Neural Collapse with Normalized Features: A Geometric Analysis over the Riemannian Manifold

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recently, it has been empirically observed that optimization of overparameterized deep networks for classification tasks often leads to a so-called "neural collapse" phenomenon. Specifically, it has been observed that the learned features (the output of the penultimate layer) of within-class samples converge to their means, and the means of different classes exhibit a certain tight frame structure, which is also aligned with the last layer's classifier. As feature normalization in the last layer becomes a common practice in modern representation learning, we theoretically justify the neural collapse phenomenon for normalized features in this work. Based on an unconstrained feature model, we simplify the empirical loss function in a multi-class classification task and obtain a nonconvex optimization problem over the Riemannian manifold by constraining all features and classifiers over the sphere. In this context, we analyze the nonconvex landscape of the Riemannian optimization problem over the product of spheres, showing a benign global landscape in the sense that the only global minimizers are the neural collapse solutions while all other critical points are strict saddles with negative curvature. Experimental results on practical deep networks are reported to corroborate our theory and demonstrate that better representations can be learned faster via feature normalization.

1 Introduction

Depsite the tremendous success of deep learning in engineering and scientific applications over the past decades, the underlying mechanism of deep neural networks (DNNs) still largely remains mysterious. Towards the goal of understanding the learned deep representations, a recent seminal line of works [1, 2, 3] presents an intriguing phenomenon that persists across a range of canonical classification problems during the terminal phase of training. Specifically, last-layer features and last-layer linear classifiers of a trained DNN exhibit simple but elegant mathematical structures, in the sense that

- **(NC1) Variability Collapse:** the individual features of each class concentrate to their class-means.
- **(NC2) Convergence to Simplex ETF:** the class-means have the same length and are maximally distant. In other words, they form a Simplex Equiangular Tight Frame (ETF).
- **(NC3) Convergence to Self-duality:** the last-layer linear classifiers perfectly match their class-means.

Such a phenomenon is referred to as *Neural Collapse* ($\mathcal{NC}$) [1], which has been shown empirically to persist across a broad range of canonical classification problems, on different loss functions (e.g., cross-entropy (CE) [1, 4, 3], mean-squared error (MSE) [5, 6], and supervised contrasive (SC) losses

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

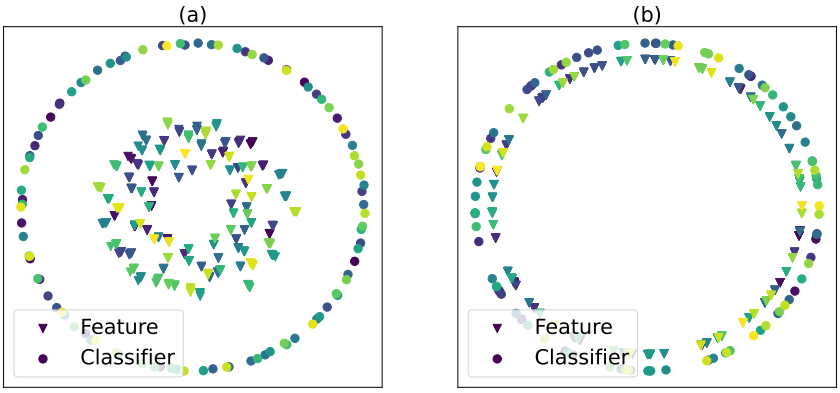

Figure 1: **Comparison of features learned with and without normalization.** $K = 100$ classes with $n = 5$ samples per class. Features and classifiers are found through optimizing the cross-entropy loss with a UFM, where features are embedded in 2-dimensional space, i.e., $d = 2$. (a) No normalization of features or classifiers. (b) Features and classifiers are constrained to the unit sphere (features are scaled down for visualization purposes).

Table 1: Average UFM feature loss and accuracy over 10 trials with and without normalization to sphere, with the same set-up as in Figure 1.

	Average CE Loss	**Average Accuracy**
No Normalization	813.4 ± 4.53	$49.9\% \pm 2.39\%$
Normalization	1920.5 ± 0.00	$100.0\% \pm 0.00\%$

[7]), different neural network architectures (e.g., VGG [8], ResNet [9], and DenseNet [10]), and on a variety of standard datasets (such as MNIST [11], CIFAR [12], and ImageNet [13], etc). Related phenomena have further been observed in a number of recent works [14, 15, 16, 17, 18, 19, 20, 21], including state-of-the-art natural language models (such as BERT, RoBERTa, and GPT) [15].

Motivations & contributions. In this work, we try to understand why $\mathcal{NC}$ happens with a common practice that the last-layer features are normalized. Our work has the following motivations.

- *Feature normalization is a common practice in training model deep networks.* Recently, many existing results demonstrated that training with feature normalization often improves the quality of learned representation with better class separation [22, 23, 24, 7]. As illustrated in Fig. 1 and Table 1, experimental results visualized in low-dimensional space show that features learned with normalization are more uniformly distributed over the sphere and hence are more linearly separable than those learned without normalization. As a result, the better learned representations lead to improved test performances as shown in [25, 7]. Additionally, recent works [22, 23] also demonstrate improved robustness when learning with feature normalization.
- *Many existing works study $\mathcal{NC}$ with norm constraints.* As shown in [26, 27, 3, 7, 28], many existing theoretical studies for $\mathcal{NC}$ consider the formulations with the norms of both features and classifiers constrained. Based upon unconstrained feature models [29], it has been shown that the only global solutions are $\mathcal{NC}$ for a variety of loss functions (e.g., MSE, CE, SC losses). Nonetheless, these works only focused on the global optimality conditions without looking into their nonconvex landscapes, and therefore failed to explain why global $\mathcal{NC}$ solutions can be efficiently reached by a training algorithm such as stochastic gradient descent (SGD).

Thus, in this work we study the global nonconvex landscape of the training deep networks with norm constraints on the features and classifiers. We consider the commonly used cross-entropy (CE) loss and formulate the problem as a Riemannian optimization problem over the product of unit spheres (i.e., oblique manifold). Our study is also based upon the assumption of the so-called *unconstrained feature model* (UFM) [29] or *layer-peeled model* [30], where the last-layer features of the deep

network are treated as free optimization variables to simplify the nonlinear interactions across layers. The underlying reasoning is that modern deep networks are often highly overparameterized with the capacity of learning any representations [31, 32, 33], so that the last-layer features can approximate, or interpolate, any point in the feature space.

Assuming the UFM, we show that the Riemannian optimization problem has a benign global landscape, in the sense that the loss with respect to (w.r.t.) the features and classifers is a strict saddle function [34, 35] over the manifold. We show that every local minimizer is a $\mathcal{NC}$ solution, and all the other critical points exhibit directions with negative curvature. Our analysis for the manifold setting is based upon a nontrivial extension of recent studies for the unconstrained formulation, which could be of independent interest. Our work brings new tools from Riemannian optimization for analyzing deep learning. At the same time, we empirically show the advantages of our constrained formulation over its penalized counterpart for solving the training problems, and also demonstrate its practical benefits for training deep networks. Lastly, we believe the benign landscape over the manifold is not limited only to CE loss. It could potentially hold for other popular training losses such as the contrastive loss [7], which is an exciting direction for future work.

