Learning Latent Space Models with Angular Constraints

Pengtao Xie 1 2  Yuntian Deng 3  Yi Zhou 4  Abhimanu Kumar 5  Yaoliang Yu 6  James Zou 7  Eric P. Xing 2

Abstract
The large model capacity of latent space models (LSMs) enables them to achieve great performance on various applications, but meanwhile renders LSMs to be prone to overfitting. Several recent studies investigate a new type of regularization approach, which encourages components in LSMs to be diverse, for the sake of alleviating overfitting. While they have shown promising empirical effectiveness, in theory why larger “diversity” results in less overfitting is still unclear. To bridge this gap, we propose a new diversity-promoting approach that is both theoretically analyzable and empirically effective. Specifically, we use near-orthogonality to characterize “diversity” and impose angular constraints (ACs) on the components of LSMs to promote diversity. A generalization error analysis shows that larger diversity results in smaller estimation error and larger approximation error. An efficient ADMM algorithm is developed to solve the constrained LSM problems. Experiments demonstrate that ACs improve generalization performance of LSMs and outperform other diversity-promoting approaches.

1. Introduction
Latent space models (LSMs), such as sparse coding (Olshausen & Field, 1997), topic models (Blei et al., 2003) and neural networks, are widely used in machine learning to extract hidden patterns and learn latent representations of data. An LSM consists of a set of components. Each component aims at capturing one latent pattern and is parameterized by a weight vector. For instance, in a topic model (Blei et al., 2003), the components are referred to as topics, aiming at discovering the semantics underlying documents. Each topic is associated with a weight vector. The modeling power of LSMs can be very large when the number of components is large and the dimension of weight vectors is high. For instance, in the LightLDA (Yuan et al., 2015) topic model, the number of topics is 1 million and the dimension of topic vector is 50000, resulting in a topic matrix with 50 billion parameters. The vast model capacity of LSMs enables them to flexibly adapt to the complex patterns underlying data and achieve great predictive performance therefrom.

While highly expressive, LSMs are prone to overfitting, because of their large amount of model parameters. A key ingredient to successfully train LSMs is regularization, which imposes certain control over the model parameters to reduce model complexity and improve the generalization performance on unseen data. Many regularizers have been proposed, including $\ell_2$ regularization, $\ell_1$ regularization (Tibshirani, 1996), nuclear norm (Recht et al., 2010), Dropout (Srivastava et al., 2014) and so on.

Recently, a new type of regularization approaches (Yu et al., 2011; Zou & Adams, 2012a; Xie et al., 2015; 2016a; Rodríguez et al., 2016), which aim at encouraging the weight vectors of components in LSMs to be “diverse”, are emerging. Zou & Adams (2012b) apply Determinantal Point Process (Kulesza & Taskar, 2012) to encourage the topic vectors in LDA to be “diverse”. Bao et al. (2013) develop a softmax regularizer to promote incoherence among hidden units in neural network. Xie et al. (2015) propose an angle-based regularizer to “diversify” the weight vectors in Restricted Boltzmann Machine (RBM). While these approaches have demonstrated promising effectiveness on a wide range of empirical studies, in theory how they reduce overfitting is still unclear. One intuitive explanation could be: promoting diversity imposes a structural constraint on model parameters, which reduces the model capacity of LSMs and therefore alleviates overfitting. However, how to make this formal is challenging. In this paper, we aim to bridge this gap, by proposing a diversity-promoting approach that is both empirically effective and theoretically analyzable. We use near-orthogonality to represent “diversity” and propose to learn LSMs with angular constraints.
constraints (ACs) where the angle between components is constrained to be close to $\pi/2$, which hence encourages the components to be close to orthogonal (therefore “diverse”). Using sparse coding and neural network as study cases, we analyze how ACs affect the generalization performance of these two LSMs. The analysis shows that the more close to $\pi/2$ the angles are, the smaller the estimation error is and the larger the approximation error is. The best tradeoffs of these two errors can be explored by properly tuning the angles. We develop an alternating direction method of multipliers (ADMM) (Boyd et al., 2011) algorithm to solve the angle-constrained LSM (AC-LSM) problems. In various experiments, we demonstrate that ACs improve the generalization performance of LSMs and outperform other diversity-promoting regularization approaches.

The major contributions of this work are:

- We propose a new approach to promote diversity in LSMs, by imposing angular constraints (ACs) on components, for the sake of alleviating overfitting.
- We perform theoretical analysis on how ACs affect the generalization error of two exemplar LSMs: sparse coding and neural networks.
- We develop an efficient ADMM algorithm to solve the AC-LSM problems.
- Empirical evaluation demonstrates that ACs are very effective in reducing overfitting and outperforms other diversity-promoting approaches.

The rest of the paper is organized as follows. Section 2 reviews related works. Second 3 introduces the angle-constrained LSMs and Section 4 gives the theoretical analysis. Section 5 presents experimental results and Section 6 concludes the paper.

