Understanding Deep Contrastive Learning via Coordinate-wise Optimization

Yuandong Tian
Meta AI (FAIR)
yuandong@meta.com

Abstract

We show that Contrastive Learning (CL) under a broad family of loss functions (including InfoNCE) has a unified formulation of coordinate-wise optimization on the network parameter $\boldsymbol{\theta}$ and pairwise importance α, where the *max player* $\boldsymbol{\theta}$ learns representation for contrastiveness, and the *min player* α puts more weights on pairs of distinct samples that share similar representations. The resulting formulation, called **$\boldsymbol{\alpha}$-CL**, unifies not only various existing contrastive losses, which differ by how sample-pair importance α is constructed, but also is able to extrapolate to give novel contrastive losses beyond popular ones, opening a new avenue of contrastive loss design. These novel losses yield comparable (or better) performance on CIFAR10, STL-10 and CIFAR-100 than classic InfoNCE. Furthermore, we also analyze the max player in detail: we prove that with fixed α, max player is equivalent to Principal Component Analysis (PCA) for deep linear network, and almost all local minima are global and rank-1, recovering optimal PCA solutions. Finally, we extend our analysis on max player to 2-layer ReLU networks, showing that its fixed points can have higher ranks.

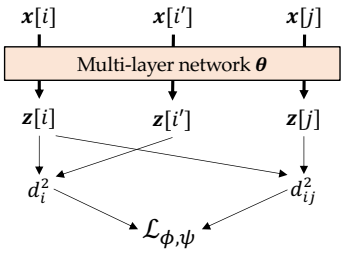
1 Introduction

While contrastive self-supervised learning has been shown to learn good features (Chen et al., 2020; He et al., 2020; Oord et al., 2018) and in many cases, comparable with features learned from supervised learning, it remains an open problem what features it learns, in particular when deep nonlinear networks are used. Theory on this is quite sparse, mostly focusing on loss function (Arora et al., 2019) and treating the networks as a black-box function approximator.

In this paper, we present a novel perspective of contrastive learning (CL) for a broad family of contrastive loss functions $\mathcal{L}(\boldsymbol{\theta})$: minimizing $\mathcal{L}(\boldsymbol{\theta})$ corresponds to a *coordinate-wise optimization* procedure on an objective $\mathcal{E}_\alpha(\boldsymbol{\theta}) - \mathcal{R}(\alpha)$ with respect to network parameter $\boldsymbol{\theta}$ and *pairwise importance* α on batch samples, where $\mathcal{E}_\alpha(\boldsymbol{\theta})$ is an energy function and $\mathcal{R}(\alpha)$ is a regularizer, both associated with the original contrastive loss $\mathcal{L}$. In this view, the *max player* $\boldsymbol{\theta}$ learns a representation to maximize the contrastiveness of different samples and keep different augmentation view of the same sample similar, while the *min player* α puts more weights on pairs of different samples that appear similar in the representation space, subject to regularization. Empirically, this formulation, named Pair-weighed Contrastive Learning (**$\boldsymbol{\alpha}$-CL**), when coupled with various regularization terms, yields novel contrastive losses that show comparable (or better) performance in CIFAR10 (Krizhevsky et al., 2009) and STL-10 (Coates et al., 2011).

We then focus on the behavior of the max player who does *representation learning* via maximizing the energy function $\mathcal{E}_\alpha(\boldsymbol{\theta})$. When the underlying network is deep linear, we show that $\max_{\boldsymbol{\theta}} \mathcal{E}_\alpha(\boldsymbol{\theta})$ is the loss function (under re-parameterization) of Principal Component Analysis (PCA) (Wold et al., 1987), a century-old unsupervised dimension reduction method. To further show they are equivalent, we prove that the nonlinear training dynamics of CL with a linear multi-layer feedforward network (MLP) enjoys nice properties: with proper weight normalization, almost all its local optima are

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



Contrastive Loss	$\phi(x)$	$\psi(x)$
InfoNCE (Oord et al., 2018)	$\tau \log(\epsilon + x)$	$e^{x/\tau}$
MINE (Belghazi et al., 2018)	$\log(x)$	e^x
Triplet (Schroff et al., 2015)	x	$[x + \epsilon]_+$
Soft Triplet (Tian et al., 2020c)	$\tau \log(1 + x)$	$e^{x/\tau+\epsilon}$
N+1 Tuplet (Sohn, 2016)	$\log(1 + x)$	e^x
Lifted Structured (Oh Song et al., 2016)	$[\log(x)]_+^2$	$e^{x+\epsilon}$
Modified Triplet Eqn. 10 (Coria et al., 2020)	x	$\mathrm{sigmoid}(cx)$
Triplet Contrastive Eqn. 2 (Ji et al., 2021)	linear	linear

Figure 1: Problem Setting. **Left**: Data points (i-th sample $\boldsymbol{x}[i]$ and its augmented version $\boldsymbol{x}[i']$, j-th sample $\boldsymbol{x}[j]$) are sent to networks with weights $\boldsymbol{\theta}$, to yield outputs $\boldsymbol{z}[i]$, $\boldsymbol{z}[i']$ and $\boldsymbol{z}[j]$. From the outputs $\boldsymbol{z}$, we compute pairwise squared distance d_{ij}^2 between $\boldsymbol{z}[i]$ and $\boldsymbol{z}[j]$ and intra-class squared distance d_i^2 between $\boldsymbol{z}[i]$ and $\boldsymbol{z}[i']$ for contrastive learning with a general family of contrastive loss $\mathcal{L}_{\phi,\psi}$ (Eqn. 1). **Right**: Different existing loss functions corresponds to different monotonous functions ϕ and ψ. Here $[x]_+ := \max(x, 0)$.

global, achieving optimal PCA objective, and are rank-1. The only difference here is that the data augmentation provides negative eigen-directions to avoid.

Furthermore, we extend our analysis to 2-layer ReLU network, to explore the difference between the rank-1 PCA solution and the solution learned by a nonlinear network. Assuming the data follow an orthogonal mixture model, the 2-layer ReLU networks enjoy similar dynamics as the linear one, except for a special *sticky weight rule* that keeps the low-layer weights to be non-negative and stays zero when touching zero. In the case of one hidden node, we prove that the solution in ReLU always picks a single mode from the mixtures. In the case of multiple hidden nodes, the resulting solution is not necessarily rank-1.

2 Related Work

Contrastive learning. While many contrastive learning techniques (e.g., SimCLR (Chen et al., 2020), MoCo (He et al., 2020), PIRL (Misra & Maaten, 2020), SwAV (Caron et al., 2020), DeepCluster (Caron et al., 2018), Barlow Twins (Zbontar et al., 2021), InstDis (Wu et al., 2018), etc) have been proposed empirically and able to learn good representations for downstream tasks, theoretical study is relatively sparse, mostly focusing on loss function itself (Tian et al., 2020b; HaoChen et al., 2021; Arora et al., 2019), e.g., the relationship of loss functions with mutual information (MI). To our knowledge, there is no analysis that combines the property of neural network and that of loss functions.

Theoretical analysis of deep networks. Many works focus on analysis of deep linear networks in supervised setting, where label is given. (Baldi & Hornik, 1989; Zhou & Liang, 2018; Kawaguchi, 2016) analyze the critical points of linear networks. (Saxe et al., 2014; Arora et al., 2018) also analyze the training dynamics. On the other hand, analyzing nonlinear networks has been a difficult task. Existing works mostly lie in supervised learning, e.g., teacher-student setting (Tian, 2020; Allen-Zhu et al., 2018), landscape (Safran & Shamir, 2018). For contrastive learning, recent work (Wen & Li, 2021) analyzes the dynamics of 1-layer ReLU networks with a specific weight structure, and (Jing et al., 2022) analyzes the collapsing behaviors in 2-layer linear network for CL. To our best knowledge, we are not aware of such analysis on deep networks (> 2 layers, linear or nonlinear) in the context of CL.

Connection between Principal Component Analysis (PCA) and Self-supervised Learning. (Lee et al., 2021) establishes the statistical connection between non-linear Canonical Component Analysis (CCA) and SimSiam (Chen & He, 2020) for any zero-mean encoder, without considering the aspect of training dynamics. In contrast, we reformulate contrastive learning as coordinate-wise optimization procedure with min/max players, in which the max player is a reparameterization of PCA optimized with gradient descent, and analyze its training dynamics in the presence of specific neural architectures.

3 Contrastive Learning as Coordinate-wise Optimization

Notation. Suppose we have N pairs of samples $\{\boldsymbol{x}[i]\}_{i=1}^N$ and $\{\boldsymbol{x}[i']\}_{i=1}^N$. Both $\boldsymbol{x}[i]$ and $\boldsymbol{x}[i']$ are augmented samples from sample i and $\boldsymbol{x}$ represents the input batch. These samples are sent to

neural networks and $\boldsymbol{z}[i]$ and $\boldsymbol{z}[i']$ are their outputs. The goal of contrastive learning (CL) is to find the representation to maximize the squared distance $d_{ij}^2 := \|\boldsymbol{z}[i] - \boldsymbol{z}[j]\|_2^2/2$ between distinct samples i and j, and minimize the squared distance $d_i^2 := \|\boldsymbol{z}[i] - \boldsymbol{z}[i']\|_2^2/2$ between different data augmentations $\boldsymbol{x}[i]$ and $\boldsymbol{x}[i']$ of the same sample i.

3.1 A general family of contrastive loss

We consider minimizing a general family of loss functions $\mathcal{L}_{\phi,\psi}$, where ϕ and ψ are monotonously increasing and differentiable scalar functions (define $\xi_i := \sum_{j\neq i}\psi(d_i^2 - d_{ij}^2)$ for notation brevity):

$$\min_{\boldsymbol{\theta}} \mathcal{L}_{\phi,\psi}(\boldsymbol{\theta}) := \sum_{i=1}^{N}\phi(\xi_i) = \sum_{i=1}^{N}\phi\left(\sum_{j\neq i}\psi(d_i^2 - d_{ij}^2)\right) \tag{1}$$

Both i and j run from 1 to N. With different ϕ and ψ, Eqn. 1 covers many loss functions (Tbl. 1). In particular, setting $\phi(x) = \tau\log(\epsilon + x)$ and $\psi(x) = \exp(x/\tau)$ gives a generalized version of InfoNCE loss (Oord et al., 2018):

$$\mathcal{L}_{nce} := -\tau\sum_{i=1}^{N}\log\frac{\exp(-d_i^2/\tau)}{\epsilon\exp(-d_i^2/\tau) + \sum_{j\neq i}\exp(-d_{ij}^2/\tau)} = \tau\sum_{i=1}^{N}\log\left(\epsilon + \sum_{j\neq i}e^{\frac{d_i^2 - d_{ij}^2}{\tau}}\right) \tag{2}$$

where $\epsilon > 0$ is some constant not related to $\boldsymbol{z}[i]$ and $\boldsymbol{z}[i']$. $\epsilon = 1$ has been used in many works (He et al., 2020; Tian et al., 2020a). Setting $\epsilon = 0$ yields SimCLR setting (Chen et al., 2020) where the denominator doesn't contains $\exp(-d_i^2/\tau)$. This is also used in (Yeh et al., 2021).

3.2 The other side of gradient descent of contrastive loss

To minimize $\mathcal{L}_{\phi,\psi}$, gradient descent follows its negative gradient direction. As a first discovery of this work, it turns out that the gradient descent of the loss function $\mathcal{L}$ is the *gradient ascent* direction of another energy function $\mathcal{E}_\alpha$:

Theorem 1. *For any differential mapping $\boldsymbol{z} = \boldsymbol{z}(\boldsymbol{x};\boldsymbol{\theta})$, gradient descent of $\mathcal{L}_{\phi,\psi}$ is equivalent to* ***gradient ascent*** *of the objective $\mathcal{E}_\alpha(\boldsymbol{\theta}) := \mathrm{tr}(\mathbb{C}_\alpha[\boldsymbol{z}(\boldsymbol{\theta}), \boldsymbol{z}(\boldsymbol{\theta})])$:*

$$\frac{\partial\mathcal{L}_{\phi,\psi}}{\partial\boldsymbol{\theta}} = -\frac{1}{2}\frac{\partial\mathcal{E}_\alpha}{\partial\boldsymbol{\theta}}\Big|_{\alpha=\alpha(\boldsymbol{\theta})} \tag{3}$$

Here the pairwise importance $\alpha = \alpha(\boldsymbol{\theta}) := \{\alpha_{ij}(\boldsymbol{\theta})\}$ *is a function of input batch $\boldsymbol{x}$, defined as:*

$$\alpha_{ij}(\boldsymbol{\theta}) := \phi'(\xi_i)\psi'(d_i^2 - d_{ij}^2) \geq 0 \tag{4}$$

where $\phi', \psi' \geq 0$ are derivatives of ϕ, ψ. The contrastive covariance $\mathbb{C}_\alpha[\cdot,\cdot]$ *is defined as:*

$$\mathbb{C}_\alpha[\boldsymbol{a},\boldsymbol{b}] := \sum_{i=1}^{N}\sum_{j\neq i}\alpha_{ij}(\boldsymbol{a}[i]-\boldsymbol{a}[j])(\boldsymbol{b}[i]-\boldsymbol{b}[j])^\top - \sum_{i=1}^{N}\left(\sum_{j\neq i}\alpha_{ij}\right)(\boldsymbol{a}[i]-\boldsymbol{a}[i'])(\boldsymbol{b}[i]-\boldsymbol{b}[i'])^\top \tag{5}$$

That is, ***minimizing*** *the loss function $\mathcal{L}_{\phi,\psi}(\boldsymbol{\theta})$ can be regarded as* ***maximizing*** *the energy function $\mathcal{E}_{\alpha=\mathrm{sg}(\alpha(\boldsymbol{\theta}))}(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. Here $\mathrm{sg}(\cdot)$ means stop-gradient, i.e., the gradient of $\boldsymbol{\theta}$ is not backpropagated into $\alpha(\boldsymbol{\theta})$.*

Please check Supplementary Materials (SM) for all proofs. From the definition of energy $\mathcal{E}_\alpha(\boldsymbol{\theta})$, it is clear that α_{ij} determines the importance of each sample pair $\boldsymbol{x}[i]$ and $\boldsymbol{x}[j]$. For (i,j)-pair that "deserves attention", α_{ij} is large so that it plays a large role in the contrastive covariance term. In particular, for InfoNCE loss with $\epsilon = 0$, the pairwise importance α takes the following form:

$$\alpha_{ij} = \frac{\exp(-d_{ij}^2/\tau)}{\sum_{j\neq i}\exp(-d_{ij}^2/\tau)} > 0 \tag{6}$$

which means that InfoNCE focuses on (i,j)-pair with small squared distance d_{ij}^2. If both ϕ and ψ are linear, then $\alpha_{ij} = \mathrm{const}$ and $\mathcal{L}$ is a simple subtraction of positive/negative squared distances.