Prior art and related works on $\mathcal{NC}$. The empirical $\mathcal{NC}$ phenomenon has inspired a recent line of theoretical studies on understanding why it happens [7, 30, 26, 29, 5, 6, 4]. Like ours, most of these works studied the problem under the UFM. In particular, recent works showed that the *only* global solutions are $\mathcal{NC}$ solutions for a variety of nonconvex training losses (e.g., crosss-entropy (CE) [4, 30, 26], mean squared error (MSE) [5, 6], supervised contrastive (SC) [7]) and different problem formulations (e.g., penalized unconstrained, constrained). Recently, this study has been extended to deeper models under the MSE training loss [5]. More surprisingly, it has been further shown that the nonconvex losses under the UFM have benign global optimization landscapes, in the sense that every local minimizer satisfies $\mathcal{NC}$ properties and the remaining critical points are strict saddles with negative curvature. Such results have been established for both CE and MSE losses [4, 6], where the studies in [4, 6] are limited to unconstrained versions of the problem with regularization on both features and classifiers.

Additionally, there is a line of recent works deciphering generalization of deep networks in terms of $\mathcal{NC}$. The work [36] shows that $\mathcal{NC}$ also happens on test data drawn from the same distribution asymptotically, but not for finite samples [37]. Other works [37, 38] demonstrated that the variability collapse of features is actually happening progressively from shallow to deep layers, and [39] showed that test performance can be improved when enforcing variability collapse on features of intermediate layers. The works [40, 41] showed that fixing the classifier as a simplex ETF improves test performance on imbalanced training data and long-tailed classification problems.

Notation. Let $\mathbb{R}^n$ be the n-dimensional Euclidean space and $\|\cdot\|_2$ be the Euclidean norm. We write matrices in bold capital letters such as $\boldsymbol{A}$, vectors in bold lower-case such as $\boldsymbol{a}$, and scalars in plain letters such as a. For a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times K}$, we denote by $\boldsymbol{a}_k$ its k-th column and by $\boldsymbol{a}^i$ its i-th row. We use $\mathrm{diag}(\boldsymbol{A})$ to denote a vector that consists of diagonal elements of $\boldsymbol{A}$ and $\mathrm{ddiag}(\boldsymbol{A})$ to denote a diagonal matrix whose diagonal elements are the diagonal ones of $\boldsymbol{A}$. Given a positive integer n, we denote by $[n]$ the set $\{1, \ldots, n\}$.

2 Nonconvex Formulation with Spherical Constraints

In this section, we review the basic concepts of deep neural networks and introduce notation that will be used throughout the paper. Based upon this, we formally introduce the problem formulation over the Riemannian manifold with the unconstrained feature model.

2.1 Basics of Deep Neural Networks

In this work, we focus on the multi-class (K class) classification problem. Given input data $\boldsymbol{x} \in \mathbb{R}^D$, the goal of deep learning is to learn a deep hierarchical representation (or feature) $\boldsymbol{h}(\boldsymbol{x}) = \phi_{\boldsymbol{\theta}}(\boldsymbol{x}) \in \mathbb{R}^d$ of the input along with a linear classifier[1] $\boldsymbol{W} \in \mathbb{R}^{d\times K}$ such that the output $\psi_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \boldsymbol{W}^\top \boldsymbol{h}(\boldsymbol{x})$ of the network fits the input $\boldsymbol{x}$ to an one-hot training label $\boldsymbol{y} \in \mathbb{R}^K$. More precisely, in the vanilla

[1] We write $\boldsymbol{W} = \boldsymbol{W}_L^\top$ in the transposed form for the simplicity of analysis.

form an L-layer deep network can be written as

$$\psi_{\boldsymbol{\Theta}}(\boldsymbol{x}) \;=\; \underbrace{\boldsymbol{W}_L}_{\textbf{linear classifier } \boldsymbol{W}=\boldsymbol{W}_L^\top} \underbrace{\sigma\left(\boldsymbol{W}_{L-1}\cdots\sigma\left(\boldsymbol{W}_1\boldsymbol{x}+\boldsymbol{b}_1\right)+\boldsymbol{b}_{L-1}\right)}_{\textbf{feature } \boldsymbol{h}=\phi_{\boldsymbol{\theta}}(\boldsymbol{x})} + \boldsymbol{b}_L, \tag{1}$$

where each layer is composed of an affine transformation, represented by some weight matrix $\boldsymbol{W}_k$, and bias $\boldsymbol{b}_k$, followed by a nonlinear activation $\sigma(\cdot)$, and $\boldsymbol{\Theta} = \{\boldsymbol{W}_k, \boldsymbol{b}_k\}_{k=1}^{L}$ and $\boldsymbol{\theta} = \{\boldsymbol{W}_k, \boldsymbol{b}_k\}_{k=1}^{L-1}$ denote the weights for *all* the network parameters and those up to the last layer, respectively. Given training samples $\{\boldsymbol{x}_{k,i}, \boldsymbol{y}_k\} \in \mathbb{R}^d \times \mathbb{R}^K$ drawn from the same data distribution, we learn the network parameters $\boldsymbol{\Theta}$ via minimizing the empirical risk over these samples,

$$\min_{\boldsymbol{\Theta}} \sum_{k=1}^{K}\sum_{i=1}^{n_k} \mathcal{L}_{\mathrm{CE}}\left(\psi_{\boldsymbol{\Theta}}(\boldsymbol{x}_{k,i}), \boldsymbol{y}_k\right), \quad \text{s.t.} \quad \boldsymbol{\Theta} \in \mathcal{C}, \tag{2}$$

where $\boldsymbol{y}_k \in \mathbb{R}^K$ is a one-hot vector with only the kth entry equal to unity and the remaining ones equal to zero for all $1 \le k \le K$, $\boldsymbol{x}_{k,i} \in \mathbb{R}^D$ is the i-th sample in the k-th class, $\{n_k\}_{k=1}^{K}$ denotes the number of training samples in each class, and the set $\mathcal{C}$ denotes the constraint set of the network parameters $\boldsymbol{\Theta}$ that we will specify later. Here, we study the cross-entropy loss of the form

$$\mathcal{L}_{\mathrm{CE}}(\boldsymbol{z}, \pi_k(\boldsymbol{z})) \;:=\; -\log\left(\frac{\exp(z_k)}{\sum_{i=1}^{K}\exp(z_i)}\right),$$

where $\pi_k(\boldsymbol{z})$ denotes that $\boldsymbol{z}$ belong to the k-the class.