2. Related Works

Diversity-promoting learning of latent space models has been widely studied recently. Ramirez et al. (2010) define a regularizer based on squared Frobenius norm to encourage the dictionary in sparse coding to be incoherent. Zou & Adams (2012a) use the determinantal point process (DPP) (Kulesza & Taskar, 2012) to encourage the location vectors in Gaussian mixture model (GMM) and topics in latent Dirichlet allocation (Blei et al., 2003) to be “diverse”. In practice, this hard constraint might be too restrictive and hurts performance, as we will confirm in experiments. Bao et al. (2013) propose a softmax regularizer to encourage the weight vectors of hidden units to have small cosine similarity. Cogswell et al. (2015) propose to decorrelate hidden activations by minimizing their covariance. In convolutional neural networks (CNNs) where the number of activations is much larger than that of weight parameters, this regularizer is computationally prohibitive since it is defined over activations rather than weights. Henaff et al. (2016) perform a study to show that random orthogonal initialization of the weight matrices in recurrent neural networks improves its ability to perform long-memory tasks. Xiong et al. (2016) propose a structured decorrelation constraint which groups hidden units and encourages units within the same group to have strong connections during the training procedure and forces units in different groups to learn nonredundant representations by minimizing the cross-covariance between them. Rodríguez et al. (2016) show that regularizing negatively correlated features inhibits effective diversity and propose a solution which locally enforces feature orthogonality. Chen et al. (2017) propose a group orthogonal CNN which leverages side information to learn diverse feature representations. Mao et al. (2017) impose a stochastic decorrelation constraint based on covariance to reduce the co-adaptation of hidden units. Xie et al. (2017) define a kernel-based regularizer to promote diversity and analyze how it affects the generalization performance of neural networks (NNs). It is unclear how to generalize the analysis to other LSMs.

In the literature of neural networks, many works have studied the “diversification” of hidden units. Le et al. (2010) apply a strict-orthogonality constraint over the weight parameters to make the hidden units uncorrelated (therefore “diverse”). In practice, this hard constraint might be too restrictive and hurts performance, as we will confirm in experiments. Bao et al. (2013) propose a softmax regularizer to encourage the weight vectors of hidden units to have small cosine similarity. Cogswell et al. (2015) propose to decorrelate hidden activations by minimizing their covariance. In convolutional neural networks (CNNs) where the number of activations is much larger than that of weight parameters, this regularizer is computationally prohibitive since it is defined over activations rather than weights. Henaff et al. (2016) perform a study to show that random orthogonal initialization of the weight matrices in recurrent neural networks improves its ability to perform long-memory tasks. Xiong et al. (2016) propose a structured decorrelation constraint which groups hidden units and encourages units within the same group to have strong connections during the training procedure and forces units in different groups to learn nonredundant representations by minimizing the cross-covariance between them. Rodríguez et al. (2016) show that regularizing negatively correlated features inhibits effective diversity and propose a solution which locally enforces feature orthogonality. Chen et al. (2017) propose a group orthogonal CNN which leverages side information to learn diverse feature representations. Mao et al. (2017) impose a stochastic decorrelation constraint based on covariance to reduce the co-adaptation of hidden units. Xie et al. (2017) define a kernel-based regularizer to promote diversity and analyze how it affects the generalization performance of neural networks (NNs). It is unclear how to generalize the analysis to other LSMs.

Diversity-promoting learning has been investigated in non-LSM models as well. In multi-class classification, Malkin & Bilmes (2008) encourage the coefficient vectors of different classes to be diverse by maximizing the determi-
nant of the covariance matrix of the coefficient vectors. In classifiers ensemble, Yu et al. (2011) develop a regularizer to encourage the coefficient vectors of support vector machines (SVMs) to have small cosine similarity and analyze how this regularizer affects the generalization performance. The analysis is specific to SVM ensemble. It is unclear how to generalize it to latent space models.

3. Methods

In this section, we propose Angle-Constrained Latent Space Models (AC-LSMs) and present an ADMM algorithm to solve them.

3.1. Latent Space Models with Angular Constraints

An LSM consists of $m$ components and these components are parameterized by vectors $W = \{w_j\}_{j=1}^m$. Let $\mathcal{L}(W)$ denote the objective function of this LSM. Similar to (Bao et al., 2013; Xie et al., 2015; Rodríguez et al., 2016), we use angle to characterize diversity: the components are considered to be more diverse if they are close to being orthogonal, i.e., their angles are close to $\frac{\pi}{2}$. To encourage this, we require the absolute value of cosine similarity between each pair of components to be less than a small value $\tau$, which leads to the following angle-constrained LSM (AC-LSM) problem

$$\min_W \mathcal{L}(W) \quad \text{s.t.} \quad 1 \leq i < j \leq m, \frac{|w_i \cdot w_j|}{\|w_i\|_2 \|w_j\|_2} \leq \tau \tag{1}$$

The parameter $\tau$ controls the level of near-orthogonality (or diversity). A smaller $\tau$ indicates that the vectors are more close to being orthogonal, and hence are more diverse. As will be shown later, representing diversity using the angular constraints facilitates theoretical analysis and is empirically effective as well.

3.2. Case Studies

In this section, we apply the ACs to two LSMs.

**Sparse Coding** Given a set of data samples $\{x_i\}_{i=1}^n$, where $x \in \mathbb{R}^d$, sparse coding (SC) (Olshausen & Field, 1997) aims to use a set of “basis” vectors (referred to as dictionary) $W = \{w_j\}_{j=1}^m$ to reconstruct the data samples. Each data sample $x$ is reconstructed by taking a sparse linear combination of the basis vectors $x \approx \sum_{j=1}^m \alpha_j w_j$ where $\{\alpha_j\}_{j=1}^m$ are the linear coefficients (referred to as sparse codes) and most of them are zero. The reconstruction error is measured using the squared $\ell_2$ norm $\|x - \sum_{j=1}^m \alpha_j w_j\|_2^2$. To achieve sparsity among the coefficients, $\ell_1$-regularization is utilized: $\sum_{j=1}^m |\alpha_j|$. To avoid the degenerated case where most coefficients are zero and the basis vectors are of large magnitude, $\ell_2$-regularization is applied to the basis vectors: $\|w_j\|_2^2$. Putting these pieces together, we learn the basis vectors and sparse codes (denoted by $A$) by minimizing the following objective function:

$$\min_W \mathcal{L}(W, A) \quad \text{s.t.} \quad \sum_i \frac{|w_i \cdot w_j|}{\|w_i\|_2 \|w_j\|_2} \leq \tau \tag{2}$$

