Deep Metric Learning: The Generalization Analysis and an Adaptive Algorithm

Mengdi Huai[1] , **Hongfei Xue**[2] , **Chenglin Miao**[2] , **Liuyi Yao**[2] ,
Lu Su[2] , **Changyou Chen**[2] and **Aidong Zhang**[1]

[1]Department of Computer Science, University of Virginia, VA, USA
[2]Department of Computer Science and Engineering, SUNY at Buffalo, NY, USA
mh6ck@virginia.edu, {hongfeix, cmiao, liuyiyao, lusu, changyou}@buffalo.edu, aidong@virginia.edu

Abstract

As an effective way to learn a distance metric between pairs of samples, deep metric learning (DML) has drawn significant attention in recent years. The key idea of DML is to learn a set of hierarchical nonlinear mappings using deep neural networks, and then project the data samples into a new feature space for comparing or matching. Although DML has achieved practical success in many applications, there is no existing work that theoretically analyzes the generalization error bound for DML, which can measure how good a learned DML model is able to perform on unseen data. In this paper, we try to fill up this research gap and derive the generalization error bound for DML. Additionally, based on the derived generalization bound, we propose a novel DML method (called ADroDML), which can adaptively learn the retention rates for the DML models with dropout in a theoretically justified way. Compared with existing DML works that require predefined retention rates, ADroDML can learn the retention rates in an optimal way and achieve better performance. We also conduct experiments on real-world datasets to verify the findings derived from the generalization error bound and demonstrate the effectiveness of the proposed adaptive DML method.

1 Introduction

Measuring the similarity between data samples plays an important role in many machine learning and data mining algorithms. Although some simple metrics (e.g., Euclidean distances) can be used to measure the similarity between samples, they have no capability to capture the statistical regularities in the data, and thus largely degrade the performance of the algorithms [Weinberger *et al.*, 2006]. To address this challenge, metric learning, whose goal is to learn a distance metric that can capture the important relationships among data samples, has drawn significant attention [Huai *et al.*, 2018a; Weinberger *et al.*, 2006; Huang *et al.*, 2015; St Amand and Huan, 2017; Zadeh *et al.*, 2016; Huai *et al.*, 2018b; Suo *et al.*, 2018]. The basic idea of most metric learning methods is first to learn a Mahalanobis distance metric, which is a linear mapping to project the original samples into a new feature space, and then determine the similarity of samples in the new feature space. However, these conventional Mahalanobis-based methods usually have inherent limits on their mapping capability, and thus fail to achieve good performance when handling data with nonlinear structures.

Given that deep learning has good capability of modeling the nonlinearity of samples, there has been significant effort [Huang *et al.*, 2016; Law *et al.*, 2017; Song *et al.*, 2016; Wang *et al.*, 2017; Ni *et al.*, 2017; Sohn, 2016] studying deep metric learning (DML), which unifies deep learning and metric learning into a joint learning framework. The key idea of DML is to explicitly train a deep neural network and derive a set of hierarchical nonlinear mappings, based on which the data samples can be projected into a new feature space for comparing. The derived nonlinear mappings are capable of guaranteeing that the distance between similar samples is close and the distance between dissimilar samples is far in the new feature space [Sohn, 2016]. Additionally, compared with the traditional metric learning methods, DML has shown better scaling properties when handling massive data. Although DML has achieved practical success in many applications, there is no existing work that theoretically analyzes the generalization error of DML, which is the difference between the empirical and expected errors, and it can measure how good a learned model is able to perform on unseen data. A comprehensive theoretical generalization analysis is essential for DML as it can not only provide much important information about the practical performance of DML but also guide the design of effective network architectures for DML.

In this paper, we try to fill up this research gap and derive the generalization bound for DML. Here we consider a general case where the DML models adopt the dropout strategies, in which each connection is kept in the neural network with a predefined retention rate during the training process. In particular, when these retention rates are set to ones, it reduces to the generalization bound of the DML model without dropout. Based on the derived generalization bound, we can have a good understanding of the generalization properties of DML in many applications, especially in the settings where dropout is used to train DML models with the goal of achieving good generalization performance [Qian *et al.*, 2014]. In practice, specifying these predefined retention rates for dropout is usually difficult as it requires significant levels of experience and

domain knowledge. However, the derived generalization error bound for DML can be treated as a function related to the weight parameters of the neural networks and the retention rates for dropout. Based on this fact, we propose a novel Adaptive Dropout based DML method (**ADroDML**) by incorporating the obtained generalization bound to the objective function of DML models as a regularizer. The goal of incorporating the bound-based regularizer is to reduce the model complexity for DML to give a lower error on future unseen data. ADroDML allows us to jointly learn the weight parameters and the retention rates for DML in a theoretically justified way, and it can achieve better performance compared with existing DML works that require predefined retention rates. Extensive experiments on real-world datasets verify the findings derived from the error bound and show the effectiveness of the proposed ADroDML.