2.2 Riemannian Optimization over the Product of Spheres

For the K-class classification problem, let us consider a simple case where the number of training samples in each class is balanced (i.e., $n = n_1 = n_2 = \cdots = n_K$) and let $N = nK$, and we assume that all the biases $\{\boldsymbol{b}_k\}_{k=1}^{L}$ are zero with the last activation function $\sigma(\cdot)$ before the output to be linear. Analyzing deep networks $\psi_{\boldsymbol{\Theta}}(\boldsymbol{x})$ is a tremendously difficult task mainly due to the *nonlinear interactions* across a large number of layers. Fortunately, based upon previous study on *unconstrained feature models* [28, 29, 7, 4], we can simplify the nonlinear interactions by treating the last-layer features $\boldsymbol{h}_{k,i} = \phi_{\boldsymbol{\theta}}(\boldsymbol{x}_{k,i}) \in \mathbb{R}^d$ as *free* optimization variables, where the underlying reasoning is that modern deep networks are often highly overparameterized to approximate any continuous function [31, 32, 33]. Specifically, we rewrite all the features in a matrix form

$$\boldsymbol{H} \;=\; [\boldsymbol{H}_1 \quad \boldsymbol{H}_2 \quad \cdots \quad \boldsymbol{H}_K] \in \mathbb{R}^{d\times N},\; \boldsymbol{H}_k \;=\; [\boldsymbol{h}_{k,1} \quad \boldsymbol{h}_{k,2} \quad \cdots \quad \boldsymbol{h}_{k,n}] \in \mathbb{R}^{d\times n},\; \forall\, k \in [K],$$

and correspondingly denote the classifier $\boldsymbol{W}$ by

$$\boldsymbol{W} \;=\; [\boldsymbol{w}_1 \quad \boldsymbol{w}_2 \quad \cdots \quad \boldsymbol{w}_K] \in \mathbb{R}^{d\times K}, \quad \boldsymbol{w}_k \in \mathbb{R}^d, \quad \forall\, k \in [K].$$

We assume that both the features $\boldsymbol{H}$ and the classifiers $\boldsymbol{W}$ are normalized, in the sense that $\|\boldsymbol{h}_{k,i}\|_2 = 1$, $\|\boldsymbol{w}_k\|_2 = \tau$, for all $k \in [K]$ and all $i \in [n]$, where $\tau > 0$ is the temperature parameter. As a result, we study $\mathcal{NC}$ with a *constrained* formulation over a Riemannian manifold

$$\min_{\boldsymbol{W},\boldsymbol{H}} \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{n} \mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{W}^\top\boldsymbol{h}_{k,i}, \boldsymbol{y}_k\right) \;\text{s.t.}\; \|\boldsymbol{w}_k\|_2 \;=\; \tau,\; \|\boldsymbol{h}_{k,i}\|_2 \;=\; 1,\; \forall\, i \in [n],\; \forall\, k \in [K].$$

Since the temperature parameter τ can be absorbed into the loss function, we can replace $\boldsymbol{w}_k$ by $\tau\boldsymbol{w}_k$ and change the original constraint into $\|\boldsymbol{w}_k\|_2 = 1$. Moreover, the product of spherical constraints forms an *oblique manifold* [42] embedded in Euclidean space, i.e.,

$$\mathcal{OB}(d, K) \;:=\; \left\{\boldsymbol{Z} \in \mathbb{R}^{d\times K} \mid \boldsymbol{z}_k \in \mathbb{S}^{d-1},\; \forall\, k \in [K]\right\}.$$

Consequently, we can rewrite the constrained problem as a Riemannian optimization problem over the oblique manifold w.r.t. $\boldsymbol{W}$ and $\boldsymbol{H}$ as

$$\min_{\boldsymbol{W}\in\mathbb{R}^{d\times K},\boldsymbol{H}\in\mathbb{R}^{d\times N}} f(\boldsymbol{W},\boldsymbol{H}) \;:=\; \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{n} \mathcal{L}_{\mathrm{CE}}\left(\tau\boldsymbol{W}^\top\boldsymbol{h}_{k,i}, \boldsymbol{y}_k\right), \tag{3}$$
$$\text{s.t.} \quad \boldsymbol{H} \in \mathcal{OB}(d, N),\; \boldsymbol{W} \in \mathcal{OB}(d, K).$$

We will show that all global solutions satisfy $\mathcal{NC}$ properties, and the function (3) is a strict saddle function [43, 44] of $(\boldsymbol{W}, \boldsymbol{H})$ over the oblique manifold so that the $\mathcal{NC}$ solution can be efficiently achieved.

Riemannian derivatives over the oblique manifold. In the following, we will use tools from Riemannian optimization to characterize the global optimality condition and the geometric properties of the optimization landscape of the problem (3). Thus, let us introduce some basic derivations of the Riemannian gradient and Hessian, defined on the tangent space of the oblique manifold. For more details, we refer the readers to the appendices.

According to [45, Chapter 3 & 5] and [46, 47], we can calculate the Riemannian gradients and Hessian of Problem (3) as follows. According to [45, Example 3.18], we have the tangent space to $\mathcal{OB}(d, K)$ at $\boldsymbol{W}$ as

$$\mathrm{T}_{\boldsymbol{W}}\mathcal{OB}(d,K) \;=\; \left\{\boldsymbol{Z} \in \mathbb{R}^{d\times K} \mid \operatorname{diag}\left(\boldsymbol{W}^\top \boldsymbol{Z}\right) = \boldsymbol{0}\right\}.$$

We can similarly derive the tangent space for $\boldsymbol{H}$. Let us define

$$\boldsymbol{M} := \tau \boldsymbol{W}^\top \boldsymbol{H},\; g(\boldsymbol{M}) := f(\boldsymbol{W}, \boldsymbol{H}).$$

First, we can derive the Riemannian gradient of $f(\boldsymbol{W}, \boldsymbol{H})$ of Problem (3) as

$$\operatorname{grad}_{\boldsymbol{H}} f(\boldsymbol{W},\boldsymbol{H}) \;=\; \tau \boldsymbol{W}\nabla g(\boldsymbol{M}) - \tau \boldsymbol{H}\operatorname{ddiag}\left(\boldsymbol{H}^\top \boldsymbol{W}\nabla g(\boldsymbol{M})\right), \tag{4}$$

$$\operatorname{grad}_{\boldsymbol{W}} f(\boldsymbol{W},\boldsymbol{H}) \;=\; \tau \boldsymbol{H}\nabla g(\boldsymbol{M})^\top - \tau \boldsymbol{W}\operatorname{ddiag}\left(\boldsymbol{W}^\top \boldsymbol{H}\nabla g(\boldsymbol{W})^\top\right). \tag{5}$$

