise levels as random variables, with each dimension denoting the importance of the corresponding feature, and all the noise levels share a common prior to avoid overparameterization. By doing this, our model captures both the local and global properties, that is, each feature dimension or feature group has its own noise level (i.e., locality), while all the noise levels share a common prior (i.e., globality).

3.1 The feature-specific case

We first consider the most flexible case, where each feature dimension has its own noise level. The extension to considering grouping structures will be presented in next section.

We follow the similar setup as in the MCF framework to corrupt the features. Here, we treat $\boldsymbol{\eta}$ as random variables,

[1]The factor 2 doesn't affect; it is simply for simplicity.

following some prior $p_0(\boldsymbol{\eta})$. The graphical structure of a Bayesian feature noising model is illustrated in Fig. 1. Then, the joint distribution of the Bayesian noising model is:

$$p(y, \boldsymbol{\eta}, \boldsymbol{\alpha}|\mathbf{x}, \boldsymbol{\theta}) = p_0(\boldsymbol{\eta}) \prod_{i=1}^{n} p(y_i|\mathbf{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{\theta}) p(\boldsymbol{\alpha}_i|\boldsymbol{\eta}), \quad (7)$$

where $p(y_i|\mathbf{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{\theta}) = p(y_i|\tilde{\mathbf{x}}_i, \boldsymbol{\theta})$ is per-sample likelihood, either logistic or the one derived from the hinge-loss. Let $p_0(\boldsymbol{\eta})$ be a non-informative prior. Then, we have the marginal likelihood (therefore the expected empirical risk $R(\mathbf{y}; \mathbf{X}, \boldsymbol{\theta})$):

$$\log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) = \mathbb{E}_{p(\boldsymbol{\eta}|\mathbf{y},\mathbf{X},\boldsymbol{\theta})} \left[\log \frac{p(\mathbf{y}, \boldsymbol{\eta}|\mathbf{X}, \boldsymbol{\theta})}{p(\boldsymbol{\eta}|\mathbf{y}, \mathbf{X}, \boldsymbol{\theta})} \right], \quad (8)$$

which is unfortunately intractable due to the non-conjugacy between the prior and likelihood and the high-dimensional integral. Here, we introduce a lower bound for the log-likelihood. Then, we can use the coordinate ascent (EM) algorithm for finding a local maxima. Specifically, using the Jensen inequality, we can get a lower bound of Eq. (8):

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}) = H(\boldsymbol{\eta}) + \mathbb{E}_{p(\boldsymbol{\eta}|\mathbf{y},\mathbf{X},\boldsymbol{\theta})} \Bigg[& \log p_0(\boldsymbol{\eta}) \\ & + \sum_{i=1}^{n} \mathbb{E}_{p(\boldsymbol{\alpha}_i|\boldsymbol{\eta})} \left[\log p(y_i|\mathbf{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{\theta}) \right] \Bigg], \end{aligned} \quad (9)$$

where $H(\boldsymbol{\eta})$ is the entropy of the posterior, i.e.,

$$p(\boldsymbol{\eta}|\mathbf{y}, \mathbf{X}, \boldsymbol{\theta}) = \frac{p_0(\boldsymbol{\eta}) \prod_{i=1}^{n} \mathbb{E}_{p(\boldsymbol{\alpha}_i|\boldsymbol{\eta})} \left[p(y_i|\mathbf{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{\theta}) \right]}{p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})}. \quad (10)$$

But it is still hard to compute the true posterior $p(\boldsymbol{\eta}|\mathbf{y}, \mathbf{X}, \boldsymbol{\theta})$ and the expectation over it. We solve this problem via Laplace methods: we approximate the posterior with a Dirac delta function $\delta_{\hat{\boldsymbol{\eta}}}(\boldsymbol{\eta})$, centered at the MAP estimator of $\boldsymbol{\eta}$, i.e., $\hat{\boldsymbol{\eta}} = \operatorname{argmax}_{\boldsymbol{\eta}} \log p(\boldsymbol{\eta}|\mathbf{y}, \mathbf{X}, \boldsymbol{\theta})$.