**Neural Networks** In a neural network (NN) with $L$ hidden layers, each hidden layer $l$ is equipped with $m^{(l)}$ units and each unit $i$ is connected with all units in layer $l - 1$. Hidden unit $i$ at layer $l$ is parameterized by a weight vector $w_i^{(l)}$. These hidden units aim at capturing latent features underlying data. Applying ACs to the weight vectors of hidden units, we obtain the following AC-SC problem:

$$\min_W \mathcal{L}(W) \quad \text{s.t.} \quad \forall 1 \leq l \leq L, 1 \leq i < j \leq m^{(l)}, \frac{|w_i^{(l)} \cdot w_j^{(l)}|}{\|w_i^{(l)}\|_2 \|w_j^{(l)}\|_2} \leq \tau$$

where $W$ denotes weight vectors in all layers and $\mathcal{L}(W)$ is the objective function of this NN.

3.3. Algorithm

In this section, we develop an ADMM-based algorithm to solve the AC-LSM problem. To make it amenable for optimization, we first factorize each weight vector $w$ into its $\ell_2$ norm $g = \|w\|_2$ and direction $\tilde{w} = \frac{w}{\|w\|_2}$. Under such a factorization, $w$ can be reparameterized as $w = g \tilde{w}$, where $g > 0$ and $\|\tilde{w}\|_2 = 1$. Then the problem defined in Eq.(1) can be transformed into

$$\min_{\tilde{W}, \tilde{G}} \mathcal{L}({\tilde{W}}, {\tilde{G}}) \quad \text{s.t.} \quad \forall j, g_j \geq 0, \|\tilde{w}_j\|_2 = 1, \forall i \neq j, |\tilde{w}_i \cdot \tilde{w}_j| \leq \tau \tag{3}$$

where $\tilde{W} = \{\tilde{w}_j\}_{j=1}^m$ and $\tilde{G} = \{g_j\}_{j=1}^m$. We solve this new problem by alternating between $W$ and $G$. Fixing $\tilde{W}$, the problem defined over $G$ is: $\min_G \mathcal{L}(G)$ s.t. $\forall j, g_j \geq 0$, which can be solved using projected gradient descent. Fixing $G$, the sub-problem defined over $\tilde{W}$ is

$$\min_{\tilde{W}} \mathcal{L}({\tilde{W}}) \quad \text{s.t.} \quad \forall j, \|\tilde{w}_j\|_2 = 1, \forall i \neq j, |\tilde{w}_i \cdot \tilde{w}_j| \leq \tau \tag{4}$$

which we solve using an ADMM algorithm. There are $R = m(m - 1)$ pairwise constraints $|\tilde{w}_i \cdot \tilde{w}_j| \leq \tau$. For the $r$-th constraint, let $p(r)$ and $q(r)$ be the index of the first and second vector respectively, i.e., the $r$-th constraint is $|\tilde{w}_{p(r)} \cdot \tilde{w}_{q(r)}| \leq \tau$. First, we introduce auxiliary variables $\{v_1^{(r)}\}_{r=1}^R$ and $\{v_2^{(r)}\}_{r=1}^R$, to rewrite the problem in
Learning Latent Space Models with Angular Constraints

Eq.(4) into an equivalent form. For each pairwise constraint: $|\tilde{w}_{p(r)} \cdot \tilde{w}_{q(r)}| \leq \tau$, we introduce two auxiliary vectors $\tilde{v}_{1(r)}$ and $\tilde{v}_{2(r)}$, and let $\tilde{w}_{p(r)} = \tilde{v}_{1(r)}$, $\tilde{w}_{q(r)} = \tilde{v}_{2(r)}$, $\|\tilde{v}_{1(r)}\|_2 = 1$, $\|\tilde{v}_{2(r)}\|_2 = 1$, $|\tilde{v}_{1(r)} \cdot \tilde{v}_{2(r)}| \leq \tau$. To this end, we obtain the following problem

$$\min_{\tilde{v}, \gamma} \mathcal{L}(\tilde{V})$$

subject to

$$\forall j, \|\tilde{v}_j\|_2 = 1$$

$$\forall r, \|\tilde{w}_{p(r)}\|_2 = 1, \|\tilde{w}_{q(r)}\|_2 = 1, |\tilde{v}_{1(r)} \cdot \tilde{v}_{2(r)}| \leq \tau$$

where $V = \{(\tilde{v}_{1(r)}, \tilde{v}_{2(r)})\}_{r=1}^R$. Then we define the augmented Lagrangian, with Lagrange multipliers $\mathcal{V} = \{(\tilde{v}_{1(r)}, \tilde{v}_{2(r)})\}_{r=1}^R$ and parameter $\rho$

$$\min_{\tilde{v}, \rho} \mathcal{L}(\tilde{V}) + \sum_{r=1}^R (\tilde{v}_{1(r)} \cdot (\tilde{w}_{p(r)} - \tilde{v}_{1(r)}) + \rho \|\tilde{w}_{p(r)} - \tilde{v}_{1(r)}\|_2^2)$$

subject to

$$\forall j, \|\tilde{v}_j\|_2 = 1$$

$$\forall r, \|\tilde{v}_{1(r)}\|_2 = 1, \|\tilde{v}_{2(r)}\|_2 = 1, |\tilde{v}_{1(r)} \cdot \tilde{v}_{2(r)}| \leq \tau$$

which can be solved by alternating between $\tilde{V}, \mathcal{V}, \mathcal{V}$.