Second, for any $\boldsymbol{\Delta} = (\boldsymbol{\Delta}_{\boldsymbol{W}}, \boldsymbol{\Delta}_{\boldsymbol{H}}) \in \mathrm{T}_{\boldsymbol{W}}\mathcal{OB}(d,K) \times \mathrm{T}_{\boldsymbol{H}}\mathcal{OB}(d,N)$, we compute the Hessian bilinear form of $f(\boldsymbol{W}, \boldsymbol{H})$ along the direction $\boldsymbol{\Delta}$ by

$$\begin{aligned}\nabla^2 f(\boldsymbol{W},\boldsymbol{H})[\boldsymbol{\Delta},\boldsymbol{\Delta}] \;&= \nabla^2 g(\boldsymbol{M})\left[\tau\left(\boldsymbol{W}^\top\boldsymbol{\Delta}_{\boldsymbol{H}} + \boldsymbol{\Delta}_{\boldsymbol{W}}^\top \boldsymbol{H}\right), \tau\left(\boldsymbol{W}^\top\boldsymbol{\Delta}_{\boldsymbol{H}} + \boldsymbol{\Delta}_{\boldsymbol{W}}^\top \boldsymbol{H}\right)\right] \\ &\quad + 2\tau\left\langle \nabla g(\boldsymbol{M}), \boldsymbol{\Delta}_{\boldsymbol{W}}^\top \boldsymbol{\Delta}_{\boldsymbol{H}}\right\rangle,\end{aligned} \tag{6}$$

Based on this, according to [45, Example 5.32], we can derive the Riemannian Hessian bilinear form of $f(\boldsymbol{W}, \boldsymbol{H})$ along the direction $\boldsymbol{\Delta}$ as

$$\begin{aligned}\operatorname{Hess} f(\boldsymbol{W},\boldsymbol{H})[\boldsymbol{\Delta},\boldsymbol{\Delta}] \;&= \nabla^2 f(\boldsymbol{W},\boldsymbol{H})[\boldsymbol{\Delta},\boldsymbol{\Delta}] - \left\langle \boldsymbol{\Delta}_{\boldsymbol{W}}\operatorname{ddiag}\left(\boldsymbol{M}\nabla g(\boldsymbol{M})^\top\right), \boldsymbol{\Delta}_{\boldsymbol{W}}\right\rangle \\ &\quad - \left\langle \boldsymbol{\Delta}_{\boldsymbol{H}}\operatorname{ddiag}\left(\boldsymbol{M}^\top\nabla g(\boldsymbol{M})\right), \boldsymbol{\Delta}_{\boldsymbol{H}}\right\rangle.\end{aligned}$$

In the following, we will use the Riemannian gradient and Hessian to characterize the optimization landscape of (3).

3 Main Theoretical Analysis

In this section, we first characterize the structure of the global solution set of Problem (3). Based upon this, we analyze the global landscape of Problem (3) via characterizing its Riemannian derivatives.

3.1 Global Optimality Condition

For the feature $\boldsymbol{H}$, let us denote the class mean for each class as

$$\overline{\boldsymbol{H}} \;:=\; \begin{bmatrix}\overline{\boldsymbol{h}}_1 & \cdots & \overline{\boldsymbol{h}}_K\end{bmatrix} \in \mathbb{R}^{d\times K}, \quad \text{and} \quad \overline{\boldsymbol{h}}_k \;:=\; \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{h}_{k,i}, \quad 1 \le k \le K. \tag{7}$$

Based upon this, we show any global solution of the problem (3) exhibits $\mathcal{NC}$ properties in the sense that it satisfies (**NC1**) variability collapse, (**NC2**) convergence to simplex ETF, and (**NC3**) convergence to self-duality.

Theorem 1 (Global Optimality Condition)**.** *Suppose that the feature dimension is no smaller than the number of classes (i.e., $d \ge K$) and the training labels are balanced in each class (i.e., $n = n_1 = \cdots = n_K$). Then, it holds for the CE loss $f(\boldsymbol{W}, \boldsymbol{H})$ in problem (3) that*

$$f(\boldsymbol{W},\boldsymbol{H}) \;\ge\; \log\left(1 + (K-1)\exp\left(-\frac{K\tau}{K-1}\right)\right)$$

for all $\boldsymbol{W} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K] \in \mathbb{R}^{d\times K}$ and $\boldsymbol{H} = [\boldsymbol{h}_{1,1}, \ldots, \boldsymbol{h}_{K,n}] \in \mathbb{R}^{d\times N}$. In particular, equality holds if and only if there are $\overline{\boldsymbol{h}}_1, \overline{\boldsymbol{h}}_2, \cdots, \overline{\boldsymbol{h}}_K \in \mathbb{R}^d$ such that

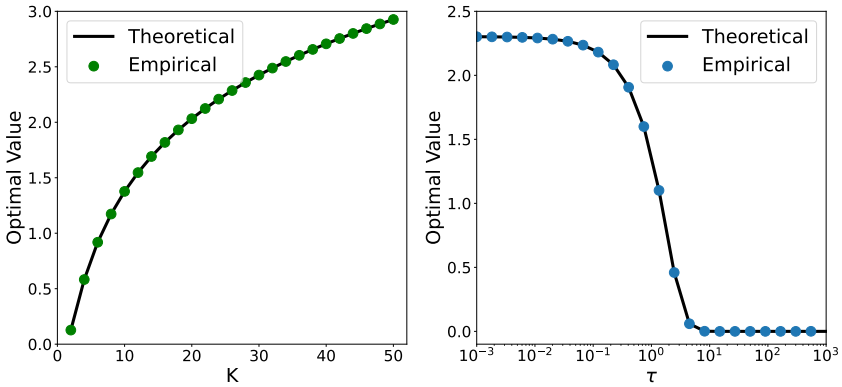

Figure 2: **Global optimization of** (3) **under UFM with** $d = 100$ **and** $n = 5$**.** Theoretical line refers to lower bound (global minimum) from Theorem 1. Left: Lower bound (black) against number of classes K while fixing $\tau = 1$, optimal value found using gradient descent with random initialization shown with green markers. Right: Lower bound (black) against temperature τ while fixing $K = 10$, optimal value found using gradient descent with random initialization shown with blue markers. The same empirical values are achieved over many trials due to the benign global landscape.

- **(NC1)** ***Variability collapse:*** $\boldsymbol{h}_{k,i} = \overline{\boldsymbol{h}}_k, \ \forall\, i \in [n]$;
- **(NC2)** ***Simplex ETF structure:*** $\{\overline{\boldsymbol{h}}_k\}_{k=1}^K$ *form a simplex ETF in the sense that*

$$\overline{\boldsymbol{H}}^\top \overline{\boldsymbol{H}} \ = \ \frac{1}{K-1}\left(K\boldsymbol{I}_K - \boldsymbol{1}_K\boldsymbol{1}_K^\top\right).$$

- **(NC3)** ***Self-duality:*** $\boldsymbol{w}_k = \overline{\boldsymbol{h}}_k, \ \forall\, k \in [K]$.

