Contrasting the landscape of contrastive and non-contrastive learning

Ashwini Pokle* Carnegie Mellon University
Jinjin Tian* Carnegie Mellon University
Yuchen Li* Carnegie Mellon University
Andrej Risteski Carnegie Mellon University

(apokle, jinjint, yuchenl4, aristesk)@andrew.cmu.edu
*Equal contribution

Abstract

A lot of recent advances in unsupervised feature learning are based on designing features which are invariant under semantic data augmentations. A common way to do this is *contrastive learning*, which uses positive and negative samples. Some recent works however have shown promising results for *non-contrastive learning*, which does not require negative samples. However, the non-contrastive losses have obvious "collapsed" minima, in which the encoders output a constant feature embedding, independent of the input. A folk conjecture is that so long as these collapsed solutions are avoided, the produced feature representations should be good. In our paper, we cast doubt on this story: we show through theoretical results and controlled experiments that even on simple data models, non-contrastive losses have a preponderance of *non-collapsed* bad minima. Moreover, we show that the training process does not avoid these minima. Code for this work can be found at https://github.com/ashwinipokle/contrastive_landscape.

1 INTRODUCTION

Recent improvements in representation learning without supervision were driven by self-supervised learning approaches, in particular contrastive learning (CL), which constructs positive and negative samples out of unlabeled dataset via data augmentation (Chen et al., 2020; He et al., 2020; Caron et al., 2020; Ye et al., 2019; Oord et al., 2018; Wu et al., 2018). Subsequent works based on data augmentation also showed promising results for methods based on non-contrastive learning (non-CL), which do not require explicit negative samples (Grill et al., 2020; Richemond et al., 2020; Chen & He, 2021; Zbontar et al., 2021; Tian et al., 2021).

However, understanding of how these approaches work, especially of how the learned representations compare—qualitatively and quantitatively—is lagging behind. In this paper, via a combination of empirical and theoretical results, we provide evidence that non-contrastive methods based on data augmentation can lead to substantially worse representations.

Most notably, avoiding the collapsed representations has been the key ingredient in prior successes in non-contrastive learning. The collapses was first referred as the *complete collapse*, that is, all representation vectors shrink into a single point; later a new type of collapses, *dimension collapse* (Hua et al., 2021; Jing et al., 2022) caught attention as well, that is the embedding vectors only span a lower-dimensional subspace. It is naturally to conjunct that avoiding those kinds of collapsed representations suffices for non-contrastive learning to succeed. We provide strong evidence in contrast to this: namely, we show that even under a very simple, but natural data model, the non-contrastive loss has a prevalence of bad optima that are not collapsed (in neither way, complete or dimension collapse). Moreover, we supplement this with a training dynamics analysis: we prove that the training dynamics can remedy this situation—however, this is crucially tied to a careful choice of a predictor network in the model architecture.

Our methodology is largely a departure from other works on self-supervised learning in general: rather than comparing the performance of different feature learning methods on specific datasets, we provide extensive theoretical and experimental results for *synthetic* data generated by a natural data-generative process in which there is a "ground-truth" representation. The

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

more structured form of the data allows us to make much more precise statements on the quality of the representations produced by the methods we consider.

2 OVERVIEW OF RESULTS

We will study two aspects of contrastive learning (CL) and non-contrastive learning (non-CL): the set of optima of the objective, and the gradient descent training dynamics.

In order to have a well-defined notion of a "ground truth" we will consider data generated by a sparse-coding inspired model Olshausen & Field (1997); Arora et al. (2015); Wen & Li (2021):

$$\boldsymbol{x} = \boldsymbol{M}\boldsymbol{z} + \boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma_0^2 \mathbf{I}_p).$$

Moreover, the encoder(s) used in the various settings will be simple one-layer ReLU networks. The motivation comes from classical theoretical work on sparse coding Arora et al. (2015), which shows that such an encoder can recover a representation $\tilde{z}$ which has the same support as the z.

2.1 Landscape Analysis

The first aspect we will turn to is an analysis of the minima for both contrastive and non-contrastive losses. More precisely, we will show that even for simple special cases of the sparse coding data generative model, non-contrastive losses exhibit abundant *non-collapsed bad global optima*, whereas the contrastive loss is minimized *only at the ground truth solution*:

Theorem (informal, non-collapse bad global optima)**.** *On a noise-less sparse coding data distribution augmented by random masking, with encoders given by single-layer ReLU networks,*

- *The non-contrastive loss function has infinitely many non-collapsed global optima that are far away from the ground truth.*
- *Any global optima of the contrastive loss function is (up to a permutation) equal to the ground truth solution.*

In this case, the ground truth refers to learning the features z underlying the sparse coding data distribution, which is formally defined in Section 4. For the formal theorem statement, see Section 5.1. The proof is deferred to Appendix A.

2.2 Training Dynamics Analysis

In the previous section, we have shown that bad optima exists for non-contrastive loss, the next reasonable question to ask is whether the training dynamics of the algorithm is capable of circumventing them. We answer this question using both theoretical analyses and experimental results.

Theoretically, we show that for non-contrastive loss, under certain conditions that we will specify,

- (Negative result) A one-layer linear dual network (equation 12) optimized via gradient descent cannot improve beyond a linear combination of its initialization. (Theorem 2 and Corollary 1)
- (Positive result) However, a one-layer ReLU network (equation 13) provably converges to the groundtruth using alternating optimization with warm-start initialization and appropriate normalization. (Theorem 3)

These results show that the results of non-contrastive learning is very sensitive to the interaction between training algorithm, non-linearity, and normalization scheme.

Remark 1. *In comparison, for contrastive loss, Wen & Li (2021) proves that training dynamics learns a good representation under a similar sparse-coding inspired data generating process, even without the warm-start assumption.*

Moreover, we provide detailed empirical evidence in support of the following conclusions:

1. The gradient descent dynamics on the contrastive loss is capable of recovering the ground truth representation z. These results are robust to changes in architecture of the encoder, as well as the parameters for the data generative model.
2. The gradient descent dynamics on the two most popular non-contrastive losses, SimSiam and BYOL (Grill et al., 2020; Chen & He, 2021), are not able to avoid the poor minima, if the architecture does not include a linear predictor on top of the encoder(s). However, if a predictor is included, both SimSiam and BYOL tend to converge to a solution close to the ground truth. We also find that this predictor is optional if weights of the encoder are initialized close to the ground truth minimum, and these weights are row-normalized or column normalized after every gradient descent update.

3 RELATED WORK

Contrastive learning (CL) The idea of learning representations so that a pair of similar samples (also

known as "positive pairs") are closer and dissimilar pairs (known as "negative samples") are farther is widespread both in NLP (Gao et al. (2021); Giorgi et al. (2020)) and Vision (Chen et al. (2020); He et al. (2020); Caron et al. (2020); Ye et al. (2019); Oord et al. (2018); Wu et al. (2018); Tian et al. (2020a); Li et al. (2020); Henaff (2020)). One of the earliest works based on this principle was proposed by Hadsell et al. (2006) which used Euclidean distance based contrastive loss. Recently, normalized temperature-scaled cross entropy loss or *NT-Xent* loss (Chen et al. (2020); Wu et al. (2018); Sohn (2016)) has gained more popularity. Wang & Isola (2020) suggest that CL should optimize for both the alignment and uniformity features on the hypersphere. Most of these CL approaches often require additional tricks like maintaining a large memory bank (Wu et al. (2018); Misra & Maaten (2020)), momentum encoder (He et al. (2020)) or using a large batch sizes (Chen et al. (2020)) to learn useful representations and prevent collapse, which makes them computationally intensive. There has been some recent work to overcome these drawbacks. Caron et al. (2020) propose contrasting soft cluster assignments for augmented views of the image instead of directly comparing features which helps them to avoid most of these tricks.

Non-contrastive learning (Non-CL) Another line of proposed SSL-approaches, such as BYOL (Grill et al., 2020), question whether these negative examples are indispensable to prevent collapsing while preserving high performance. The authors instead propose a framework that outperforms previous state-of-the-art approaches (Chen et al. (2020), He et al. (2020), Tian et al. (2020a)) without any use of negative examples. To understand how non-contrastive learning works, existing works mostly focus on analyzing what elements help it avoid learning collapsed representations (Grill et al., 2020; Richemond et al., 2020; Chen & He, 2021; Zbontar et al., 2021; Tian et al., 2021; Wang et al., 2021). Our work contributes to these prior efforts by pointing out another subtle difference between contrastive and non-contrastive learning, which has not been formalized in prior theoretical or empirical works. Specifically, as we show both theoretically and empirically, the non-contrastive loss has abundant *non-collapse bad global optima*.

Theoretical understanding Methodologically, our work is closely related to a long line of works aiming to understand algorithmic behaviors by analyzing a simple model in controlled settings. Our model architecture gets inspirations from Tian et al. (2021) and Wen & Li (2021). Namely, Tian et al. (2021) adopts a linear network and calculates the optimization dynamics of non-contrastive learning, while Wen & Li (2021) analyzes the feature learning process of contrastive learning on a single-layer linear model with ReLU activation. Another relevant work by Tian et al. (2020b) calculates the optimization dynamics for a more complex N-layered dual-deep linear networks with ReLU activation for SimCLR architecture and different variants of contrastive loss like InfoNCE, soft-triplet loss etc. In a similar spirit, we base our comparison of contrastive and non-contrastive learning on single-layer dual networks, but instead of discussing the optimization process, we focus on the final features learned by these different training approaches. Our work is also related to Arora et al. (2019) in which the authors assess the ability of self-supervised learning methods to encode the latent structure of the data. To theoretically evaluate the learned features, we use a variant of the sparse coding data generating model presented in Olshausen & Field (1997); Lewicki & Sejnowski (2000); Arora et al. (2015); Wen & Li (2021), so that the ground truth dictionary defines the evaluation metrics that characterize the quality of features learned by our simple model. Namely, in order for a learned model to represent the latent structure of the data, each feature in the ground truth dictionary is expected to be picked up by a set of learned neurons. (Ma & Collins, 2018; Zimmermann et al., 2021) are on the related topics as well but use different data models or setups.

4 PROBLEM SETUP

Notation For a matrix $\boldsymbol{M} \in \mathcal{R}^{m\times n}$, we use $\boldsymbol{M}_{i*}$ to represent its i-th row and $\boldsymbol{M}_{*j}$ to represent its j-th column. We denote $[d] := \{1, \dots, d\}$. We use the notation *poly(d)* to represent a polynomial in d. We denote the set of matrices with unit column norm as $\mathcal{U}$ — we will omit the implied dimensions if clear from context. The subset of $\mathcal{U}$ with positive entries and non-negative entries is denoted by $\mathcal{U}_>$ and $\mathcal{U}_{\geq}$. We use $\mathbb{O}$ to denote the set of orthogonal matrices (again, we will omit the dimension if clear from context), and the subset of orthogonal matrices with positive entries and non-negative entries is denoted by $\mathbb{O}_>$ and $\mathbb{O}_{\geq}$.