Solve $\tilde{V}$ The sub-problem defined over $\tilde{V}$ is

$$\min_{\tilde{v}} \mathcal{L}(\tilde{V}) + \sum_{r=1}^R (\tilde{v}_{1(r)} \cdot (\tilde{w}_{p(r)} + \tilde{y}_{2(r)} \cdot \tilde{w}_{q(r)}) + \rho \|\tilde{w}_{p(r)} - \tilde{v}_{1(r)}\|_2^2)$$

subject to

$$\forall j, \|\tilde{v}_j\|_2 = 1$$

For sparse coding, we solve this sub-problem using coordinate descent. At each iteration, we update $\tilde{w}_j$ by fixing the other variables. Please refer to the supplements for details. For neural network, this sub-problem can be solved using projected gradient descent which iteratively performs the following three steps: (1) compute the gradient of $\tilde{w}_j$ using backpropagation; (2) perform a gradient descent update of $\tilde{w}_j$; (3) project each vector onto the unit sphere: $\tilde{w}_j \leftarrow \frac{\tilde{w}_j}{\|\tilde{w}_j\|_2}$.

Solve $v_{1(r)}, v_{2(r)}$ The corresponding sub-problem is

$$\min_{v_{1(r)}, v_{2(r)}} -y_{1(r)} \cdot v_{1(r)} - y_{2(r)} \cdot v_{2(r)} + \frac{\rho}{2} \|\tilde{w}_{p(r)} - v_{1(r)}\|_2^2 + \frac{\rho}{2} \|\tilde{w}_{q(r)} - v_{2(r)}\|_2^2$$

subject to

$$\|v_{1(r)}\|_2 = 1, \|v_{2(r)}\|_2 = 1$$

$$v_{1(r)} \cdot v_{2(r)} \leq \tau, -v_{1(r)} \cdot v_{2(r)} \leq \tau$$

Let $\gamma_1, \gamma_2, \lambda_1 \geq 0$, $\lambda_2 \geq 0$ be the KKT multipliers associated with the four constraints in this sub-problem. According to the KKT conditions, we have

$$-y_{1(r)} + \lambda_1 \tilde{v}_{1(r)} + \lambda_2 \tilde{v}_{2(r)} = 0$$

$$-y_{2(r)} + \gamma_1 \tilde{v}_{1(r)} + \gamma_2 \tilde{v}_{2(r)} = 0$$

We solve these two equations by examining four cases.

Case 1 First, we assume $\gamma_1 = 0, \gamma_2 = 0$, then $(\rho + 2\gamma_1)\tilde{v}_{1(r)} + \gamma_2 = \tilde{v}_{1(r)} + \gamma_2$, and $(\rho + 2\gamma_2)\tilde{v}_{2(r)} = \tilde{v}_{2(r)} + \gamma_2$. According to the primal feasibility $\|y_{1(r)}\|_2 = 1$ and $\|y_{2(r)}\|_2 = 1$, we know

$$\tilde{v}_{1(r)} = \frac{y_{1(r)}}{\|y_{1(r)}\|_2}, \tilde{v}_{2(r)} = \frac{y_{2(r)}}{\|y_{2(r)}\|_2}$$

Then we check whether the constraint $\|v_{1(r)} \cdot v_{2(r)}\|_2 \leq \tau$ is satisfied. If so, then $v_{1(r)}$ and $v_{2(r)}$ are the optimal solution.

Case 2 We assume $\gamma_1 > 0$ and $\gamma_2 = 0$, then

$$(\rho + 2\gamma_1)\tilde{v}_{1(r)} + \lambda_2 = \tilde{v}_{1(r)} + \lambda_2$$

$$\gamma_1 \tilde{v}_{1(r)} = \tilde{v}_{1(r)}$$

According to the complementary slackness condition, we know $\|\tilde{v}_{1(r)} \cdot \tilde{v}_{2(r)}\|_2 = \tau$. For the vectors on both sides of Eq.(8), taking the square of their $\ell_2$ norm, we get

$$(\rho + 2\gamma_1)^2 + \lambda_2^2 + 2(\rho + 2\gamma_1)\lambda_2 \tau = \|y_{1(r)}\|_2^2 + \gamma_2 \|y_{2(r)}\|_2^2$$

Similarly, from Eq.(9), we get

$$(\rho + 2\gamma_2)^2 + \lambda_2^2 + 2(\rho + 2\gamma_2)\lambda_2 \tau = \|y_{2(r)}\|_2^2 + \gamma_2 \|y_{1(r)}\|_2^2$$

Taking the inner product of the two vectors on the left hand sides of Eq.(8,9), and that on the right hand sides, we get

$$2\gamma_1 \gamma_2 + 2(\rho + 2\gamma_1)\lambda_2 = (\gamma_1 - \gamma_2)\lambda_2$$

Solving the system of equations consisting of Eq.(10-12), we obtain the optimal values of $\gamma_1, \gamma_2$ and $\lambda_1$. Plugging them into Eq.(8) and Eq.(9), we obtain a solution of $v_{1(r)}$ and $v_{2(r)}$. Then we check whether this solution satisfies $v_{1(r)} \cdot v_{2(r)} \leq \tau$. If so, this is an optimal solution.