The proof follows similar ideas of a line of recent works [7, 30, 26, 5, 6, 4], and we refer the readers to the appendices for the proof. The result of our problem (3) is closely related to [7, Theorem 1] (i.e., spherical constraints vs. that of ball constraints). In fact, our problem and that in [7] share the same global solution set. Additionally, compared to the unconstrained regularized problems in [6, 4], it is worth noting that the regularization parameters there influence the structure of global solutions, while the temperature parameter τ only affects the optimization landscape but not the global solutions.

3.2 Global Landscape Analysis

Due to the nonconvex nature of Problem (3), the characterization of global optimality alone in Theorem 1 is not sufficient for guaranteeing efficient optimization to those desired global solutions. Thus, we further study the global landscape of Problem (3) by characterizing all the *Riemannian critical points* $(\boldsymbol{W}, \boldsymbol{H})$ with

$$\operatorname{grad}_{\boldsymbol{H}} f(\boldsymbol{W}, \boldsymbol{H}) \ = \ \boldsymbol{0}, \quad \operatorname{grad}_{\boldsymbol{W}} f(\boldsymbol{W}, \boldsymbol{H}) \ = \ \boldsymbol{0}.$$

As such, we state our major result below.

Theorem 2 (Global Landscape Analysis)**.** *Assume that the number of training samples in each class is balanced* $n = n_1 = \cdots = n_K$*. If the feature dimension is larger than the number of samples* $d > N$*, the function* $f(\boldsymbol{W}, \boldsymbol{H})$ *is a strict saddle function that has no spurious local minimum, in the sense that*

- *Any local minimizer of Problem* (3) *is a global minimizer of the form shown in Theorem 1.*
- *Any Riemannian critical point* $(\boldsymbol{W}, \boldsymbol{H})$ *of Problem* (3) *that is not a local minimizer is a Riemannian strict saddle point with negative curvature, in the sense that the Riemannian Hessian* $\operatorname{Hess} f(\boldsymbol{W}, \boldsymbol{H})$ *at the critical point* $(\boldsymbol{W}, \boldsymbol{H})$ *is non-degenerate, and there exists a direction* $\boldsymbol{\Delta} = (\boldsymbol{\Delta}_{\boldsymbol{W}}, \boldsymbol{\Delta}_{\boldsymbol{H}}) \in \mathrm{T}_{\boldsymbol{W}}\mathcal{OB}(d, K) \times \mathrm{T}_{\boldsymbol{H}}\mathcal{OB}(d, N)$ *such that*

$$\operatorname{Hess} f(\boldsymbol{W}, \boldsymbol{H})[\boldsymbol{\Delta}, \boldsymbol{\Delta}] \ < \ 0.$$

In other words, $\lambda_{\min}\left(\operatorname{Hess} f(\boldsymbol{W}, \boldsymbol{H})\right) < 0$ *at the corresponding Riemannian critical point.*

The major challenge of our analysis is to find the negative curvature for $\operatorname{Hess} f(\boldsymbol{W}, \boldsymbol{H})$. Our key observation is that, because $d > N$ and $\boldsymbol{H} \in \mathbb{R}^{d \times N}$, there always exists a non-zero $\boldsymbol{a} \in \mathbb{R}^d$ such that it lies the null space of $\boldsymbol{H}$, i.e., $\boldsymbol{H}^\top \boldsymbol{a} = \boldsymbol{0}$. According to this, (4), (5), and the properties of $\nabla g(\boldsymbol{M})$, we can show that the same $\boldsymbol{a} \in \mathbb{R}^d$ is also in the null space of $\boldsymbol{W}^\top$, i.e., $\boldsymbol{W}^\top \boldsymbol{a} = \boldsymbol{0}$. This allows us to construct a direction along which the leading bilinear form of $\nabla^2 g(\boldsymbol{M})$ in (6) vanishes. Based on this and the optimality condition for Problem (3), we can construct a rank-1 negative curvature direction $(\boldsymbol{\Delta}_{\boldsymbol{W}}, \boldsymbol{\Delta}_{\boldsymbol{H}})$ for $\boldsymbol{W}$ and $\boldsymbol{H}$, respectively. For more details of the proof, we refer readers to the appendices. In the following, we discuss the implications, relationship, and limitations of our results.

- *Efficient global optimization to $\mathcal{NC}$ solutions.* Our theorem implies that, for optimizing $(\boldsymbol{W}, \boldsymbol{H})$ via Problem (3), the $\mathcal{NC}$ solutions can be efficiently reached by Riemannian first-order methods (e.g., Riemannian stochastic gradient descent) with random initialization [43, 48, 49]; see Figure 2 for a demonstration. For training practical deep networks, this can be efficiently implemented by normalizing last-layer features when running SGD.
- *Relation to existing works on $\mathcal{NC}$.* Most existing results have only studied the global minimizers under UFM [7, 30, 26, 5], which has limited implication for optimization. On the other hand, our landscape analysis is based upon a nontrivial extension of that with the unconstrained problem formulation [6, 4]. Compared to those works, Problem (3) is much more challenging for analysis, due to the fact that the set of critical points of our problem is essentially much larger than that of [6, 4]. Moreover, we empirically demonstrate the advantages of the manifold formulation over its regularized counterpart, in terms of representation quality and training speed.
- *Limitations on the assumptions of the feature dimension d.* The major limitation of our current result is that we require relatively large $d > N$,[2] which could hold for modern highly overparameterized networks [50, 31, 32, 51, 33]. We believe that this bound can be improved to $d > K + 1$ by a more delicate analysis; this is also corroborated by our experimental results (see Figure 2). Furthermore, through numerical simulations we conjecture the global landscape also holds even when $d \ll K$, while the global solutions are uniform over the sphere [26] rather than being simplex ETFs (see Figure 1). The analysis on $d \ll K$ is left for future work.
- *Relation to other Riemannian nonconvex problems.* Our result joins a recent line of work on the study of global nonconvex landscapes over Riemannian manifolds, such as orthogonal tensor decomposition [34], dictionary learning [52, 53, 54, 55, 56], and sparse blind deconvolution [57, 58, 59, 60]. For all these problems constrained over a Riemannian manifold, it can be shown that they exhibit "equivalently good" global minimizers due to symmetries and intrinsic low-dimensional structures, and the loss functions are usually strict saddles [34, 35, 61]. As we can see, the global minimizers (i.e., simplex ETFs) of our problem here also exhibit a similar rotational symmetry, in the sense that $\boldsymbol{W}^\top \boldsymbol{H} = (\boldsymbol{Q}\boldsymbol{W})^\top (\boldsymbol{Q}\boldsymbol{H})$ for any orthogonal matrix $\boldsymbol{Q}$. Additionally, by adapting tools from Riemmanian optimization, our result showed how these tools can be powerful for the study of deep learning.

4 Experiments

In this section, we support our theoretical results in previous sections and provide further motivation with experimental results on practical deep network training. In the first experiment, we validate the assumption of UFM introduced in Section 2 for analyzing $\mathcal{NC}$, by demonstrating that $\mathcal{NC}$ occurs for increasingly overparameterized deep networks. In the second experiment, we further motivate feature normalization with empirical results showing that feature normalization can lead to faster training and better collapse than the unconstrained counterpart with regularization. This occurs not only with the UFM but also with practical overparameterized networks. Before that, let us introduce some basics of the experimental setup and metrics for evaluating $\mathcal{NC}$.