From Thm. 1, an important observation is that when propagating gradient w.r.t. $\boldsymbol{\theta}$ using the objective $\mathcal{E}_\alpha(\boldsymbol{\theta})$ during the backward pass, the gradient does not propagate into $\alpha(\boldsymbol{\theta})$, even if $\alpha(\boldsymbol{\theta})$ is a function of $\boldsymbol{\theta}$ in the forward pass. In fact, in Sec. 6 we show that propagating gradient through $\alpha(\boldsymbol{\theta})$ yields worse empirical performance. This suggests that α should be treated as an *independent* variable when optimizing $\boldsymbol{\theta}$. It turns out that if $\psi(x)$ is an exponential function (as in most cases of Tbl. 1), this is indeed true and α can be determined by a separate optimization procedure:

Theorem 2. *If $\psi(x) = e^{x/\tau}$, then the corresponding pairwise importance α (Eqn. 4) is the solution to the minimization problem:*

$$\alpha(\boldsymbol{\theta}) = \arg\min_{\alpha\in\mathcal{A}} \mathcal{E}_\alpha(\boldsymbol{\theta}) - \mathcal{R}(\alpha), \qquad \mathcal{A} := \left\{ \alpha : \quad \forall i,\ \sum_{j\neq i} \alpha_{ij} = \tau^{-1}\xi_i\phi'(\xi_i),\ \ \alpha_{ij} \geq 0 \right\} \tag{7}$$

Here the regularization $\mathcal{R}(\alpha) = \mathcal{R}_H(\alpha) := 2\tau \sum_{i=1}^N H(\alpha_{i\cdot}) = -2\tau \sum_{i=1}^N \sum_{j\neq i} \alpha_{ij} \log \alpha_{ij}$.

For InfoNCE, the feasible set $\mathcal{A}$ becomes $\{\alpha : \alpha \geq 0, \sum_{j\neq i} \alpha_{ij} = \xi_i/(\xi_i + \epsilon)\}$. This means that if i-th sample is already well-separated (small intra-augmentation distance d_i and large inter-augmentation distance d_{ij}), then ξ_i is small, the summation of weights $\sum_{j\neq i} \alpha_{ij}$ associated with sample i is also small and such a sample is overall discounted. Setting $\epsilon = 0$ reduces to sample-agnostic constraint (i.e., $\sum_{j\neq i} \alpha_{ij} = 1$).

Thm. 2 leads to a novel perspective of *coordinate-wise optimization* for Contrastive Learning (CL):

Corollary 1 (Contrastive Learning as Coordinate-wise Optimization)**.** *If $\psi(x) = e^{x/\tau}$, minimizing $\mathcal{L}_{\phi,\psi}$ is equivalent to the following iterative procedure:*

$$\textit{(Min-player } \alpha\textit{)} \qquad \alpha_t = \arg\min_{\alpha\in\mathcal{A}} \mathcal{E}_\alpha(\boldsymbol{\theta}_t) - \mathcal{R}(\alpha) \tag{8a}$$

$$\textit{(Max-player } \boldsymbol{\theta}\textit{)} \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta\nabla_{\boldsymbol{\theta}}\mathcal{E}_{\alpha_t}(\boldsymbol{\theta}) \tag{8b}$$

Intuitively, the max player $\boldsymbol{\theta}$ (Eqn. 8b) performs one-step gradient ascent for the objective $\mathcal{E}_\alpha(\boldsymbol{\theta}) - \mathcal{R}(\alpha)$, *learns a representation* to maximize the distance of different samples and minimize the distance of the same sample with different augmentations (as suggested by $\mathbb{C}_\alpha[\boldsymbol{z}, \boldsymbol{z}]$). On the other hand, the "min player" α (Eqn. 8a) finds optimal α analytically, assigning high weights on confusing pairs for "max player" to solve.

Relation to max-min formulation. While Corollary 1 looks very similar to max-min formulation, important differences exist. Different from traditional max-min formulation, in Corollary 1 there is asymmetry between $\boldsymbol{\theta}$ and α. First, $\boldsymbol{\theta}$ only follows one step update along gradient ascent direction of $\max_{\boldsymbol{\theta}} \mathcal{E}_\alpha(\boldsymbol{\theta})$, while α is solved analytically. Second, due to the stop-gradient operator, the gradient of $\boldsymbol{\theta}$ contains no knowledge on how $\boldsymbol{\theta}$ changes α. This prevents $\boldsymbol{\theta}$ from adapting to α's response on changing $\boldsymbol{\theta}$. Both give advantages to min-player α to find the confusing sample pairs more effectively.

Relation to hard-negative samples. While many previous works (Kalantidis et al., 2020; Robinson et al., 2021) focus on seeking and putting more weights on hard samples, Corollary 1 shows that contrastive losses already have such mechanism at the batch level, focusing on "hard-negative pairs" beyond hard-negative samples.

From this formulation, different pairwise importance α corresponds to different loss functions within the loss family specified by Eqn. 1, and choosing among this family (i.e., different ϕ and ψ) can be regarded as choosing different α when optimizing the *same* objective $\mathcal{E}_\alpha(\boldsymbol{\theta})$. Based on this observation, we now propose the following training framework called $\boldsymbol{\alpha}$**-CL**:

Definition 1 (Pair-weighed Contrastive Learning (α-CL))**.** *Optimize $\boldsymbol{\theta}$ by gradient ascent: $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta\nabla_{\boldsymbol{\theta}}\mathcal{E}_{\mathrm{sg}(\alpha_t)}(\boldsymbol{\theta})$, with the energy $\mathcal{E}_\alpha(\boldsymbol{\theta})$ defined in Thm. 1 and pairwise importance $\alpha_t = \alpha(\boldsymbol{\theta}_t)$.*

In α-CL, choosing α can be achieved by either implicitly specifying a regularizer $\mathcal{R}(\alpha)$ and solve Eqn. 8a, or by a direct mapping $\alpha = \alpha(\boldsymbol{\theta})$ without any optimization. This opens a novel revenue for CL loss design. Initial experiments (Sec. 6) show that α-CL gives comparable (or even better) downstream performance in CIFAR10 and STL-10, compared to vanilla InfoNCE loss.

4 Representation Learning in Deep Linear CL is PCA

In Corollary 1, optimizing over α is well-understood, since $\mathcal{E}_\alpha(\boldsymbol{\theta})$ is *linear* w.r.t. α and $\mathcal{R}(\alpha)$ in general is a (strong) concave function. As a result, α has a unique optimal. On the other hand,

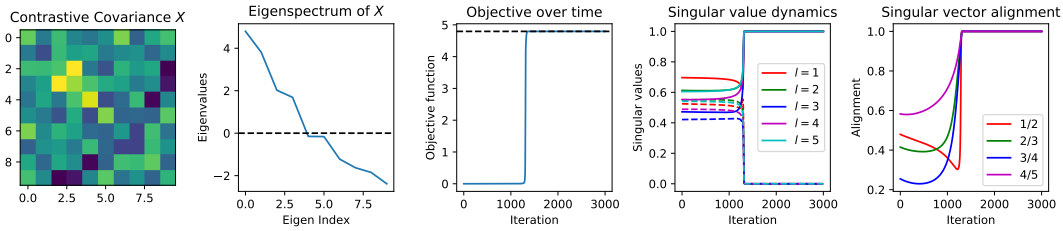


Figure 2: Dynamics of CL with multilayer ($L = 5$) linear network (`DeepLin`) with fixed α. Running the training dynamics (Lemma 1) quickly leads to convergence towards the maximal eigenvalue of X_α. For dynamics of singular value of W_l, the largest singular values (solid lines) converges to 1 while the second largest singular values (dashed lines) decay to 0.

understanding the max player $\max_{\boldsymbol{\theta}} \mathcal{E}_\alpha(\boldsymbol{\theta})$ is important since it performs *representation learning* in CL. It is also a hard problem because of non-convex optimization.

We start with a specific case when $\boldsymbol{z}$ is a deep linear network, i.e., $\boldsymbol{z} = W(\boldsymbol{\theta})\boldsymbol{x}$, where W is the equivalent linear mapping for the deep linear network, and $\boldsymbol{\theta}$ is the parameters to be optimized. Note that this covers many different kinds of deep linear networks, including VGG-like (Saxe et al., 2014), ResNet-like (Hardt & Ma, 2017) and DenseNet-like (Huang et al., 2017). For notation brevity, we define $\mathbb{C}_\alpha[\boldsymbol{x}] := \mathbb{C}_\alpha[\boldsymbol{x}, \boldsymbol{x}]$.

Corollary 2 (Representation learning in Deep Linear CL reparameterizes Principal Component Analysis (PCA))**.** *When $\boldsymbol{z} = W(\boldsymbol{\theta})\boldsymbol{x}$ with a constraint $WW^\top = I$, $\mathcal{E}_\alpha$ is the objective of Principal Component Analysis (PCA) with reparameterization $W = W(\boldsymbol{\theta})$:*

$$\max_{\boldsymbol{\theta}} \mathcal{E}_\alpha(\boldsymbol{\theta}) = \mathrm{tr}(W(\boldsymbol{\theta})X_\alpha W^\top(\boldsymbol{\theta})) \quad \text{s.t. } WW^\top = I \tag{9}$$

here $X_\alpha := \mathbb{C}_\alpha[\boldsymbol{x}]$ is the contrastive covariance of input $\boldsymbol{x}$.

As a comparison, in traditional Principal Component Analysis, the objective is (Kokiopoulou et al., 2011): $\max_W \mathrm{tr}(W\mathbb{V}_{\text{sample}}[\boldsymbol{x}]W^\top)$ subject to the constraint $WW^\top = I$, where $\mathbb{V}_{\text{sample}}[\boldsymbol{x}]$ is the empirical covariance of the dataset (here it is one batch). Therefore, X_α can be regarded as a generalized covariance matrix, possibly containing negative eigenvalues. In the case of supervised CL (i.e,. pairs from the same/different labels are treated as positive/negative (Khosla et al., 2020)), then it is connected with Fisher's Linear Discriminant Analysis (Fisher, 1936).

Here we show a mathematically rigorous connection between CL and dimensional reduction, as suggested intuitively in (Hadsell et al., 2006). Unlike traditional PCA, due to the presence of data augmentation, while symmetric, the contrastive covariance X_α is not necessarily a PSD matrix. Nevertheless, the intuition is the same: to find the direction that corresponds to maximal variation of the data.

While it is interesting to discover that CL with deep linear network is essentially a reparameterization of PCA, it remains elusive that such a reparameterization leads to the same solution of PCA, in particular when the network is deep (and may contain local optima). Also, PCA has an overall end-to-end constraint $WW^\top = I$, while in network training, we instead use normalization layers and it is unclear whether they are equivalent or not.

In this section, we show for a specific deep linear model, almost all its local maxima of Eqn. 9 are global and it indeed solves PCA.

4.1 A concrete deep linear model

We study a concrete deep linear network with parameters/weights $\boldsymbol{\theta} := \{W_l\}_{l=1}^L$:

$$\boldsymbol{z}[i] := W_L W_{L-1} \dots W_1 \boldsymbol{x}[i] \tag{10}$$

Here $W_l \in \mathbb{R}^{n_l \times n_{l-1}}$, n_l is the number of nodes at layer l, $\boldsymbol{z}[i]$ is the output of $\boldsymbol{x}[i]$ and similarly $\boldsymbol{z}[i']$ for $\boldsymbol{x}[i']$. We use $\boldsymbol{\theta}$ to represent the collection of weights at all layers. For convenience, we define the l-th layer activation $\boldsymbol{f}_l[i] = W_l \boldsymbol{f}_{l-1}[i]$. With this notation $\boldsymbol{f}_0[i] = \boldsymbol{x}[i]$ is the input and $\boldsymbol{z}[i] = W_L \boldsymbol{f}_{L-1}[i]$.

We call this setting `DeepLin`. The Jacobian matrix $W_{>l} := W_L W_{L-1} \dots W_{l+1}$ and $W := W_{>0} = W_L W_{L-1} \dots W_1$.

Lemma 1. *The training dynamics in* `DeepLin` *is* $\dot{W}_l = W_{>l}^\top W_{>l} W_l \mathbb{C}_\alpha[\boldsymbol{f}_{l-1}]$

Note that $\mathbb{C}_\alpha[\boldsymbol{f}_0] = \mathbb{C}_\alpha[\boldsymbol{x}] = X_\alpha$. Similar to supervised learning (Arora et al., 2018; Du et al., 2018b), nearby layers are also balanced: $\frac{\mathrm{d}}{\mathrm{d}t}\left(W_l W_l^\top - W_{l+1}^\top W_{l+1}\right) = 0$.

4.2 Normalization Constraints

Note that if we just run the training dynamics (Lemma 1) without any constraints, $\|W_l\|_F$ will go to infinity. Fortunately, empirical works already suggest various ways of normalization to stabilize the network training.

One popular technique in CL is ℓ_2 normalization. It is often put right after the output of the network and before the loss function $\mathcal{L}$ (Chen et al., 2020; Grill et al., 2020; He et al., 2020), i.e., $\hat{\boldsymbol{z}}[i] = \boldsymbol{z}[i]/\|\boldsymbol{z}[i]\|_2$. Besides, LayerNorm (Ba et al., 2016) (i.e., $\hat{\boldsymbol{f}}[i] = (\boldsymbol{f}[i] - \mathrm{mean}(\boldsymbol{f}[i]))/\mathrm{std}(\boldsymbol{f}[i])$) is extensively used in Transformer-based models (Xiong et al., 2020). Here we show that for gradient flow dynamics of MLP models, such normalization layers conserve $\|W_l\|_F$ for any l below it, regardless of loss function.

Lemma 2. *For MLP, if the weight W_l is below a ℓ_2-norm or LayerNorm layer, then $\frac{\mathrm{d}}{\mathrm{d}t}\|W_l\|_F^2 = 0$.*

Note that Lemma 2 also holds for nonlinear MLP with reversible activations, which includes ReLU (see SM). Therefore, without loss of generality, we consider the following complete objective for max player with `DeepLin` (here Θ is the constraint set of the weights due to normalization):

$$\max_{\boldsymbol{\theta}\in\Theta} \mathcal{E}_\alpha(\boldsymbol{\theta}) := \mathrm{tr}(W X_\alpha W^\top), \quad \Theta := \{\boldsymbol{\theta} : \ \|W_l\|_F = 1, 1 \le l \le L\} \tag{11}$$

4.3 Representation Learning with `DeepLin` is PCA

As one of our main contributions, the following theorem asserts that almost all local optimal solutions of Eqn. 11 are global, and the optimal objective corresponds to the PCA objective. Note that (Kawaguchi, 2016; Laurent & Brecht, 2018) proves no bad local optima for deep linear network in supervised learning, while here we give similar results for CL, and additionally we also give the (simple) rank-1 structure of all local optima.