In Case 3, we discuss $\lambda_1 = 0, \lambda_2 > 0$. In Case 4, we discuss $\lambda_1 > 0, \lambda_2 > 0$. The corresponding problems can be solved in a similar way as Case 2. Please refer to the supplements for details.
Solve $y_1^{(r)}, y_2^{(r)}$ We simply perform the following:

\[
y_1^{(r)} = y_1^{(r)} + \rho(\tilde{w}_p^{(r)} - v_1^{(r)})
\]

\[
y_2^{(r)} = y_2^{(r)} + \rho(\tilde{w}_q^{(r)} - v_2^{(r)})
\]

Compared with a vanilla backpropagation algorithm, the major extra cost in this ADMM algorithm comes from solving the $R = m(m-1)$ pairs of vectors $\{y_1^{(r)}, y_2^{(r)}\}_{r=1}^R$. Solving each pair incurs $O(m)$ cost. The $R$ pairs brings in a total cost of $O(m^3)$. Such a cubic cost is also incurred in other decoupling methods such as (Le et al., 2010; Bao et al., 2013). In practice, $m$ is typically less than 1000. This $O(m^3)$ cost does not substantially bottleneck computation, as we will validate in experiments.

4. Analysis

In this section, we discuss how the parameter $\tau$ which controls the level of diversity affects the generalization performance of sparse coding and neural network.

4.1. Sparse Coding

Following (Vainsencher et al., 2011), we assume the data example $x \in \mathbb{R}^d$ and basis vector $w \in \mathbb{R}^d$ are both of unit length, and the linear coefficient vector $a \in \mathbb{R}^m$ is at most $k$ sparse, i.e., $\|a\|_0 \leq k$. The estimation error of dictionary $\mathcal{W}$ is defined as

\[
L(\mathcal{W}) = \mathbb{E}_{x \sim p} [\min_{\|a\|_0 \leq k} \|x - \sum_{j=1}^m a_j w_j\|_2].
\]

Let $\tilde{L}(\mathcal{W}) = \frac{1}{n} \sum_{i=1}^n \min_{\|a\|_0 \leq k} \|x_i - \sum_{j=1}^m a_j w_j\|_2$ be the empirical reconstruction error on $n$ samples. We have the following theorem.

**Theorem 1** Assume $\tau < \frac{1}{k}$, with probability at least $1 - \delta$:

\[
L(\mathcal{W}) \leq \tilde{L}(\mathcal{W}) + \sqrt{\frac{d\ln(1/\delta)}{2n}} + \sqrt{\frac{\ln 1/\delta}{2n}} + \frac{\sqrt{4}}{n}.
\]

Note that the right hand side is an increasing function w.r.t $\tau$. As expected, a smaller $\tau$ (implying more diversity) would induce a lower estimation error bound.

4.2. Neural Network

The generalization error of a hypothesis $f$ represented with a neural network is defined as $L(f) = \mathbb{E}_{(x,y) \sim p}(\ell(f(x), y))$, where $p$ is the distribution of input-output pairs $(x, y)$ and $\ell(\cdot, \cdot)$ is the loss function. The training error is $L(f) = \frac{1}{n} \sum_{i=1}^n \ell(f(x_i^{(t)}, y_i^{(t)})$, where $n$ is the number of training samples. Let $f^* \in \text{argmin}_{f \in \mathcal{F}} L(f)$ be the true risk minimizer and $\hat{f} \in \text{argmin}_{f \in \mathcal{F}} \hat{L}(f)$ be the empirical risk minimizer. We aim to analyze the generalization error $L(f)$ of the empirical risk minimizer $f$. $L(f)$ can be decomposed into $L(f) = L(f) - L(f^*) + L(f^*)$, where $L(f) - L(f^*)$ is the estimation error and $L(f^*)$ is the approximation error.

For simplicity, we start with a “simple” fully connected network with one hidden layer of $m$ units, used for univariate regression (one output unit) with squared loss. Analysis for more complicated fully connected NNs with multiple hidden layers can be achieved in a straightforward way by cascading our analysis for this “simple” fully connected NN. Let $x \in \mathbb{R}^d$ be the input vector and $y$ be the response value. For simplicity, we assume $\text{max}\{\|x\|_2, |y|\} \leq 1$. Let $w_j \in \mathbb{R}^d$ be the weights connecting the $j$-th hidden unit with input units, with $\|w_j\|_2 \leq C$.

Let $\alpha$ be a vector where $\alpha_j$ is the weight connecting hidden unit $j$ to the output unit, with $\|\alpha\|_2 \leq B$. We assume the activation function $h(t)$ applied on the hidden units is Lipschitz continuous with constant $L$. Commonly used activation functions such as rectified linear $h(t) = \max(0, t)$, tanh $h(t) = (e^t - e^{-t})/(e^t + e^{-t})$, and sigmoid $h(t) = 1/(1 + e^{-t})$ are all Lipschitz continuous with $L = 1, 1, 0.25$, respectively. Consider the hypothesis set

\[
\mathcal{F} = \{x \mapsto \sum_{j=1}^m \alpha_j h(w_j^\top x) \mid \|\alpha\|_2 \leq B, \|w_j\|_2 \leq C, \forall i \neq j, \|w_i \cdot w_j\| \leq \tau \|w_i\|_2 \|w_j\|_2\}.
\]

The estimation error given in Theorem 2 below indicates how well the algorithm is able to learn from the samples.

**Theorem 2** Let the activation function $h$ be $L$-Lipschitz continuous and the loss $\ell(y, y) = \frac{1}{2}(y - y)^2$. Then, with probability at least $1 - \delta$:

\[
L(f^*) - L(f^*) \leq \frac{25}{\tau} + \frac{4\gamma B(2CL + |h(0)|)\sqrt{m}}{\sqrt{n}}
\]

where $\gamma = 1 + 2CL\sqrt{(m-1)\tau} + 1 + \sqrt{mB|h(0)|}$.