For maximizing $\log p(\boldsymbol{\eta}|\mathbf{y}, \mathbf{X}, \boldsymbol{\theta})$, we can introduce a variational bound as previous works [Maaten *et al.*, 2013; Chen *et al.*, 2014], since $-\sum_{i=1}^{n} \log \mathbb{E}_{p(\boldsymbol{\alpha}_i|\boldsymbol{\eta})} \left[p(y_i|\mathbf{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{\theta}) \right]$ has the same formulation as the loss function in the MCF framework. However, it will be quite time-consuming. Here we use a loose but easy to compute and very informative lower bound. Specifically, omitting the irrelevant terms, we can rewrite $\log p(\boldsymbol{\eta}|\mathbf{y}, \mathbf{X}, \boldsymbol{\theta})$ as

$$\log p_0(\boldsymbol{\eta}) + \sum_{i=1}^{n} \log \left\{ \sum_{\boldsymbol{\alpha}_i} p(\boldsymbol{\alpha}_i|\boldsymbol{\eta}) p(y_i|\mathbf{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{\theta}) \right\}. \quad (11)$$

Recall that in the corrupted likelihood $\mathbf{x}_i \circ \boldsymbol{\alpha}_i$ is an entity, which implies $p(y_i|\mathbf{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{\theta})$ stays unchanged for any $\alpha_{id} \in \{0, 1\}$ when $x_{id} = 0$. So we can rewrite the dropout distribution as $p(\boldsymbol{\alpha}_i|\boldsymbol{\eta}) = \prod_{d=1}^{D} (1 - \eta_d)^{\alpha_{id}\mathbb{I}(x_{id} \neq 0)} \eta_d^{(1-\alpha_{id})\mathbb{I}(x_{id} \neq 0)}$. Then, let $\hat{\boldsymbol{\alpha}}_i = \operatorname{argmax}_{\boldsymbol{\alpha}_i} \log p(y_i|\mathbf{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{\theta})$. We get a lower bound of Eq. (11):

$$\mathcal{L}(\boldsymbol{\eta}) = \log p_0(\boldsymbol{\eta}) + \sum_{i=1}^{n} \log \left\{ p(\hat{\boldsymbol{\alpha}}_i|\boldsymbol{\eta}) p(y_i|\mathbf{x}_i, \hat{\boldsymbol{\alpha}}_i, \boldsymbol{\theta}) \right\}. (12)$$

The lower bound (11) has a nice property that if a loss function has no coupling among the features (e.g., hinge loss/logistic loss), the solution $\hat{\boldsymbol{\eta}}$ is in closed-form. For hinge loss, we solve the following problem to estimate $\hat{\boldsymbol{\alpha}}_i$:

$$\begin{aligned} \hat{\boldsymbol{\alpha}}_i &= \underset{\boldsymbol{\alpha}_i \in \{0,1\}^D}{\operatorname{argmax}} \left\{ -(\ell - y_i \boldsymbol{\theta}^{\mathrm{T}} (\mathbf{x}_i \circ \boldsymbol{\alpha}_i))_+ \right\} \\ &= \underset{\boldsymbol{\alpha}_i \in \{0,1\}^D}{\operatorname{argmax}} \; y_i \boldsymbol{\theta}^{\mathrm{T}} (\mathbf{x}_i \circ \boldsymbol{\alpha}_i), \end{aligned} \quad (13)$$

whose solution is:

$$\hat{\alpha}_{id} = \begin{cases} \mathbb{I}(y_i \theta_d x_{id} > 0), x_{id} \neq 0 \\ \text{Any value}, \quad x_{id} = 0 \end{cases}. \quad (14)$$

Then, maximizing the bound $\mathcal{L}(\boldsymbol{\eta})$ in Eq. (12) leads to the closed-form solution:

$$\hat{\eta}_d = \frac{\sum_{i=1}^{n} \mathbb{I}(y_i \theta_d x_{id} < 0)}{\sum_{i=1}^{n} \mathbb{I}(x_{id} \neq 0)}. \quad (15)$$