2 Preliminary

In this section, we introduce the DML model that takes dropout into account. Without loss of generality, we use the widely adopted Siamese network [Huang *et al.*, 2016; Law *et al.*, 2017; Sun *et al.*, 2014] as our example. Suppose $\boldsymbol{z} = \{\boldsymbol{z}_i = (\boldsymbol{x}_i, y_i)\}_{i=1}^n$ denote the labeled training dataset, where $\boldsymbol{x}_i \in \mathbb{R}^d$ is a d dimensional feature vector and $y_i \in \{-1, 1\}$ is the class label. DML aims to train a L-layer neural network to predict whether two input samples (i.e., two feature vectors) are similar or not. Assume that the trained L-layer neural network is parameterized by the weights $\boldsymbol{W} = \{\boldsymbol{W}^l \in \mathbb{R}^{h_l * h_{l-1}}\}_{l=1}^L$ (note that the biases are included in $\boldsymbol{W}$ with a corresponding fixed input of 1 for simplicity), where h_l represents the number of neurons in the l-th layer of the network and $h_0 = d$. We denote the retention rates for dropout as $\boldsymbol{\rho} = \{\rho^l\}_{l=1}^L$, where ρ^l represents the retention rate for the l-th layer. Then, given the input sample $\boldsymbol{x}_i \in \mathbb{R}^d$, the output of the l-th layer in the network can be written as

$$
\begin{aligned}
&f^l(\boldsymbol{x}_i; \boldsymbol{W}^{1:l}, \boldsymbol{M}^{1:l}) \qquad (1)\\
&= (\boldsymbol{W}^l \odot \boldsymbol{M}^l)\sigma(f^{l-1}(\boldsymbol{x}_i, \boldsymbol{W}^{1:l-1}, \boldsymbol{M}^{1:l-1}))\\
&= (\boldsymbol{W}^l \odot \boldsymbol{M}^l)\sigma((\boldsymbol{W}^{l-1} \odot \boldsymbol{M}^{l-1})\sigma(\cdots\sigma((\boldsymbol{W}^1 \odot \boldsymbol{M}^1)\boldsymbol{x}_i))),
\end{aligned}
$$

where $\sigma(\cdot)$ denotes the activation function, $\boldsymbol{M}^{1:l} = \{\boldsymbol{M}^1, \boldsymbol{M}^2, ..., \boldsymbol{M}^l\}$, and $\boldsymbol{M}^l = \{M_{ij}^l\}_{i=1,j=1}^{h_l, h_{l-1}} \in \mathbb{R}^{h_l * h_{l-1}}$ is a binary matrix where each element $M_{ij}^l \in \{0, 1\}$ is drawn from the distribution $Bern(\rho^l)$. The term $(\boldsymbol{W}^l \odot \boldsymbol{M}^l)$ corresponds to dropping each of the weight parameters $\boldsymbol{W}^l = \{W_{ij}^l\}_{i=1,j=1}^{h_l, h_{l-1}}$ independently with probability $1 - \rho^l$. In particular, $f^1(\boldsymbol{x}_i, \boldsymbol{W}^1, \boldsymbol{M}^1) = (\boldsymbol{W}^1 \odot \boldsymbol{M}^1)\boldsymbol{x}_i$. Note that the most top level representation of the input $\boldsymbol{x}_i$, i.e.,$f^L(\boldsymbol{x}_i; \boldsymbol{W}^{1:L}, \boldsymbol{M}^{1:L})$, is a random vector due to the adopted Bernoulli random variable $\boldsymbol{M}$. Thus, following [Ma *et al.*, 2016; Zhai and Wang, 2018; Wan *et al.*, 2013], we use the expected value $f^L(\boldsymbol{x}_i; \boldsymbol{W}, \boldsymbol{\rho}) = E_{\boldsymbol{M}}[f^L(\boldsymbol{x}_i; \boldsymbol{W}, \boldsymbol{M})]$ as the deterministic output of the neural network with dropout.

Specifically, DML aims to seek the nonlinear embedding function $f^L : \mathbb{R}^d \to \mathbb{R}^{h_L}$, which guarantees that the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is smaller than a pre-specified margin $\gamma > 0$ in the transformed space if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are similar, and larger than γ in the transformed space if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are dissimilar. To learn a good embedding function f^L with such desirable properties, the widely adopted method is to minimize the following empirical risk with dropout over the given training samples

$$
\begin{aligned}
\mathcal{R}_{\boldsymbol{z}}(\boldsymbol{W}) = &\frac{2}{n(n-1)} \sum_{i<j} g(1 + y_{ij}(D(f^L(\boldsymbol{x}_i; \boldsymbol{W}, \boldsymbol{\rho}),\\
&f^L(\boldsymbol{x}_j; \boldsymbol{W}, \boldsymbol{\rho})) - \gamma)), \qquad (2)
\end{aligned}
$$

where $y_{ij} = y_i y_j \in \{-1, 1\}$ is the similarity label, γ is the unit margin and $g(\cdot)$ is the hinge loss. $D(f^L(\boldsymbol{x}_i, \boldsymbol{W}, \boldsymbol{\rho}), f^L(\boldsymbol{x}_j; \boldsymbol{W}, \boldsymbol{\rho}))$ is defined as follows:

$$
\begin{aligned}
&D(f^L(\boldsymbol{x}_i, \boldsymbol{W}, \boldsymbol{\rho}), f^L(\boldsymbol{x}_j; \boldsymbol{W}, \boldsymbol{\rho})) \qquad (3)\\
&= (f^L(\boldsymbol{x}_i; \boldsymbol{W}, \boldsymbol{\rho}) - f^L(\boldsymbol{x}_j; \boldsymbol{W}, \boldsymbol{\rho}))^T (f^L(\boldsymbol{x}_i; \boldsymbol{W}, \boldsymbol{\rho})\\
&- f^L(\boldsymbol{x}_j; \boldsymbol{W}, \boldsymbol{\rho})) = \sum_{k=1}^{h_L} (f_k^L(\boldsymbol{x}_i; \boldsymbol{W}, \boldsymbol{\rho}) - f_k^L(\boldsymbol{x}_j; \boldsymbol{W}, \boldsymbol{\rho}))^2,
\end{aligned}
$$