4.1 A Sparse coding model set up

Data generating process The data samples $\boldsymbol{x}$ in our model are generated i.i.d. through a sparse coding generative model as:

$$\boldsymbol{x} = \boldsymbol{M}\boldsymbol{z} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma_0^2 \mathbf{I}_p) \tag{1}$$

where, the *sparse* latent variable $\mathbf{z} = (\boldsymbol{z}_1, \dots \boldsymbol{z}_d) \in \{0,1\}^d$ and $\boldsymbol{\epsilon} \in \mathbb{R}^{p\times 1}$. We assume that $\mathbf{z}_1, \dots, \mathbf{z}_d$ are i.i.d. with $\beta := Pr(\mathbf{z}_i \neq 0) \ll 1$ for all $i \in [d]$. The *dictionary matrix* $\mathbf{M} \in \mathbb{O}^{p\times d}$, is a column-orthonormal

matrix. Further, we assume $p = poly(d)$. We note that the noise $\boldsymbol{\epsilon}$ is optional in our setting and there are some theoretical results in section 5 that assume that $\boldsymbol{\epsilon} = 0$. All empirical results assume a small Gaussian noise with $\sigma_0 = \Theta(\frac{\log d}{d})$.

Augmentation We augment an input sample $\boldsymbol{x}$ through random masking. These random masks are generated as follows: Consider two independent diagonal matrices $\boldsymbol{D_1}, \boldsymbol{D_2} \in \mathbb{R}^{p\times p}$ with $\{0, 1\}$ as their diagonal entries. Each of the diagonal entries are sampled i.i.d from $Bernoulli(\alpha)$. For an input $\boldsymbol{x}$, the augmented views $\mathbf{a}_1, \mathbf{a}_2 \in \mathbb{R}^{p\times 1}$ are generated as *:

$$\mathbf{a}_1 := \boldsymbol{D_1}\mathbf{x}; \;\; \mathbf{a}_2 := \boldsymbol{D_2}\mathbf{x} \tag{3}$$

Network architecture We use a dual network architecture inline with prior work Arora et al. (2015); Tian et al. (2021); Wen & Li (2021). In contrastive learning setting, we assume that the encoder shares weights between the two input views. However, in non-contrastive learning setting, we assume two separate independently initialized networks, called online network and target network. Both the networks are a single-layer neural network with ReLU activation, namely:

$$h_{\boldsymbol{W},\boldsymbol{b}}(\boldsymbol{x}) := ReLU(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}) \in \mathbb{R}^{m\times 1}.$$

For brevity, we will skip the bias $\boldsymbol{b}$ in the notation and instead refer to $h_{\boldsymbol{W},\boldsymbol{b}}(\boldsymbol{x})$ simply as $h_{\boldsymbol{W}}(\boldsymbol{x})$. For results in section 5.2, in addition to the linear encoder, we assume a linear prediction head $\boldsymbol{W}^p \in \mathbb{R}^{m\times m}$ which transforms the output of one of the encoders to match it to the output of the other encoder.

$$g_{\boldsymbol{W}^p,\boldsymbol{b}^p,\boldsymbol{W},\boldsymbol{b}}(\boldsymbol{x}) := \boldsymbol{W}^p h_{\boldsymbol{W},\boldsymbol{b}}(\boldsymbol{x}) + \boldsymbol{b}^p \in \mathbb{R}^{m\times 1}.$$

For brevity, we will skip the biases $\boldsymbol{b}$ and $\boldsymbol{b}^p$ in the notation and refer to $g_{\boldsymbol{W}^p,\boldsymbol{b}^p,\boldsymbol{W},\boldsymbol{b}}(\boldsymbol{x})$ simply as $g_{\boldsymbol{W}^p,\boldsymbol{W}}(\boldsymbol{x})$.

Non-contrastive learning algorithm and loss function Our non-contrastive learning algorithm is motivated from the recent works in non-contrastive SSL methods like BYOL (Grill et al. (2020)) and SimSiam (Chen & He (2021)). Given an input data sample $\boldsymbol{x}$, we first augment it through random masking to obtain two different views of the input $\boldsymbol{x}$. We then extract representation vectors of these views with two different randomly initialized encoders (described above), called the online network and the target network, with weights $\boldsymbol{W}^o \in \mathbb{R}^{m\times p}$ and $\boldsymbol{W}^t \in \mathbb{R}^{m\times p}$, respectively. The output of these encoders are normalized and loss on the resulting representations is computed as:

$$L_{\text{non-CL-l2}}(\mathbf{W}^o, \mathbf{W}^t) := \tag{4}$$
$$\mathbb{E}\left\| \frac{a(h_{\boldsymbol{W}^o}(\boldsymbol{D}_1\boldsymbol{x}))}{\|a(h_{\boldsymbol{W}^o}(\boldsymbol{D}_1\boldsymbol{x}))\|_2} - SG\left(\frac{a(h_{\boldsymbol{W}^t}(\boldsymbol{D}_2\boldsymbol{x}))}{\|a(h_{\boldsymbol{W}^t}(\boldsymbol{D}_2\boldsymbol{x}))\|_2}\right)\right\|_2^2$$

where, $a(\cdot)$ refers to the activation function, and the stop gradient operator SG indicates that we do not compute the gradients of the operand during optimization. In our experiments, we use ReLU activation, and during the training process, we alternate between $L_{\text{non-CL-l2}}(\mathbf{W}^o, \mathbf{W}^t)$ and $L_{\text{non-CL-l2}}(\mathbf{W}^t, \mathbf{W}^o)$ to update the online and target networks. This alternating optimization can be viewed as a simple approximation of the combination of stop gradient and exponential moving average techniques used in prior works Grill et al. (2020), so that some theoretical analysis on the training dynamics can be carried out conveniently.

For the architecture with a linear prediction head, we minimize the following loss:

$$L_{\text{non-CL-l2-pred}}(\mathbf{W}^o, \mathbf{W}^t, \mathbf{W}^p) := \tag{5}$$
$$\mathbb{E}\left\| \frac{g_{\boldsymbol{W}^t,\boldsymbol{W}^p}(\boldsymbol{D}_1\boldsymbol{x})}{\|g_{\boldsymbol{W}^t,\boldsymbol{W}^p}(\boldsymbol{D}_1\boldsymbol{x})\|_2} - SG\left(\frac{a(h_{\boldsymbol{W}^t}(\boldsymbol{D}_2\boldsymbol{x})}{\|a(h_{\boldsymbol{W}^t}(\boldsymbol{D}_2\boldsymbol{x}))\|_2}\right)\right\|_2^2$$

For our theoretical results, we use a closely related loss function defined as:

$$L_{\text{non-CL}}(\mathbf{W}^o, \mathbf{W}^t)^\dagger := \tag{6}$$
$$\left(2 - 2\mathbb{E}\langle ReLU(\boldsymbol{W}^o\boldsymbol{D}_1\boldsymbol{x}), SG(ReLU(\boldsymbol{W}^t\boldsymbol{D}_2\boldsymbol{x}))\rangle\right)$$

For the architectures with a prediction head, we define the non-contrastive loss function as:

$$L_{\text{non-CL-pred}}(\mathbf{W}^o, \mathbf{W}^t, \mathbf{W}^p)^\dagger := \tag{7}$$
$$\left(2 - 2\mathbb{E}\langle \mathbf{W}^p ReLU(\boldsymbol{W}^o\boldsymbol{D}_1\boldsymbol{x}), SG(ReLU(\boldsymbol{W}^t\boldsymbol{D}_2\boldsymbol{x}))\rangle\right)$$

Contrastive learning algorithm and loss function For contrastive learning, we use a simplified version of SimCLR algorithm (Chen et al. (2020)). We assume a linear encoder with weight $\boldsymbol{W} \in \mathbb{R}^{m\times p}$ that extracts representation vectors of an augmented input. Given positive augmented samples $\boldsymbol{D}_1\boldsymbol{x}$ and $\boldsymbol{D}_2\boldsymbol{x}$ and a batch

*Prior work (Wen & Li, 2021) motivates another masking scheme which we refer to as "dependent masking" in which the diagonal entries of a diagonal matrix $\boldsymbol{D}$ are sampled i.i.d. from from $Bernoulli(\alpha)$. The augmentation views are computed as

$$\boldsymbol{a}_1 := 2\boldsymbol{D}\boldsymbol{x}; \;\; \boldsymbol{a}_2 := 2(\boldsymbol{I} - \boldsymbol{D})\boldsymbol{x} \tag{2}$$

† The contrastive loss in Eq. (8) and non-contrastive loss Eq. (5, 7) use the unnormalized representations instead of the normalized ones as it is simpler to analyze theoretically. For empirical experiments, we use normalized representations.

of augmented negative data samples $\mathbb{B} = \{\boldsymbol{D}'\boldsymbol{x}'\}$ with $\boldsymbol{x}' \neq \boldsymbol{x}$, the contrastive loss is defined as:

$$L_{CL}(\mathbf{W}) := \tag{8}$$
$$-\mathbb{E}\left[\log \frac{\exp\{\tau S^+\}}{\exp\{\tau S^+\} + \sum_{\boldsymbol{x}' \in \mathbb{B}} \exp\{\tau S^-\}}\right].$$

where

$$S^+ := \text{sim}(h(\boldsymbol{D}_1\boldsymbol{x}), h(\boldsymbol{D}_2\boldsymbol{x})),$$
$$S^- := \text{sim}(h(\boldsymbol{D}_1\boldsymbol{x}), h(\boldsymbol{D}_3'\boldsymbol{x}')),$$

and

$$\text{sim}(\boldsymbol{y}_1, \boldsymbol{y}_2)^\dagger := \langle \boldsymbol{y}_1, \boldsymbol{y}_2 \rangle = \boldsymbol{y}_1^\top \boldsymbol{y}_2$$

representing the similarity of two vectors $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ and τ is the temperature parameter and $\boldsymbol{D}_3'$ is an augmentation matrix for the negative sample $\boldsymbol{x}'$. In our theoretical results, we will assume that $|\mathbb{B}| \to \infty$.

4.2 Experimental set up

We provide details of our experimental setup here.

Dataset We use the data generating process described in section 4.1 to generate a synthetic dataset with 1000 data samples. The dictionary $\mathbf{M}$ is generated by applying QR decomposition on a randomly generated matrix $\boldsymbol{G} \in \mathbb{R}^{p \times d}$ whose entries are i.i.d standard Gaussian. The sparse coding latent variable $\boldsymbol{z} \in \{0, \pm 1\}^d$ is set to $\{-1, +1\}$ with an equal probability of $\beta/2$[‡]. The noise $\boldsymbol{\epsilon} \in \mathbb{R}^p$ is sampled i.i.d. from $\mathcal{N}\left(0, \frac{\log d}{d}\right)$.

Augmentation We try both augmentation schemes described in in Eq. 2 and in Eq. 3. Empirically, we find that the second augmentation scheme outperforms the first augmentation scheme. Therefore we report empirical results only on the second augmentation scheme.

Training setting We randomly initialize the weights of encoder and predictor networks for all the experiments (except for warm start). We sample each of the entries of these weight matrices from a Gaussian distribution $\mathcal{N}(0, \Theta(\frac{1}{pd}))$; and we always initialize the bias to be zero. For optimization, we use stochastic gradient descent (SGD) with a learning rate of 0.025 to train the model for 8000 epochs with batch size of 512. By default, the masking probability of random masks is 0.5, unless specified otherwise. In general, we use the dimensions m=50, p=50 and d=10, unless specified otherwise. As the latents $\boldsymbol{z} \in \{-1, 0, 1\}$, we use symmetric ReLU activation (Wen & Li (2021)) instead of the standard ReLU activation.

[‡]In addition, we also ensure that the sparse coding vectors $\mathbf{z}$ are generated such that at least one of the entries is non-zero.

5 RESULTS

In this section we formalize the differences between contrastive loss and non-contrastive loss under our simple model.