For the logistic loss, the optimization problem will be

$$\begin{aligned} \hat{\boldsymbol{\alpha}}_i &= \underset{\boldsymbol{\alpha} \in \{0,1\}^D}{\operatorname{argmin}} \sum_{i=1}^{n} \log(1 + e^{-y_i \boldsymbol{\theta}^{\mathrm{T}} (\mathbf{x}_i \circ \boldsymbol{\alpha}_i)}) \\ &= \underset{\boldsymbol{\alpha} \in \{0,1\}^D}{\operatorname{argmax}} \; y_i \boldsymbol{\theta}^{\mathrm{T}} (\mathbf{x}_i \circ \boldsymbol{\alpha}_i), \end{aligned} \quad (16)$$

which has the same solution as in Eq. (14), and thereby the same update rule for $\boldsymbol{\eta}$ in Eq. (15).

The update rule in Eq. (15) has an intuitive interpretation. In the numerator, $y_i \theta_d x_{id}$ is the contribution of the d-th feature to the prediction score of the i-th training data. Since the task is binary classification, we only consider the sign of this contribution instead of its magnitude. In the denominator, the indicator function $\mathbb{I}(x_{id} \neq 0)$ acts as a normalizer — if $x_{id} = 0$, there is no need to consider its contribution since this feature will certainly be deleted (i.e., $\tilde{x}_{id} = 0$); but if $x_{id} \neq 0$, the corresponding feature-wise score should normalize the importance to 1 because of the indicator function we use in the numerator. The basic idea of Eq. (15) is that if the d-th feature always makes positive contributions, its dropout level will be very low, implying this feature is more discriminative in the classification task than other features.

In Eq. (15) we omit $p_0(\boldsymbol{\eta})$ since we assume it non-informative. In fact, there are many cases in which there is some prior knowledge we can use. For example, in the "nightmare at test time" scenario [Globerson and Roweis, 2006], the test data is the corrupted version of the training data where the features of the test data are dropped with some fixed probability. Our framework can handle these cases by introducing an informative prior (e.g., the beta distribution $\mathrm{Beta}(\mathbf{a}, \mathbf{b})$). Formally, let $\mu_d = \frac{a_d - 1}{a_d + b_d - 2}$ be the prior mode and $m_d = a_d + b_d - 2$ be the measure of importance of the prior. Then we can parametrize $\mathrm{Beta}(\mathbf{a}, \mathbf{b})$ with $(\boldsymbol{\mu}, \mathbf{m})$ and the update rule will be

$$\hat{\eta}_d = \frac{\sum_{i=1}^{n} \mathbb{I}(y_i \theta_d x_{id} < 0) + \mu_d m_d}{\sum_{i=1}^{n} \mathbb{I}(x_{id} \neq 0) + m_d}, \quad (17)$$

which is a weighted summation between the prior mode and empirical mode. By doing so, we can incorporate the prior knowledge into our model with the price of introducing $2D$ extra hyperparameters. The number of hyperparameters can be reduced to 2 if we assign all the features with the same Beta prior. We use the latter setting in the experiments.

Once we have an estimate $\hat{\boldsymbol{\eta}}$ of the noise levels, plug it into Eq. (9), omit the unrelated terms and reverse the sign, we get the corrupted empirical loss

$$R(\mathbf{y}; \mathbf{X}, \boldsymbol{\theta}, \hat{\boldsymbol{\eta}}) = -\sum_{i=1}^{n} \mathbb{E}_{p(\boldsymbol{\alpha}_i|\hat{\boldsymbol{\eta}})} \left[\log p(y_i|\mathbf{x}_i, \boldsymbol{\alpha}_i, \boldsymbol{\theta})\right]. \quad (18)$$

This is exactly the loss function (4) in the MCF framework, except that we are using more flexible feature-specific noise levels $\boldsymbol{\eta}$ learnt from data. We can follow the idea for solving Eq. (4) over $\boldsymbol{\theta}$. By doing so, we get a coordinate ascent algorithm for finding MLE of Eq. (8). Specifically, for hinge loss, our algorithm iteratively performs the following steps:

For $\boldsymbol{\eta}$: update the noise levels $\boldsymbol{\eta}$ with Eq. (15) for the non-informative prior case, or Eq. (17) for the Beta prior case.