where $f_k^L(\cdot)$ represents the output of the k-th neuron in the L-th layer of the network. In practice, the deterministic output $f^L(\boldsymbol{x}_i; \boldsymbol{W}, \boldsymbol{\rho})$ can be derived by introducing a deterministic scaling factor (i.e., $E[\boldsymbol{M}^l]$) for each layer to replace the random dropout variable [Zhai and Wang, 2018; Srivastava *et al.*, 2014]. Then, the empirical loss $\mathcal{R}_{\boldsymbol{z}}(\boldsymbol{W})$ can be easily computed since it only involves a single deterministic network. Note that $\mathcal{R}_{\boldsymbol{z}}(\boldsymbol{W})$ is also known as the contrastive loss and can measure how well f^L is able to place similar samples nearby and keep dissimilar samples separated.

3 Generalization Analysis for Deep Metric Learning

In this section, we derive the generalization bound for DML, which is the difference between the expected and empirical risks. We derive the generalization bound by analyzing

$$
\text{the generalization bound} \triangleq \mathcal{R}(\boldsymbol{W}_{\boldsymbol{z}}) - \mathcal{R}_{\boldsymbol{z}}(\boldsymbol{W}_{\boldsymbol{z}}), \qquad (4)
$$

where $\boldsymbol{W}_{\boldsymbol{z}} = \arg\min_{\boldsymbol{W}} \mathcal{R}_{\boldsymbol{z}}(\boldsymbol{W})$ denotes the empirical risk minimizer and $\mathcal{R}(\boldsymbol{W}_{\boldsymbol{z}}) = E_{\boldsymbol{z}}[\mathcal{R}_{\boldsymbol{z}}(\boldsymbol{W}_{\boldsymbol{z}})]$ represents the expected risk of the trained model on the whole space of possible data. Note that the empirical risk $\mathcal{R}_{\boldsymbol{z}}(\boldsymbol{W}_{\boldsymbol{z}})$ is also known as a U-statistic [Clémençon *et al.*, 2005] in the statistic literature, and is no longer an empirical average of independent random samples from $\boldsymbol{z}$ as in the standard deep learning setting, but rather an average of *pairs* of random samples from $\boldsymbol{z}$. Thus, it is more challenging to perform generalization analysis for DML. To address this challenge, we develop a novel analysis method by extending Rademacher complexity analysis [Shalev-Shwartz and Ben-David, 2014] to the setting of DML, and then we derive Theorem 1 that describes the generalization bound for DML with dropout.

Theorem 1. ***(Generalization Bound for DML with Dropout)*** *Suppose the deep neural network needed to be learned has L layers and the weight parameter $\boldsymbol{W}^l$ ($l \in [L]$) satisfies $\|\boldsymbol{W}^l\|_F \leq B^l$. Let $\sigma(\cdot)$ be a 1-Lipschitz activation function (e.g., ReLU) and $\boldsymbol{\mathcal{X}} \in [0, 1]^d$ denote the feature space. Then for any $\delta \in (0, 1)$, with probability $1 - \delta$ we have*

$$\mathcal{R}(\boldsymbol{W}_{\boldsymbol{z}}) - \mathcal{R}_{\boldsymbol{z}}(\boldsymbol{W}_{\boldsymbol{z}}) \leq 3V_1\sqrt{\frac{2\log(1/\delta)}{n}} + 6\sqrt{1/\lfloor\frac{n}{2}\rfloor} \quad (5)$$
$$+ 32h_L B^L V^L(\sqrt{2L\log 2}+1)(\prod_{l=1}^{L} B^l)(\prod_{l=1}^{L}\sqrt{\rho^l})\sqrt{d/\lfloor\frac{n}{2}\rfloor},$$

where $|g(\cdot)| \leq V_1$, and $\|\sigma(f^{L-1}(\cdot))\| \leq V^L$. Note that $g(\cdot)$ denotes the loss function, and $f^{L-1}(\cdot)$ represents the output of the $(L-1)$-th layer of the learned neural network.

Proof. Due to space limitation, the detailed proof for this theorem is provided in the full version. □

Observations. With Theorem 1, we can derive the following observations, which can help explain the behaviors of existing DML models and guide the design of good neural networks for DML.