Network architectures, datasets, and training details. In our experiments, we use ResNet [9] architectures for the feature encoder. For the normalized network, we project the output of the encoder onto the sphere of radius τ (as done in [7]) and also project the weight classifiers to the unit sphere after each optimization step to maintain constraints. In all experiments, we set $\tau = 1$. For the non-normalized UFM and network, we use a weight and feature decay of 10^{-4} (using the loss in [4]).

[2] This is compared to the requirement of $d > K$ in [6, 4].

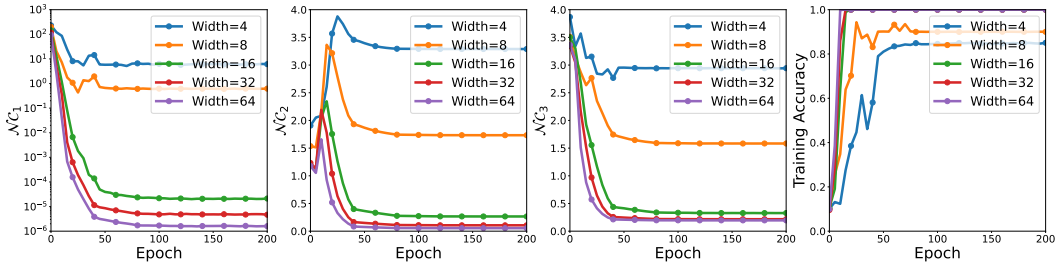

Figure 3: **Validation of UFM on ResNet with varying network width.** $\mathcal{NC}$ metrics and training accuracy of ResNet-18 networks of various widths on CIFAR10 with $n = 200$ over 200 epochs.

We do not use a bias term for the classifier for either architecture. For both experiments, we use the CIFAR dataset[3] [12], where we use CIFAR10 and CIFAR100 for the first and second experiment, respectively. In both experiments, we train the networks using SGD with a batch size of 128 and momentum 0.9 with an initial learning rate of 0.05, and we decay the learning rate by a factor of 0.1 after every 40 epochs - these hyperparameters are chosen to be the same as those in [4] for fair comparisons. All networks are trained on Nvidia Tesla V100 GPUs with 16G of memory.

Neural collapse metrics. For measuring different aspects of neural collapse, we adopt similar $\mathcal{NC}$ metrics from [1, 4, 6], given by

$$\mathcal{NC}_1 := \frac{1}{K}\text{trace}(\boldsymbol{\Sigma}_W \boldsymbol{\Sigma}_B^{\dagger})$$
$$\mathcal{NC}_2 := \left\| \frac{\boldsymbol{W}^\top \boldsymbol{W}}{\|\boldsymbol{W}^\top \boldsymbol{W}\|_F} - \frac{1}{\sqrt{K-1}} \left(\boldsymbol{I}_K - \mathbf{1}_K \mathbf{1}_K^\top\right) \right\|_F$$
$$\mathcal{NC}_3 := \left\| \frac{\boldsymbol{W}^\top \overline{\boldsymbol{H}}}{\|\boldsymbol{W}^\top \overline{\boldsymbol{H}}\|_F} - \frac{1}{\sqrt{K-1}} \left(\boldsymbol{I}_K - \mathbf{1}_K \mathbf{1}_K^\top\right) \right\|_F$$

where $\boldsymbol{\Sigma}_W$ and $\boldsymbol{\Sigma}_B$ are the within-class and between-class covariance [1], $\boldsymbol{\Sigma}_B^{\dagger}$ denotes pseudo inverse of $\boldsymbol{\Sigma}_B$, and $\overline{\boldsymbol{H}}$ is the centered class mean matrix in (7). More specifically, $\mathcal{NC}_1$ measures **NC1** within class variability collapse, $\mathcal{NC}_2$ measures **NC2** convergence to the simplex ETF, and $\mathcal{NC}_3$ measures **NC3** duality collapse.

4.1 Validation of the UFM for training networks with feature normalization

In Section 2, our study of the Riemannian optimization problem (3) is based upon the UFM, where we assume $\boldsymbol{H}$ is a free optimization variable. Here, we justify this assumption by showing that $\mathcal{NC}$ happens for training overparameterized networks even when the training labels are completely random. By using random labels, we disassociate the input from their class labels, by which we can characterize the approximation power of the features of overparmeterized models. To show this, we train ResNet-18 on CIFAR10 dataset with random labels, with normalized features and classifiers with varying widths (i.e., the number of feature maps resulting from the first convolutional layer).

As shown in Figure 3, we observe that increasing the width of the network allows for perfect classification on the training data even when the labels are random. Furthermore, increasing the network width also leads to better $\mathcal{NC}$, measured by the decrease in each $\mathcal{NC}$ metric. This corroborates that (*i*) our assumption of UFM is reasonable given that $\mathcal{NC}$ seems to be independent of the input data, and (*ii*) $\mathcal{NC}$ happens under the our constraint formulation (3) on practical networks.

4.2 Improved training speed and better representation quality with feature normalization

We now investigate the benefits of using feature normalization, more specifically for improved training speed and better representation quality. First, we consider the UFM formulation, where we optimize Problem (3) and compare to the regularized UFM in [4]. The results are shown in

[3] Both CIFAR10 and CIFAR100 are publicly available and are licensed under the MIT license.

Figure 4. We can see that normalizing features over the sphere consistently results in reaching perfect classification and greater feature collapse (i.e., smaller $\mathcal{NC}_1$) quicker than penalizing the features. To demonstrate that these behaviors are reflected in training practical deep networks, we train both ResNet-18 and ResNet-50 architectures on a reduced CIFAR100 [12] dataset with $N = 3000$ total samples, comparing the training accuracy and neural collapse with and without feature normalization. The results are shown in Figure 5.

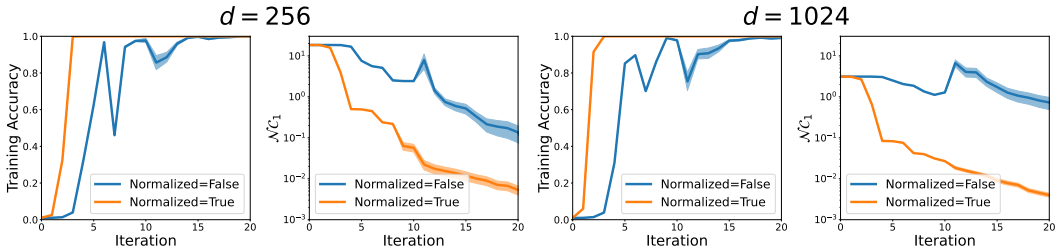

Figure 4: **Faster training/feature collapse of UFM with feature normalization.** Average (deviation denoted by shaded area) training accuracy and $\mathcal{NC}_1$ of UFM over 10 trials of (Riemmanian) gradient descent with exact line search. We set $K = 100$ classes, $n = 30$ samples per class.