For $q(\boldsymbol{\lambda})$: infer the variational distribution $q(\boldsymbol{\lambda})$ by optimizing Eq. (6), which is an upper bound of Eq. (18). The solution is

$$q(\lambda_i) \sim \mathcal{GIG}\left(\lambda_i; \frac{1}{2}, 1, \mathbb{E}_p[\zeta_i^2]\right), \quad (19)$$

where $\mathcal{GIG}(x; p, a, b) \propto x^{p-1} \exp(-\frac{1}{2}(\frac{b}{x} + ax))$ is the generalized inverse Gaussian distribution and the term $\mathbb{E}_p[\zeta_i^2]$ has an analytical form for the dropout noise [Chen *et al.*, 2014].

For $\boldsymbol{\theta}$: plug the results of $q(\lambda_i)$ into Eq. (6), adding the regularizer and ignoring irrelevant terms, we get the following objective:

$$\mathcal{L}_{[\boldsymbol{\theta}]} = c \cdot ||\boldsymbol{\theta}||_2^2 + \frac{1}{2}\sum_{i=1}^{n} \mathbb{E}_{p(\alpha_i|\boldsymbol{\eta})} \left[\zeta_i + \frac{1}{2}\mathbb{E}_q[\lambda_i^{-1}]\zeta_i^2\right], \quad (20)$$

where $\mathbb{E}_q[\lambda_i^{-1}] = 1/\mathbb{E}_p[\zeta_i^2]$ according to the distribution (19). Then, we update $\boldsymbol{\theta}$ by minimizing $\mathcal{L}_{[\boldsymbol{\theta}]}$ over $\boldsymbol{\theta}$. As noted by [Chen *et al.*, 2014], the objective $\mathcal{L}_{[\boldsymbol{\theta}]}$ can be denoted as a re-weighted quadratic loss and has the closed-form solution. However, to avoid high-dimensional matrix inversion, the numerical methods (e.g., L-BFGS [Liu and Nocedal, 1989]) are often more efficient to solve this problem. For logistic loss, the update equations for $q(\boldsymbol{\lambda})$ and $\boldsymbol{\theta}$ are slightly different and can be found in [Chen *et al.*, 2014].

In summary, our algorithm runs the above three steps iteratively until convergence or reaching some threshold. Compared to the iteratively re-weighted least square (IRLS) algorithm in MCF framework, our algorithm adds the extra step of updating $\boldsymbol{\eta}$ in each iteration. Since the cost of updating $\boldsymbol{\eta}$ is linear to the data size and feature dimension, the overall complexity is almost the same as the ordinary MCF method with a fixed noise level.

3.2 The group-specific case

We now extend the above ideas to the case where the features are partitioned into M groups and the features within the same group share the same noise level. For simplicity, we consider the case where groups have no overlapping. Specifically, let $G_1, \ldots, G_M$ represents a partition of the features. The dropout noise distribution is

$$p(\boldsymbol{\alpha}_i|\boldsymbol{\eta}) = \prod_{j=1}^{M} \prod_{k \in G_j} (1-\eta_j)^{\alpha_{ik}} \eta_j^{1-\alpha_{ik}}. \quad (21)$$

We can build a similar Bayesian feature noising model as before, and the loss function over $\boldsymbol{\theta}$ has the same formulation. The only difference is on $\boldsymbol{\eta}$. Now, the update rule for $\boldsymbol{\eta}$ is

$$\hat{\eta}_j = \frac{\sum_{i=1}^{n} \sum_{k \in G_j} \mathbb{I}(y_i \theta_k x_{ik} < 0)}{\sum_{i=1}^{n} \sum_{k \in G_j} \mathbb{I}(x_{ik} \neq 0)}. \quad (22)$$

This update rule is a smoothing version of Eq. (15) for the feature-specific case. Intuitively, the group dropout level averages the empirical dropout level of each feature, representing the common trend of features in the same group and mitigating the influence of the extreme cases (i.e. the feature-specific dropout level of some feature is very different from the other most features in the same group). This property reflects our intuition that the features within the same group should have very similar behaviour: to assign them with the same dropout level η_j, we model the common properties; To let each feature has its own indicator variable $\alpha_{ik}, k \in G_j$, we allow them to have their own behaviour to some extent.