Note that $\gamma$, hence the above bound on estimation error, decreases as $\tau$ becomes smaller. The bound goes to zero as $n$ (sample size) goes to infinite. The inverse square root dependence on $n$ matches existing results (Bartlett & Mendelson, 2003). We note that it is straightforward to extend our bound to any bounded Lipschitz continuous loss $\ell$.

The approximation error indicates how capable the hypothesis set $\mathcal{F}$ is to approximate a target function $g = \mathbb{E}[y|x]$, where the error is measured by $\text{min}_{f \in \mathcal{F}} \|f - g\|_2$ and $\|f - g\|_2^2 = \int (f(x) - g(x))^2 p(dx)$. Following (Baron, 1993), we assume the target function $g$ satisfies certain smoothness condition that is expressed in the first moment of its Fourier representation: $\int \|\omega\|_2 \hat{g}(\omega) d\omega \leq B/2$.
Learning Latent Space Models with Angular Constraints

where \( \hat{g}(\omega) \) is the Fourier representation of \( g(x) \). For such function the following theorem states its approximation error. (In order to derive explicit constants we restrict \( h \) to be the sigmoid function, but other Lipschitz continuous activation function can be similarly handled.)

**Theorem 3** Let \( C > 1 \), \( m \leq 2(\lceil \frac{\pi}{\tau} \rceil + 1) \), where \( \theta = \arccos(\tau) \), and \( h(t) = 1/(1 + e^{-t}) \). Then, there is a function \( f \in \mathcal{F} \) such that

\[
\|f - g\|_{L^2} \leq B\left(\frac{1}{\sqrt{m}} + \frac{1 + 2\ln C}{h} + 2\sqrt{mBC} \sin\left(\frac{\min(3m\theta, \pi)}{2}\right)\right).
\]

(18)

This theorem implies that whether to use the angular constraint (AC) or not has a significant influence on the approximate error bound: without using AC (\( \tau = 1 \)), the bound is a decreasing function of \( m \) (the number of hidden units); using AC (\( \tau < 1 \)), the bound increases with \( m \). This striking phrase-change indicates the impact of AC. Given a fixed \( m \), the bound decreases with \( \tau \), implying that a stronger regularization (smaller \( \tau \)) incurs larger approximation error. When \( \tau = 1 \), the second term in the bound vanishes and the bound is reduced to the one in (Barron, 1993), which is a decreasing function of \( m \) (and \( C \), the upper bound on the weights). When \( \tau < 1 \), the second term increases with \( m \) up to the upper bound \( 2(\lceil \frac{\pi}{\tau} \rceil + 1) \). This is because a larger number of hidden units bear a larger difficulty in satisfying the pairwise ACs, which causes the function space \( \mathcal{F} \) to shrink rapidly; accordingly, the approximation power of \( \mathcal{F} \) decreases quickly.

The analysis in the two theorems shows that \( \tau \) incurs a tradeoff between the estimation error and the approximation error: decreasing \( \tau \) reduces the estimation error and enlarges the approximation error. Since the generalization error is the sum of the estimation error and the approximation error, \( \tau \) has an optimal value to yield the minimal generalization error.

**5. Experiments**

In this section, we present experimental results. Due to space limit, we put some results into supplements.

### 5.1. Sparse Coding

Following (Yang et al., 2009), we applied sparse coding for image feature learning. We used three datasets in the experiments: Scenes-15 (Lazebnik et al., 2006), Caltech-256 (Griffin et al., 2007) and UIUC-Sport (Li & Fei-Fei, 2007). For each dataset, five random train/test splits are performed and the results are averaged over the five runs. We extract pixel-level dense SIFT (Lowe, 2004) features where the step size and patch size are 8 and 16 respectively. On top of the SIFT features, we use sparse coding methods to learn a dictionary and represent each SIFT feature into a sparse code. To obtain image-level features, we apply max-pooling (Yang et al., 2009) and spatial pyramid matching (Lazebnik et al., 2006; Yang et al., 2009) over the pixel-level sparse codes. Then a linear SVM is applied to classify the images. We compare with other diversity-promoting regularizers including determinant of covariance matrix (DCM) (Malkin & Bilmes, 2008), cosine similarity (CS) (Yu et al., 2011), determinantal point process (DPP) (Kulesza & Taskar, 2012; Zou & Adams, 2012b), InCoherence (IC) (Bao et al., 2013) and mutual angles (MA) (Xie et al., 2015). We use 5-fold cross validation to tune \( \tau \) in \( \{0.3, 0.4, \cdots, 1\} \) and the number of basis vectors in \( \{50, 100, 200, \cdots, 500\} \). The parameter \( \rho \) in ADMM is set to 1.

Table 1 shows the classification accuracy on three datasets, from which we can see that compared with unregularized SC, AC-SC greatly improves performance. For example, on the Sports dataset, AC improves the accuracy from 87.4% to 90.9%. This suggests that AC is effective in reducing overfitting and improving generalization performance. Compared with other diversity-promoting regularizers, AC achieves better performance, demonstrating its better efficacy in promoting diversity.

### 5.2. Neural Networks

We evaluate AC on three types of neural networks: fully-connected NN (FNN) for phone recognition (Hinton et al., 2012), CNN for image classification (Krizhevsky et al., 2012), and RNN for question answering (Seo et al., 2017). In the main paper, we report results on four datasets: TIMIT\(^1\), CIFAR-10\(^2\), CNN (Hermann et al., 2015), Daily Mail (Hermann et al., 2015). Please refer to the supplements for results on other datasets.