Theorem 3 (Representation Learning with `DeepLin` is PCA)**.** *If $\lambda_{\max}(X_\alpha) > 0$, then for any local maximum $\boldsymbol{\theta} \in \Theta$ of Eqn. 11 whose $W_{>1}^\top W_{>1}$ has distinct maximal eigenvalue:*

- *there exists a set of unit vectors $\{\boldsymbol{v}_l\}_{l=0}^L$ so that $W_l = \boldsymbol{v}_l \boldsymbol{v}_{l-1}^\top$ for $1 \le l \le L$, in particular, $\boldsymbol{v}_0$ is the unit eigenvector corresponding to $\lambda_{\max}(X_\alpha)$,*
- *$\boldsymbol{\theta}$ is global optimal with objective $\mathcal{E}^* = \lambda_{\max}(X_\alpha)$.*

Corollary 3. *If we additionally use per-filter normalization (i.e., $\|\boldsymbol{w}_{lk}\|_2 = 1/\sqrt{n_l}$), then Thm. 3 holds and $\boldsymbol{v}_l$ is more constrained: $[\boldsymbol{v}_l]_k = \pm 1/\sqrt{n_l}$ for $1 \le l \le L-1$.*

Remark. Here we prove that given fixed α, maximizing $\mathcal{E}_\alpha(\boldsymbol{\theta})$ gives rank-1 solutions for deep linear network. This conclusion is an extension of (Jing et al., 2022), which shows weight collapsing happens if $\boldsymbol{\theta}$ is 2-layer linear network and α is fixed. If the pairwise importance α is adversarial, then it may not lead to a rank-1 solution. In fact, α can magnify minimal eigen-directions and change the eigenstructure of X_α continuously. We leave it for future work.

Note that the condition that "$W_{>1}^\top W_{>1}$ has distinct maximal eigenvalue" is important. Otherwise there are counterexamples. For example, consider 1-layer linear network $\boldsymbol{z} = W_1 \boldsymbol{x}$, and X_α has duplicated maximal eigenvalues (with $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ being corresponding orthogonal eigenvectors), then $W_{>1}^\top W_{>1} = I$ (i.e., it has degenerated eigenvalues), and for any local maximal W_1, its row vector can be arbitrary linear combinations of $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ and thus W_1 is not rank-1.

Compared to recent works (Ji et al., 2021) that also relates CL with PCA in linear representation setting using constant α, our Theorem 3 has no statistical assumptions on data distribution and augmentation, and operates on vanilla InfoNCE loss and deep architectures.

5 How Representation Learning Differs in Two-layer ReLU Network

So far we have shown that the max player $\max_{\boldsymbol{\theta}} \mathcal{E}_\alpha(\boldsymbol{\theta}) := \mathrm{tr}(\mathbb{C}_\alpha[\boldsymbol{z}(\boldsymbol{\theta})])$ is essentially a PCA objective when the input-output mapping $\boldsymbol{z} = W(\boldsymbol{\theta})\boldsymbol{x}$ is linear. A natural question arises. What is the benefit of CL if its representation learning component has such a simple nature? Why can it learn a good representation in practice beyond PCA?

For this, nonlinearity is the key but understanding its role is highly nontrivial. For example, when the neural network model is nonlinear, Thm. 1 and Corollary 1 holds but *not* Corollary 2. Therefore, there is not even a well-defined X_α due to the fact that multiple hidden nodes can be switched on/off given different data input. Previous works (Safran & Shamir, 2018; Du et al., 2018a) also show that with nonlinearity, in supervised learning spurious local optima exist.

Here we take a first step to analyze nonlinear cases. We study 2-layer models with ReLU activation $h(x) = \max(x, 0)$. We show that with a proper data assumption, the 2-layer model shares a *modified* version of dynamics with its linear version, and the contrastive covariance term X_α (and its eigenstructure) remains well-defined and useful in nonlinear case.

5.1 The 2-layer ReLU network and data model

We consider the bottom-layer weight $W_1 = [\boldsymbol{w}_{11}, \boldsymbol{w}_{12}, \ldots, \boldsymbol{w}_{1K}]^\top$ with $\boldsymbol{w}_{1k}$ being the k-th filter. For brevity, let $K = n_1$ be the number of hidden nodes. We still consider solution in the constraint set Θ (Eqn. 11), since Lemma 2 still holds for ReLU networks. This model is named `ReLU2Layer`.

In addition, we assume the following data model:

Assumption 1 (Orthogonal mixture model within receptive field R_k)**.** *There exists a set of orthonormal bases $\{\bar{\boldsymbol{x}}_m\}_{m=1}^M$ so that any input data $\boldsymbol{x}[i] = \sum_m a_m[i]\bar{\boldsymbol{x}}_m$ satisfies the property that $a_m[i]$ is **Nonnegative**: $a_m[i] \geq 0$, **One-hot**: for any k, $a_m[i] > 0$ for at most one m and **Augmentation** only scales $\boldsymbol{x}_k$ by a (sample-dependent) factor, i.e., $\boldsymbol{x}[i'] = \gamma[i]\boldsymbol{x}[i]$ with $\gamma[i] > 0$.*

Since all $\boldsymbol{x}$ appears in the inner-product with the weight vectors $\boldsymbol{w}_{1k}$, with a rotation of coordination, we can just set $\bar{\boldsymbol{x}}_m = \boldsymbol{e}_m$, where $\boldsymbol{e}_m$ is the one-hot vector with m-th component being 1. In this case, $\boldsymbol{x} \geq 0$ is always a one-hot vector with only at most only one positive entry.

Intuitively, the model is motivated by sparsity: in each instantiation of $\boldsymbol{x}$, there are very small number of activated modes and their linear combination becomes the input signal $\boldsymbol{x}$. As we shall see, even with this simple model, the dynamics of ReLU network behaves very differently from the linear case.

With this assumption, we only need to consider nonnegative low-layer weights and X_α is still a valid quantity for `ReLU2Layer`:

Lemma 3 (Evaluation of `ReLU2Layer`)**.** *If Assumption 1 holds, setting $\boldsymbol{w}'_{1k} = \max(\boldsymbol{w}_{1k}, 0)$ won't change the output of `ReLU2Layer`. Furthermore, if $W_1 \geq 0$, then the formula for linear network $\mathcal{E}_\alpha = \mathrm{tr}(W_2 W_1 X_\alpha W_1^\top W_2^\top)$ still works for `ReLU2Layer`.*

On the other hand, sharing the energy function $\mathcal{E}_\alpha$ does not mean `ReLU2Layer` is completely identical to its linear version. In fact, the dynamics follows its linear counterparts, but with important modifications:

Theorem 4 (Dynamics of `ReLU2Layer`)**.** *If Assumption 1 holds, then the dynamics of `ReLU2Layer` with $\boldsymbol{w}_{1k} \geq 0$ is equivalent to linear dynamics with the **Sticky Weight rule**: any component that reaches 0 stays 0.*

As we will see, this modification leads to very different dynamics and local optima in `ReLU2Layer` from linear cases, even when there is only one ReLU node.

5.2 Dynamics in One ReLU node

Now we consider the dynamics of the simplest case: `ReLU2Layer` with only 1 hidden node. In this case, $W_{>1}^\top W_{>1}$ is a scalar and thus $W_2^\top W_2 = \mathrm{tr}(W_2^\top W_2) = 1$. We only need to consider $\boldsymbol{w}_1 \in \mathbb{R}^{n_1}$, which is the only weight vector in the lower layer, under the constraint $\|W_1\|_F = \|\boldsymbol{w}_1\|_2 = 1$ (Eqn. 11). We denote this setting as `ReLU2Layer1Hid`.

The dynamics now becomes very different from linear setting. Under linear network, according to Theorem 3, $\boldsymbol{w}_1$ converges to the largest eigenvector of $X_\alpha = \mathbb{C}_\alpha[\boldsymbol{x}_1]$. For `ReLU2Layer1Hid`, situation differs drastically:

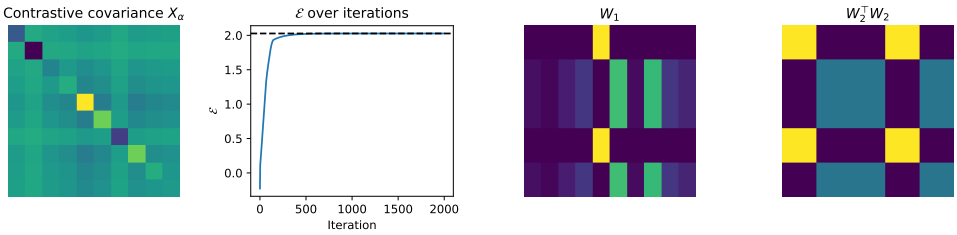


Figure 3: Theorem 6 shows that training `ReLU2Layer` could lead to more diverse hidden weight patterns beyond rank-1 solution obtained in the linear case (shown in right two figures: converged W_1 and $W_2^\top W_2$).

Theorem 5. *If Assumption 1 holds, then in* `ReLU2Layer1Hid`*,* $\boldsymbol{w}_1 \to \boldsymbol{e}_m$ *for certain* m*.*

Intuitively, this theorem is achieved by closely tracing the dynamics. When the number of positive entries of $\boldsymbol{w}_1$ is more than 1, the linear dynamics always hits the boundary of the polytope $\boldsymbol{w}_1 \geq 0$, making one of its entry be zero, and stick to zero due to sticky weight rule. This procedure repeats until there is only one survival positive entry in $\boldsymbol{w}_1$.

Overall, this simple case already shows that nonlinear landscape can lead to many local optima: for any m, $\boldsymbol{w}_1 = \boldsymbol{e}_m$ is one local optimal. Which one the training falls into depends on weight initialization, and critically affects the properties of per-trained models.

5.3 Multiple hidden nodes

For complicated situations like multiple hidden units, completely characterizing the training dynamics like Theorem 5 becomes hard (if not impossible). Instead, we focus on fixed point analysis.

For deep linear model, using multiple hidden units does not lead to any better solutions. According to Thm. 3, at local optimal, $W_1 = \boldsymbol{v}_1 \boldsymbol{v}_0^\top$. This means that the weights $\boldsymbol{w}_{1k}$, which are row vectors of W_1, are just a scaled version of the maximal eigenvector $\boldsymbol{v}_0$ of X_α. Moreover, this is independent of the eigenstructure of X_α as long as $\lambda_{\max}(X_\alpha) > 0$.

In `ReLU2Layer`, the situation is a bit different. Thm. 6 shows that these hidden nodes are (slightly) more diverse. Fig. 3 shows one such example. The intuition here is that in nonlinear case, rank-1 structure of the critical points may be replaced with low-rank structures.

Theorem 6 (`ReLU2Layer` encourages diversity)**.** *If Assumption 1 holds, then for any local optimal* $(W_2, W_1) \in \Theta$ *of* `ReLU2Layer` *with* $\mathcal{E} > 0$*, either* $W_1 = \boldsymbol{v}\boldsymbol{e}_m^\top$ *for some* m *and* $\boldsymbol{v} \geq 0$*, or* $\operatorname{rank}(W_1) > 1$*.*

6 Experiments

We evaluate our α-CL framework (Def. 1) in CIFAR10 (Krizhevsky et al., 2009) and STL-10 (Coates et al., 2011) with ResNet18 (He et al., 2016), and compare the downstream performance of multiple losses, with regularizers taking the form of $\mathcal{R}(\alpha) = \sum_i \sum_{j \neq i} r(\alpha_{ij})$ with a constraint $\sum_{j \neq i} \alpha_{ij} = 1$. Here r can be different concave functions:

- (α-CL-r_H) Entropy regularizer $r_H(\alpha_{ij}) = -2\tau\alpha_{ij} \log \alpha_{ij}$;
- (α-CL-r_γ) Inverse regularizers $r_\gamma(\alpha_{ij}) = \frac{2\tau}{1-\gamma}\alpha_{ij}^{1-\gamma}$ ($\gamma > 1$).
- (α-CL-r_s) Square regularizer $r_s(\alpha_{ij}) = -\frac{\tau}{2}\alpha_{ij}^2$.

Besides, we also compare with the following:

- Minimizing InfoNCE or quadratic loss: $\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$ for $\mathcal{L} \in \{\mathcal{L}_{nce}, \mathcal{L}_{quadratic}\}$.
- Setting α as InfoNCE (Eqn. 6) and backpropagates through $\alpha = \alpha(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$.
- (α-CL-direct) Directly setting α (here $p > 1$):

$$\alpha_{ij} = \frac{\exp(-d_{ij}^p/\tau)}{\sum_j \exp(-d_{ij}^p/\tau)} \tag{12}$$

	CIFAR-10			*STL-10*		
	100 epochs	300 epochs	500 epochs	100 epochs	300 epochs	500 epochs
$\mathcal{L}_{quadratic}$	63.59 ± 2.53	73.02 ± 0.80	73.58 ± 0.82	55.59 ± 4.00	64.97 ± 1.45	67.28 ± 1.21
$\mathcal{L}_{nce}$	84.06 ± 0.30	87.63 ± 0.13	87.86 ± 0.12	78.46 ± 0.24	82.49 ± 0.26	83.70 ± 0.12
backprop $\alpha(\boldsymbol{\theta})$	83.42 ± 0.25	87.18 ± 0.19	87.48 ± 0.21	77.88 ± 0.17	81.86 ± 0.30	83.19 ± 0.16
α-CL-r_H	84.27 ± 0.24	87.75 ± 0.25	87.92 ± 0.24	78.53 ± 0.35	82.62 ± 0.15	83.74 ± 0.18
α-CL-r_γ	83.72 ± 0.19	87.51 ± 0.11	87.69 ± 0.09	78.22 ± 0.28	82.19 ± 0.52	83.47 ± 0.34
α-CL-r_s	84.72 ± 0.10	86.62 ± 0.17	86.74 ± 0.15	76.95 ± 1.06	80.64 ± 0.77	81.65 ± 0.59
α-CL-direct	$\mathbf{85.11 \pm 0.19}$	$\mathbf{87.93 \pm 0.16}$	$\mathbf{88.09 \pm 0.13}$	$\mathbf{79.32 \pm 0.36}$	$\mathbf{82.95 \pm 0.17}$	$\mathbf{84.05 \pm 0.20}$

Table 1: Comparison over multiple loss formulations (ResNet18 backbone, batchsize 128). Top-1 accuracy with linear evaluation protocol. Temperature $\tau = 0.5$ and learning rate is 0.01. **Bold** is highest performance and blue is second highest. Each setting is repeated 5 times with different random seeds.