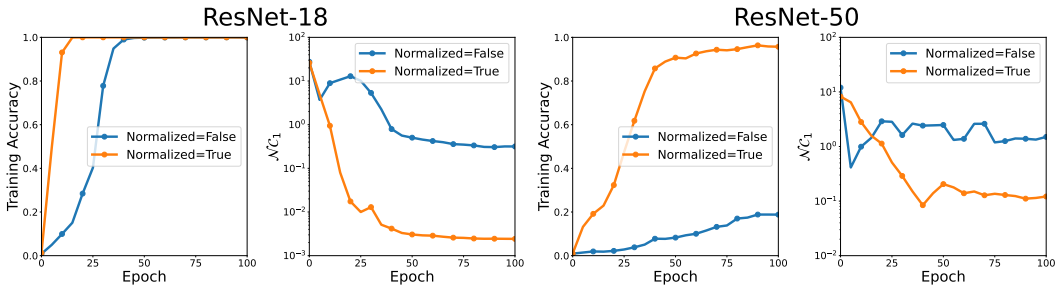

Figure 5: **Faster training/feature collapse with ResNet on CIFAR100 with feature normalization.** Training accuracy and $\mathcal{NC}_1$ of ResNet-18 and ResNet-50 on CIFAR100 with $n = 30$ over 100 epochs.

From Figure 5 (left), we can see that for the ResNet-18 network, we reach perfect classification of the training data about 10-20 epochs sooner by using feature normalization compared to that of the unconstrained formulation. From Figure 5 (right), training the ResNet-50 network without feature normalization for 100 epochs shows slow convergence with poor training accuracy, whereas using feature normalization arrives at above 90% training accuracy in the same number of epochs. By keeping the size of the dataset the same and increasing the number of parameters, it is reasonable that optimizing the ResNet-50 network is more challenging due to the higher degree of overparameterization, yet this effect is mitigated by using feature normalization.

At the same time, for both architectures, using feature normalization leads to greater feature collapse (i.e., smaller $\mathcal{NC}_1$) compared to that of the unconstrained counterpart. As shown in recent work [1, 6, 36] , better $\mathcal{NC}$ often leads to better generalization performance. Last but not least, we believe the benefits of feature normalization are not limited to the evidence that we showed here, it could also lead to better generalization [25, 7] and robustness [22, 23] that are worth further exploration.

5 Conclusion & Discussion

In this work, we study the prevalence of the $\mathcal{NC}$ phenomenon with normalized features. Based upon the assumption of UFM, we formulate the problem as a Riemannian optimization problem over the product of spheres. We showed that the loss function is a strict saddle function over the manifold with respect to the last-layer features and classifiers, with no other spurious local minimizers. We demonstrate this on practical deep network training, and show practical benefits of feature normalization in terms of training speed and learned representation quality. As future work, we would like to expand the study to other popular loss functions such as contrastive loss and study the settings $d \ll K$, which could be of great importance for studying self-supervised learning.

References

[1] Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020.

[2] XY Han, Vardan Papyan, and David L Donoho. Neural collapse under mse loss: Proximity to and dynamics on the central path. *arXiv preprint arXiv:2106.02073*, 2021.

[3] Cong Fang, Hangfeng He, Qi Long, and Weijie J Su. Exploring deep neural networks via layer-peeled model: Minority collapse in imbalanced training. *Proceedings of the National Academy of Sciences*, 118(43), 2021.

[4] Zhihui Zhu, Tianyu Ding, Jinxin Zhou, Xiao Li, Chong You, Jeremias Sulam, and Qing Qu. A geometric analysis of neural collapse with unconstrained features. *Advances in Neural Information Processing Systems*, 34, 2021.

[5] Tom Tirer and Joan Bruna. Extended unconstrained features model for exploring deep neural collapse. *arXiv preprint arXiv:2202.08087*, 2022.

[6] Jinxin Zhou, Xiao Li, Tianyu Ding, Chong You, Qing Qu, and Zhihui Zhu. On the optimization landscape of neural collapse under mse loss: Global optimality with unconstrained features. *arXiv preprint arXiv:2203.01238*, 2022.

[7] Florian Graf, Christoph Hofer, Marc Niethammer, and Roland Kwitt. Dissecting supervised constrastive learning. In *International Conference on Machine Learning*, pages 3821–3830. PMLR, 2021.

[8] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[11] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. at&t labs, 2010.

[12] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[14] Uri Cohen, SueYeon Chung, Daniel D Lee, and Haim Sompolinsky. Separability and geometry of object manifolds in deep neural networks. *Nature communications*, 11(1):1–13, 2020.

[15] Jonathan Mamou, Hang Le, Miguel Del Rio, Cory Stephenson, Hanlin Tang, Yoon Kim, and Sueyeon Chung. Emergence of separable manifolds in deep language representations. In *International Conference on Machine Learning*, pages 6713–6723. PMLR, 2020.

[16] Diego Doimo, Aldo Glielmo, Alessio Ansuini, and Alessandro Laio. Hierarchical nucleation in deep neural networks. *Advances in Neural Information Processing Systems*, 33:7526–7536, 2020.

[17] Gregory Naitzat, Andrey Zhitnikov, and Lek-Heng Lim. Topology of deep neural networks. *J. Mach. Learn. Res.*, 21(184):1–40, 2020.

[18] Stefano Recanatesi, Matthew Farrell, Madhu Advani, Timothy Moore, Guillaume Lajoie, and Eric Shea-Brown. Dimensionality compression and expansion in deep neural networks. *arXiv preprint arXiv:1906.00443*, 2019.

[19] Nicholas Frosst, Nicolas Papernot, and Geoffrey Hinton. Analyzing and improving representations with the soft nearest neighbor loss. In *International conference on machine learning*, pages 2012–2020. PMLR, 2019.

[20] Akshaya Jha, Peter H Matthews, and Nicholas Z Muller. Does environmental policy affect income inequality? evidence from the clean air act. In *AEA Papers and Proceedings*, volume 109, pages 271–76, 2019.

[21] Christoph Hofer, Florian Graf, Marc Niethammer, and Roland Kwitt. Topologically densified distributions. In *International Conference on Machine Learning*, pages 4304–4313. PMLR, 2020.

[22] Kwan Ho Ryan Chan, Yaodong Yu, Chong You, Haozhi Qi, John Wright, and Yi Ma. Deep networks from the principle of rate reduction. *arXiv preprint arXiv:2010.14765*, 2020.

[23] Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. *arXiv preprint arXiv:2006.08558*, 2020.

[24] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.

[25] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020.

[26] Jianfeng Lu and Stefan Steinerberger. Neural collapse with cross-entropy loss. *arXiv preprint arXiv:2012.08465*, 2020.