We note that the group-specific case is the intermediate version. When $M = 1$, all the features share the same dropout level — this is the assumption that previous work makes. When $M = D$, each feature has its own dropout level — this is the assumption that the feature-specific model makes.

4 Experiments

We present the experimental results of our Bayesian feature noising models, with comparison to the ordinary methods with dropout noise on a wide range of tasks, including: (1) binary classification on the Amazon review datasets [Blitzer *et al.*, 2007] using the feature-specific noising model and on the MEMset donor dataset[2] using the group-specific noising model; (2) multi-class classification on the 20 Newsgroup dataset[3]; and (3) the "nightmare at test time" scenario [Globerson and Roweis, 2006] on the MNIST dataset[4].

4.1 Binary classification: Feature-specific case

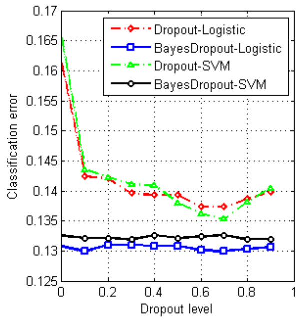
The Amazon review datasets have been widely adopted to evaluate the MCF methods [Maaten *et al.*, 2013; Chen *et al.*, 2014]. We consider the Books and DVD datasets. Each dataset consists of review documents in a 20,000 dimensional feature space under a bag-of-words representation, and the corresponding label denotes the altitude of the review. Following the previous settings, we choose 2,000 documents for training and the other about 4,000 documents for testing.

We compare with the Dropout-Logistic and Dropout-SVM algorithm [Chen *et al.*, 2014], which use an ℓ_2-norm regularizer and have proven to achieve state-of-the-art performance on this task. We apply our Bayesian update rule on them to build our BayesDropout-Logistic and BayesDropout-SVM algorithms. All the models are learned by an EM algorithm with the same optimizing iteration using data augmentation

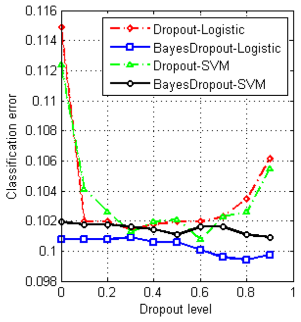
[2]Available at: http://genes.mit.edu/burgelab/maxent/ssdata/

[3]Available at: http://people.csail.mit.edu/jrennie/20Newsgroups/

[4]Available at: http://yann.lecun.com/exdb/mnist/



(a) books



(b) dvd

Figure 2: Dropout noise level versus classification error on the Amazon review datasets

bounds [Chen *et al.*, 2014], implemented in C++ using L-BFGS methods [Liu and Nocedal, 1989]. We run the algorithms with 640 iterations, which are sufficiently large for all methods to converge. Fig. 2 shows the test errors of various methods with different initialized dropout levels, where for Dropout-Logistic/SVM these noise levels are fixed while for our methods they are updated as Eq. (15) in each iteration. When the dropout level is zero, Dropout-Logistic/SVM corresponds to the standard logistic regression/SVM with an ℓ_2-norm regularizer. We can see that no matter what the initial noise level is, our Bayesian methods converge to a good enough local minima, which is almost independent of the initial values. Therefore, there is no need for us to tune the dropout levels. Furthermore, we may get better results because of the flexibility on capturing the difference among features by using feature-specific noise levels, instead of the single one shared by all features in Dropout-Logistic/SVM. Finally, if we use cross-validation to find the best dropout level for Dropout-Logistic/SVM, the computational cost will be much higher than that of our Bayesian methods since a single run of each method has a similar cost (e.g., the training time for our model is 9.1s, while the baseline needs 6.8s by running 640 iterations with a single thread) but doing cross-validation needs learning multiple models, i.e., multiple runs of the method, as detailed in Sec. 3.1.

4.2 Binary classification: Group-specific case

The MEMset Donor dataset was built for donor splice site detection, which plays an important role in gene finding algorithms. Splice sites are the regions between coding (exons) and noncoding (introns) DNA segments. The 5 end of an intron is called a donor splice site and the 3 end an acceptor splice site. The training set consists of 8,415 true and 179,438 false human donor sites, and the testing set contains 4,208 true and 89,717 false donor sites. Each instance is of length 7 and factored with 4 levels $\{A, T, G, C\}$. We refer the readers to [Yeo and Burge, 2004] for more details.