	ResNet18 Backbone			*ResNet50 Backbone*		
	CIFAR-100					
	100 epochs	300 epochs	500 epochs	100 epochs	300 epochs	500 epochs
$\mathcal{L}_{nce}$	55.70 ± 0.37	59.71 ± 0.36	59.89 ± 0.34	60.16 ± 0.48	65.40 ± 0.31	65.53 ± 0.30
α-CL-direct	$\mathbf{57.63 \pm 0.07}$	$\mathbf{60.12 \pm 0.26}$	$\mathbf{60.27 \pm 0.29}$	$\mathbf{62.93 \pm 0.28}$	$\mathbf{65.84 \pm 0.14}$	$\mathbf{65.87 \pm 0.21}$
	CIFAR-10					
$\mathcal{L}_{nce}$	84.06 ± 0.30	87.63 ± 0.13	87.86 ± 0.12	86.39 ± 0.16	89.97 ± 0.14	90.19 ± 0.23
α-CL-direct	$\mathbf{85.11 \pm 0.19}$	$\mathbf{87.93 \pm 0.16}$	$\mathbf{88.09 \pm 0.13}$	$\mathbf{87.79 \pm 0.25}$	$\mathbf{90.41 \pm 0.18}$	$\mathbf{90.50 \pm 0.21}$
	STL-10					
$\mathcal{L}_{nce}$	78.46 ± 0.24	82.49 ± 0.26	83.70 ± 0.12	81.64 ± 0.24	86.57 ± 0.17	$\mathbf{87.90 \pm 0.22}$
α-CL-direct	$\mathbf{79.32 \pm 0.36}$	$\mathbf{82.95 \pm 0.17}$	$\mathbf{84.05 \pm 0.20}$	$\mathbf{83.20 \pm 0.25}$	$\mathbf{87.17 \pm 0.14}$	87.85 ± 0.21

Table 2: More experiments with ResNet18/ResNet50 backbone on CIFAR-10, STL-10 and CIFAR-100. Batchsize is 128. For ResNet18, learning rate is 0.01; for ResNet50, learning rate is 0.001.

For inverse regularizer r_γ, we pick $\gamma = 2$ and $\tau = 0.5$; for direct-set α, we pick $p = 4$ and $\tau = 0.5$; for square regularizer, we use $\tau = 5$. All training is performed with Adam (Kingma & Ba, 2014) optimizer. Code is written in PyTorch and a single modern GPU suffices for the experiments.

The results are shown in Tbl. 1. We can see that (1) backpropagating through $\alpha(\boldsymbol{\theta})$ is worse, justifying our perspective of coordinate-wise optimization, (2) our proposed α-CL works for different regularizers, (3) using different regularizer leads to comparable or better performance than original InfoNCE $\mathcal{L}_{nce}$, (4) the pairwise importance α does not even need to come from a minimization process. Instead, we can directly set α based on pairwise squared distances d_{ij}^2 and d_i^2. For α-CL-direct, the performance is slightly worse if we do not normalize α_{ij} (i.e., $\alpha_{ij} := \exp(-d_{ij}^p/\tau)$). It seems that for strong performance, $\frac{\mathrm{d}r}{\mathrm{d}\alpha_{ij}}$ should go to $+\infty$ when $\alpha_{ij} \to 0$. Regularizers that do not satisfy this condition (e.g., squared regularizer r_s) may not work as well.

Tbl. 2 shows more experiments with different backbones (e.g., ResNet50) and more complicated datasets (e.g., CIFAR-100). Overall, we see consistent gains of α-CL over InfoNCE in early stages of the training (e.g., 1-2 point of absolute percentage gain) and comparable performance at 500 epoch. More ablations on batchsizes and exponent p in Eqn. 12 are provided in Appendix B.

7 Conclusion and Future Work

We provide a novel perspective of contrastive learning (CL) via the lens of coordinate-wise optimization and propose a unified framework called α-CL that not only covers a broad family of loss functions including InfoNCE, but also allows a direct set of importance of sample pairs. Preliminary experiments on CIFAR10/STL-10/CIFAR100 show comparable/better performance with the new loss than InfoNCE. Furthermore, we prove that with deep linear networks, the representation learning part is equivalent to Principal Component Analysis (PCA). In addition, we also extend our analysis to representation learning in 2-layer ReLU network, shedding light on the important difference in representation learning for linear/nonlinear cases.

Future work. Our framework α-CL turns various loss functions into a unified framework with different choices of pairwise importance α and how to find good choices remains open. Also, we mainly focus on representation learning with fixed pairwise importance α. However, in the actual training, α and $\boldsymbol{\theta}$ change concurrently. Understanding their interactions is an important next step. Finally, removing Assumption 1 in ReLU analysis is also an open problem to be addressed later.

References

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*, 2018.

Arora, S., Cohen, N., and Hazan, E. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International Conference on Machine Learning*, pp. 244–253. PMLR, 2018.

Arora, S., Khandeparkar, H., Khodak, M., Plevrakis, O., and Saunshi, N. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Baldi, P. and Hornik, K. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58, 1989.

Belghazi, M. I., Baratin, A., Rajeshwar, S., Ozair, S., Bengio, Y., Courville, A., and Hjelm, D. Mutual information neural estimation. In *International Conference on Machine Learning*, pp. 531–540. PMLR, 2018.

Caron, M., Bojanowski, P., Joulin, A., and Douze, M. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Chen, X. and He, K. Exploring simple siamese representation learning. In *CVPR*, 2020.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *International conference on artificial intelligence and statistics*, 2011.

Coria, J. M., Bredin, H., Ghannay, S., and Rosset, S. A comparison of metric learning loss functions for end-to-end speaker verification. In *International Conference on Statistical Language and Speech Processing*, pp. 137–148. Springer, 2020.

Du, S., Lee, J., Tian, Y., Singh, A., and Poczos, B. Gradient descent learns one-hidden-layer cnn: Don't be afraid of spurious local minima. In *International Conference on Machine Learning*, pp. 1339–1348. PMLR, 2018a.

Du, S. S., Hu, W., and Lee, J. D. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. *arXiv preprint arXiv:1806.00900*, 2018b.

Fisher, R. A. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2): 179–188, 1936.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., et al. Bootstrap your own latent: A new approach to self-supervised learning. *NeurIPS*, 2020.

Hadsell, R., Chopra, S., and LeCun, Y. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pp. 1735–1742. IEEE, 2006.

HaoChen, J. Z., Wei, C., Gaidon, A., and Ma, T. Provable guarantees for self-supervised deep learning with spectral contrastive loss. *NeurIPS*, 2021.

Hardt, M. and Ma, T. Identity matters in deep learning. *ICLR*, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. B. Momentum contrast for unsupervised visual representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9726–9735, 2020.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

Ji, W., Deng, Z., Nakada, R., Zou, J., and Zhang, L. The power of contrast for feature learning: A theoretical analysis. *arXiv preprint arXiv:2110.02473*, 2021.

Jing, L., Vincent, P., LeCun, Y., and Tian, Y. Understanding dimensional collapse in contrastive self-supervised learning. *ICLR*, 2022.

Kalantidis, Y., Sariyildiz, M. B., Pion, N., Weinzaepfel, P., and Larlus, D. Hard negative mixing for contrastive learning. *NeurIPS*, 2020.

Kawaguchi, K. Deep learning without poor local minima. *NeurIPS*, 2016.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. Supervised contrastive learning. *NeurIPS*, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kokiopoulou, E., Chen, J., and Saad, Y. Trace optimization and eigenproblems in dimension reduction methods. *Numerical Linear Algebra with Applications*, 18(3):565–602, 2011.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Laurent, T. and Brecht, J. Deep linear networks with arbitrary loss: All local minima are global. In *International conference on machine learning*, pp. 2902–2907. PMLR, 2018.

Lee, J. D., Lei, Q., Saunshi, N., and Zhuo, J. Predicting what you already know helps: Provable self-supervised learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Misra, I. and Maaten, L. v. d. Self-supervised learning of pretext-invariant representations. In *CVPR*, 2020.

Oh Song, H., Xiang, Y., Jegelka, S., and Savarese, S. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4004–4012, 2016.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Robinson, J., Chuang, C.-Y., Sra, S., and Jegelka, S. Contrastive learning with hard negative samples. *ICLR*, 2021.

Safran, I. and Shamir, O. Spurious local minima are common in two-layer relu neural networks. In *International Conference on Machine Learning*, pp. 4433–4441. PMLR, 2018.

Saxe, A. M., McClelland, J. L., and Ganguli, S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *ICLR*, 2014.

Schroff, F., Kalenichenko, D., and Philbin, J. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.

Sohn, K. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pp. 1857–1865, 2016.

Tian, Y. A theoretical framework for deep locally connected relu network. *arXiv preprint arXiv:1809.10829*, 2018.

Tian, Y. Student specialization in deep relu networks with finite width and input dimension. *ICML*, 2020.

Tian, Y., Krishnan, D., and Isola, P. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pp. 776–794. Springer, 2020a.

Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., and Isola, P. What makes for good views for contrastive learning? *NeurIPS*, 2020b.

Tian, Y., Yu, L., Chen, X., and Ganguli, S. Understanding self-supervised learning with dual deep networks. *arXiv preprint arXiv:2010.00578*, 2020c.

Wen, Z. and Li, Y. Toward understanding the feature learning process of self-supervised contrastive learning. *arXiv preprint arXiv:2105.15134*, 2021.

Wold, S., Esbensen, K., and Geladi, P. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

Wu, Z., Xiong, Y., Yu, S. X., and Lin, D. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3733–3742, 2018.

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., and Liu, T. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pp. 10524–10533. PMLR, 2020.

Yeh, C.-H., Hong, C.-Y., Hsu, Y.-C., Liu, T.-L., Chen, Y., and LeCun, Y. Decoupled contrastive learning. *arXiv preprint arXiv:2110.06848*, 2021.

Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arxiv:2103.03230*, 2021.

Zhou, Y. and Liang, Y. Critical points of linear neural networks: Analytical forms and landscape properties. In *International Conference on Learning Representations*, 2018.

A Proofs

A.1 Section 3

Theorem 1. *For any differential mapping $\boldsymbol{z} = \boldsymbol{z}(\boldsymbol{x}; \boldsymbol{\theta})$, gradient descent of $\mathcal{L}_{\phi,\psi}$ is equivalent to* ***gradient ascent*** *of the objective $\mathcal{E}_\alpha(\boldsymbol{\theta}) := \mathrm{tr}(\mathbb{C}_\alpha[\boldsymbol{z}(\boldsymbol{\theta}), \boldsymbol{z}(\boldsymbol{\theta})])$:*

$$\frac{\partial \mathcal{L}_{\phi,\psi}}{\partial \boldsymbol{\theta}} = -\frac{1}{2}\frac{\partial \mathcal{E}_\alpha}{\partial \boldsymbol{\theta}}\Big|_{\alpha=\alpha(\boldsymbol{\theta})} \tag{3}$$

Here the pairwise importance $\alpha = \alpha(\boldsymbol{\theta}) := \{\alpha_{ij}(\boldsymbol{\theta})\}$ *is a function of input batch $\boldsymbol{x}$, defined as:*

$$\alpha_{ij}(\boldsymbol{\theta}) := \phi'(\xi_i)\psi'(d_i^2 - d_{ij}^2) \geq 0 \tag{4}$$

where $\phi', \psi' \geq 0$ are derivatives of ϕ, ψ. The contrastive covariance $\mathbb{C}_\alpha[\cdot,\cdot]$ *is defined as:*

$$\mathbb{C}_\alpha[\boldsymbol{a}, \boldsymbol{b}] := \sum_{i=1}^N \sum_{j\neq i} \alpha_{ij}(\boldsymbol{a}[i]-\boldsymbol{a}[j])(\boldsymbol{b}[i]-\boldsymbol{b}[j])^\top - \sum_{i=1}^N \left(\sum_{j\neq i} \alpha_{ij}\right)(\boldsymbol{a}[i]-\boldsymbol{a}[i'])(\boldsymbol{b}[i]-\boldsymbol{b}[i'])^\top \tag{5}$$

That is, ***minimizing*** *the loss function $\mathcal{L}_{\phi,\psi}(\boldsymbol{\theta})$ can be regarded as* ***maximizing*** *the energy function $\mathcal{E}_{\alpha=\mathrm{sg}(\alpha(\boldsymbol{\theta}))}(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. Here $\mathrm{sg}(\cdot)$ means stop-gradient, i.e., the gradient of $\boldsymbol{\theta}$ is not backpropagated into $\alpha(\boldsymbol{\theta})$.*

Proof. By the definition of gradient descent, we have for any component θ in a high-dimensional vector $\boldsymbol{\theta}$:

$$-\frac{\partial \mathcal{L}}{\partial \theta} = -\sum_{i=1}^N \frac{\partial \boldsymbol{z}[i]}{\partial \theta}\frac{\partial \mathcal{L}}{\partial \boldsymbol{z}[i]} + \frac{\partial \boldsymbol{z}[i']}{\partial \theta}\frac{\partial \mathcal{L}}{\partial \boldsymbol{z}[i']} \tag{13}$$

Here we use the "Denominator-layout notation" and treat $\frac{\partial \mathcal{L}}{\partial \boldsymbol{z}[i]}$ as a column vector while $\frac{\partial \boldsymbol{z}[i]}{\partial \theta}$ as a row vector. Using Lemma 4, we have:

$$-\frac{\partial \mathcal{L}}{\partial \theta} = \mathbb{C}_\alpha\left[\frac{\partial \boldsymbol{z}}{\partial \theta}, \boldsymbol{z}^\top\right] \tag{14}$$

On the other hand, treating α as independent variables of $\boldsymbol{\theta}$, we compute (here o_k is the k-th component of $\boldsymbol{z}$):

$$\frac{\partial \mathcal{E}_\alpha}{\partial \theta} = \sum_k \mathbb{C}_\alpha\left[\frac{\partial o_k}{\partial \theta}, o_k\right] + \mathbb{C}_\alpha\left[o_k, \frac{\partial o_k}{\partial \theta}\right] \tag{15}$$

For scalar x and y, $\mathbb{C}_\alpha[x, y] = \mathbb{C}_\alpha[y, x]$ and $\sum_k \mathbb{C}_\alpha[a_k, b_k] = \mathbb{C}_\alpha[\boldsymbol{a}, \boldsymbol{b}^\top]$ for row vector $\boldsymbol{a}$ and column vector $\boldsymbol{b}$. Therefore,

$$\frac{\partial \mathcal{E}_\alpha}{\partial \boldsymbol{\theta}} = 2\mathbb{C}_\alpha\left[\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}, \boldsymbol{z}^\top\right] \tag{16}$$

Therefore, we have

$$\frac{\partial \mathcal{E}_\alpha}{\partial \boldsymbol{\theta}} = -2\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}} \tag{17}$$

and the proof is complete. □

Theorem 2. *If $\psi(x) = e^{x/\tau}$, then the corresponding pairwise importance α (Eqn. 4) is the solution to the minimization problem:*

$$\alpha(\boldsymbol{\theta}) = \arg\min_{\alpha\in\mathcal{A}} \mathcal{E}_\alpha(\boldsymbol{\theta}) - \mathcal{R}(\alpha), \qquad \mathcal{A} := \left\{\alpha : \quad \forall i,\ \sum_{j\neq i} \alpha_{ij} = \tau^{-1}\xi_i\phi'(\xi_i),\ \ \alpha_{ij} \geq 0\right\} \tag{7}$$

Here the regularization $\mathcal{R}(\alpha) = \mathcal{R}_H(\alpha) := 2\tau\sum_{i=1}^N H(\alpha_{i\cdot}) = -2\tau\sum_{i=1}^N \sum_{j\neq i} \alpha_{ij}\log\alpha_{ij}$.

Proof. We just need to solve the internal minimizer w.r.t. α. Note that each α_i can be optimized independently.