- The generalization bound (i.e., $\mathcal{R}(\boldsymbol{W}_{\boldsymbol{z}}) - \mathcal{R}_{\boldsymbol{z}}(\boldsymbol{W}_{\boldsymbol{z}})$) decreases monotonically at the rate of $\mathcal{O}(\sqrt{1/n})$ when the training data size n increases. In particular, $(\mathcal{R}(\boldsymbol{W}_{\boldsymbol{z}}) - \mathcal{R}_{\boldsymbol{z}}(\boldsymbol{W}_{\boldsymbol{z}})) \to 0$ when $n \to +\infty$, which indicates that the DML models can achieve good generalization performance when the training data size is sufficiently large.
- The generalization bound is also related to the dimensionality of the feature vector (i.e., d). In the cases where the value of d is extremely small (e.g., $d = 0$), we may get a small generalization bound. However, the empirical loss may be very large since the learned model cannot capture the particular characteristics of the data [Zhai and Wang, 2018]. On the other hand, high-dimensional input features (i.e., d is extremely large) usually contain much noisy information, which can hide the relationship between the learning task and the most relevant features [Mason *et al.*, 2017] and thus incurs large generalization bounds. Thus, a proper feature set that dominates the underlying learning task should be selected, and are then fed into the network.
- The magnitude of the weight parameters (i.e., $\|\boldsymbol{W}^l\|_F$) at the end of the learning process is critical to the generalization performance, and small magnitude of the weights is preferred. By observing this, explicit regularizers can be imposed on the weight parameters, which is achieved by penalizing the norm of the optimal solution. In this way, the generalization bound can be dramatically reduced when the magnitude of the weight parameters are very large. Also, weight-decay can be adopted to avoid choosing large-magnitude weights, which can improve the generalization performance.
- The multiplicative term $\prod_{l=1}^{L}\sqrt{\rho^l}$ which is related to the retention rates of dropout helps us to understand how the dropout method works. When $\rho^l = 0$, the above bound is tight since the features from the training samples have no influence on the output [Zhai and Wang, 2018]. When $\forall l \in [L]$, $\rho^l = 1$, it reduces to the complexity of a standard model. That is to say, when these retention rates are set to ones, it reduces to the generalization bound of the DML model without dropout. For other cases where $\rho_l \in (0, 1)$, we can obtain that $\prod_{l=1}^{L}\sqrt{\rho^l} < 1$, which means that dropout can reduce the generalization bound. However, when continuously decreasing the retention rates for dropout, the quality of the learned model may deteriorate [Zhai and Wang, 2018]. The reason is that the learned model may be tuned to the particular training samples, rather than the underlying characteristics of the data. Thus, specifying optimal retention rates for dropout is very crucial in practice.
- The generalization bound is also affected by V^L, which denotes the bounded output range of the activation function $\sigma(\cdot)$ on the $(L-1)$-th layer. This theorem implies that batch normalization can improve the generalization performance as it is an operator that normalizes the output of the previous layer within each mini-batch, especially in the settings where the output range is extremely large.

Two other factors that affect the derived generalization bound are L and h_L, which can provide suggestions on non-extreme-deep neural networks and non-extreme-wide output layers, respectively. As we can see, all the above observations are consistent with the widely used network architectures in practice. Additionally, Theorem 1 can be easily generalized to other situations where different loss functions (e.g., the triplet network-based DML models), activation functions and types of dropout (e.g., dropout of both weights and units) are adopted.

4 Adaptive Dropout for Deep Metric Learning

The above generalization analysis implies that taking dropout into account during the training process can help to reduce the generalization error of DML. However, the retention rates for dropout are usually pre-defined based on the experience and domain knowledge, and fixed throughout the training process. It is still not clear how to choose the optimal retention rates such that the learned DML model can achieve the best performance. To address this challenge, in this section, we propose a novel adaptive dropout based DML method (called **ADroDML**) by incorporating the derived generalization error bound into the objective function of DML as a regularizer, and let this error bound guide the choice of the retention rates. Specifically, the retention rates $\boldsymbol{\rho} = \{\rho^l\}_{l=1}^{L}$ for dropout and the weight parameters $\boldsymbol{W} = \{\boldsymbol{W}^l\}_{l=1}^{L}$ of the network are unified into one objective function, which is defined as

$$\min_{\boldsymbol{W},\boldsymbol{\rho}} \mathcal{L}(\boldsymbol{W},\boldsymbol{\rho}) = \frac{2}{n(n-1)}\sum_{i<j} g(1 + y_{ij}(D(f^L(\boldsymbol{x}_i,\boldsymbol{W},\boldsymbol{\rho}), f^L(\boldsymbol{x}_j,\boldsymbol{W},\boldsymbol{\rho})) - \gamma)) + \beta\Omega(\boldsymbol{W},\boldsymbol{\rho}), \quad (6)$$

where $\beta > 0$ is the associated regularization parameter, and the regularization term $\Omega(\boldsymbol{W},\boldsymbol{\rho})$ is derived from the generalization bound given in Eq. (5). There are two terms in the right hand side of Eq. (6). The first term is used to penalize the large distance between similar sample pairs and penalize the small distance between dissimilar instance pairs. The goal of the second term is to reduce the model complexity for DML to give lower error on future unseen data. The term $\Omega(\boldsymbol{W},\boldsymbol{\rho})$ in Eq. (6) is computed as follows

$$\Omega(\boldsymbol{W},\boldsymbol{\rho}) = \Delta * (\prod_{l=1}^{L}\|\boldsymbol{W}^l\|_F)(\prod_{l=1}^{L}\sqrt{\rho^l}),$$

Dataset	Size	Dimension
MNIST 8v9	2,016	28×28
Bone disease	9,704	672
Wine quality	4,898	11

Table 1: The statistics of the adopted datasets.

Dataset	# units in the three layers
MNIST 8v9	(784, 64, 10)
Bone disease	(672, 64, 10)
Wine quality	(11, 8, 4)

Table 2: The number of units in each layer of the neural networks.

where $\Delta = 32h_L B^L V^L \sqrt{d/\lfloor \frac{n}{2} \rfloor}(\sqrt{2L\log 2}+1)$. Note that the first two terms in the right hand of Eq. (5) are omitted since they do not contain either the weight parameters or the retention probability parameters.