Here we follow the setup in [Meier *et al.*, 2008], which splits the original training set into a balanced training set (5,610 true and 5,610 false instances) and an unbalanced validation set (2,805 true and 59,804 false instances) which has the same true/false ratio as the testing set. The data are represented as a collection of all factor interactions up to degree 2, leading to a 2,604-dimensional feature space. The features are partitioned into 63 groups of sizes varying from 4 to 4^3. Each instance is sparse and has 63 non-zero features. We train the models on the balanced training set, then use the validation set to choose the best threshold τ for the classifier output over the trained model. That is, we assign the instance x_i with label 1 when $p(y_i = 1|x_i, \theta) > \tau$ and 0 otherwise. The performance is measured with correlation coefficient. Details can be found in [Meier *et al.*, 2008]. We remove the ℓ_2-norm regularizer from Dropout-Logistic and BayesDropout-Logistic here to compare our methods with group lasso which uses the group-wise $\ell_{1,2}$-norm as regularizer equally. Fig. 3 shows that the dropout noise is helpful for improving the performance, and the group-specific method can achieve better performance than the feature-specific method. We note that the best result using the logistic group-lasso over 2 interactions is 0.6593 [Meier *et al.*, 2008] and over 4 interactions (27,896 groups in a 22,458,100-dimensional space) is 0.663 [Roth and Fischer, 2008]. Therefore, our results are competitive with the results of group-lasso. Moreover, our methods don't need to tune the regularization parameter since there is no regularization terms.

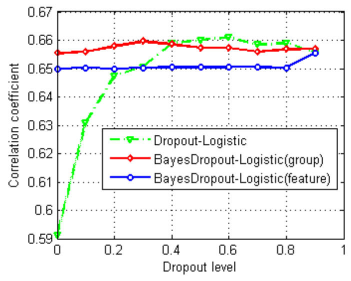


Figure 3: Results on the MEMset donor dataset.

4.3 Multi-class Classification

We report the results on the 20 Newsgroup dataset for multi-class classification. The dataset consists of news documents in 20 different newsgroups. The training set contains 11,269 instances and the testing set contains 7,505 instances. We adopt the "one-vs-all" strategy [Rifkin and Klautau, 2004], which is effective to do the multi-class classification with a binary classifier. However, in the "one-vs-all" case, the positive instances and negative instances are unbalanced when learning each binary classifier. So, we set different weights for the contribution of the positive and negative instances in Eq. (15) corresponding to the ratio r of numbers between the positive and negative instances to make a balance:

$$\hat{\eta}_d = \frac{\sum_{i=1}^{n} \mathbb{I}(y_i \theta_d x_{id} < 0) \left(\mathbb{I}(y_i > 0) + r\mathbb{I}(y_i < 0)\right)}{\sum_{i=1}^{n} \mathbb{I}(x_{id} \neq 0) \left(\mathbb{I}(y_i > 0) + r\mathbb{I}(y_i < 0)\right)}.$$

Since there is no obvious grouping structure in the feature space, we only report the results of our feature-specific Bayesian model, again with comparison to the strong baselines of Dropout-Logistic and Dropout-SVM. As Fig. 4 shows, our Bayesian model can achieve almost as well as the

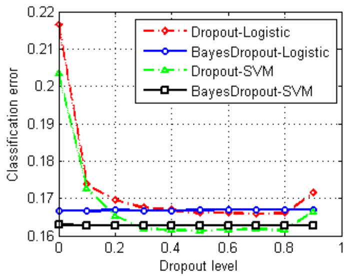
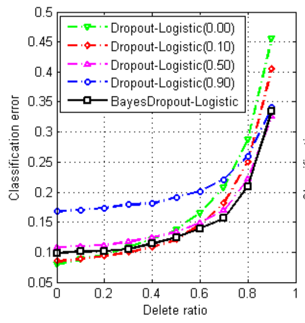
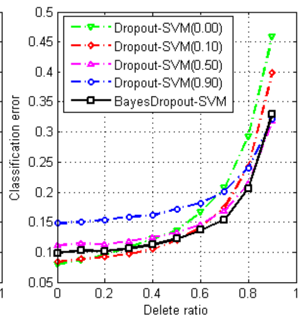


Figure 4: Classification error on the 20 Newsgroup dataset.