First, we know that $\mathcal{E}_\alpha(\boldsymbol{\theta}) := \mathrm{tr}\mathbb{C}_\alpha[\boldsymbol{z}, \boldsymbol{z}]$ can be written as:

$$\begin{aligned}
\mathcal{E}_\alpha(\boldsymbol{\theta}) &= \sum_{i\neq j} \alpha_{ij} \left[\mathrm{tr}(\boldsymbol{z}[i]-\boldsymbol{z}[j])(\boldsymbol{z}[i]-\boldsymbol{z}[j])^\top - \mathrm{tr}(\boldsymbol{z}[i]-\boldsymbol{z}[i'])(\boldsymbol{z}[i]-\boldsymbol{z}[i'])^\top\right] && (18)\\
&= \sum_{i\neq j} \alpha_{ij} \left[\|\boldsymbol{z}[i]-\boldsymbol{z}[j]\|_2^2 - \|\boldsymbol{z}[i]-\boldsymbol{z}[i']\|_2^2\right] && (19)\\
&= 2\sum_{i\neq j} \alpha_{ij} \left(d_{ij}^2 - d_i^2\right) && (20)
\end{aligned}$$

Applying Lemma 5 with $c_{ij} = 2(d_{ij}^2 - d_i^2)$, for each i, the optimal solution α is:

$$\begin{aligned}
\alpha_{ij} &= \frac{1}{\tau}\exp\left(-\frac{c_{ij}}{2\tau}\right)\phi'\left(\sum_{j\neq i}\exp\left(-\frac{c_{ij}}{2\tau}\right)\right) && (21)\\
&= \frac{1}{\tau}\exp\left(\frac{d_i^2-d_{ij}^2}{\tau}\right)\phi'\left(\sum_{j\neq i}\exp\left(\frac{d_i^2-d_{ij}^2}{\tau}\right)\right) && (22)\\
&= \psi'(d_i^2-d_{ij}^2)\phi'\left(\sum_{j\neq i}\psi(d_i^2-d_{ij}^2)\right) && (23)\\
&= \psi'(d_i^2-d_{ij}^2)\phi'(\xi_i) && (24)
\end{aligned}$$

which coincides with Eqn. 4 that is from the gradient descent rule of the loss function $\mathcal{L}_{\phi,\psi}$.

In particular, for InfoNCE, we have $\phi(x) = \tau\log(\epsilon + x)$, $\phi'(x) = \tau/(x+\epsilon)$ and therefore:

$$\alpha_{ij} = \frac{\exp((d_i^2-d_{ij}^2)/\tau)}{\epsilon + \sum_{j\neq i}\exp((d_i^2-d_{ij}^2)/\tau)} = \frac{\exp(-d_{ij}^2/\tau)}{\epsilon\exp(-d_i^2/\tau) + \sum_{j\neq i}\exp(-d_{ij}^2/\tau)} \tag{25}$$

which is exactly the coefficients α_{ij} directly computed during minimization of $\mathcal{L}_{nce}$. If $\epsilon = 0$, then the constraint becomes $\sum_{j\neq i}\alpha_{ij} = 1$ and we have:

$$\alpha_{ij} = \frac{\exp(-d_{ij}^2/\tau)}{\sum_{j\neq i}\exp(-d_{ij}^2/\tau)} \tag{26}$$

That is, the coefficients α does not depend on intra-augmentation squared distance d_i^2. □

Corollary 1 (Contrastive Learning as Coordinate-wise Optimization)**.** *If $\psi(x) = e^{x/\tau}$, minimizing $\mathcal{L}_{\phi,\psi}$ is equivalent to the following iterative procedure:*

$$\textit{(Min-player } \alpha\textit{)} \qquad \alpha_t = \arg\min_{\alpha\in\mathcal{A}} \mathcal{E}_\alpha(\boldsymbol{\theta}_t) - \mathcal{R}(\alpha) \tag{8a}$$

$$\textit{(Max-player } \boldsymbol{\theta}\textit{)} \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta\nabla_{\boldsymbol{\theta}}\mathcal{E}_{\alpha_t}(\boldsymbol{\theta}) \tag{8b}$$

Proof. The proof naturally follows from the conclusion of Theorem 1 and Theorem 2. □

A.2 Section 4

Corollary 2 (Representation learning in Deep Linear CL reparameterizes Principal Component Analysis (PCA))**.** *When $\boldsymbol{z} = W(\boldsymbol{\theta})\boldsymbol{x}$ with a constraint $WW^\top = I$, $\mathcal{E}_\alpha$ is the objective of Principal Component Analysis (PCA) with reparameterization $W = W(\boldsymbol{\theta})$:*

$$\max_{\boldsymbol{\theta}} \mathcal{E}_\alpha(\boldsymbol{\theta}) = \mathrm{tr}(W(\boldsymbol{\theta})X_\alpha W^\top(\boldsymbol{\theta})) \quad \text{s.t. } WW^\top = I \tag{9}$$

here $X_\alpha := \mathbb{C}_\alpha[\boldsymbol{x}]$ is the contrastive covariance of input $\boldsymbol{x}$.

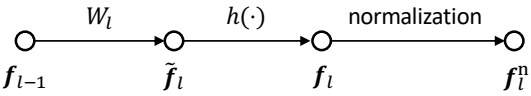


Figure 4: Notations on normalization (Sec. A.2.1).

Proof. Notice that in deep linear setting, $\boldsymbol{z} = W(\boldsymbol{\theta})\boldsymbol{x}$ where $W(\boldsymbol{\theta})$ does not dependent on specific samples. Therefore, $\mathbb{C}_\alpha[\boldsymbol{z}, \boldsymbol{z}] = W(\boldsymbol{\theta})\mathbb{C}_\alpha[\boldsymbol{x}, \boldsymbol{x}]W^\top(\boldsymbol{\theta}) = W(\boldsymbol{\theta})X_\alpha W^\top(\boldsymbol{\theta})$. □

Lemma 1. *The training dynamics in* `DeepLin` *is* $\dot{W}_l = W_{>l}^\top W_{>l} W_l \mathbb{C}_\alpha[\boldsymbol{f}_{l-1}]$

Proof. We can start from Eqn. 13 directly and takes out $J_{>l}^\top$. This leads to

$$\dot{W}_l = J_{>l}^\top \left(\sum_{i=1}^N \frac{\partial \mathcal{L}}{\partial \boldsymbol{z}[i]} \boldsymbol{f}_{l-1}^\top[i] + \frac{\partial \mathcal{L}}{\partial \boldsymbol{z}[i']} \boldsymbol{f}_{l-1}^\top[i'] \right) = J_{>l}^\top \mathbb{C}_\alpha[\boldsymbol{z}, \boldsymbol{f}_{l-1}] \tag{27}$$

Using that $\boldsymbol{z} = J_{\geq l}\boldsymbol{f}_{l-1}$ leads to the conclusion. If the network is linear, then $J_{>l}^\top[i] = J_{>l}^\top$ is a constant. Then we can take the common factor $J_{>l}^\top J_{\geq l}$ out of the summation, yield $\dot{W}_l = J_{>l}^\top J_{\geq l} F_{l-1}$. Here $F_l := \mathbb{C}_\alpha[\boldsymbol{f}_l]$ is the contrastive covariance at layer l. □

A.2.1 Section 4.2

For this we talk about more general cases where the deep network is nonlinear. Let $h(\cdot)$ be the point-wise activation function and the network architecture looks like the following:

$$\boldsymbol{z}[i] := W_L h(W_{L-1}(h(\dots W_1 \boldsymbol{x}[i]))) \tag{28}$$

We consider the case where $h(\cdot)$ satisfies the following constraints:

Definition 2 (Reversibility (Tian et al., 2020c) / Homogeneity (Du et al., 2018b))**.** *The activation function $h(x)$ satisfies $h(x) = h'(x)x$.*

This is satisfied by linear, ReLU, leaky ReLU and many polynomial activations (with an additional constant). With this condition, we have $\boldsymbol{f}_l[i] = D_l W_l \boldsymbol{f}_{l-1}[i]$, where $D_l = D_l(\boldsymbol{x}[i]) := \mathrm{diag}[h'(\boldsymbol{w}_{lk}^\top \boldsymbol{f}_{l-1}[i])] \in \mathbb{R}^{n_l \times n_l}$ is a diagonal matrix. For ReLU activation, the diagonal entry of D_l is binary.

Definition 3 (Reversible Layers (Tian et al., 2020c))**.** *A layer is reversible if there exists $J[i]$ so that $\boldsymbol{f}_{\mathrm{out}}[i] = J[i]\boldsymbol{f}_{\mathrm{in}}[i]$ and $\boldsymbol{g}_{\mathrm{in}}[i] = J^\top[i]\boldsymbol{g}_{\mathrm{out}}[i]$ for each sample i.*

It is clear that linear layers, ReLU and leaky ReLU are reversible. Lemma 6 tells us that ℓ_2-normalization and LayerNorm are also reversible.

Lemma 2. *For MLP, if the weight W_l is below a ℓ_2-norm or LayerNorm layer, then $\frac{\mathrm{d}}{\mathrm{d}t}\|W_l\|_F^2 = 0$.*

Proof. See Lemma 7 that proves more general cases. □

A.2.2 Section 4.3

Definition 4 (Aligned-rank-1 solution)**.** *A solution $\boldsymbol{\theta} = \{W_l\}_{l=1}^L$ is called aligned-rank-1, if there exists a set of unit vectors $\{\boldsymbol{v}_l\}_{l=0}^L$ so that $W_l = \boldsymbol{v}_l \boldsymbol{v}_{l-1}^\top$ for $1 \leq l \leq L$.*

Theorem 3 (Representation Learning with `DeepLin` is PCA)**.** *If $\lambda_{\max}(X_\alpha) > 0$, then for any local maximum $\boldsymbol{\theta} \in \Theta$ of Eqn. 11 whose $W_{>1}^\top W_{>1}$ has distinct maximal eigenvalue:*

- *there exists a set of unit vectors $\{\boldsymbol{v}_l\}_{l=0}^L$ so that $W_l = \boldsymbol{v}_l \boldsymbol{v}_{l-1}^\top$ for $1 \leq l \leq L$, in particular, $\boldsymbol{v}_0$ is the unit eigenvector corresponding to $\lambda_{\max}(X_\alpha)$,*
- *$\boldsymbol{\theta}$ is global optimal with objective $\mathcal{E}^* = \lambda_{\max}(X_\alpha)$.*

Proof. A necessary condition for $\boldsymbol{\theta}$ to be the local maximum is the critical point condition (here λ_{l-1} is some constant):

$$W_{>l}^\top W_{>l} W_l F_{l-1} = \lambda_{l-1} W_l \tag{29}$$

Right multiplying W_l on both sides of the critical point condition for W_l, and taking matrix trace, we have:

$$\mathcal{E}(\boldsymbol{\theta}) = \mathrm{tr}(W_{>l}^\top W_{>l} W_l F_{l-1} W_l^\top) = \mathrm{tr}(\lambda_{l-1} W_l W_l^\top) = \lambda_{l-1} \tag{30}$$

Therefore, all λ_l are the same, denoted as λ, and they are equal to the objective value.

Now let's consider $l = 1$. Then we have:

$$W_{>1}^\top W_{>1} W_1 X = \lambda W_1 \tag{31}$$

Applying $\mathrm{vec}(AXB) = (B^\top \otimes A)\mathrm{vec}(X)$, we have:

$$(X \otimes W_{>1}^\top W_{>1})\mathrm{vec}(W_1) = \lambda \mathrm{vec}(W_1) \tag{32}$$

with the constraint that $\|\mathrm{vec}(W_1)\|_2^2 = \|W_1\|_F^2 = 1$.

We then prove that λ is the largest eigenvalue of $X \otimes W_{>1}^\top W_{>1}$. We prove by contradiction. If not, then $\mathrm{vec}(W_1)$ is not the largest eigenvector, then there is always a direction W_1 can move, while respecting the constraint $\|W_1\|_F = 1$ and keeping W_{-1} fixed, to make $\mathcal{E}(\boldsymbol{\theta})$ strictly larger. Therefore, for any local maximum $\boldsymbol{\theta}$, λ has to be the largest eigenvalue of $X \otimes W_{>1}^\top W_{>1}$.

Let $\{\boldsymbol{v}_{0m}\}$ be the orthonormal basis of the eigenspace of $\lambda_{\max}(X)$ and $\boldsymbol{u}$ be the (unique!) unit eigenvector of $W_{>1}^\top W_{>1}$. Then $\mathrm{vec}(W_1) = \sum_m c_m \boldsymbol{v}_{0m} \otimes \boldsymbol{u}$ where $\sum_m c_m^2 = 1$, or $\mathrm{vec}(W_1) = \boldsymbol{v}_0 \otimes \boldsymbol{u}$ where the unit vector $\boldsymbol{v}_0 := \sum_m c_m \boldsymbol{v}_{0m}$, and $\lambda = \lambda_{\max}(X)\|W_{>1}\boldsymbol{u}\|_2^2$.

Now we show that $\lambda_{\max}(W_{>1}^\top W_{>1}) = \|W_{>1}\|_2 = 1$. If not, then by Statement 3-4 of Lemma 9, $W_{L:2}$ is not a local maximum and there exists $W'_{L:2}$ in its neighborhood so that (1) $W'_{L:2}$ satisfy the F-norm constraints and (2) $\|W'_{>1}\|_2 > \|W_{>1}\|_2$, or more specifically, $\|W'_{>1}\boldsymbol{u}\|_2 > \|W_{>1}\boldsymbol{u}\|_2$. Let $\boldsymbol{\theta}' := \{W'_{L:2}, W_1\}$, we have:

$$\begin{aligned}
\mathcal{J}(\boldsymbol{\theta}') &= \mathrm{vec}^\top(W_1)(X \otimes W'^\top_{>1} W'_{>1})\mathrm{vec}(W_1) && (33)\\
&= (\boldsymbol{v}_0^\top \otimes \boldsymbol{u}^\top)(X \otimes W'^\top_{>1} W'_{>1})(\boldsymbol{v}_0 \otimes \boldsymbol{u}) && (34)\\
&= \lambda_{\max}(X)\|W'_{>1}\boldsymbol{u}\|_2^2 && (35)\\
&> \lambda_{\max}(X)\|W_{>1}\boldsymbol{u}\|_2^2 && (36)\\
&= \lambda = \mathcal{J}(\boldsymbol{\theta}) && (37)
\end{aligned}$$

This means that $\boldsymbol{\theta}$ is not a local maximum. Note that $\boldsymbol{\theta}'$ is not necessarily a critical point (and Eqn. 29 may not hold for $\boldsymbol{\theta}'$). Therefore, $\lambda_{\max}(W_{>1}^\top W_{>1}) = \|W_{>1}\|_2 = 1$ and thus $\mathcal{E}(\boldsymbol{\theta}) = \lambda = \lambda_{\max}(X)$.