[27] Stephan Wojtowytsch et al. On the emergence of simplex symmetry in the final and penultimate layers of neural network classifiers. *arXiv preprint arXiv:2012.05420*, 2020.

[28] Wenlong Ji, Yiping Lu, Yiliang Zhang, Zhun Deng, and Weijie J Su. An unconstrained layer-peeled perspective on neural collapse. *arXiv preprint arXiv:2110.02796*, 2021.

[29] Dustin G Mixon, Hans Parshall, and Jianzong Pi. Neural collapse with unconstrained features. *arXiv preprint arXiv:2011.11619*, 2020.

[30] Cong Fang, Hangfeng He, Qi Long, and Weijie J Su. Layer-peeled model: Toward understanding well-trained deep neural networks. *arXiv preprint arXiv:2101.12699*, 2021.

[31] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks: a view from the width. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6232–6240, 2017.

[32] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

[33] Uri Shaham, Alexander Cloninger, and Ronald R Coifman. Provable approximation properties for deep neural networks. *Applied and Computational Harmonic Analysis*, 44(3):537–557, 2018.

[34] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Proceedings of The 28th Conference on Learning Theory*, pages 797–842, 2015.

[35] Ju Sun, Qing Qu, and John Wright. When are nonconvex problems not scary? *arXiv preprint arXiv:1510.06096*, 2015.

[36] Tomer Galanti, András György, and Marcus Hutter. On the role of neural collapse in transfer learning. In *International Conference on Learning Representations*, 2022.

[37] Like Hui, Mikhail Belkin, and Preetum Nakkiran. Limitations of neural collapse for understanding generalization in deep learning. *arXiv preprint arXiv:2202.08384*, 2022.

[38] Vardan Papyan. Traces of class/cross-class structure pervade deep learning spectra. *Journal of Machine Learning Research*, 21(252):1–64, 2020.

[39] Ido Ben-Shaul and Shai Dekel. Nearest class-center simplification through intermediate layers. *arXiv preprint arXiv:2201.08924*, 2022.

[40] Liang Xie, Yibo Yang, Deng Cai, Dacheng Tao, and Xiaofei He. Neural collapse inspired attraction-repulsion-balanced loss for imbalanced learning. *arXiv preprint arXiv:2204.08735*, 2022.

[41] Yibo Yang, Liang Xie, Shixiang Chen, Xiangtai Li, Zhouchen Lin, and Dacheng Tao. Do we really need a learnable classifier at the end of deep neural network? *arXiv preprint arXiv:2203.09081*, 2022.

[42] Nicolas Boumal. An introduction to optimization on smooth manifolds. *Available online, May*, 3, 2020.

[43] C Jin, MI Jordan, R Ge, P Netrapalli, and SM Kakade. How to escape saddle points efficiently. In *International Conference on Machine Learning*, pages 2727–2752, 2017.

[44] Simon S Du, Chi Jin, Jason D Lee, Michael I Jordan, Aarti Singh, and Barnabas Poczos. Gradient descent can take exponential time to escape saddle points. In *Advances in neural information processing systems*, pages 1067–1077, 2017.

[45] Nicolas Boumal. An introduction to optimization on smooth manifolds. To appear with Cambridge University Press, Jan 2022.

[46] Jiang Hu, Xin Liu, Zai-Wen Wen, and Ya-Xiang Yuan. A brief introduction to manifold optimization. *Journal of the Operations Research Society of China*, 8(2):199–248, 2020.

[47] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[48] Christopher Criscitiello and Nicolas Boumal. Efficiently escaping saddle points on manifolds. *Advances in Neural Information Processing Systems*, 32, 2019.

[49] Yue Sun, Nicolas Flammarion, and Maryam Fazel. Escaping from saddle points on riemannian manifolds. *Advances in Neural Information Processing Systems*, 32, 2019.

[50] Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[51] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *Advances in Neural Information Processing Systems*, 32, 2019.

[52] Qing Qu, Ju Sun, and John Wright. Finding a sparse vector in a subspace: Linear sparsity using alternating directions. In *Advances in Neural Information Processing Systems*, pages 3401–3409, 2014.

[53] Ju Sun, Qing Qu, and John Wright. Complete dictionary recovery over the sphere I: Overview and the geometric picture. *IEEE Transactions on Information Theory*, 63(2):853–884, 2016.

[54] Ju Sun, Qing Qu, and John Wright. Complete dictionary recovery over the sphere II: Recovery by Riemannian trust−region method. *IEEE Transactions on Information Theory*, 63(2):853–884, 2017.

[55] Qing Qu, Yuexiang Zhai, Xiao Li, Yuqian Zhang, and Zhihui Zhu. Geometric analysis of nonconvex optimization landscapes for overcomplete learning. In *International Conference on Learning Representations*, 2020.

[56] Qing Qu, Zhihui Zhu, Xiao Li, Manolis C. Tsakiris, John Wright, and René Vidal. Finding the sparsest vectors in a subspace: Theory, algorithms, and applications. *arXiv preprint arXiv:2001.06970*, 2020.

[57] Han-Wen Kuo, Yenson Lau, Yuqian Zhang, and John Wright. Geometry and symmetry in short-and-sparse deconvolution. In *International Conference on Machine Learning*, pages 3570–3580. PMLR, 2019.

[58] Qing Qu, Xiao Li, and Zhihui Zhu. Exact recovery of multichannel sparse blind deconvolution via gradient descent. *SIAM Journal on Imaging Sciences*, 13(3):1630–1652, 2020.

[59] Yenson Lau, Qing Qu, Han-Wen Kuo, Pengcheng Zhou, Yuqian Zhang, and John Wright. Short and sparse deconvolution — a geometric approach. In *International Conference on Learning Representations*, 2020.

[60] Yuqian Zhang, Han-Wen Kuo, and John Wright. Structured local optima in sparse blind deconvolution. *IEEE Transactions on Information Theory*, 66(1):419–452, 2019.

[61] Yuqian Zhang, Qing Qu, and John Wright. From symmetry to geometry: Tractable nonconvex problems. *arXiv preprint arXiv:2007.06753*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We state in Section 2 that our analysis uses the unconstrained feature model, and in Section 3 we discuss the strong requirement of $d > N$ for Theorem 2.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] This paper mainly focuses on understanding the neural collapse phenomena observed in practical neural networks. We believe there is no potential negative societal impact in this work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] We explicitly mention the assumptions in Section 3.
   (b) Did you include complete proofs of all theoretical results? [Yes] We include all the proofs in the Appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The CIFAR datasets used here are cited and easily accessible. The GitHub code repository will be made available after review.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Figure 2 does not show error bars since the attained value is always the same independent of initialization due to the benign global landscape. In Section 4, we used shaded area as error bars for Figure 4, but regardless we have observed that the results are similar and reproducible independent of the random seed.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 4.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cite them in Section 4.
   (b) Did you mention the license of the assets? [Yes] This is mentioned in Section 4.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] This is mentioned in Section 4.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]