**FNN for Phone Recognition** The TIMIT dataset contains a total of 6300 sentences (5.4 hours), divided into a training set (462 speakers), a validation set (50 speakers) and a core test set (24 speakers). We used the Kaldi (Povey et al., 2011) toolkit to train the monophone system which was utilized to do forced alignment and to get labels for speech frames. The toolkit was also utilized to preprocess the data into log-filter banks. Among methods based on FNN, Karel’s recipe in Kaldi achieves state

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Scene</th>
<th>Caltech</th>
<th>Sports</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC</td>
<td>83.6 ± 0.2</td>
<td>42.3 ± 0.4</td>
<td>87.4 ± 0.5</td>
</tr>
<tr>
<td>DCM-SC</td>
<td>85.4 ± 0.5</td>
<td>44.7 ± 0.8</td>
<td>89.6 ± 0.1</td>
</tr>
<tr>
<td>CS-SC</td>
<td>84.8 ± 0.6</td>
<td>45.4 ± 0.5</td>
<td>88.3 ± 0.3</td>
</tr>
<tr>
<td>DPP-SC</td>
<td>84.6 ± 0.3</td>
<td>43.5 ± 0.3</td>
<td>88.1 ± 0.2</td>
</tr>
<tr>
<td>IC-SC</td>
<td>85.5 ± 0.1</td>
<td>43.9 ± 0.7</td>
<td>90.2 ± 0.7</td>
</tr>
<tr>
<td>MA-SC</td>
<td>86.1 ± 0.5</td>
<td>45.6 ± 0.4</td>
<td>89.7 ± 0.4</td>
</tr>
<tr>
<td>AC-SC</td>
<td>86.5 ± 0.7</td>
<td>46.1 ± 0.3</td>
<td>90.9 ± 0.3</td>
</tr>
</tbody>
</table>

Table 1. Classification accuracy (%) on three datasets.  

---

\(^1\)https://catalog.ldc.upenn.edu/LDC93S1  
\(^2\)https://www.cs.toronto.edu/ kriz/cifar.html
of the art performance. We apply AC to the FNN in this recipe. The inputs of the FNN are the FMLLR (Gales, 1998) features of the neighboring 21 frames, which are mean centered and normalized to have unit variance. The number of hidden layers is 4. Each layer has 1024 hidden units. Stochastic gradient descent (SGD) is used to train the network. The learning rate is set to 0.008. We compare with four diversity-promoting regularizers: CS, IC, MA and DeCorrelation (DC) (Cogswell et al., 2015). The regularization parameter in these methods are tuned in $\{10^{-6}, 10^{-5}, \ldots, 10^{5}\}$. The $\beta$ parameter in IC is set to 1.

Table 2 shows state of the art phone error rate (PER) on the TIMIT core test set. Methods in the first panel are mostly based on FNN, which perform less well than Kaldi. Methods in the second panel are all based on RNNs which in general perform better than FNN since they are able to capture the temporal structure in speech data. In the second panel, we compare AC with other diversity-promoting regularizers. Without regularization, the error is 18.53%. With AC, the error is reduced to 18.41%, which is very close to a strong RNN-based baseline Connectionist Temporal Classification (CTC) (Graves et al., 2013). Besides, AC outperforms other regularizers.

**CNN for Image Classification** The CIFAR-10 dataset contains 32x32 color images from 10 categories, with 50,000 images for training and 10,000 for testing. We used 5000 training images as the validation set to tune hyperparameters. The data is augmented by first zero-padding the images with 4 pixels on each side, then randomly cropping the padded images to reproduce 32x32 images. We apply AC to wide residual network (WideResNet) (Zagoruyko & Komodakis, 2016) where the depth is set to 28 and the width is set to 10. SGD is used for training, with epoch number 200, initial learning rate 0.1, minibatch size 128, Nesterov momentum 0.9, dropout probability 0.3 and weight decay 0.0005. The learning rate is dropped by 0.2 at 60, 120 and 160 epochs. The performance is the median of 5 runs. We compare with CS, IC, MA, DC and an Orthogonality-Promoting (OP) regularizer (Rodríguez et al., 2016).

Table 3 shows state of the art classification error on the test set. Compared with the unregularized WideResNet which achieves an error of 3.89%, applying AC reduces the error to 3.63%. AC achieves lower error than other regularizers.

**LSTM for Question Answering** We apply AC to long short-term memory (LSTM) (Hochreiter & Schmidhuber, 1997) network, which is a type of RNN. Given the input $x_t$ at timestamp $t$, LSTM produces a hidden state $h_t$ based on the following transition equations:

$$ i_t = \sigma(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}) $$
$$ f_t = \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}) $$
$$ o_t = \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}) $$
$$ c_t = i_t \odot \tanh(W^{(c)}x_t + U^{(c)}h_{t-1} + b^{(c)}) + f_t \odot c_{t-1} $$
$$ h_t = o_t \odot \tanh(c_t) $$

where $W$s are U$s are gate-specific weight matrices. On the row vectors of each weight matrix, the AC is applied. The LSTM is used for a question answering (QA) task on two datasets: CNN and DailyMail (Hermann et al., 2015).