By Statement 1 of Lemma 9, $W_{L:2}$ is aligned-rank-1 and $W_{>1} = \boldsymbol{v}_L \boldsymbol{v}_1^\top$ is also a rank-1 matrix. $W_{>1}^\top W_{>1} = \boldsymbol{v}_1 \boldsymbol{v}_1^\top$ has a unique maximal eigenvector $\boldsymbol{v}_1$. Therefore $\mathrm{vec}(W_1) = \boldsymbol{v}_0 \otimes \boldsymbol{v}_1$, or $W_1 = \boldsymbol{v}_1 \boldsymbol{v}_0^\top$. As a result, $\boldsymbol{\theta} := \{W_{L:2}, W_1\}$ is aligned-rank-1.

Finally, since all local maxima have the same objective functions, they are all global maxima. □

Remarks. Leveraging similar proof techniques, we can also show that with BatchNorm layers, the local maxima are more constrained. From Lemma 10 we knows that if each hidden node is covered with BatchNorm, then its fan-in weights are conserved. Therefore, without loss of generality, we could set the per-filter normalization: $\|\boldsymbol{w}_{lk}\|_2 = 1$. In this case we have:

Definition 5 (Aligned-uniform solution)**.** *A solution $\boldsymbol{\theta}$ is called aligned-uniform, if it is aligned-rank-1, and $[\boldsymbol{v}_l]_k = \pm 1/\sqrt{n_l}$ for $1 \le l \le L-1$. The two end-point unit vectors ($\boldsymbol{v}_0$ and $\boldsymbol{v}_L$) can still be arbitrary.*

Corollary 3. *If we additionally use per-filter normalization (i.e., $\|\boldsymbol{w}_{lk}\|_2 = 1/\sqrt{n_l}$), then Thm. 3 holds and $\boldsymbol{v}_l$ is more constrained: $[\boldsymbol{v}_l]_k = \pm 1/\sqrt{n_l}$ for $1 \le l \le L-1$.*

Proof. Leveraging Lemma 11 in Theorem 3 yields the conclusion. □

Remark. We could see that with BatchNorm, the optimization problem is more constrained, and the set of local maxima have less degree of freedom. This makes optimization better behaved.

A.3 Section 5

Lemma 3 (Evaluation of `ReLU2Layer`)**.** *If Assumption 1 holds, setting $\boldsymbol{w}'_{1k} = \max(\boldsymbol{w}_{1k}, 0)$ won't change the output of `ReLU2Layer`. Furthermore, if $W_1 \geq 0$, then the formula for linear network $\mathcal{E}_\alpha = \mathrm{tr}(W_2 W_1 X_\alpha W_1^\top W_2^\top)$ still works for `ReLU2Layer`.*

Proof. For the first part, we just want to prove that if Assumption 1 holds, then a 2-layer ReLU network with weights $\boldsymbol{w}_{1k}$ and W_2 has the same activation as another ReLU network with $\boldsymbol{w}'_{1k} = \max(\boldsymbol{w}_{1k}, 0) \geq 0$ and $W'_2 = W_2$.

We are comparing the two activations:

$$f_{1k} = \max\left(\sum_m w_{1km} x_{km}, 0\right) \tag{38}$$

$$f'_{1k} = \max\left(\sum_m \max(w_{1km}, 0) x_{km}, 0\right) = \sum_m \max(w_{1km}, 0) x_{km} \tag{39}$$

The equality is due to the fact that $\boldsymbol{x}_k \geq 0$ (by nonnegativeness). Now we consider two cases.

Case 1. If all $\boldsymbol{w}_{1k} \geq 0$ then obviously they are identical.

Case 2. If there exists m so that $w_{1km} < 0$. The only situation that the difference could happen is for some specific $\boldsymbol{x}_k[i]$ so that $x_{km}[i] > 0$. By Assumption 1(one-hotness), for $m' \neq m$, $x_{km}[i] = 0$ so the gate $d_k[i] = \mathbb{I}(\boldsymbol{w}_{1k}^\top \boldsymbol{x}_k > 0) = 0$. On the other hand, $\boldsymbol{w'}_{1k}^\top \boldsymbol{x}_k = 0$ so $d'_k[i] = 0$.

Therefore, in all situations, $f_{1k} = f'_{1k}$.

For the second part, since $W_1 \geq 0$ and all input $\boldsymbol{x} \geq 0$ by non-negativeness, all gates are open and the energy $\mathcal{E}_\alpha$ of `ReLU2Layer` is the same as the linear model. □

Theorem 4 (Dynamics of `ReLU2Layer`)**.** *If Assumption 1 holds, then the dynamics of `ReLU2Layer` with $\boldsymbol{w}_{1k} \geq 0$ is equivalent to linear dynamics with the **Sticky Weight rule**: any component that reaches 0 stays 0.*

Proof. Let $\boldsymbol{w}_{1k} \geq 0$ be the k-th filter to be considered and $w_{1km} \geq 0$ its m-th component. Consider a linear network with the same weights ($\boldsymbol{w}'_{1k} = \boldsymbol{w}_{1k}$ and $W'_2 = W_2$) with only the ReLU activation removed.

Now we consider the gradient rule of the ReLU network and the corresponding linear network with a sticky weight rule (here $g_k[i]$ is the backpropagated gradient sent to node k for sample i, and $d_k[i]$ is the binary gating for sample i at node k):

$$\dot{w}_{1km} = \sum_i g_k[i] d_k[i] x_m[i] \tag{40}$$

$$\dot{w}'_{1km} = \mathbb{I}(w_{1km} > 0) \sum_i g'_k[i] x_m[i] \tag{41}$$

Thanks to Lemma 12, we know the forward pass between two networks are identical and thus $g_k[i] = g'_k[i]$ so we don't need to consider the difference between backpropagated gradient.

In the following, we will show that each summand of the two equations is identical.

Case 1. $x_m[i] = 0$. In that case, $g_k[i] x_m[i] = g_k[i] d_k[i] x_m[i] = 0$ regardless of whether the gate $d_k[i]$ is open or closed.

Case 2. $x_m[i] > 0$. There are two subcases:

Subcase 1: $d_k[i] = 1$. In this case, the ReLU gating of k-th filter is open, then $g'_k[i] x_m[i] = g_k[i] x_m[i] = g_k[i] d_k[i] x_m[i]$. By Assumption 1(One-hotness), for other $m' \neq m$, $x_{km'}[i] = 0$, since $d_k[i] = 1$, it must be the case that $w_{1km} > 0$ and thus $\mathbb{I}(w_{1km} > 0) = 1$. So the two summands are identical.

Subcase 2: $d_k[i] = 0$. Then w_{1km} must be 0, otherwise since $\boldsymbol{x}_k \geq 0$ (nonnegativeness), we have $\boldsymbol{w}_{1k}^\top \boldsymbol{x}_k[i] \geq w_{1km} x_m[i] > 0$ and the gating of k-th filter must open. Therefore, the two summands are both 0: the ReLU one is because $d_k[i] = 0$ and the linear one is due to $\mathbb{I}(w_{1km} > 0) = 0$. □

Theorem 5. *If Assumption 1 holds, then in* `ReLU2Layer1Hid`*,* $\boldsymbol{w}_1 \to \boldsymbol{e}_m$ *for certain* m*.*

Proof. In `ReLU2Layer1Hid`, since there is only one node, we have $X = \mathbb{C}_\alpha[\boldsymbol{x}_1, \boldsymbol{x}_1] = \mathbb{C}_\alpha[\boldsymbol{x}, \boldsymbol{x}]$. By Theorem 4, the dynamics of $\boldsymbol{w}_1$ is the linear dynamics plus the sticky weight rule, which is:

$$\dot{\boldsymbol{w}}_1 = \mathrm{diag}(\boldsymbol{w}_1 > 0) X \boldsymbol{w}_1 \tag{42}$$

By Lemma 3, the negative parts of $\boldsymbol{w}_1$ can be removed without changing the result. Let's only consider the nonnegative part of $\boldsymbol{w}$ and remove corresponding rows and columns of X.

Note that the linear dynamics $\dot{\boldsymbol{w}}_1 = X\boldsymbol{w}_1$ will converge to certain maximal eigenvector $\boldsymbol{y}$ (or its scaled version, depending on whether we have norm constraint or not). By Lemma 13, as long as X is not a scalar, $\boldsymbol{y}$ has at least one negative entry. Therefore, by continuity of the trajectory of the linear dynamics, from $\boldsymbol{w}_1$ to $\boldsymbol{y}$, the trajectory must cross the boundary of the polytope $\boldsymbol{w}_1 \geq 0$ that require all entries to be nonnegative.

After that, according to the sticky weight rule, in the ReLU dynamics, the corresponding component (say w_{1m}) stays at zero. We can remove the corresponding m-th row and column of X, and the process repeats until X becomes a scalar. Then $\boldsymbol{w}_1$ converges to that remaining dimension. Since $\boldsymbol{w}_1 \geq 0$, it must be the case that $\boldsymbol{w}_1 \to \boldsymbol{e}_m$ for some m. □

Theorem 6 (`ReLU2Layer` encourages diversity)**.** *If Assumption 1 holds, then for any local optimal* $(W_2, W_1) \in \Theta$ *of* `ReLU2Layer` *with* $\mathcal{E} > 0$*, either* $W_1 = \boldsymbol{v}\boldsymbol{e}_m^\top$ *for some* m *and* $\boldsymbol{v} \geq 0$*, or* $\mathrm{rank}(W_1) > 1$*.*

Proof. We just need to prove that if the solution (W_2, W_1) satisfies $\mathrm{rank}(W_1) = 1$, then for the solution to be a local optimal, $W_1 = \boldsymbol{v}\boldsymbol{e}_m^\top$ for some m and $\boldsymbol{v} \geq 0$.

Since $\mathrm{rank}(W_1) = 1$ and $\|W_1\|_F = 1$, by Lemma 8 we know that there exists unit vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ so that $W_1 = \boldsymbol{v}\boldsymbol{u}^\top$. Since $W_1 \geq 0$, we can pick $\boldsymbol{u} \geq 0$ and $\boldsymbol{v} \geq 0$. Otherwise if $\boldsymbol{u}$ has both positive and negative elements, then picking any nonzero element of $\boldsymbol{v}$, the corresponding rows/colums of W_1 will also have both signs, which is a contradiction.

Note that the objective function is

$$\mathcal{E} = \mathrm{tr}(W_2 F_1 W_2^\top) = \mathrm{tr}(W_2 W_1 X_\alpha W_1^\top W_2^\top) = (\boldsymbol{u}^\top X_\alpha \boldsymbol{u}) \|W_2 \boldsymbol{u}\|_2^2 > 0 \tag{43}$$

Therefore, $\boldsymbol{u}^\top X_\alpha \boldsymbol{u} > 0$ and $\|W_2\boldsymbol{u}\|_2 > 0$. Reusing the proof in Lemma 9 (Statement 3), we know that for W_2 with the constraint $\|W_2\|_F = 1$ to be an local optimal, W_2 has to be a rank-1 matrix with decomposition $W_2 = \boldsymbol{b}\boldsymbol{v}^\top$ with $\|\boldsymbol{b}\|_2 = 1$.

Then we have $\mathcal{E} = \boldsymbol{u}^\top X_\alpha \boldsymbol{u} > 0$ with $\boldsymbol{u} \geq 0$. From the proof of Lemma 13, we know that X_α has a unique *minimal* all-positive eigenvector $\boldsymbol{c} > 0$.

If there are ≥ 2 positive elements in $\boldsymbol{u}$, then we can always create a vector $\boldsymbol{a}$ (with mixed signs in its elements) so that (1) $\boldsymbol{a}$ has the same non-zero support as $\boldsymbol{u}$ and (2) $\boldsymbol{a}^\top \boldsymbol{c} = 0$. Therefore, $\boldsymbol{a}$ is in the space of orthogonal complement of $\boldsymbol{c}$. Since $\boldsymbol{c}$ is the unique minimal eigenvector, moving $\boldsymbol{u}$ along the direction of $\boldsymbol{a}$ will strictly improve $\mathcal{E}$, which contradicts with the fact that (W_2, W_1) is locally optimal.

Therefore, the unit vector $\boldsymbol{u}$ has only 1 positive entry, which is $\boldsymbol{e}_m$ for some m. Fig. 3 shows one example of learned weights with $\mathrm{rank} > 1$. □

B More Experiments

We also provide experiments with different batchsize (i.e., 256) and ablation studies on different exponent p in the direct version of α-CL. Note that we refer an unnormalized α-CL-direct as the following:

$$\alpha_{ij} = \exp(-d_{ij}^p / \tau) \tag{44}$$

while (normalized) α-CL-direct as the following (same as Eqn. 12 in the main text):

$$\alpha_{ij} = \frac{\exp(-d_{ij}^p / \tau)}{\sum_j \exp(-d_{ij}^p / \tau)} \tag{45}$$

By default, we set the exponent $p = 4$ and $\tau = 0.5$.

Dataset	Methods	100 epochs	300 epochs	500 epochs
CIFAR-10	$\mathcal{L}_{nce}$	86.84 ± 0.26	89.19 ± 0.15	$\mathbf{91.07 \pm 0.12}$
	α-CL-direct (Eqn. 44)	87.74 ± 0.28	89.76 ± 0.26	91.06 ± 0.09
	α-CL-direct (Eqn. 45)	$\mathbf{87.91 \pm 0.12}$	$\mathbf{89.89 \pm 0.18}$	91.06 ± 0.17
CIFAR-100	$\mathcal{L}_{nce}$	60.70 ± 0.40	64.22 ± 0.19	$\mathbf{66.84 \pm 0.16}$
	α-CL-direct (Eqn. 44)	63.28 ± 0.31	65.71 ± 0.20	66.73 ± 0.13
	α-CL-direct (Eqn. 45)	$\mathbf{63.47 \pm 0.06}$	$\mathbf{65.86 \pm 0.24}$	66.57 ± 0.21
STL10	$\mathcal{L}_{nce}$	82.09 ± 0.31	86.96 ± 0.19	87.31 ± 0.17
	α-CL-direct (Eqn. 44)	83.00 ± 0.28	87.35 ± 0.28	87.63 ± 0.29
	α-CL-direct (Eqn. 45)	$\mathbf{83.20 \pm 0.17}$	$\mathbf{87.36 \pm 0.12}$	$\mathbf{87.71 \pm 0.14}$

Table 3: Top-1 downstream task accuracy with ResNet50 backbone and 256 batchsize. Learning rate is 0.001. We also compare unnormalized α-CL-direct (Eqn. 44) versus (normalized) α-CL-direct (Eqn. 45). Normalized version, which is used in the main text of the paper, performs slightly better.

Exponent p	$p=2$	$p=4$	$p=6$	$p=8$	$p=10$
Top-1 accuracy (500 epochs)	83.74 ± 0.18	84.06 ± 0.24	$\mathbf{84.08 \pm 0.42}$	83.91 ± 0.28	83.56 ± 0.13

Table 4: Ablation study on different exponent p in STL10 for the normalized pairwise importance (Eqn. 45) in α-CL-direct.