(a) Logistic loss



(b) Hinge loss

Figure 5: The nightmare at test time phenonmenon on MNIST dataset. Numbers in brackets are dropout levels.

best performance of the baselines, without the need for tuning the dropout levels. An interesting thing to note is that the performance of the baselines is insensitive to the dropout level in a wide range. That may be one reason why there is no explicit improvement using the Bayesian model.

4.4 Nightmare at test time

The "nightmare at test time" scenario is one of the common settings for testing the robustness of learning [Globerson and Roweis, 2006]. We follow the setup of [Maaten *et al.*, 2013; Chen *et al.*, 2014] to evaluate our Bayesian methods on MNIST dataset, which has 60,000 training and 10,000 testing handwritten character images with labels from 0 to 9. Each image is represented by 28×28 pixels (thus 784 dimensional features), and most of the pixel values are close to zero. We train the model on the full training set with "one-vs-all" strategy, and evaluate the performance on different versions of test set in which a certain level of the features are randomly deleted (i.e., set to zero) while keeping the label invariant.

Delete ratio	0	0.1	0.3	0.5	0.7	0.9
Dropout-Logistic	**7.94**	8.77	9.94	12.07	16.58	32.23
Bayes-Logistic-I	9.84	10.10	10.50	12.43	15.63	33.38
Bayes-Logistic-II	7.97	8.74	**9.68**	11.98	16.02	31.76
Dropout-SVM	8.14	8.55	9.78	12.04	15.63	31.74
Bayes-SVM-I	9.86	10.24	10.57	12.25	**15.41**	32.89
Bayes-SVM-II	7.99	**8.40**	9.73	**11.77**	15.51	**31.03**

Table 1: Error rates (%) on the MNIST dataset with different delete ratios, where Bayes-SVM/Logistic-I denotes our Bayesian models with uninformative prior and Bayes-SVM/Logistic-II uses a Beta prior as detailed in text.

Fig. 5(a) and 5(b) present the results of our Bayesian methods with an uninformative prior, comparing to the baselines (i.e., Dropout-Logistic/SVM) with different dropout levels. We can see that there is an obvious improvement when the delete ratio is around 0.5, while the performance for too low delete rates is not very good. There are three possible reasons. First, the noise levels learnt by our methods are on the relative sense. That is, they only reflect the relative importance among features since we do not know the ground level (i.e., the delete ratio). With no information about the ground level, the learnt levels will approximately center around 0.5. As shown in Table 1, if we provide the information of the ground level (i.e., delete ratio) as an informative prior[5], our Bayesian methods achieve better performance than the baselines for almost every delete ratio. The second reason is that when the delete ratio is low, there is no benefit to apply dropout noise, as we can find in Fig. 5(a) and 5(b) that for Dropout-Logistic/SVM, when the delete level is less than 0.3, the performance deteriorates as the dropout level increases. Finally, unlike the document classification, the feature-specific model may overfit for modelling the image classification problem, as a single pixel doesn't contribute a lot to the classification label but some overlapped groups of features (e.g., a group of features placed as a circle in the image for classifying digit 0) does.

5 Conclusions and Future Work

We present a Bayesian feature noising model for learning with corrupted features. Our methods flexibly allow feature-specific or group-specific noise levels and adaptively update these noise level parameters. Empirical results on various datasets demonstrate the effectiveness of our methods on learning accurate classifiers without extensive tuning of the noise levels. In some cases, our methods achieve improved accuracy due to the flexibility on capturing feature-wise differences.

For future work, we are interested in the deep connection with feature-wise boosting [O'Sullivan *et al.*, 2000]. We are also interested in extending our techniques to learning deep networks with an adaptive dropout noise.