We first formalize the notion of *ground truth features* associated with our data generating process in Section 4.1. Specifically, we say that an encoder successfully encodes our data distribution if given the observed data point $\boldsymbol{x} = \boldsymbol{M}\boldsymbol{z} + \boldsymbol{\epsilon}$, the encoder is able recover the latent variable $\boldsymbol{z}$ up to permutation, i.e.

$$\text{ReLU}(\boldsymbol{W}^o x) \approx c\boldsymbol{P}\boldsymbol{z} \tag{9}$$

for some constant c and permutation matrix P. We will sometimes assume that $\mathbf{W}^o$ is normalized (i.e. has unit row norm), in which case we have $c = 1$. Note, the indeterminacy up to permutation is unavoidable: one can easily see that permuting both the latent coordinates and the matrix M correspondingly results in the same distribution for $\boldsymbol{x}$.

In the following sections, we present two lines of results that highlight some fundamental differences between contrastive and non-contrastive loss. Specifically, we conduct *landscape analysis* (Section 5.1), which focuses on the properties of global optima; and *training dynamics analysis* (Section 5.2), which focuses on the properties of the points that the training dynamics converge to. Both lines of analysis reveal the fragility of non-contrastive loss: its global optima contain bad optima, from which the training process cannot easily navigate through without careful architectural engineering — in particular, the inclusion of a predictor $\boldsymbol{W}^p$. We provide a combination of theoretical results with extensive experiments. The network architecture and the values of hyper-parameters cover a large spread for thoroughness, and we specify these wherever appropriate.

Evaluation metric Traditionally, the success of a self-supervised learning (SSL) algorithm is determined by evaluating the learnt representations of the encoder on a downstream auxiliary task. For example, the representations learnt on images are typically evaluated through a linear evaluation protocol (Kolesnikov et al. (2019); Bachman et al. (2019); Chen et al. (2020); Grill et al. (2020)) on a standard image classification datasets like ImageNet. A higher accuracy on the downstream tasks is indicative of the better quality of the learnt representations.

In our experimental setup, a self-supervised learning algorithm will be considered successful if we are able to find the following separation through the learnt weights

$\boldsymbol{W}$ of the encoder.

$$\langle \mathbf{W}_{i*}, \boldsymbol{M}_{*j} \rangle_{j \in \mathcal{N}_i} \gg \langle \mathbf{W}_{i*}, \boldsymbol{M}_{*j} \rangle_{j \in [d] \setminus \mathcal{N}_i} \tag{10}$$

where $\mathcal{N}_i \subseteq [d]$ is the subset of dictionary bases (i.e. $\{M_{*1}, \ldots, M_{*d}\}$) that neuron i (approximately) lies in.

Motivated by this goal, we propose an alternate approach to evaluate the success of a SSL algorithm under the sparse coding setup. Specifically, we assess the quality of learnt representations by computing the maximum, median and minimum values of the following expression,

$$\max_i \left| \left\langle \frac{\boldsymbol{W_{i*}}}{\|\boldsymbol{W_{i*}}\|_2}, \frac{\boldsymbol{M_{*j}}}{\|\boldsymbol{M_{*j}}\|_2} \right\rangle \right| \quad \forall j \in [m], \tag{11}$$

referred to as *Maximum max-cosine*, *Medium max-cosine* and *Minimum max-cosine* respectively, for each $j \in [d]$ and use these values to determine if the SSL algorithm has correctly recovered the ground truth dictionary matrix $\boldsymbol{M}$. Ideally, we want the SSL algorithm to learn the correct weights $\boldsymbol{W}$ of its linear encoder such that the values of the above dot product are close to 1 for all the three metrics as this would indicate near-perfect recovery of the ground truth support. Of these three metrics, intuitively high *Minimum max-cosine* is the most indicative of the success of a SSL algorithm, as it suggests that even the worst alignment of dictionary and neurons is good. We therefore include plots corresponding to *Minimum max-cosine* and training loss in the main paper, and defer the plots of the other metrics to Appendix A (for additional empirical results of Landscape analysis) and Appendix B (for additional empirical results of training dynamics analysis).

5.1 Landscape analysis

In this section, we provide results which show that non-contrastive loss has infinitely many non-collapsed global optima that are far from the ground truth. By contrast, contrastive learning loss guarantees recovery of the correct ground truth support. In particular, this happens even in the extremely simple setting in which there is no noise in the data generating process (i.e. $\sigma_0^2 = 0$) and $M = I$.

Precisely, we show:

Theorem 1 (Landscape of contrastive and non-contrastive loss)**.** *Let the data generating process and network architecture be specified as in Section 4, and consider the setting $d = p = m, \boldsymbol{M} = \mathbf{I}$, and $\sigma_0^2 = 0$. Moreover, let the latent vectors $\{\boldsymbol{z}_j \in \mathbb{R}^{d \times 1}\}_{j=1}^d$ be chosen by a uniform distribution over 1-sparse vectors.*

Then, we have:

(a) $\mathcal{U}_{\geq} \subseteq \operatorname{argmin}_{W \in \mathcal{U}} L_{non\text{-}CL}(W, W)$

(b) $\operatorname{argmin}_{W \in \mathcal{U}} L_{CL}(W, W)$ *is the set of permutation matrices.*

Part (a) of Theorem 1 shows, since any matrix $\boldsymbol{W} \in \mathcal{U}_{>}$ is a global optimum of the non-contrastive loss, optimizing it does not necessarily lead to learning of the groundtruth features described in equation 9. Indeed, there are clearly abundant elements $\boldsymbol{W} \in \mathcal{U}_{>}$ such that $\mathrm{ReLU}(\boldsymbol{W}x)$ and $c\boldsymbol{P}\boldsymbol{z}$ are very different by any measure. On the other hand, part (b) of Theorem 1 shows optimizing the contrastive loss objective guarantees learning the groundtruth features up to a permutation.

The proof of this theorem is deferred to Appendix A, though one core idea of the proof is relatively simple. For 1a), it's easy to see that so long as all the inputs to the ReLU are non-negative, the objective is minimized — from which the result easily follows. For 1b) the result follows by noting that in order to minimize the contrastive loss, it suffices that the columns of W are non-negative and orthogonal — which is only satisfied when W is a permutation matrix.

Remark 2. *For landscape analysis, it suffices to show the abundance of non-collapsed bad global optima. Therefore, for simplicity, we remove the bias and use the (asymmetric) ReLU activation. Note that without the bias, a symmetric ReLU would be the same as the identity function, which would lose the non-linearity.*

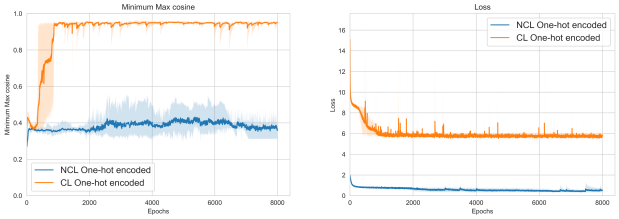

Figure 1: (NCL vs CL with one-hot latents) Minimum Max-Cosine (left) and Loss (right) curves for non-contrastive loss (NCL) and contrastive loss (CL) on an architecture with a randomly initialized linear encoder. We normalize the representations before computing the loss and use a symmetric ReLU after the linear encoder. The latent $\boldsymbol{z}$ is one-hot encoded. Reported numbers are averaged over 5 different runs.

5.2 Training dynamics analysis

5.2.1 Theoretical results

We analyze the training dynamics under two simple models, namely dual linear networks and dual ReLU networks.

First, for the dual linear networks, we explicitly write the optimization process and characterize the limita-

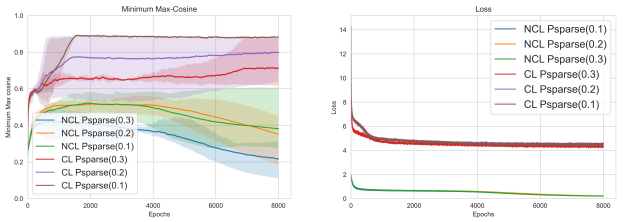

Figure 2: (NCL vs CL loss with k-sparse latents) Minimum Max-Cosine (left) and Loss (right) curves for non-contrastive loss (NCL) and contrastive loss (CL) on an architecture with a randomly initialized linear encoder. We include Batch Normalization (Ioffe & Szegedy (2015)) layer and symmetric ReLU activation after the linear encoder. Psparse indicates $Pr(\boldsymbol{z}_i = \pm 1), i \in [d]$ in the sparse coding vector $\boldsymbol{z}$. Reported numbers are averaged over 5 different runs.

tion of non-contrastive training process in learning the groundtruth even in our simple setting:

Theorem 2 (non-contrastive loss on linear network)**.** *Let the data generating process be specified as in Section 4, and let the online and target networks be linear, with weights initialized to $\boldsymbol{W}_0^o$ and $\boldsymbol{W}_0^t$, respectively. In step t, denote the learning rate as $\eta_t << 1$, and the weight-decay factor as $\lambda_t \in (0,1)$. Then, for the non-contrastive loss function*

$$L_{linear\text{-}non\text{-}CL}(\mathbf{W}^o, \mathbf{W}^t) := 2 - 2\mathbb{E}_{\boldsymbol{x},\boldsymbol{D}_1,\boldsymbol{D}_2}\langle \boldsymbol{W}^o\boldsymbol{D}_1\boldsymbol{x}, \boldsymbol{W}^t\boldsymbol{D}_2\boldsymbol{x}\rangle \tag{12}$$

the running gradient descent whose t-th step is

$$\boldsymbol{W}_{t+1}^o = \lambda_t \boldsymbol{W}_t^o - \eta_t \nabla_{\boldsymbol{W}_t^o} L_{linear\text{-}non\text{-}CL}(\boldsymbol{W}_t^o, \boldsymbol{W}_t^t)$$
$$\boldsymbol{W}_{t+1}^t = \lambda_t \boldsymbol{W}_t^t - \eta_t \nabla_{\boldsymbol{W}_t^t} L_{linear\text{-}non\text{-}CL}(\boldsymbol{W}_t^o, \boldsymbol{W}_t^t)$$

will lead to

$$\boldsymbol{W}_t^o\boldsymbol{M} = C_{1,t}\boldsymbol{W}_0^o\boldsymbol{M} + C_{2,t}(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M})$$
$$\boldsymbol{W}_t^t\boldsymbol{M} = C_{1,t}\boldsymbol{W}_0^t\boldsymbol{M} + C_{2,t}(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M})$$

for some scalars $C_{1,t} \in (0,1), C_{2,t} > 0$ depending on the number of steps t (if $\boldsymbol{W}_t^o$ and $\boldsymbol{W}_t^t$ neither explode nor vanish, which is controlled by η_t and λ_t).

Corollary 1 (limitation of dual linear networks)**.** *In the above setting, the learned encoders $\boldsymbol{W}_t^o$ and $\boldsymbol{W}_t^t$ do not have better max-cosine metrics (defined in equation 11) than the best linear combination of $\boldsymbol{W}^o$ and $\boldsymbol{W}^t$.*

The proof of this Theorem 2 is deferred to Appendix C.1.