C Other Lemmas

Lemma 4 (Gradient Formula of contrastive Loss (Eqn. 1) (extension of Lemma 2 in (Jing et al., 2022))**.** *Consider the loss function*

$$\min_{\boldsymbol{\theta}} \mathcal{L}_{\phi,\psi}(\boldsymbol{\theta}) := \sum_{i=1}^{N} \phi\left(\sum_{j\neq i} \psi(d_i^2 - d_{ij}^2)\right) \tag{46}$$

Then for any matrix (or vector) variable A, we have:

$$\sum_{i=1}^{N} \frac{\partial \mathcal{L}_{\phi,\psi}}{\partial \boldsymbol{z}[i]} A^\top[i] + \frac{\partial \mathcal{L}_{\phi,\psi}}{\partial \boldsymbol{z}[i']} A^\top[i'] = -\mathbb{C}_\alpha[\boldsymbol{z}, A] \tag{47}$$

and

$$\sum_{i=1}^{N} A[i] \frac{\partial \mathcal{L}_{\phi,\psi}}{\partial \boldsymbol{z}[i]} + A[i'] \frac{\partial \mathcal{L}_{\phi,\psi}}{\partial \boldsymbol{z}[i']} = -\mathbb{C}_\alpha[A, \boldsymbol{z}^\top] \tag{48}$$

where $\mathbb{C}_\alpha[\cdot,\cdot]$ is the contrastive covariance *defined as (here $\beta_i := \sum_{j\neq i} \alpha_{ij}$):*

$$\mathbb{C}_\alpha[\boldsymbol{x}, \boldsymbol{y}] := \sum_{i,j=1}^{N} \alpha_{ij} (\boldsymbol{x}[i] - \boldsymbol{x}[j])(\boldsymbol{y}[i] - \boldsymbol{y}[j])^\top - \sum_{i=1}^{N} \beta_i (\boldsymbol{x}[i] - \boldsymbol{x}[i'])(\boldsymbol{y}[i] - \boldsymbol{y}[i'])^\top \tag{49}$$

and α is defined as the following:

$$\alpha_{ij} := \phi'\left(\sum_{j\neq i} \psi(d_i^2 - d_{ij}^2)\right) \psi'(d_i^2 - d_{ij}^2) \geq 0 \tag{50}$$

where ϕ', ψ' are derivatives of ϕ, ψ.

Proof. Taking derivative of the loss function $\mathcal{L} = \mathcal{L}_{\phi,\psi}$ w.r.t. $\boldsymbol{z}[i]$ and $\boldsymbol{z}[i']$, we have:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{z}[i]} = \sum_{j\neq i} \alpha_{ij} (\boldsymbol{z}[j] - \boldsymbol{z}[i']) + \sum_{j\neq i} \alpha_{ji} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) \tag{51}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{z}[i']} = \sum_{j\neq i} \alpha_{ij} (\boldsymbol{z}[i'] - \boldsymbol{z}[i]) = \beta_i (\boldsymbol{z}[i'] - \boldsymbol{z}[i]) \tag{52}$$

We just need to check the following:

$$\sum_i \left(\sum_{j \neq i} \alpha_{ij} (\boldsymbol{z}[j] - \boldsymbol{z}[i']) + \sum_{j \neq i} \alpha_{ji} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) \right) A^\top[i] + \sum_i \beta_i (\boldsymbol{z}[i'] - \boldsymbol{z}[i]) A^\top[i'] \tag{53}$$

To see this, we only need to check whether the following is true:

$$-\Sigma_0 = \sum_i \left(\sum_{j \neq i} \alpha_{ij} (\boldsymbol{z}[j] - \boldsymbol{z}[i']) + \sum_{j \neq i} \alpha_{ji} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) \right) A^\top[i] + \sum_i \beta_i (\boldsymbol{z}[i'] - \boldsymbol{z}[i]) A^\top[i] \tag{54}$$

which means that

$$-\Sigma_0 = \sum_i \left(\sum_{j \neq i} \alpha_{ij} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) + \sum_{j \neq i} \alpha_{ji} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) \right) A^\top[i] \tag{55}$$

Since $\alpha_{ii}(\boldsymbol{z}[i] - \boldsymbol{z}[i]) = 0$ for arbitrarily defined α_{ii}, j can also take the value of i, this leads to

$$-\Sigma_0 = \sum_{i,j} \alpha_{ij} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) A^\top[i] + \sum_{i,j} \alpha_{ji} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) A^\top[i] \tag{56}$$

Swapping indices for the second term, we have:

$$\begin{aligned} -\Sigma_0 &= \sum_{i,j} \alpha_{ij} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) A^\top[i] + \sum_{i,j} \alpha_{ij} (\boldsymbol{z}[i] - \boldsymbol{z}[j]) A^\top[j] && (57) \\ &= \sum_{i,j} \alpha_{ij} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) A^\top[i] - \sum_{i,j} \alpha_{ij} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) A^\top[j] && (58) \\ &= -\sum_{i,j} \alpha_{ij} (\boldsymbol{z}[j] - \boldsymbol{z}[i]) (A^\top[j] - A^\top[i]) && (59) \end{aligned}$$

and the conclusion follows. □

Lemma 5. *The following minimization problem:*

$$\min_{p_j} \sum_j c_j p_j - \tau H(p) \quad \text{s.t.} \sum_j p_j = \frac{1}{\tau} x_0 \phi'(x_0) \tag{60}$$

where $H(p) := -\sum_j p_j \log p_j$ is the entropy and $x_0 := \sum_j e^{-c_j/\tau}$, has close-form solution:

$$p_j = \frac{1}{\tau} \exp(-c_j/\tau) \phi' \left(\sum_j \exp(-c_j/\tau) \right) \tag{61}$$

Proof. Define the following Lagrangian multiplier:

$$\mathcal{J}(\alpha, \boldsymbol{\theta}) := \sum_j c_j p_j - \tau H(p) + \mu \left(\sum_j p_j - \frac{1}{\tau} x_0 \phi'(x_0) \right) \tag{62}$$

Taking derivative w.r.t p_j and we have:

$$\frac{\partial \mathcal{J}}{\partial p_j} = c_j + \tau(\log p_j + 1) - \mu = 0 \tag{63}$$

which gives the solution

$$p_j = \exp\left(\frac{\mu}{\tau} - 1\right) \exp\left(-\frac{c_j}{\tau}\right) := Z \exp\left(-\frac{c_j}{\tau}\right) \tag{64}$$

where Z can be computed via the constraint:

$$Z = \frac{1}{\tau} \frac{x_0 \phi'(x_0)}{\sum_j e^{-c_j/\tau}} = \frac{1}{\tau} \phi'(x_0) \tag{65}$$

□

Lemma 6. *The normalization function* $\boldsymbol{y} = (\boldsymbol{x} - \text{mean}(\boldsymbol{x}))/\|\boldsymbol{x}\|_2$ *has the following forward/backward rule:*

$$\boldsymbol{y} = J(\boldsymbol{x})\boldsymbol{x}, \quad \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = J^\top(\boldsymbol{x}) \tag{66}$$

where $J(\boldsymbol{x}) := \frac{1}{\|P_{\mathbf{1}}^\perp \boldsymbol{x}\|_2} P_{\boldsymbol{x},\mathbf{1}}^\perp$ *is a symmetric matrix. For* $\boldsymbol{y} = \boldsymbol{x}/\|\boldsymbol{x}\|_2$*, the relationship still holds with* $J(\boldsymbol{x}) = \frac{1}{\|\boldsymbol{x}\|_2} P_{\boldsymbol{x}}^\perp$.

Proof. See Theorem 5 in (Tian, 2018). □

Lemma 7. *Suppose the output of a linear layer (with a weight matrix* W_l*) connects to a* ℓ_2 *regularization or LayerNorm through reversible layers, then* $\frac{\mathrm{d}}{\mathrm{d}t}\|W_l\|_F^2 = 0$.

Proof. From Lemma, for each sample i, we have its gradient before/after the normalization layer (say it is layer m) to be the following:

$$\boldsymbol{g}_m[i] = J_m^{\mathrm{n}}[i]^\top \boldsymbol{g}_m^{\mathrm{n}}[i] \tag{67}$$

where $\boldsymbol{g}_m[i]$ is the gradient after back-propagating through normalization, and $\boldsymbol{g}_m^{\mathrm{n}}[i]$ is the gradient sending from the top level.

Here $J_m^{\mathrm{n}}[i] = \frac{1}{\|P_{\mathbf{1}}^\perp \boldsymbol{f}_m[i]\|_2} P_{\boldsymbol{f}_m[i],\mathbf{1}}^\perp$ for LayerNorm and $J_m^{\mathrm{n}}[i] = \frac{1}{\|\boldsymbol{f}_m[i]\|_2} P_{\boldsymbol{f}_m[i]}^\perp$ for ℓ_2 normalization. For W_l, its gradient update rule is:

$$\dot{W}_l = \sum_i \tilde{\boldsymbol{g}}_l[i] \boldsymbol{f}_{l-1}^\top[i] \tag{68}$$

By reversibility, we know that $\tilde{\boldsymbol{g}}_l[i] = J_{(\tilde{l},m]}^\top[i]\boldsymbol{g}[i]$, where $J_{(\tilde{l},m]}[i]$ is the Jacobian after the linear layer $\tilde{l}$ till layer m, right before the normalization layer. Therefore, we have:

$$\begin{aligned}
\mathrm{tr}(W_l^\top \dot{W}_l) &= \sum_i \mathrm{tr}(W_l^\top J_{(\tilde{l},m]}^\top[i] J_m^{\mathrm{n}}[i]^\top \boldsymbol{g}_m^{\mathrm{n}}[i] \boldsymbol{f}_{l-1}^\top[i]) && (69)\\
&= \sum_i \mathrm{tr}(\boldsymbol{f}_{l-1}^\top[i] W_l^\top J_{(\tilde{l},m]}^\top[i] J_m^{\mathrm{n}}[i]^\top \boldsymbol{g}_m^{\mathrm{n}}[i]) && (70)\\
&= \sum_i \mathrm{tr}(\boldsymbol{f}_m^\top[i] J_m^{\mathrm{n}}[i]^\top \boldsymbol{g}_m^{\mathrm{n}}[i]) && (71)\\
&= 0 && (72)
\end{aligned}$$

The last two equality is due to reversibility $\boldsymbol{f}_m[i] = J_{(\tilde{l},m]}[i] W_l \boldsymbol{f}_{l-1}[i]$ and the property of normalization layers: $J_m^{\mathrm{n}}[i]\boldsymbol{f}_m[i] = 0$, since a vector projected to its own complementary space is always zero $P_{\boldsymbol{f}_m[i]}^\perp \boldsymbol{f}_m[i] = 0$.

Then we have

$$\frac{\mathrm{d}}{\mathrm{d}t}\|W_l\|_F^2 = \frac{\mathrm{d}}{\mathrm{d}t}\mathrm{tr}(W_l^\top W_l) = \mathrm{tr}(\dot{W}_l^\top W_l) + \mathrm{tr}(W_l^\top \dot{W}_l) = 0 \tag{73}$$

□

Lemma 8. *For every rank-1 matrix A with* $\|A\|_F = 1$*, there exists* $\|\boldsymbol{u}\|_2 = \|\boldsymbol{v}\|_2 = 1$ *so that* $A = \boldsymbol{u}\boldsymbol{v}^\top$.

Proof. It is clear that there exists $\boldsymbol{u}'$ and $\boldsymbol{v}'$ so that $A = \boldsymbol{u}'\boldsymbol{v}'^\top$. Since $\|A\|_F = 1$, we have $\mathrm{tr}(AA^\top) = \|\boldsymbol{u}'\|_2^2\|\boldsymbol{v}'\|_2^2 = 1$. Therefore, taking $\boldsymbol{u} = \boldsymbol{u}'/\|\boldsymbol{u}'\|_2$ and $\boldsymbol{v} = \boldsymbol{v}'/\|\boldsymbol{v}'\|_2$, we have $A = \boldsymbol{u}\boldsymbol{v}^\top$. □

Lemma 9. *For the following optimization problem*

$$\max_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}) := \|W_L W_{L-1} \dots W_1\|_2 \quad \text{s.t. } \|W_l\|_F = 1, \tag{74}$$

we have

- *(*Statement 1*) For any solution* $\boldsymbol{\theta}$ *with* $\mathcal{J}(\boldsymbol{\theta}) = 1$*,* $\boldsymbol{\theta}$ *is an aligned rank-1 solution.*

- *(*Statement 2*) Any solution $\boldsymbol{\theta}$ with $\mathcal{J}(\boldsymbol{\theta}) = 0$ cannot be a local maximum.*
- *(*Statement 3*) Every local maximum is an aligned-rank-1 solution (Def. 4).*
- *(*Statement 4*) All local maxima are global with an optimal value of 1.*

Proof. Statement 1. Note that we have:

$$\mathcal{J}(\boldsymbol{\theta}) := \|W_L W_{L-1} \dots W_1\|_2 \leq \prod_{l=1}^{L} \|W_l\|_2 \leq \prod_{l=1}^{L} \|W_l\|_F = 1 \tag{75}$$

and the equality only holds when all W_l are rank-1. By Lemma 8, for any l, there exists unit vectors $\boldsymbol{v}_l'$, $\boldsymbol{v}_{l-1}$ so that $W_l = \boldsymbol{v}_l' \boldsymbol{v}_{l-1}^\top$. To show that they must be aligned (i.e. $\boldsymbol{v}_l = \pm \boldsymbol{v}_l'$), we prove by contradiction.

Suppose for some l, $\boldsymbol{v}_l' \neq \pm \boldsymbol{v}_l$ and thus $|\boldsymbol{v}_l^\top \boldsymbol{v}_l'| < 1$. Then $W_{l+1} W_l = (\boldsymbol{v}_l^\top \boldsymbol{v}_l') \boldsymbol{v}_{l+1} \boldsymbol{v}_{l-1}^\top$ and $\|W_{l+1} W_l\|_2 \leq \|W_{l+1} W_l\|_F < 1$. Therefore, $\mathcal{J}(\boldsymbol{\theta}) < 1$. Note that for $W_l = \pm \boldsymbol{v}_l \boldsymbol{v}_{l-1}^\top$, we can always move around the signs to either $\boldsymbol{v}_0$ or $\boldsymbol{v}_L$ to fit into the definition of aligned-rank-1.

Statement 2. Note that $\mathcal{J}(\boldsymbol{\theta}) = 0$ means that $W_L W_{L-1} \dots W_1 = 0$. Since $\|W_1\|_F = 1$, either $W_{>1} = 0$, or there exists one non-zero row in $W_{>1}$ and a non-zero column in W_1 so that their inner product is 0. For the latter, we can make small change (certain column) of W_1 to W_1' so that $W_{>1} W_1' \neq 0$ and thus $\mathcal{J}(\boldsymbol{\theta}') > 0$; for the former, we repeat this until there exists l so that $W_{>l} \neq 0$, then we could slightly change W_l to W_l' so that $W_{>l-1}' = W_{>l} W_l' \neq 0$ and slightly change W_{l-1} to W_{l-1}' so that $W_{>l-2}' \neq 0$, until $W_{>1} W_1' \neq 0$ and thus $\mathcal{J}(\boldsymbol{\theta}') > 0$. Therefore, $\boldsymbol{\theta}$ cannot be a local maximum.