Acknowledgement

This work is supported by the National Basic Research Program (973 Program) of China (Nos. 2013CB329403, 2012CB316301), National NSF of China (Nos. 61322308, 61332007), Tsinghua Initiative Scientific Research Program (Nos. 20121088071, 20141080934).

[5]We use Beta prior in which we set the prior mode for each feature μ_d according to the delete ratio and the importance $m_d = \lambda \sum_{i=1}^{n} \mathbb{I}(x_{id} \neq 0)$ to be proportional to the number of non-zero entries of one feature, and tune λ via cross-validation. That is, we do a linear combination of the trained dropout level and the prior mode, which is independent with the number of non-zero entries of one feature, and the update rule Eq. (17) can be rewritten as $\hat{\eta}_d = \frac{1}{1+\lambda}\eta_d + \frac{\lambda}{1+\lambda}\mu_d$.

References

[Ba and Frey, 2013] Jimmy Ba and Brendan Frey. Adaptive dropout for training deep neural networks. In *Advances in Neural Information Processing Systems*, pages 3084–3092, 2013.

[Bishop, 1995] Chris M Bishop. Training with noise is equivalent to tikhonov regularization. *Neural Computation*, 7(1):108–116, 1995.

[Blitzer *et al.*, 2007] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *the Annual Meeting of the Association for Computational Linguistics*, volume 7, pages 440–447. Citeseer, 2007.

[Bovik, 2010] Alan C Bovik. *Handbook of image and video processing*. Academic press, 2010.

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

[Chen *et al.*, 2014] Ning Chen, Jun Zhu, Jianfei Chen, and Bo Zhang. Dropout training for support vector machines. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[Geras and Sutton, 2014] Krzysztof J. Geras and Charles Sutton. Scheduled denoising autoencoders. In *Deep Learning and Representation Learning Workshop, NIPS*, 2014.

[Globerson and Roweis, 2006] Amir Globerson and Sam Roweis. Nightmare at test time: robust learning by feature deletion. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 353–360. ACM, 2006.

[Hinton *et al.*, 2012] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[Lee *et al.*, 2010] Seunghak Lee, Jun Zhu, and Eric P Xing. Adaptive multi-task lasso: with application to eqtl detection. In *Advances in Neural Information Processing Systems*, pages 1306–1314, 2010.

[Liu and Nocedal, 1989] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989.

[Maaten *et al.*, 2013] Laurens Maaten, Minmin Chen, Stephen Tyree, and Kilian Q Weinberger. Learning with marginalized corrupted features. In *Proceedings of the 30th International Conference on Machine Learning*, pages 410–418, 2013.

[Meier *et al.*, 2008] Lukas Meier, Sara Van De Geer, and Peter Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008.

[O'Sullivan *et al.*, 2000] Joseph O'Sullivan, John Langford, Richard Caruana, and Avrim Blum. Featureboost: A meta learning algorithm that improves model robustness. *Tech. Report, Computer Science Department, CMU*, 2000.

[Polson *et al.*, 2011] Nicholas G Polson, Steven L Scott, et al. Data augmentation for support vector machines. *Bayesian Analysis*, 6(1):1–23, 2011.

[Rifkin and Klautau, 2004] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

[Roth and Fischer, 2008] Volker Roth and Bernd Fischer. The group-lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *Proceedings of the 25th International Conference on Machine Learning*, pages 848–855. ACM, 2008.

[Vincent *et al.*, 2008] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1096–1103. ACM, 2008.

[Wager *et al.*, 2013] Stefan Wager, Sida Wang, and Percy Liang. Dropout training as adaptive regularization. In *Advances in Neural Information Processing Systems*, pages 351–359, 2013.

[Wang and Manning, 2013] Sida Wang and Christopher Manning. Fast dropout training. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 118–126, 2013.

[Yeo and Burge, 2004] Gene Yeo and Christopher B Burge. Maximum entropy modeling of short sequence motifs with applications to rna splicing signals. *Journal of Computational Biology*, 11(2-3):377–394, 2004.

[Zhu *et al.*, 2014] Jun Zhu, Ning Chen, Hugh Perkins, and Bo Zhang. Gibbs max-margin topic models with data augmentation. *Journal of Machine Learning Research*, 15(1):1073–1110, 2014.