Next, for the ReLU network with normalization case, under the non-contrastive loss

$$L(\boldsymbol{W}^o, \boldsymbol{W}^t) = 2 - 2\mathbb{E}_{\boldsymbol{x},\boldsymbol{D}_1,\boldsymbol{D}_2}\langle \mathrm{SReLU}_{\boldsymbol{b}^o}(\boldsymbol{W}^o\boldsymbol{D}_1\boldsymbol{x}), \mathrm{SReLU}_{\boldsymbol{b}^t}(\boldsymbol{W}^t\boldsymbol{D}_2\boldsymbol{x})\rangle \tag{13}$$

where $\mathrm{SReLU}_{\boldsymbol{b}}(\boldsymbol{x}) := ReLU(\boldsymbol{x} - \boldsymbol{b}) - ReLU(-\boldsymbol{x} - \boldsymbol{b})$, for some positive bias vector $\boldsymbol{b}$ i.e.

$$(\mathrm{SReLU}_{\boldsymbol{b}}(\boldsymbol{x}))_i = \begin{cases} x_i - b_i, & \text{if } x_i > b_i \\ 0, & \text{if } -b_i \le x_i \le b_i \\ x_i + b_i, & \text{if } x_i < -b_i \end{cases} \tag{14}$$

We study the case $\boldsymbol{M} = \boldsymbol{I}$, under a series of assumptions that we state formally and discuss their significance in the appendix (Appendix C.2). Our following theorem characterizes the convergence point of alternating optimization on the non-contrastive loss:

Theorem 3 (ReLU network with normalization, warm start)**.** *Under Assumptions 1, 2, 3, 4 in Appendix C.2, running:*

Repeat until both $\boldsymbol{W}^o$ and $\boldsymbol{W}^t$ converges:

- *Repeat until $\boldsymbol{W}^o$ converges:*

  $$\boldsymbol{W}^o \leftarrow normalize(\boldsymbol{W}^o - \eta \nabla_{\boldsymbol{W}^o} L(\boldsymbol{W}^o, \boldsymbol{W}^t))$$

- *Repeat until $\boldsymbol{W}^t$ converges:*

  $$\boldsymbol{W}^t \leftarrow normalize(\boldsymbol{W}^t - \eta \nabla_{\boldsymbol{W}^t} L(\boldsymbol{W}^o, \boldsymbol{W}^t))$$

will make $\boldsymbol{W}^o$ and $\boldsymbol{W}^t$ both converge to I

The proof is deferred to Section C.3.

5.2.2 Experimental results

We analyze properties of the training dynamics via extensive experimental results. We follow the set up described in section 4.2.

NCL from random initialization First, we empirically show that the training dynamics for the non-contrastive loss (Eq. 6) fails to learn the correct ground truth representations whereas contrastive loss (Eq. 8) is able to successfully learn the correct representations. We include plots of *Minimum Max-Cosine* and training loss in Figure 1 and 2. Both of these figures show that when encoder is a one layer ReLU network, NCL has lower values of *Minimum Max-Cosine* which indicates that it has failed to recover the correct support of ground truth dictionary $\boldsymbol{M}$, whereas CL is able to learn good representations across different levels of sparsity as shown by its high values of *Minimum Max-Cosine*.

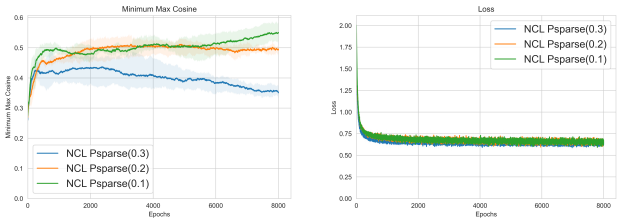

Figure 3: (NCL with 2 layered encoder does not work) Minimum Max-Cosine (left) and Loss (right) curves for non-contrastive loss (NCL) with a two-layered linear encoder with batch-normalization and symmetric ReLU activation. Psparse indicates $Pr(\boldsymbol{z}_i = \pm 1), i \in [d]$ in the sparse coding vector $\boldsymbol{z}$. Reported numbers are averaged over 5 different runs.

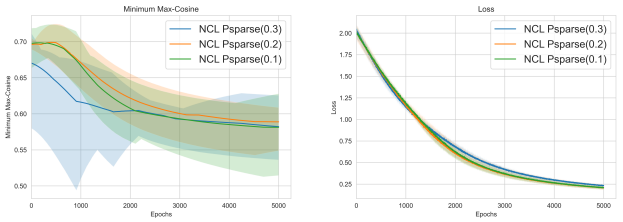

Figure 4: (NCL with warm start does not work) Minimum Max-Cosine (left) and Loss (right) curves for non-contrastive loss (NCL) with a warm-started linear encoder. Reported numbers are averaged over 5 different runs.

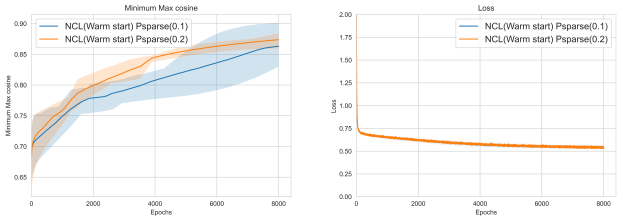

Figure 5: (NCL with a linear prediction head and warm-start works) Minimum Max-Cosine (left) and Loss (right) curves for non-contrastive loss (NCL) with warm-started linear encoder and a linear predictor. Reported numbers are averaged over 3 different runs.

We include additional plots for *Maximum Max-Cosine* and *Median Max-Cosine* in Appendix A. Finally, from Figure 3, we note that introducing additional linear layers in the base encoder does not aid non-contrastive loss to learn the correct ground truth representations.

Next, we provide several experiments that demonstrate the importance of including a *prediction head* $\boldsymbol{W}_p$. Namely, we will consider starting the training dynamics from a *warm start* — i.e. a point nearby the optimum. Precisely, we initialize the columns of the matrix $\boldsymbol{W}$ with random columns of the dictionary matrix $\boldsymbol{M}$ and add Gaussian noise $\mathcal{N}(0, \sigma^2 \mathbf{I})$ with $\sigma = \frac{1}{p^{c/2}}$ where the choice of σ determines the closeness of the matrix $\boldsymbol{W}$ to the dictionary matrix. For the results in this section, we use $c = 1$. As we use $p = 50$ for these experiments, we have $\sigma \approx 0.141$. We include additional results for other values of σ in Appendix B.

NCL with warm start fails to learn good representations We provide empirical evidence that even with warm-start, non-contrastive loss without a prediction head $\boldsymbol{W}_p$ fails to learn the correct ground truth representations. Figure 4 shows that for non-contrastive loss with warm start, *Minimum Max-Cosine* decreases as the training proceeds which points towards failure of this model to learn good representations.

NCL with warm start and prediction head learns good representations On the other hand, from Figure 5, we can conclude that if we do include a linear predictor, from a warm start non-contrastive loss objective can learn the correct ground truth representations. We observe that *Minimum Max-Cosine* increases on average from around 0.7 to 0.87 as the training proceeds. Finally, Figure 6 shows that in the absence of "warm-started" encoder, even with a predictor, NCL without a projection head fails to learn the ground truth representations. This experiment, together with Figure 6 indicate that the inclusion of a predictor head W_p is of paramount importance to training; however even with the inclusion of the predictor head, the training dynamics seem unable to escape the plethora of bad minima of the optimization landscape. We provide some theoretical evidence for these phenomena as well in Appendix B.

As a technical implementation detail, we note for all the experiments in this subsection the encoder includes a batch normalization layer. There is no symmetric ReLU activation or Batch Normalization layer after the predictor.

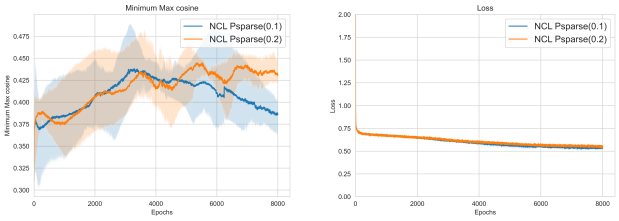

Figure 6: (NCL with a linear prediction head does not work without warm-start) Minimum Max-Cosine (left) and Loss (right) curves for non-contrastive loss (NCL) with non warm-started linear encoder and a linear predictor. Reported numbers are averaged over 3 different runs.

Model	Pr(sparse)	Maximum Max cosine ↑	Median Max cosine ↑	Minimum Max cosine ↑
Simplified-SimCLR	0.1	0.94 ± 0.005	0.92 ± 0.004	0.83 ± 0.16
Simplified-SimCLR	0.2	0.94 ± 0.004	0.93 ± 0.003	0.90 ± 0.01
Simplified-SimCLR	0.3	0.94 ± 0.006	0.92 ± 0.007	0.57 ± 0.019
Simplified-SimSiam	0.1	0.94 ± 0.007	0.87 ± 0.08	0.47 ± 0.07
Simplified-SimSiam	0.2	0.93 ± 0.008	0.82 ± 0.01	0.45 ± 0.005
Simplified-SimSiam	0.3	0.678 ± 0.2	0.45 ± 0.01	0.37 ± 0.03

Table 1: Comparison of the cosine values learnt by the simplified-SimCLR and simplified-SimSiam. Pr(sparse) indicates the probability $Pr(\boldsymbol{z}_i = \pm 1), i \in [d]$ in the sparse coding vector $\boldsymbol{z}$. We report mean $\pm$ std. deviation over 5 runs. ↑ symbol indicates that the higher value is better for the associated metric. Note that we sample the diagonal entries in random masks from $Bernoulli(0.1)$ for these experiments.

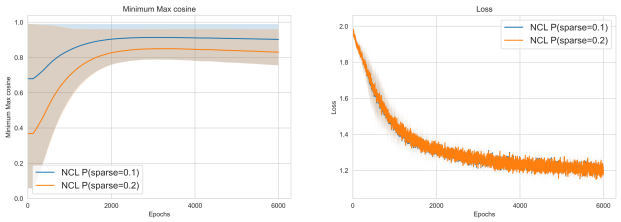

Figure 7: (NCL [Eq. 6] with warm-start and row normalized encoders $\mathbf{W}^o$, $\mathbf{W}^t$ works even in the absence of a prediction head) Minimum Max-Cosine (left) and Loss (right) curves for non-contrastive loss (NCL) with warm-started linear encoder. Reported numbers are averaged over 3 different runs. The warm start parameter $c = 2$ and probability of random masking is 0.9

NCL with warm start, and row or column normalized encoder learns good representations Another interesting observation from our experiments is that we can skip a linear predictor if the encoder is sufficiently "warm-started", and the rows or columns of the encoder $\mathbf{W}^o$ and $\mathbf{W}^t$ are normalized after every gradient descent update. In Figure 7, where we normalize the rows of encoder, we observe that *Minimum Max-Cosine* increases on average from around 0.62 to 0.85 as the training proceeds. We include some additional results with column normalization of the weights of encoder in Appendix B. Note that we also normalize the outputs of the encoder in these experiments.

Robustness across architectures and loss functions In the previous subsections, we used a relatively simple architecture for contrastive and non-contrastive learning. Our empirical observations in the previous sections generalize to the cases where we use a slightly more complex architecture and a different variant of loss. Table 1 lists the cosine values learnt by two architectures *Simplified-SimCLR* and *Simplified-Simsiam*. *Simplified-SimCLR* uses a linear projector in addition to a linear encoder, and optimizes the normalized temperature-scaled cross entropy loss (Chen et al. (2020)). *Simplified-Simsiam* architecture uses a linear predictor, and minimizes negative cosine similarity as proposed in (Chen & He (2021)). The encoders of both the architectures are randomly initialized. Both the architectures use batch-normalization layer and ReLU activation after the encoder. We observe that *Simplified-SimCLR* successfully recovers the ground-truth support but *Simplified-Simsiam* fails to do so, as evident by its low value of *Minimum Max-Cosine*. This is consistent with our observations in previous subsections.