<table>
<thead>
<tr>
<th>Network</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kaldi</td>
<td>18.53</td>
</tr>
<tr>
<td>CS-Kaldi</td>
<td>18.48</td>
</tr>
<tr>
<td>IC-Kaldi</td>
<td>18.46</td>
</tr>
<tr>
<td>MA-Kaldi</td>
<td>18.51</td>
</tr>
<tr>
<td>DC-Kaldi</td>
<td>18.50</td>
</tr>
<tr>
<td>AC-Kaldi</td>
<td>18.41</td>
</tr>
<tr>
<td>CTC</td>
<td>18.4</td>
</tr>
<tr>
<td>RNN Transducer</td>
<td>17.7</td>
</tr>
<tr>
<td>Attention RNN</td>
<td>17.6</td>
</tr>
<tr>
<td>CTC+SCRF</td>
<td>17.4</td>
</tr>
<tr>
<td>Segmental RNN</td>
<td>17.3</td>
</tr>
<tr>
<td>RNNDrop</td>
<td>16.9</td>
</tr>
<tr>
<td>CNN</td>
<td>16.7</td>
</tr>
</tbody>
</table>

Table 2. Phone error rate (%) on the TIMIT test set.

<table>
<thead>
<tr>
<th>Network</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maxout</td>
<td>9.38</td>
</tr>
<tr>
<td>NiN</td>
<td>8.81</td>
</tr>
<tr>
<td>DSN</td>
<td>7.97</td>
</tr>
<tr>
<td>Highway Network</td>
<td>7.60</td>
</tr>
<tr>
<td>All-CNN</td>
<td>7.25</td>
</tr>
<tr>
<td>ResNet</td>
<td>6.61</td>
</tr>
<tr>
<td>ELU-Network</td>
<td>6.55</td>
</tr>
<tr>
<td>LSVU</td>
<td>5.84</td>
</tr>
<tr>
<td>Fract. Max-Pooling</td>
<td>4.50</td>
</tr>
<tr>
<td>WideResNet</td>
<td>3.89</td>
</tr>
<tr>
<td>CS-WideResNet</td>
<td>3.81</td>
</tr>
<tr>
<td>IC-WideResNet</td>
<td>3.85</td>
</tr>
<tr>
<td>MA-WideResNet</td>
<td>3.68</td>
</tr>
<tr>
<td>DC-WideResNet</td>
<td>3.77</td>
</tr>
<tr>
<td>OP-WideResNet</td>
<td>3.69</td>
</tr>
<tr>
<td>AC-WideResNet</td>
<td>3.63</td>
</tr>
<tr>
<td>ResNeXt</td>
<td>3.58</td>
</tr>
<tr>
<td>PyramidNet</td>
<td>3.48</td>
</tr>
<tr>
<td>DenseNet</td>
<td>3.46</td>
</tr>
<tr>
<td>PyramidSepDrop</td>
<td>3.31</td>
</tr>
</tbody>
</table>

Table 3. Classification error (%) on CIFAR-10 test set.
each containing a training, development and test set with 300k/4k/3k and 879k/65k/53k examples respectively. Each example consists of a passage, a question and an answer. The question is a cloze-style task where an entity is replaced by a placeholder and the goal is to infer this missing entity (answer) from all the possible entities appearing in the passage. The LSTM architecture and experimental settings follow the Bidirectional Attention Flow (BIDAF) (Seo et al., 2017) model, which consists of the following layers: character embedding, word embedding, contextual embedding, attention flow, modeling and output. LSTM is applied in the contextual embedding and modeling layer. Character embedding is based on one-dimensional convolutional neural network, where the number of filters is set to 100 and the width of receptive field is set to 5. In LSTM, the size of hidden state is set to 100. Optimization is based on AdaDelta (Zeiler, 2012), where the minibatch size and initial learning rate are set to 48 and 0.5. The model is trained for 8 epochs. Dropout (Srivastava et al., 2014) with probability 0.2 is applied. We compare the computational time of neural networks under different regularizers. Table 5 shows the total runtime of FNNs on TIMIT and CNNs on CIFAR-10 with a single GTX TITAN X GPU, and the runtime of LSTM networks on the CNN dataset with 2 TITAN X GPUs. Compared with no regularization, AC incurs a 18.2% extra time on TIMIT, 12.7% on CIFAR-10 and 14.8% on CNN. The runtime of AC is comparable to that under other diversity-promoting regularizers.

5.4. Computational Time

We compare the computational time of neural networks under different regularizers. Table 5 shows the total runtime of FNNs on TIMIT and CNNs on CIFAR-10 with a single GTX TITAN X GPU, and the runtime of LSTM networks on the CNN dataset with 2 TITAN X GPUs. Compared with no regularization, AC incurs a 18.2% extra time on TIMIT, 12.7% on CIFAR-10 and 14.8% on CNN. The runtime of AC is comparable to that under other diversity-promoting regularizers.

6. Conclusions

In this paper, we propose Angled-Constrained Latent Space Models (AC-LSMs) that aim at promoting diversity among components in LSMs for the sake of alleviating overfitting. Compared with previous diversity-promoting methods, AC has two benefits. First, it is theoretically analyzable: the generalization error analysis shows that larger diversity leads to smaller estimation error and larger approximation error. Second, it is empirically effective, as validated in various experiments.
Acknowledgements

We would like to thank the anonymous reviewers for the suggestions and comments that help to improve this work a lot, and thank Yajie Miao for helping with some of the experiments. P.X and E.X are supported by National Institutes of Health P30DA035778, R01GM114311, National Science Foundation IIS1617583, DARPA FA872105C0003 and Pennsylvania Department of Health BD4BH4100070287.

References


Learning Latent Space Models with Angular Constraints


Prabhavalkar, Rohit, Sainath, Tara N, Nahamoo, David, Ramabhadran, Bhuvana, and Kanevsky, Dimitri. An evaluation of posterior modeling techniques for phonetic