Optimization. Next, we discuss how to solve the optimization problem formulated in Eq. (6), where we have two sets of parameters that need to be learned, i.e., $\boldsymbol{W} = \{\boldsymbol{W}^l\}_{l=1}^L$ and $\boldsymbol{\rho} = \{\rho^l\}_{l=1}^L$. Here we solve this optimization problem using the block-coordinate descent algorithm [Bertsekas, 1999], which starts with an initial setting of the parameters, and then optimize $\boldsymbol{W}$ and $\boldsymbol{\rho}$ in an alternating fashion. Specifically, we iteratively conduct the following two steps:

Step 1: Weights update. With an initial estimate of the retention rates $\boldsymbol{\rho} = \{\rho^l\}_{l=1}^L$, we first update the weight parameters $\boldsymbol{W} = \{\boldsymbol{W}^l\}_{l=1}^L$ by minimizing $\mathcal{L}(\boldsymbol{W}, \boldsymbol{\rho})$ with fixed $\boldsymbol{\rho} = \{\rho^l\}_{l=1}^L$. By solving this optimization problem, we can then obtain the set of weight parameters $\boldsymbol{W} = \{\boldsymbol{W}^l\}_{l=1}^L$ which minimize $\mathcal{L}(\boldsymbol{W}, \boldsymbol{\rho})$ with the fixed retention rates.

Step 2: Retention rates update. In this step, we fix the weight matrices $\boldsymbol{W} = \{\boldsymbol{W}^l\}_{l=1}^L$ for different layers, and then calculate the retention rates $\boldsymbol{\rho} = \{\rho^l\}_{l=1}^L$ through minimizing the objective function $\mathcal{L}(\boldsymbol{W}, \boldsymbol{\rho})$ given in Eq. (6).

The above two steps are iteratively conducted until the convergence criterion is satisfied. In this paper, the convergence criterion is that the difference between the objective function values in two consecutive iterations is less than a threshold. Compared with the dropout strategy that requires to specify the retention rates in advance, the bound-based regularizer enables us to adaptively optimize the objective and adjust the retention rates for dropout in a theoretically justified way.

5 Experiments

5.1 Experimental Setup

Datasets. We adopt the following real-world datasets for our experiments: *the MNIST 8v9 dataset*[1], *the bone disease dataset*[2], and *the wine quality dataset* [3]. The statistics of these datasets are provided in Table 1.

Model settings. Unless otherwise specified, all the neural networks adopted in the experiments have 3 layers. For each dataset, the number of the units in each layer of the neural network is provided in Table 2. We implement the DML model using Google Tensorflow, and the training process is done locally using NVIDIA GeForce GTX 1060 GPU. Additionally, Adam optimizer is used in the training process for DML and the learning rate is set as $1e-4$. As for the activation function, we use ReLU because it is a 1-Lipschitz activation function and satisfies the Lipschitz-continuous condition.

[1]http://yann.lecun.com/exdb/mnist/

[2]https://sofonline.epi-ucsf.org/interface/

[3]https://archive.ics.uci.edu/ml/datasets.php

5.2 Experiments for Theoretical Observations

We first conduct experiments to verify the derived theoretical observations in Section 3. Specifically, we evaluate the effect of the training data size, batch normalization, regularization, dropout and the input feature dimension on the generalization behavior of DML. Note that the input for DML is a set of sample pairs instead of individual samples. For each dataset, 4,950 sample pairs are selected as the test set (no overlap with training set). Unless otherwise specified, the training data sizes for the MNIST 8v9 dataset, the bone disease dataset and the wine quality dataset are set as 160, 220 and 100, respectively. Correspondingly, the numbers of the generated training sample pairs for the MNIST 8v9 dataset, the bone dataset and the wine quality dataset are 12,720, 24,090 and 4,950, respectively. We do not use the validation set to tune parameters, but assign values by standard settings. When evaluating the effect of the training data size, batch normalization, regularization and dropout, we report the testing loss because the generalization bound is mainly used to measure how well the learned ML model performs on the unseen data (test data). Additionally, we also evaluate the impact of the input feature dimension on the empirical training loss.

The effect of the training data size. To investigate the effect of the training data size (i.e., n) on the generalization behavior of DML, we train the model with different training data sizes and then calculate the testing loss. Here we consider three cases where the training data sizes are set as 20, 50 and 110, respectively. Figure 1a and Figure 1d show the results on the MNIST 8v9 and bone disease datasets when the number of batches varies. From the two figures, we can see that the larger the training data size, the smaller the testing loss. This verifies that increasing the training data size can potentially improve the generalization performance on unseen data.

The effect of batch normalization. Then we evaluate the effect of batch normalization on the generalization behavior of DML. Here we still adopt the MNIST 8v9 and bone disease datasets. For each dataset, we train the model with and without batch normalization, respectively, and then calculate the testing loss. The results for the two datasets are shown in Figure 1b and Figure 1e, from which we can see the testing loss of the model trained with batch normalization is lower than that of the model trained without batch normalization. The results verify that batch normalization play an important role to generalize DML.