6 CONCLUSION

In this work, we present some interesting theoretical results that highlight some fundamental differences in the representations learnt by contrastive learning and non-contrastive learning loss objectives in the sparse coding model setting, provided that the encoder architecture is fixed. We use a simple dual network architecture with ReLU activation. We theoretically prove that in this setting, non-contrastive loss objective has an ample amount of non-collapsed global minima that might not learn the correct ground truth features and therefore fail to recover the correct ground truth dictionary matrix. In contrast, optimizing the contrastive loss objective guarantees recovery of the correct ground truth dictionary matrix. We provide additional empirical results which show that even non-contrastive training process cannot avoid these bad non-collapsed global minima. We then empirically show that using warm-start and a linear predictor aids non-contrastive loss to learn the correct ground truth representations. While we worked in a relatively simple setting, we unearthed some fundamental key differences in the quality of representations learnt by contrastive and non-contrastive loss objectives. We hope that these results will motivate further studies to understand the qualitative and quantitative differences in representations learnt by contrastive and non-contrastive loss objectives.

ACKNOWLEDGEMENTS

The authors would like to thank Bingbin Liu for helpful comments and a careful proofread.

References

Sanjeev Arora, Rong Ge, Tengyu Ma, and Ankur Moitra. Simple, efficient, and neural algorithms for sparse coding. In Peter Grünwald, Elad Hazan, and Satyen Kale (eds.), *Proceedings of The 28th Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pp. 113–149, Paris, France, 03–06 Jul 2015. PMLR. URL https://proceedings.mlr.press/v40/Arora15.html.

Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.

Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15750–15758, June 2021.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

John M Giorgi, Osvald Nitski, Gary D Bader, and Bo Wang. Declutr: Deep contrastive learning for unsupervised textual representations. *arXiv preprint arXiv:2006.03659*, 2020.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 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.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

Olivier Henaff. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Machine Learning*, pp. 4182–4192. PMLR, 2020.

Tianyu Hua, Wenxiao Wang, Zihui Xue, Sucheng Ren, Yue Wang, and Hang Zhao. On feature decorrelation in self-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9598–9608, 2021.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR, 2015.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=YevsQ05DEN7.

Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1920–1929, 2019.

Michael S Lewicki and Terrence J Sejnowski. Learning overcomplete representations. *Neural computation*, 12(2):337–365, 2000.

Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*, 2020.

Zhuang Ma and Michael Collins. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3698–3707, 2018.

Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6707–6717, 2020.

Bruno A. Olshausen and David J. Field. Sparse coding with an overcomplete basis set: A

strategy employed by v1? *Vision Research*, 37(23):3311–3325, 1997. ISSN 0042-6989. doi: https://doi.org/10.1016/S0042-6989(97)00169-7. URL https://www.sciencedirect.com/science/article/pii/S0042698997001697.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Pierre H Richemond, Jean-Bastien Grill, Florent Altché, Corentin Tallec, Florian Strub, Andrew Brock, Samuel Smith, Soham De, Razvan Pascanu, Bilal Piot, et al. Byol works even without batch statistics. *arXiv preprint arXiv:2010.10241*, 2020.

Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 1857–1865, 2016.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 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.

Yuandong Tian, Lantao Yu, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning with dual deep networks. *arXiv preprint arXiv:2010.00578*, 2020b.

Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. *Proceedings of the 38th International Conference on Machine Learning*, 18–24 Jul 2021.

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

Xiang Wang, Xinlei Chen, Simon S Du, and Yuandong Tian. Towards demystifying representation learning with non-contrastive self-supervision. *arXiv preprint arXiv:2110.04947*, 2021.

Zixin Wen and Yuanzhi Li. Toward understanding the feature learning process of self-supervised contrastive learning. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11112–11122. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/wen21c.html.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3733–3742, 2018.

Mang Ye, Xu Zhang, Pong C Yuen, and Shih-Fu Chang. Unsupervised embedding learning via invariant and spreading instance feature. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6210–6219, 2019.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*, 2021.

Roland S Zimmermann, Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. Contrastive learning inverts the data generating process. In *International Conference on Machine Learning*, pp. 12979–12990. PMLR, 2021.

Supplementary Material: Contrasting the landscape of contrastive and non-contrastive learning

A LANDSCAPE ANALYSIS

A.1 Proof for Theorem 1

Theorem (Landscape of contrastive and non-contrastive loss (Theorem 1 restated))**.** *Let the data generating process and network architecture be specified as in Section 4, and consider the setting* $d = p = m, \boldsymbol{M} = \mathbf{I}$*, and* $\sigma_0^2 = 0$*. Moreover, let the latent vectors* $\{\boldsymbol{z}_j \in \mathbb{R}^{d\times 1}\}_{j=1}^d$ *be chosen by a uniform distribution over 1-sparse vectors.*

Then, we have:

(a) $\mathcal{U}_{\geq} \subseteq \operatorname{argmin}_{W\in\mathcal{U}} L_{non\text{-}CL}(W, W)$

(b) $\operatorname{argmin}_{W\in\mathcal{U}} L_{CL}(W, W)$ *is the set of permutation matrices.*

Notation : Recall that $\mathcal{U} := \{V \in \mathbb{R}^{m\times p} : \|V_{*j}\|_2 = 1\}$; $\mathcal{U}_{\geq} := \{V \in \mathcal{U}, V \geq 0\}$ We will also denote by $\boldsymbol{e}_j \in \mathbb{R}^{p\times 1}$ the vector which is one at j-th entry and zero elsewhere.

Proof. We proceed to claim (a) first. By the definition of L_{non-CL}, we have

$$\begin{aligned}\operatorname{argmin}_{\boldsymbol{W}\in\mathcal{U}} L_{non-CL}(\boldsymbol{W}, \boldsymbol{W}) &= \operatorname{argmin}_{\boldsymbol{W}\in\mathcal{U}} \sum_{j\in[d]} (2 - 2\mathbb{E}_{\boldsymbol{D}_1,\boldsymbol{D}_2}\langle Relu(\boldsymbol{W}\boldsymbol{D}_1\boldsymbol{e}_j), Relu(\boldsymbol{W}\boldsymbol{D}_2\boldsymbol{e}_j)\rangle) \\ &= \sum_{j\in[d]} \operatorname{argmax}_{\boldsymbol{W}\in\mathcal{U}} \mathbb{E}_{\boldsymbol{D}_1,\boldsymbol{D}_2}\langle Relu(\boldsymbol{W}\boldsymbol{D}_1\boldsymbol{e}_j), Relu(\boldsymbol{W}\boldsymbol{D}_2\boldsymbol{e}_j)\rangle\end{aligned}$$

Furthermore, we have

$$\begin{aligned}\langle Relu(\boldsymbol{W}\boldsymbol{D}_1\boldsymbol{e}_j), Relu(\boldsymbol{W}\boldsymbol{D}_2\boldsymbol{e}_j)\rangle &= \boldsymbol{D}_{1,jj}\boldsymbol{D}_{2,jj}\langle ReLU(\boldsymbol{W}_{*j}), ReLU(\boldsymbol{W}_{*j})\rangle \\ &\leq^{(1)} \boldsymbol{D}_{1,jj}\boldsymbol{D}_{2,jj}\|\boldsymbol{W}_{*j}\|^2 \\ &= \boldsymbol{D}_{1,jj}\boldsymbol{D}_{2,jj}\end{aligned}$$

where (1) follows since $(ReLU(x))^2 \leq x^2, \forall x \in \mathbb{R}$ and the last equality follows since for any $\boldsymbol{W} \in \mathcal{U}$, we have $\|\boldsymbol{W}_{*j}\| = 1$. Hence,

$$\max_{\boldsymbol{W}\in\mathcal{U}} \sum_{j\in[d]} \mathbb{E}_{\boldsymbol{D}_1,\boldsymbol{D}_2}\langle Relu(\boldsymbol{W}\boldsymbol{D}_1\boldsymbol{e}_j), Relu(\boldsymbol{W}\boldsymbol{D}_2\boldsymbol{e}_j)\rangle \leq \max_{\boldsymbol{W}\in\mathcal{U}} \sum_{j\in[d]} \mathbb{E}_{\boldsymbol{D}_1,\boldsymbol{D}_2}\boldsymbol{D}_{1,jj}\boldsymbol{D}_{2,jj} = d\alpha^2$$

Moreover, for any $\boldsymbol{W} \geq 0$, the inequality (1) is an equality, as $Relu(x) = x$ — thus any such $\boldsymbol{W}$ is a maximum of the objective, which is what we wanted to show.

□

Next, we proceed to (b).

Recall our definition of contrastive loss:

$$L_{CL}(\mathbf{W}) := -\mathbb{E}\left[\log \frac{\exp\{\tau S^+\}}{\exp\{\tau S^+\} + \sum_{\boldsymbol{x}'\in\mathbb{B}} \exp\{\tau S^-\}}\right].$$

Assume that $|\mathbb{B}| \to \infty$, according to law of large numbers, we have

$$L_{CL}(\mathbf{W}) \approx -\mathbb{E}\left[\log \frac{\exp\{\tau S^+\}}{\exp\{\tau S^+\} + B\mathbb{E}\exp\{\tau S^-\}}\right],$$

where $B := |\mathbb{B}|$.

Plug in the definition of S^+ and S^-, we have

$$L_{CL}(\mathbf{W}) \approx -\frac{1}{d}\sum_{i\in[d]} \mathbb{E}_{D_1,D_2}\left[\log\left[\frac{\exp\{\tau\langle ReLU(\boldsymbol{W}\boldsymbol{D}_1^i\boldsymbol{e}_i), ReLU(\boldsymbol{W}\boldsymbol{D}_2^i\boldsymbol{e}_i)\}\rangle}{\sum_{j\in[d]}\exp\{\tau\langle ReLU(\boldsymbol{W}\boldsymbol{D}_1^i\boldsymbol{e}_i), ReLU(\boldsymbol{W}\boldsymbol{D}_2^j\boldsymbol{e}_j)\rangle\}}\right]\right] + \log\frac{B}{d} \tag{15}$$

We can drop the constant $\log\frac{B}{d}$ and $\frac{1}{d}$, and consider the surrogate loss function

$$\widetilde{L}_{CL}(\boldsymbol{W}) := -\sum_{i\in[d]} \mathbb{E}_{D_1,D_2}\left[\log\left[\frac{\exp\{\tau\langle ReLU(\boldsymbol{W}\boldsymbol{D}_1^i\boldsymbol{e}_i), ReLU(\boldsymbol{W}\boldsymbol{D}_2^i\boldsymbol{e}_i)\}\rangle}{\sum_{j\in[d]}\exp\{\tau\langle ReLU(\boldsymbol{W}\boldsymbol{D}_1^i\boldsymbol{e}_i), ReLU(\boldsymbol{W}\boldsymbol{D}_2^j\boldsymbol{e}_j)\rangle\}}\right]\right]. \tag{16}$$

Note that the minimizer of the L_{CL} is exactly the same as those of $\widetilde{L}_{CL}$.