Statement 3. Suppose $\boldsymbol{\theta}^*$ is a local maximum solution. By Statement 2, $J(\boldsymbol{\theta}^*) > 0$. Since $J(\boldsymbol{\theta}^*)$ is the spectral norm, by its definition, there exists a unit vector $\boldsymbol{u}$ so that $\|W_L^* W_{L-1}^* \dots W_1^* \boldsymbol{u}\|_2 = J(\boldsymbol{\theta}^*) > 0$.

Now let $\boldsymbol{v}_{L-1}' := W_{L-1}^* W_{L-2}^* \dots W_1^* \boldsymbol{u}$. Note that $\boldsymbol{v}_{L-1}' \neq 0$ (otherwise $J(\boldsymbol{\theta}^*)$ would be zero). Consider the following optimization subproblem (here we optimize over W_L and treat $\boldsymbol{v}_{L-1}'$ as a fixed vector).

$$\max_{W_L} \mathcal{J}(W_L; W_{-L}^*) = \|W_L \boldsymbol{v}_{L-1}'\|_2 \quad \text{s.t. } \|W_L\|_F = 1 \tag{76}$$

By local optimality of $\boldsymbol{\theta}^*$, W_L^* must be the local maximum of Eqn. 76. Note that all critical points of Eqn. 76 must satisfy

$$W_L \boldsymbol{v}_{L-1}' \boldsymbol{v}'^\top_{L-1} = \lambda W_L \tag{77}$$

for some constant λ. Notice that to satisfy this condition, each row of W_L must be an eigenvector of $\boldsymbol{v}_{L-1}' \boldsymbol{v}'^\top_{L-1}$. For local maximal solutions, λ is the largest eigenvalue of $\boldsymbol{v}_{L-1}' \boldsymbol{v}'^\top_{L-1}$, and each row of W_L is the corresponding eigenvector. It is clear that the rank-1 matrix $\boldsymbol{v}_{L-1}' \boldsymbol{v}'^\top_{L-1}$ has a unique maximum eigenvalue $\|\boldsymbol{v}_{L-1}'\|_2^2 > 0$ with its corresponding one-dimensional eigenspace span by $\boldsymbol{v}_{L-1} = \boldsymbol{v}_{L-1}' / \|\boldsymbol{v}_{L-1}'\|_2$ (while all other eigenvalues are zeros). Therefore, W_L^* as the local maximum of Eqn. 76, must have:

$$W_L^* = \boldsymbol{v}_L \boldsymbol{v}_{L-1}^\top \tag{78}$$

for some $\|\boldsymbol{v}_L\|_2 = 1$.

Now let $\boldsymbol{v}_{L-2}' := W_{L-2}^* \dots W_1^* \boldsymbol{u} \neq 0$. Then $\boldsymbol{v}_{L-1}' = W_{L-1}^* \boldsymbol{v}_{L-2}'$. Treating $\boldsymbol{v}_{L-2}'$ as a fixed vector and varying W_{L-1} and W_L simultaneously, then since $\boldsymbol{\theta}^*$ is a local maximal solution, W_L^* must take the form of Eqn. 78 given any W_{L-1}^*, which means that the objective function now becomes

$$\mathcal{J}(W_{L-1}; W_{-(L-1)}^*) = \|W_L^* \boldsymbol{v}_{L-1}'\|_2 = \|\boldsymbol{v}_L \boldsymbol{v}_{L-1}^\top \boldsymbol{v}_{L-1}'\|_2 = \|\boldsymbol{v}_{L-1}'\|_2 = \|W_{L-1} \boldsymbol{v}_{L-2}'\|_2 \tag{79}$$

and the subproblem becomes:

$$\max_{W_{L-1}} \|W_{L-1} \boldsymbol{v}_{L-2}'\|_2 \quad \text{s.t. } \|W_{L-1}\|_F = 1 \tag{80}$$

Repeating this process, we know W_{L-1}^* must satisfy:

$$W_{L-1}^* = \boldsymbol{v}_{L-1} \boldsymbol{v}_{L-2}^\top \tag{81}$$

for $\boldsymbol{v}_{L-2} := \boldsymbol{v}'_{L-2}/\|\boldsymbol{v}'_{L-2}\|_2$. This procedure can be repeated until W_1 and the prove is complete.

Statement 4. Due to Statement 3, for any local optimal $\boldsymbol{\theta}^*$, we know that $W_L^* W_{L-1}^* \dots W_1^* = \boldsymbol{v}_L \boldsymbol{v}_0^\top$, so $\mathcal{J}(\boldsymbol{\theta}^*) = 1$. By the upper bound of the objective (Eqn. 75), we know they are all globally optimal. □

Lemma 10. $\frac{\mathrm{d}}{\mathrm{d}t}\|\boldsymbol{w}_k\|_2^2 = 0$*, if node* k *is under BatchNorm.*

Proof. For BN, it is a layer with reversibility on each filter k. We use $\boldsymbol{f}_k, \boldsymbol{g}_k \in \mathbb{R}^N$ to represent the activation/gradient at node k in a batch of size N. The forward/backward operation of BN can be written as:

$$\boldsymbol{f}_k^{\mathrm{n}} = J_k \boldsymbol{f}_k, \quad \boldsymbol{g}_k = J_k^\top \boldsymbol{g}_k^{\mathrm{n}} \tag{82}$$

Here $J_k = J_k^\top = \frac{1}{\|P_{\mathbf{1}}^\perp \boldsymbol{f}_k\|_2} P^\perp_{\boldsymbol{f}_k, \mathbf{1}}$ is the Jacobian matrix at each node k.

We check how the weight $\boldsymbol{w}_k$ changes under BatchNorm. Here we have $\boldsymbol{f}_k = h(F_{l-1}\boldsymbol{w}_k)$ where h is a reversible activation and $F_{l-1} \in \mathbb{R}^{N \times n_{l-1}}$ contains all output from the last layer. Then we have:

$$\dot{\boldsymbol{w}}_k = \sum_i h'_i g_k[i] \boldsymbol{f}_{l-1}[i] = F_{l-1}^\top D_k \boldsymbol{g}_k = F_{l-1}^\top D_k J_k^\top \boldsymbol{g}_k^{\mathrm{n}} \tag{83}$$

where $D_k := \mathrm{diag}([h'_i]_{i=1}^N) \in \mathbb{R}^{N \times N}$. Due to reversibility, we have $\boldsymbol{f}_k = h(F_{l-1}\boldsymbol{w}_k) = D_k F_{l-1} \boldsymbol{w}_k$. Therefore,

$$\boldsymbol{w}_k^\top \dot{\boldsymbol{w}}_k = \boldsymbol{w}_k^\top F_{l-1}^\top D_k J_k^\top \boldsymbol{g}_k^{\mathrm{n}} = \boldsymbol{f}_k^\top J_k^\top \boldsymbol{g}_k^{\mathrm{n}} = 0 \tag{84}$$

□

Lemma 11 (BatchNorm regularization)**.** *Consider the following optimization problem*

$$\max_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}) := \|W_L W_{L-1} \dots W_1\|_2 \quad \text{s.t. } \|W_L\|_F = 1, \quad \|\boldsymbol{w}_{lk}\|_2 = 1/\sqrt{n_l} \tag{85}$$

where $\boldsymbol{w}_{lk}$ *are rows of* W_l *(i.e., weight of the* k*-th filter at layer* l*). Then Lemma 9 still holds by replacing aligned-ranked-one with aligned-uniform condition.*

Proof. The proof is basically the same. The only difference here is that the sub-problem (Eqn. 80) becomes:

$$\max_{W_l} \|W_l \boldsymbol{v}'_{l-1}\|_2 \quad \text{s.t. } \|\boldsymbol{w}_{lk}\|_2 = 1/\sqrt{n_l} \tag{86}$$

for $1 \le l \le L-1$. The critical point condition now becomes (here Λ is a diagonal matrix):

$$W_l \boldsymbol{v}'_{l-1} \boldsymbol{v'}_{l-1}^\top = \Lambda W_l \tag{87}$$

That is, each row of W_l now has a different constant. Since the eigenvalue of $\boldsymbol{v}'_{l-1}\boldsymbol{v'}_{l-1}^\top$ can only be 0 or 1, and 0 won't work (otherwise the corresponding row of W_l would be a zero vector, violating the row-norm constraint), all diagonal element of λ has to be 1. Therefore, $W_l = \boldsymbol{v}_l \boldsymbol{v}_{l-1}^\top$. Due to row-normalization, we have $[\boldsymbol{v}_l]_k = \pm 1/\sqrt{n_l}$ for $1 \le l \le L-1$, while $\boldsymbol{v}_L$ and $\boldsymbol{v}_0$ can still take arbitrary unit vector. □

Lemma 12. *If Assumption 1(Nonnegativeness) holds, then a 2-layer ReLU network with weights* $\boldsymbol{w}_{1k} \ge 0$ *and* W_2 *has the same activations (i.e.,* $\boldsymbol{f}_l = \boldsymbol{f}'_l$*) as its linear network counterpart with the same weights* $\boldsymbol{w}'_{1k} = \boldsymbol{w}_{1k}$ *and* $W'_2 = W_2$*.*

Proof. Since $W'_2 = W_2$, we only need to prove $\boldsymbol{f}_1 = \boldsymbol{f}'_1$. For each filter k, we have its activation $f_{1k} = \max(\sum_m w_{1km} x_{km}, 0)$ and $f'_{1k} = \sum_m w'_{1km} x_{km} = \sum_m w_{1km} x_{km}$. By Assumption 1(non-negativeness), all $x_{km} \ge 0$. Since $w_{1km} \ge 0$, $\sum_m w_{1km} x_{km} \ge 0$ and $f_{1k} = f'_{1k}$. □

Lemma 13. *If Assumption 1 holds,* $M \ge 2$*,* $\boldsymbol{x}_1$ *covers all* M *modes, and* $\alpha_{ij} > 0$*, then the maximal eigenvector of* X_α *always contains at least one negative entry.*

Proof. Let $X_k := \mathbb{C}_\alpha[\boldsymbol{x}_k, \boldsymbol{x}_k]$. By Lemma 14, all off-diagonal elements of X_k are negative. Then X_k can be written as $X_k = \beta I - X'_k$ for some β where X'_k is a symmetric matrix whose entries are all positive. By Perron–Frobenius theorem, X'_k has a unique maximal eigenvector $\boldsymbol{u}_k > 0$ (with all positive entries) and its associated positive eigenvalue $\lambda_k > 0$. Therefore, $\boldsymbol{u}_k > 0$ is also the unique(!) minimal eigenvector of X_k. Since $M \geq 2$, there exists a maximal eigenspace, in which any maximal eigenvector $\boldsymbol{y}_k$ satisfies $\boldsymbol{y}_k^\top \boldsymbol{u}_k = 0$. By Lemma 15, the theorem holds. □

Lemma 14. *If the receptive field R_k satisfies Assumption 1, and the collection of N vectors $\{\boldsymbol{x}_k[i]\}_{i=1}^N$ contains all M modes, then all off-diagonal elements of $\mathbb{C}_\alpha[\boldsymbol{x}_k, \boldsymbol{x}_k]$ are negative.*

Proof. We check every entry of $X_k := \mathbb{C}_\alpha[\boldsymbol{x}_k, \boldsymbol{x}_k]$. Let $\beta_i := \sum_{j \neq i} \alpha_{ij}$. Note that for off-diagnoal element $[X_k]_{ml}$ with $m \neq l$, we have:

$$[X_k]_{ml} = \sum_{ij} \alpha_{ij}(x_{km}[i] - x_{km}[j])(x_{kl}[i] - x_{kl}[j]) - \sum_i \beta_i (x_{km}[i] - x_{km}[i'])(x_{kl}[i] - x_{kl}[i']) \quad (88)$$

Let $A_m := \{i : x_{km}[i] > 0\}$ be the sample set in which the m-th component is strictly positive, and $A_m^c := \{1, 2, \ldots, N\} \backslash A_m$ its complement. By Assumption 1(one-hotness), if $i \in A_m$ then $i \in A_{m'}^c$ for any $m' \neq m$.

Now we consider several cases for sample i and j:

Case 1, $i, j \in A_m$. Then $i, j \in A_l^c$ for $l \neq m$. This means that $x_{kl}[i] - x_{kl}[j] = 0$.

Case 2, $i, j \in A_m^c$. Then $x_{km}[i] - x_{km}[j] = 0$.

Case 3, $i \in A_m$ **and** $j \in A_m^c$. Since $j \in A_m^c$, we have $x_{km}[i] - x_{km}[j] = x_{km}[i] > 0$. On the other hand, since $i \in A_m$, $i \in A_l^c$, we have $x_{kl}[i] - x_{kl}[j] = -x_{kl}[j] \leq 0$. Therefore, $(x_{km}[i] - x_{km}[j])(x_{kl}[i] - x_{kl}[j]) \leq 0$.

Case 4. $i \in A_m^c$ **and** $j \in A_m$. This is similar to Case 3.

Putting them all together, since $\alpha_{ij} > 0$, we know that

$$\sum_{ij} \alpha_{ij}(x_{km}[i] - x_{km}[j])(x_{kl}[i] - x_{kl}[j]) \leq 0 \quad (89)$$

Furthermore, it is strictly negative since for $i \in A_m$ and $j \in A_l$, we have

$$(x_{km}[i] - x_{km}[j])(x_{kl}[i] - x_{kl}[j]) = -x_{km}[i]x_{kl}[j] < 0 \quad (90)$$

By our assumption that the N vectors $\{\boldsymbol{x}_k[i]\}_{i=1}^N$ contains all M modes, both A_m and A_l are not empty so this is achievable.

For the second summation, by Assumption 1(Augmentation), either $i, i' \in A_m$ or $i, i' \in A_m^c$, it is always zero for $m \neq l$. □

Lemma 15. *If $\boldsymbol{v} > 0$ is an all positive d-dimensional vector, $\boldsymbol{u}^\top \boldsymbol{v} = 0$, then*

$$\min_m u_m \leq -\frac{\min_m v_m}{d - k} \frac{\|\boldsymbol{u}\|_\infty}{\|\boldsymbol{v}\|_\infty} \quad (91)$$

where k is the number of nonnegative entries in $\boldsymbol{u}$.

Proof. Let $m_0 := \arg\max_m |u_m|$. If $u_{m_0} = -\|\boldsymbol{u}\|_\infty = \min_m u_m$ then we have proven the theorem. Otherwise $u_0 := u_{m_0} \geq 0$. u_{m_0} is the largest entry of $\{u_m\}$.

Since $\min_m u_m < 0$, by Rearrangement inequality we have:

$$0 = \boldsymbol{u}^\top \boldsymbol{v} = \sum_m u_m v_m \geq \left(\min_m v_m\right) u_0 + (d - k) \left(\max_m v_m\right) \left(\min_m u_m\right) \quad (92)$$

The conclusion follows. □