The effect of regularization. We also evaluate the effect of regularization for DML through explicitly comparing the performance of the DML models with and without regularization. Then we report the calculated testing losses on the MNIST 8v9 and bone disease datasets in Figure 1c and Figure 1f, respectively. From the two figures we can see the models with

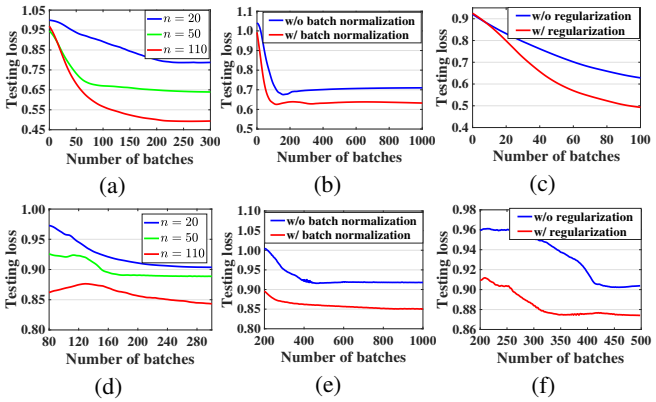


Figure 1: The testing loss of the DML model on the MNIST 8v9 dataset (a-c) and the bone disease dataset (d-f). (a) and (d): The effect of the training data size. (b) and (e): The effect of batch normalization. (c) and (f): The effect of regularization.

regularization perform much better than those without regularization. That is to say, regularization can help to improve the generalization performance.

The effect of dropout. Next, we analyze the effect of dropout on the performance of the DML model. In this experiment, we adopt the MNIST 8v9, bone disease and wine quality datasets. We vary the retention rate from 0.3 to 1.0 for the MNIST 8v9 dataset, from 0.1 to 1.0 for the bone disease dataset and from 0.1 to 0.9 for the wine quality dataset. Note that when the retention rate for each layer is set as 1.0 (i.e., $\forall l \in [L], \rho^l = 1.0$), it is the case without dropout. The calculated testing losses for the three datasets are shown in Figure 2. The results show that the DML model trained with dropout performs better than that trained without dropout, which means dropout can improve the model's generalization ability. Additionally, from this figure we can see that smaller retention rate does not mean better generalization performance. This result also accords with the previous theoretical analysis.

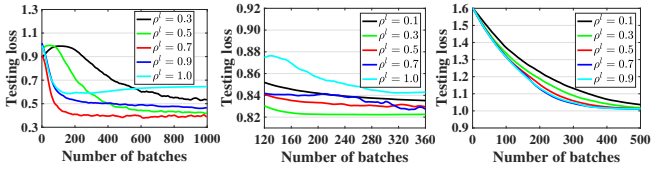


Figure 2: The testing loss of the DML model under different retention rates on the MNIST 8v9, bone disease and wine-quality datasets.

The effect of the input feature dimension. Finally, we evaluate the effect of the input feature dimension (i.e., d) on the training loss. Here we adopt the MNIST 8v9 dataset. We first randomly select a subset of features to reduce the feature dimension of this dataset. Then, for this newly derived reduced-dimensional dataset, we randomly select $1,200$ samples as the training samples (i.e., $n = 1,200$). Additionally, we consider three neural network structures in this experiment, and all of them have three layers. The numbers of the units in different layers of the three network structures are (d, 64, 10), (d, 80, 10) and (d, 128, 16), respectively. Figure 3 reports the evolution of the training loss under various input feature dimensions (i.e., d). In this figure, each line denotes the evolution of the training loss for a specific value of d. We can see that the smaller the value of of the input feature dimension (i.e., d), the larger the training loss. This is also consistent with our previous theoretical analysis in Section 3 that when the value of the input feature dimension (i.e., d) is very small, the empirical training loss could be very large. The reason is that when the number of the randomly selected features is very small compared with that in the original unreduced dataset, most of the useful information in the original dataset cannot be captured, which means the learned DML model cannot capture the particular characteristic features in the original dataset. Thus, the empirical training loss becomes large.

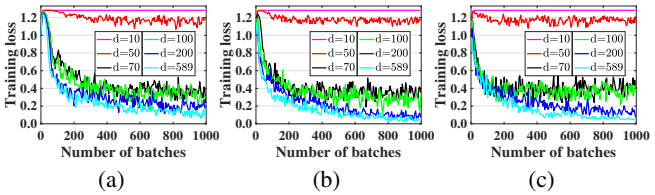


Figure 3: The training loss of the DML model for different input feature dimensions on the MNIST 8v9 dataset. The results in (a), (b) and (c) are for three different neural network structures, respectively.

5.3 Experiments for ADroDML

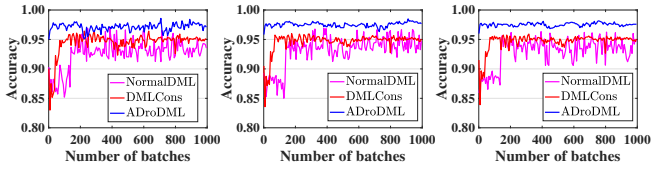
In this section, we evaluate the performance of ADroDML on the MNIST 8v9 and wine quality datasets. For each dataset, we first select 1,200 samples as the training set, and then use the remaining samples as the test set.