Denote

$$\mathbb{A}_i^+ := \exp\{\tau\langle ReLU(\boldsymbol{W}\boldsymbol{D}_1^i\boldsymbol{e}_i), ReLU(\boldsymbol{W}\boldsymbol{D}_2^i\boldsymbol{e}_i)\rangle\},$$
$$\mathbb{A}_i^- := \sum_{j\neq i}\exp\{\tau\langle ReLU(\boldsymbol{W}\boldsymbol{D}_1^i\boldsymbol{e}_i), ReLU(\boldsymbol{W}\boldsymbol{D}_2^j\boldsymbol{e}_j)\rangle\},$$

then the term inside the summation of equation 16 corresponding to i can be rewritten as

$$\mathbb{B}_i := \log\frac{1}{1+\mathbb{A}_i^-/\mathbb{A}_i^+}.$$

First, we note that again we have:

$$\begin{aligned}\langle Relu(\boldsymbol{W}\boldsymbol{D}_1\boldsymbol{e}_i), Relu(\boldsymbol{W}\boldsymbol{D}_2\boldsymbol{e}_i)\rangle &= \boldsymbol{D}_{1,ii}\boldsymbol{D}_{2,ii}\langle ReLU(\boldsymbol{W}_{*i}), ReLU(\boldsymbol{W}_{*i})\rangle \\ &\leq^{(1)} \boldsymbol{D}_{1,ii}\boldsymbol{D}_{2,ii}\|\boldsymbol{W}_{*i}\|^2 \qquad (17)\\ &= \boldsymbol{D}_{1,ii}\boldsymbol{D}_{2,ii} \qquad (18)\end{aligned}$$

where (1) follows since $(ReLU(x))^2 \leq x^2, \forall x \in \mathbb{R}$ and the last equality follows since for any $\boldsymbol{W} \in \mathcal{U}$, we have $\|\boldsymbol{W}_{*i}\| = 1$.

Additionally, since $ReLU(x) \geq 0$ we have

$$\langle ReLU(\boldsymbol{W}\boldsymbol{D}_1^i\boldsymbol{e}_i), ReLU(\boldsymbol{W}\boldsymbol{D}_2^j\boldsymbol{e}_j)\rangle \geq 0 \tag{19}$$

Let us denote $g(\boldsymbol{D}_1^i, \boldsymbol{D}_2^i) := \exp\{\tau\left(\sum_j(\boldsymbol{D}_1^i)_{jj}\right)\left(\sum_j(\boldsymbol{D}_2^i)_{jj}\right)\}$. We have by equation 18

$$\widetilde{L}_{CL}(\boldsymbol{W}) \geq -\sum_{i\in[d]}\mathbb{E}_{D_2,D_2}\log\left(\frac{1}{1+(d-1)/g(\boldsymbol{D}_1^i,\boldsymbol{D}_2^i)}\right) \tag{20}$$

We claim that the only $\boldsymbol{W}$ for which equation 20 is satisfied with an equality (thus, they are minima of L_{CL}) are $\boldsymbol{W} = \boldsymbol{P}$, for a permutation matrix $\boldsymbol{P}$.

First, we show that such $\boldsymbol{W}$ have to satisfy $\boldsymbol{W}_{*i} \geq 0, \forall i$, where the inequality is understood to apply entrywise. Indeed, for the sake of contradiction, assume otherwise. Then, consider an $i \in [d]$, and a mask $\boldsymbol{D}_1^i$, s.t. $\boldsymbol{D}_{1,ii}^i = 1$. (Note, such a mask has a non-zero probability of occurring.) For this choice of $i, \boldsymbol{D}_1^i$, we have

$$\|ReLU(\boldsymbol{W}\boldsymbol{D}_1^i\boldsymbol{e}_i)\| = \|ReLU(\boldsymbol{W}_{*i})\| < \|\boldsymbol{W}_{*i}\| \tag{21}$$

since at least one element of $\boldsymbol{W}_{*i}$ is negative. Thus, equation 17 is a strict inequality, and thus equation 20 cannot yield an equality.

Next, we show that $\forall i \neq j, \langle \boldsymbol{W}_{*i}, \boldsymbol{W}_{*j} \rangle = 0$ (i.e. the vectors $\boldsymbol{W}_{*i}$ are orthogonal). Again, we proceed by contradiction. Let us assume that there exist i, j, s.t. $\langle \boldsymbol{W}_{*i}, \boldsymbol{W}_{*j} \rangle > 0$. (Note, as we concluded before, all coordinates of $\boldsymbol{W}_{*i}$ have to be nonnegative, so if the above inner product is non-zero, it has to be nonnegative.) Consider a pair of $i \neq j \in [d]$, and two masks $\boldsymbol{D}_1^i, \boldsymbol{D}_2^j$, s.t. $\boldsymbol{D}_{1,ii}^i = \boldsymbol{D}_{2,jj}^j = 1$. (Again, such masks occur with non-zero probability.) For such choices of $i, j, \boldsymbol{D}_1^i, \boldsymbol{D}_2^j$, we have

$$\langle ReLU(\boldsymbol{W}\boldsymbol{D}_1^i\boldsymbol{e}_i), ReLU(\boldsymbol{W}\boldsymbol{D}_2^j\boldsymbol{e}_j) \rangle = \langle \boldsymbol{W}_{*i}, \boldsymbol{W}_{*j} \rangle > 0$$

which implies that equation 19 cannot be satisfied with an inequality. Thus, again equation 20 cannot yield an equality.

Thus, we concluded that, in order to achieve the minimum of $\widetilde{L}_{CL}$ over $\boldsymbol{W} \in \mathcal{U}$, the vectors $\boldsymbol{W}_{*i}$ have to be nonnegative and orthonormal for all i. Then, we claim this means $\boldsymbol{W}$ is a permutation matrix. Indeed, assume otherwise — i.e. that there exists some row, for which two columns i, i' has a non-zero element in that row. But then, these columns have a stricly positive inner product, so cannot be orthogonal. Thus, $\boldsymbol{W}$ has to be a permutation matrix.

On the other hand, for $\boldsymbol{W}$ a permutation matrix, equation 19 and equation 17 is an equality, and consequently equation 20 is an equality. Thus, any permutation matrix is a minimum of L_{CL}. Altogether, this concludes the proof of the theorem.

A.2 Additional Empirical results for Landscape analysis

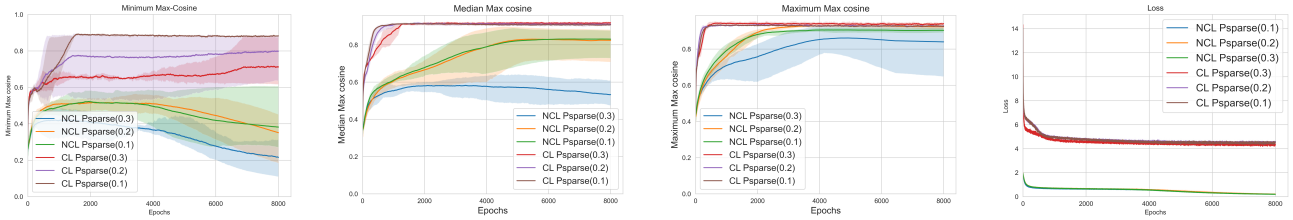

Figure 8: (NCL vs CL with k-sparse latents - Overparametrized network) (left-to-right) Minimum Max-Cosine, Median Max-Cosine, Maximum Max-Cosine, and Loss curves for non-contrastive loss (NCL) and contrastive loss (CL) on an architecture with a randomly initialized linear encoder. We normalize the representations before computing the loss and use a symmetric ReLU after the linear encoder. Reported numbers are averaged over 5 different runs. The shaded area represents the maximum and the minimum values observed across those 5 runs. We use p = 50, d=10, m=10.

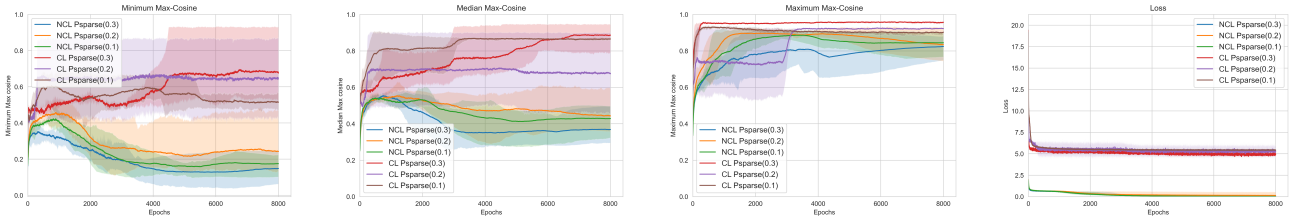

Figure 9: (NCL vs CL with k-sparse latents) (left-to-right) Minimum Max-Cosine, Median Max-Cosine, Maximum Max-Cosine, and Loss curves for non-contrastive loss (NCL) and contrastive loss (CL) on an architecture with a randomly initialized linear encoder. We normalize the representations before computing the loss and use a symmetric ReLU after the linear encoder. Reported numbers are averaged over 5 different runs. The shaded area represents the maximum and the minimum values observed across those 5 runs. We use p=50, d=10, m=10.

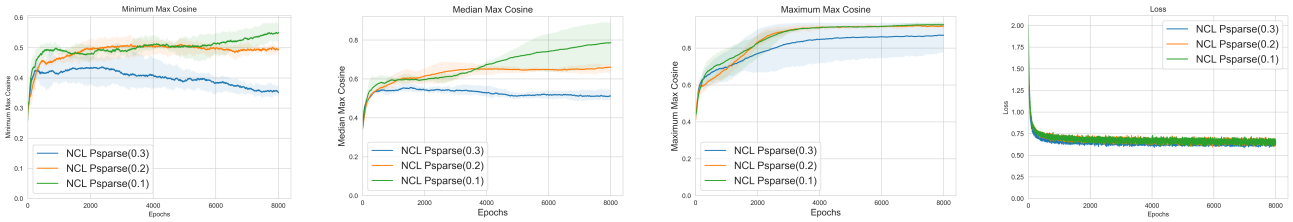

Figure 10: (NCL with 2 layered encoder does not work) (left to right) Minimum Max-Cosine, Median Max-Cosine, Maximum Max-Cosine and Loss curves for non-contrastive loss (NCL) with a two-layered linear encoder with batch-normalization and symmetric ReLU activation. Psparse indicates $Pr(\boldsymbol{z}_i = \pm 1), i \in [d]$ in the sparse coding vector $\boldsymbol{z}$. Reported numbers are averaged over 5 different runs. The shaded area represents the maximum and the minimum values observed across those 5 runs. We use p = 50, d=10, m=50.

A.2.1 Non contrastive loss collapses to bad minima

The results have been summarized in Table 2 and Table 1. We observe that the non-contrastive learning model *NCL-basic* fails to achieve good values for the metric *Minimum max-cosine* while *NCL-linear* fails to achieve good values on all the three metrics. In practice, we observe a collapse in *NCL-linear* as the weights of the encoder are driven to 0. This indicates that the simple encoder when training with non-CL loss objective fails to recover the correct ground-truth dictionary matrix $\boldsymbol{M}$. On the other hand, we observe that the encoders trained with CL loss objective *CL-linear* and *CL-basic* have high values for all the three metrics which indicates that they succeed in recovering most of the ground truth support. This raises a question *Does introducing more fully connected layers in the encoder help non-CL loss objective to learn better representations?* We test this in our model *NCL-basic-2L* that uses a two-layered encoder with batch norm and symmetric ReLU. We observe that even this model fails to match *Minimum max-cosine* values of a *CL-basic* which uses a single layered linear encoder with CL loss objective. We observe similar trends when we analyze slightly more complex architecture in Table 1 where *Simplified-SimCLR* successfully recovers the ground-truth support but *Simplified-Simsiam* fails to do so.