Baseline methods. We compare the performance of ADroDML with the following two baseline methods:

- *Normal DML training (NormalDML).* In this method, we use the standard contrastive loss to train a DML model and do not take the dropout strategy into consideration.
- *DML training with a constant dropout retention rate (DMLCons).* In this method, we consider the dropout strategy with a pre-defined retention rate. Here we set the value of the retention rate as 0.5 by following existing DML works [Hoffer and Ailon, 2015; Gouk *et al.*, 2015].

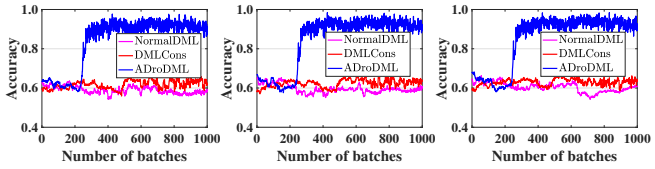
For the sake of fairness, the network structure for each of the baseline methods is the same as that of ADropDML.

Performance. In this experiment, we use the standard K-nearest neighbor algorithm (KNN) as the classifier, which means for each given test sample, its label is assigned by majority voting over its top-K nearest samples in the training set. Here we consider three cases where the value of K is set as 3, 5 and 7, respectively. In Figure 4 and Figure 5, we respectively report the classification accuracy of ADroDML on the MNIST 8v9 dataset and the wine quality dataset. The results in the two figures show that our proposed ADroDML can achieve the best performance in all cases. When $K = 3$, the classification accuracy of ADroDML on the wine quality dataset is around 97.8% while that of the two baseline methods (i.e., NormalDML and DMLCons) is around 60.0% and 62.8%, respectively. The main reason is that the proposed



(a) K=3 (b) K=5 (c) K=7

Figure 4: Classification accuracy of the proposed ADroDML on the MNIST 8v9 dataset.



(a) K=3 (b) K=5 (c) K=7

Figure 5: Classification accuracy of the proposed ADroDML on the wine quality dataset.

ADroDML can adaptively learn the optimal dropout retention rates to avoid the overfitting problem.

Convergence. Next, we evaluate the convergence of ADroDML through calculating the training loss in each batch of the training process. Figure 6 reports the experimental results on the MNIST 8v9 dataset. Here we conduct the experiment for three times. Each time the training data are randomly selected from the dataset. From this figure, we can see that the training loss gradually converges to zero with the increase of the number of batches. This confirms that the convergence can be guaranteed in our proposed method ADroDML.

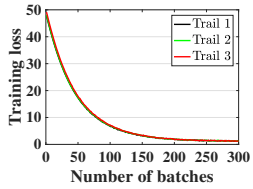


Figure 6: The training loss of AdroDML w.r.t Number of batches on the MNIST 8V9 dataset.

6 Related Work

Based on the types of neural networks, the existing DML works can be roughly divided into the following three categories: (1) **The Siamese network based DML methods** [Huang *et al.*, 2016; Law *et al.*, 2017] are trained by minimizing a contractive loss function, where the task is to minimize the distance between similar sample pairs and to push the pairwise distance between dissimilar pairs larger than a fixed margin. (2) **The triplet network based DML methods** [Song *et al.*, 2016; Wang *et al.*, 2017] are trained by minimizing a triplet loss function, and the triplets are usually generated based on the class labels of the training dataset. (3) There are also some **DML methods based on other types of networks** [Ni *et al.*, 2017; Sohn, 2016]. However, all the existing works do not provide generalization analysis for DML. Additionally, they do not consider how to derive the optimal retention rates for dropout.

Although there are some works providing the theoretical analysis for traditional linear metric learning [Jin *et al.*, 2009; Cao *et al.*, 2016], they can not be directly used for analyzing the generalization bound of DML. Even though [Cao *et al.*, 2016] also adopts the Rademacher complexity, this work is significantly different from ours. First of all, we provide the generalization bound for DML that solves the *nonlinear transformation* problem, while [Cao *et al.*, 2016] presents the bound for traditional metric learning that solves the *linear transformation* problem. Thus, the problem studied in our paper is more challenging than that in [Cao *et al.*, 2016]. Secondly, to make the theoretical analysis more general, we take into account *the dropout strategy* for DML models, which is unfortunately not considered in [Cao *et al.*, 2016].

Additionally, adding a regularization related to a specific upper bound to learn the parameters in an optimal way has many practical applications [Raghunathan *et al.*, 2018; Zhai and Wang, 2018]. However, their settings are different from ours. For example, [Zhai and Wang, 2018] considers the classification model with a final softmax layer, which assumes that the input data are i.i.d. However, we aim to derive the bound for DML, where the input data are not independent, making our problem more challenging. Moreover, *the techniques used to derive the bound* in our paper and those used in [Zhai and Wang, 2018] are significantly different. In particular, the derived bound based on the techniques in [Zhai and Wang, 2018] has an exponential dependency (i.e., 2^L) on the network length L, which is not appealing even for moderate networks. In contrast, based on our proposed techniques, the derived bound only linearly depends on L. Finally, *the dropout strategy* considered in our paper and that in [Zhai and Wang, 2018] are different. In our paper, we consider randomly dropping out each connection, whereas they considered randomly dropping out each hidden neuron, which is more restrictive than ours.

7 Conclusions

In this paper, we present the generalization error bound for DML and analyze the findings derived from this bound. Additionally, we propose a novel method (ADroDML) to adaptively adjust the dropout rates for DML based on the derived generalization bound. Compared with existing DML works that require predefined dropout rates, ADroDML can adaptively learn the dropout retention rates for DML in a theoretically justified way. We also conduct experiments on real-world datasets to verify the findings derived from the generalization bound and test the effectiveness of the proposed adaptive method. The experimental results are consistent with our theoretical analysis, and they also show that the proposed ADroDML can achieve much better performance compared with the baselines.

Acknowledgments

This work was supported in part by the US National Science Foundation (NSF) under grant IIS-1514204. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

References

[Bertsekas, 1999] Dimitri P Bertsekas. *Nonlinear programming*. Athena scientific Belmont, 1999.

[Cao *et al.*, 2016] Qiong Cao, Zheng-Chu Guo, and Yiming Ying. Generalization bounds for metric and similarity learning. *Machine Learning*, 102(1):115–132, 2016.

[Clémençon *et al.*, 2005] Stéphan Clémençon, Gábor Lugosi, and Nicolas Vayatis. Ranking and scoring using empirical risk minimization. In *Proceedings of COLT*, pages 1–15, 2005.

[Gouk *et al.*, 2015] Henry Gouk, Bernhard Pfahringer, and Michael Cree. Fast metric learning for deep neural networks. *arXiv preprint arXiv:1511.06442*, 2015.

[Hoffer and Ailon, 2015] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, 2015.

[Huai *et al.*, 2018a] Mengdi Huai, Chenglin Miao, Yaliang Li, Qiuling Suo, Lu Su, and Aidong Zhang. Metric learning from probabilistic labels. In *Proceedings of SIGKDD*, pages 1541–1550, 2018.

[Huai *et al.*, 2018b] Mengdi Huai, Chenglin Miao, Qiuling Suo, Yaliang Li, Jing Gao, and Aidong Zhang. Uncorrelated patient similarity learning. In *Proceedings of SDM*, pages 270–278, 2018.

[Huang *et al.*, 2015] Zhiwu Huang, Ruiping Wang, Shiguang Shan, Xianqiu Li, and Xilin Chen. Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *Proceedings of ICML*, pages 720–729, 2015.

[Huang *et al.*, 2016] Chen Huang, Chen Change Loy, and Xiaoou Tang. Local similarity-aware deep feature embedding. In *Proceedings of NeurIPS*, pages 1262–1270, 2016.

[Jin *et al.*, 2009] Rong Jin, Shijun Wang, and Yang Zhou. Regularized distance metric learning: Theory and algorithm. In *Proceedings of NeurIPS*, pages 862–870, 2009.

[Law *et al.*, 2017] Marc T Law, Raquel Urtasun, and Richard S Zemel. Deep spectral clustering learning. In *Proceedings of ICML*, pages 1985–1994, 2017.

[Ma *et al.*, 2016] Xuezhe Ma, Yingkai Gao, Zhiting Hu, Yaoliang Yu, Yuntian Deng, and Eduard Hovy. Dropout with expectation-linear regularization. *arXiv preprint arXiv:1609.08017*, 2016.

[Mason *et al.*, 2017] Blake Mason, Lalit Jain, and Robert Nowak. Learning low-dimensional metrics. In *Proceedings of NeurIPS*, pages 4139–4147, 2017.

[Ni *et al.*, 2017] Jiazhi Ni, Jie Liu, Chenxin Zhang, Dan Ye, and Zhirou Ma. Fine-grained patient similarity measuring using deep metric learning. In *Proceedings of CIKM*, pages 1189–1198, 2017.

[Qian *et al.*, 2014] Qi Qian, Juhua Hu, Rong Jin, Jian Pei, and Shenghuo Zhu. Distance metric learning using dropout: a structured regularization approach. In *Proceedings of SIGKDD*, pages 323–332, 2014.

[Raghunathan *et al.*, 2018] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. In *Proceedings of ICML*, 2018.

[Shalev-Shwartz and Ben-David, 2014] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[Sohn, 2016] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Proceedings of NeurIPS*, pages 1857–1865, 2016.

[Song *et al.*, 2016] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of CVPR*, pages 4004–4012, 2016.

[Srivastava *et al.*, 2014] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[St Amand and Huan, 2017] Joseph St Amand and Jun Huan. Sparse compositional local metric learning. In *Proceedings of Proceedings of SIGKDD*, pages 1097–1104, 2017.

[Sun *et al.*, 2014] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Proceedings of NeurIPS*, pages 1988–1996, 2014.

[Suo *et al.*, 2018] Qiuling Suo, Weida Zhong, Fenglong Ma, Yuan Ye, Mengdi Huai, and Aidong Zhang. Multi-task sparse metric learning for monitoring patient similarity progression. In *Proceedings of ICDM*, pages 477–486, 2018.

[Wan *et al.*, 2013] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In *Proceedings of ICML*, pages 1058–1066, 2013.

[Wang *et al.*, 2017] Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. Deep metric learning with angular loss. *arXiv preprint arXiv:1708.01682*, 2017.

[Weinberger *et al.*, 2006] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *Proceedings of NeurIPS*, pages 1473–1480, 2006.

[Zadeh *et al.*, 2016] Pourya Zadeh, Reshad Hosseini, and Suvrit Sra. Geometric mean metric learning. In *Proceedings of ICML*, pages 2464–2471, 2016.

[Zhai and Wang, 2018] Ke Zhai and Huan Wang. Adaptive dropout with rademacher complexity regularization. In *Proceedings of ICLR*, 2018.