Model	BN	SymReLU	Pr(sparse)	Maximum Max cosine ↑	Median Max cosine ↑	Minimum Max cosine ↑
NCL-linear	×	×	0.1	0.12 ± 0.03	0.10 ± 0.01	0.08 ± 0.1
NCL-linear	×	×	0.2	0.11 ± 0.02	0.09 ± 0.03	0.08 ± 0.3
NCL-linear	×	×	0.3	0.31 ± 0.006	0.29 ± 0.007	0.28 ± 0.002
CL-linear	×	×	0.1	0.77 ± 0.01	0.69 ± 0.01	0.58 ± 0.01
CL-linear	×	×	0.2	0.78 ± 0.06	0.68 ± 0.02	0.59 ± 0.01
CL-linear	×	×	0.3	0.78 ± 0.04	0.67 ± 0.01	0.6 ± 0.03
NCL-basic	✓	✓	0.1	0.9 ± 0.01	0.83 ± 0.06	0.52 ± 0.06
NCL-basic	✓	✓	0.2	0.92 ± 0.01	0.82 ± 0.07	0.51 ± 0.03
NCL-basic	✓	✓	0.3	0.84 ± 0.11	0.58 ± 0.04	0.42 ± 0.06
NCL-basic-2L	✓	✓	0.1	0.94 ± 0.01	0.91 ± 0.01	0.66 ± 0.21
NCL-basic-2L	✓	✓	0.2	0.94 ± 0.009	0.92 ± 0.002	0.51 ± 0.11
NCL-basic-2L	✓	✓	0.3	0.83 ± 0.16	0.48 ± 0.04	0.36 ± 0.01
CL-basic	✓	✓	0.1	0.93 ± 0.007	0.91 ± 0.005	0.88 ± 0.002
CL-basic	✓	✓	0.2	0.93 ± 0.002	0.91 ± 0.009	0.79 ± 0.14
CL-basic	✓	✓	0.3	0.93 ± 0.01	0.86 ± 0.07	0.76 ± 0.18

Table 2: Comparison of the cosine values learnt by contrastive vs non-contrastive losses. The column BN indicates the presence of a batch normalization layer after linear layer in the encoder. SymReLU indicates whether the encoder uses symmetric ReLU activation. Pr(sparse) indicates the probability $Pr(\boldsymbol{z}_i = \pm 1), i \in [d]$ in the sparse coding vector $\boldsymbol{z}$. We report mean ± std. deviation over 5 runs. ↑ symbol indicates that the higher value is better for the associated metric.

Model	Pr(sparse)	Maximum Max cosine ↑	Median Max cosine ↑	Minimum Max cosine ↑
Simplified-SimCLR	0.1	0.94 ± 0.005	0.92 ± 0.004	0.83 ± 0.16
Simplified-SimCLR	0.2	0.94 ± 0.004	0.93 ± 0.003	0.90 ± 0.01
Simplified-SimCLR	0.3	0.94 ± 0.006	0.92 ± 0.007	0.57 ± 0.019
Simplified-SimSiam	0.1	0.94 ± 0.007	0.87 ± 0.08	0.47 ± 0.07
Simplified-SimSiam	0.2	0.93 ± 0.008	0.82 ± 0.01	0.45 ± 0.005
Simplified-SimSiam	0.3	0.678 ± 0.2	0.45 ± 0.01	0.37 ± 0.03

Table 3: Comparison of the cosine values learnt by the simplified-SimCLR and simplified-SimSiam. Pr(sparse) indicates the probability $Pr(\boldsymbol{z}_i = \pm 1), i \in [d]$ in the sparse coding vector $\boldsymbol{z}$. We report mean ± std. deviation over 5 runs. ↑ symbol indicates that the higher value is better for the associated metric. Note that we sample the diagonal entries in random masks from $Bernoulli(0.1)$ for these experiments.

B TRAINING DYNAMICS

In this section, we include additional empirical results that elucidate the training dynamics of contrastive and non-contrastive training.

B.1 Additional Empirical results for Training dynamics

B.1.1 Non-contrastive model needs a warm start and a linear prediction head to learn better representations

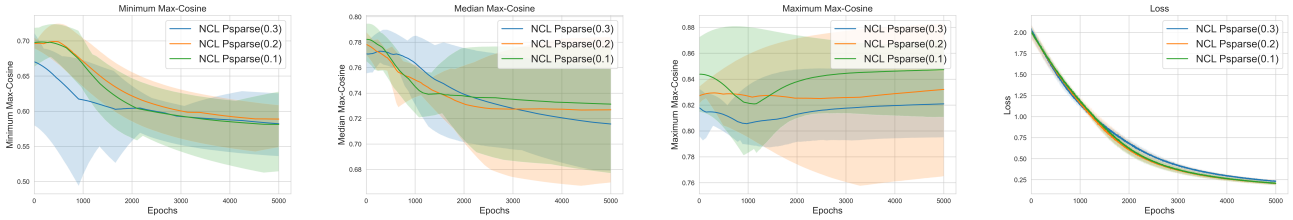

Figure 11: (NCL with warm start does not work) Minimum Max-Cosine (left) and Loss (right) curves for non-contrastive loss (NCL) with a warm-started linear encoder. Psparse indicates $Pr(\boldsymbol{z}_i = \pm 1), i \in [d]$ in the sparse coding vector $\boldsymbol{z}$. Reported numbers are averaged over 5 different runs. The shaded area represents the maximum and the minimum values observed across those 5 runs. We use p=50, m=50, d=10

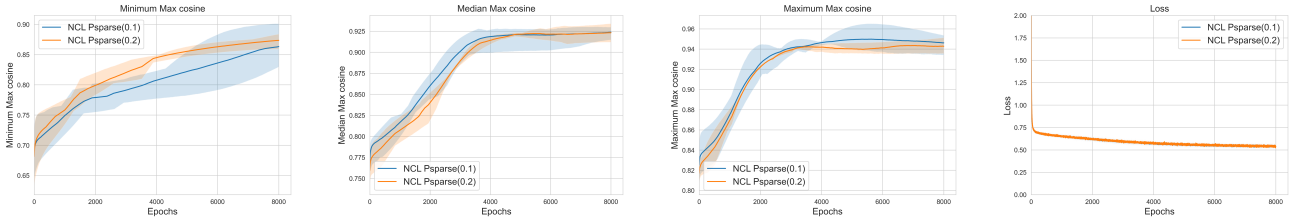

Figure 12: (NCL with a linear prediction head and warm-start works) (left-to-right) Minimum Max-Cosine, Median Max-Cosine, Maximum Max-Cosine, and Loss curves for non-contrastive loss (NCL) with warm-started linear encoder and a linear predictor. Reported numbers are averaged over 3 different runs. The shaded area represents the maximum and the minimum values observed across those 3 runs. We use p=50, m=50, d=10

B.1.2 Non-contrastive model with warm start learns good representations if weights of the encoder are row-normalized or column-normalized

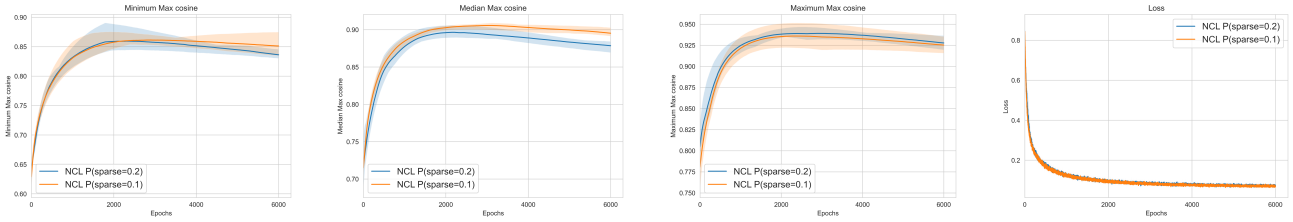

Figure 13: (NCL with warm-start and row-normalized encoder works) (left-to-right) Minimum Max-Cosine, Median Max-Cosine, Maximum Max-Cosine, and Loss curves for non-contrastive loss [BYOL Grill et al. (2020)] with warm-started linear encoder and row normalized encoder. Reported numbers are averaged over 3 different runs. The shaded area represents the maximum and the minimum values observed across those 3 runs. We use p=50, d=10, m=10. Warm start parameter $c = 1$.

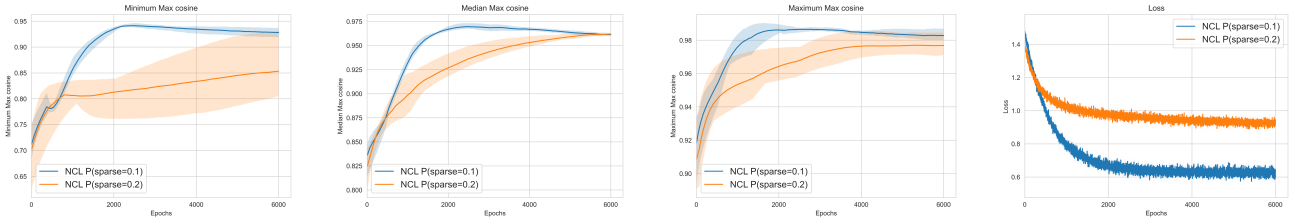

Figure 14: (NCL with warm-start and row-normalized encoder works) (left-to-right) Minimum Max-Cosine, Median Max-Cosine, Maximum Max-Cosine, and Loss curves for non-contrastive loss [BYOL Grill et al. (2020)] with warm-started linear encoder and row normalized encoder. Reported numbers are averaged over 3 different runs. The shaded area represents the maximum and the minimum values observed across those 3 runs. We use p=20, d=20, m=20. Warm start parameter $c = 1.25$.

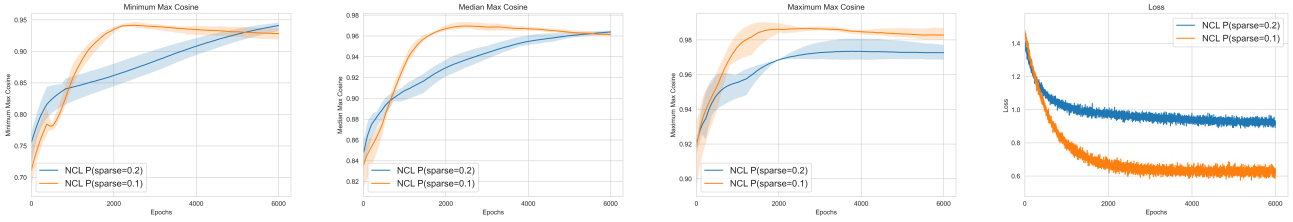

Figure 15: (NCL with warm-start and column-normalized encoder works) (left-to-right) Minimum Max-Cosine, Median Max-Cosine, Maximum Max-Cosine, and Loss curves for non-contrastive loss [BYOL Grill et al. (2020)] with warm-started linear encoder and column normalized encoder. Reported numbers are averaged over 3 different runs. The shaded area represents the maximum and the minimum values observed across those 3 runs. We use p=20, d=20, m=20. Warm start parameter $c = 1.25$ and probability of random masking is 0.5.

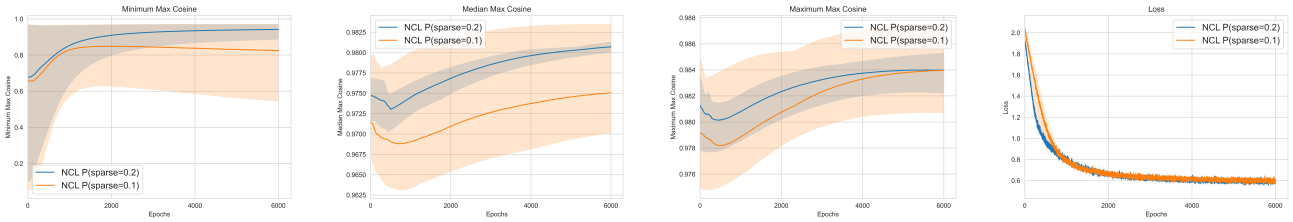

Figure 16: (NCL with warm-start and column-normalized encoder works) (left-to-right) Minimum Max-Cosine, Median Max-Cosine, Maximum Max-Cosine, and Loss curves for non-contrastive loss [BYOL Grill et al. (2020)] with warm-started linear encoder and column normalized encoder. Reported numbers are averaged over 3 different runs. The shaded area represents the maximum and the minimum values observed across those 3 runs. We use p=50, d=10, m=50. Warm start parameter $c = 2$ and probability of random masking is 0.75.

B.1.3 Robustness across architectures and loss functions

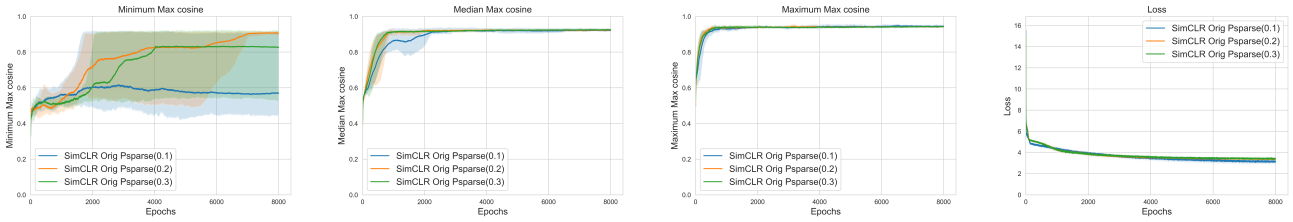

Figure 17: (Simplified SimCLR with random initialization learns good representations when the latents have low sparsity) (left-to-right) Minimum Max-Cosine, Median Max-Cosine, Maximum Max-Cosine, and Loss curves for Simplified SimCLR model discussed in Table 1. Reported numbers are averaged over 5 different runs. The shaded area represents the maximum and the minimum values observed across those 5 runs. We use p = 50, d=10, m=50. Probability of random masking is 0.1.

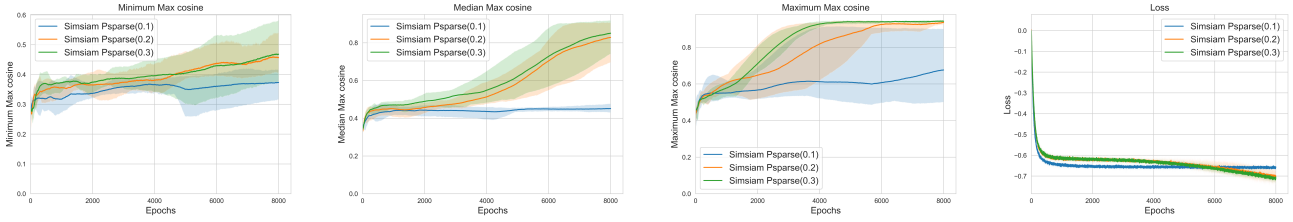

Figure 18: (Simplified Simsiam with random initialization fails to learn good representations) (left-to-right) Minimum Max-Cosine, Median Max-Cosine, Maximum Max-Cosine, and Loss curves for Simplified Simsiam model discussed in Table 1. Reported numbers are averaged over 5 different runs. The shaded area represents the maximum and the minimum values observed across those 5 runs. We use p = 50, m=50, d=10. Probability of random masking is 0.1.

B.1.4 Additional results for different levels of warm start

In our experiments we find that initializing the weights of the encoder $\boldsymbol{W}$ close to the dictionary matrix $\boldsymbol{M}$ helps the non-contrastive learning algorithm to learn better cosine values. For warm start, we initialize the columns of the matrix $\boldsymbol{W}$ with random columns of the dictionary matrix $\boldsymbol{M}$. In addition, we add Gaussian noise $\mathcal{N}(0, \sigma^2)I$ to the matrix $\boldsymbol{W}$ where the choice of σ determines the closeness of the matrix $\boldsymbol{W}$ to the dictionary matrix. The results of these experiments have been summarized in Table 4. As expected, a warmer start (corresponding to

smaller σ) helps in all settings of Pr(sparse) that we tried.

Model	Pr(sparse)	σ	Maximum Max cosine ↑	Median Max cosine ↑	Minimum Max cosine ↑
NCL-basic	0.1	0.14	0.84 ± 0.02	0.73 ± 0.04	0.58 ± 0.04
NCL-basic	0.1	0.02	0.97 ± 0.005	0.77 ± 0.01	0.69 ± 0.04
NCL-basic	0.1	0.002	0.98 ± 0.003	0.78 ± 0.04	0.69 ± 0.02
NCL-basic	0.2	0.14	0.84 ± 0.04	0.73 ± 0.05	0.57 ± 0.05
NCL-basic	0.2	0.02	0.97 ± 0.006	0.79 ± 0.07	0.69 ± 0.03
NCL-basic	0.2	0.002	0.97 ± 0.006	0.81 ± 0.08	0.67 ± 0.03
NCL-basic	0.3	0.14	0.82 ± 0.02	0.71 ± 0.03	0.58 ± 0.03
NCL-basic	0.3	0.02	0.95 ± 0.009	0.77 ± 0.06	0.66 ± 0.01
NCL-basic	0.3	0.002	0.96 ± 0.02	0.75 ± 0.04	0.66 ± 0.02

Table 4: Summary of cosine values learnt by the simple linear encoder by non contrastive algorithm with an overparameterized linear encoder (with batch normalization and symmetric ReLU) when the weights of the encoder $\boldsymbol{W}$ are initialized close to the ground truth dictionary $\boldsymbol{M}$. Pr(sparse) indicates the probability $Pr(\boldsymbol{z}_i = \pm 1), i \in [d]$ in the sparse coding vector $\boldsymbol{z}$. σ denotes the std. deviation of the Gaussian noise added to $\boldsymbol{W}$ that is initialized with random columns of the ground truth dictionary matrix $\boldsymbol{M}$. We report mean $\pm$ std. deviation over 5 runs.

C THEORETICAL RESULTS FOR TRAINING DYNAMICS

C.1 Proof of the limitation of linear networks

Theorem (non-contrastive loss on linear network, Theorem 2 restated)**.** *Let the data generating process be specified as in Section 4, and let the online and target networks be linear, with weights initialized to $\boldsymbol{W}_0^o$ and $\boldsymbol{W}_0^t$, respectively. In step t, denote the learning rate as $\eta_t << 1$, and the weight-decay factor as $\lambda_t \in (0, 1)$. Then, for the non-contrastive loss function*

$$\begin{aligned} &L_{\textit{linear-non-CL}}(\mathbf{W}^o, \mathbf{W}^t) \\ &:= 2 - 2\mathbb{E}_{\boldsymbol{x}, \boldsymbol{D}_1, \boldsymbol{D}_2} \langle \boldsymbol{W}^o \boldsymbol{D}_1 \boldsymbol{x}, \boldsymbol{W}^t \boldsymbol{D}_2 \boldsymbol{x} \rangle \end{aligned} \tag{22}$$

running gradient descent whose t-th step is

$$\begin{aligned} \boldsymbol{W}_{t+1}^o &= \lambda_t \boldsymbol{W}_t^o - \eta_t \nabla_{\boldsymbol{W}_t^o} L_{\textit{linear-non-CL}}(\boldsymbol{W}_t^o, \boldsymbol{W}_t^t) \\ \boldsymbol{W}_{t+1}^t &= \lambda_t \boldsymbol{W}_t^t - \eta_t \nabla_{\boldsymbol{W}_t^t} L_{\textit{linear-non-CL}}(\boldsymbol{W}_t^o, \boldsymbol{W}_t^t) \end{aligned}$$

results in matrices $\boldsymbol{W}^o, \boldsymbol{W}^t$ satisfying

$$\begin{aligned} \boldsymbol{W}_t^o \boldsymbol{M} &= C_{1,t} \boldsymbol{W}_0^o \boldsymbol{M} + C_{2,t} (\boldsymbol{W}_0^o \boldsymbol{M} + \boldsymbol{W}_0^t \boldsymbol{M}) \\ \boldsymbol{W}_t^t \boldsymbol{M} &= C_{1,t} \boldsymbol{W}_0^t \boldsymbol{M} + C_{2,t} (\boldsymbol{W}_0^o \boldsymbol{M} + \boldsymbol{W}_0^t \boldsymbol{M}) \end{aligned}$$

for some scalars $C_{1,t} \in (0, 1), C_{2,t} > 0$ depending on the number of steps t (if $\boldsymbol{W}_t^o$ and $\boldsymbol{W}_t^t$ neither explode nor vanish, which is controlled by η_t and λ_t).

Proof. When the ReLU works in the identity region, the loss function

$$\begin{aligned} L_{\text{non-contrastive}}(\boldsymbol{W}^o, \boldsymbol{W}^t) &= 2 - 2\mathbb{E}_{\boldsymbol{x}, \boldsymbol{D}_1, \boldsymbol{D}_2} \langle \textit{Relu}(\boldsymbol{W}^o \boldsymbol{D}_1 \boldsymbol{x}), \textit{Relu}(\boldsymbol{W}^t \boldsymbol{D}_2 \boldsymbol{x}) \rangle \\ &= 2 - 2\mathbb{E}_{\boldsymbol{x}, \boldsymbol{D}_1, \boldsymbol{D}_2} \langle \boldsymbol{W}^o \boldsymbol{D}_1 \boldsymbol{x}, \boldsymbol{W}^t \boldsymbol{D}_2 \boldsymbol{x} \rangle \end{aligned}$$

Differentiating $L_{\text{non-contrastive}}$ gives

$$\begin{aligned}
\nabla_{\boldsymbol{W}^o} L_{\text{non-contrastive}}(\boldsymbol{W}^o, \boldsymbol{W}^t) &= -2\mathbb{E}_{\boldsymbol{x},\boldsymbol{D}_1,\boldsymbol{D}_2}[(\boldsymbol{W}^t\boldsymbol{D}_2\boldsymbol{x})(\boldsymbol{D}_1\boldsymbol{x})^\top] \\
&= -2\mathbb{E}_{\boldsymbol{x},\boldsymbol{D}_1,\boldsymbol{D}_2}[(\boldsymbol{W}^t\boldsymbol{D}_2\boldsymbol{x})\boldsymbol{x}^\top\boldsymbol{D}_1] \\
&= -2\mathbb{E}_{\boldsymbol{x}}[(\boldsymbol{W}^t\mathbb{E}_{\boldsymbol{D}_2}[\boldsymbol{D}_2]\boldsymbol{x})\boldsymbol{x}^\top\mathbb{E}_{\boldsymbol{D}_1}[\boldsymbol{D}_1]] \\
&= -2\mathbb{E}_{\boldsymbol{x}}[\boldsymbol{W}^t(\alpha\boldsymbol{I})\boldsymbol{x}\boldsymbol{x}^\top(\alpha\boldsymbol{I})] \\
&= -2\alpha^2\boldsymbol{W}^t\mathbb{E}_{\boldsymbol{x}}[\boldsymbol{x}\boldsymbol{x}^\top] \\
&= -2\alpha^2\boldsymbol{W}^t\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon}}[(\boldsymbol{M}\boldsymbol{z}+\boldsymbol{\epsilon})(\boldsymbol{M}\boldsymbol{z}+\boldsymbol{\epsilon})^\top] \\
&= -2\alpha^2\boldsymbol{W}^t\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon}}[(\boldsymbol{M}\boldsymbol{z}+\boldsymbol{\epsilon})(\boldsymbol{z}^\top\boldsymbol{M}^\top+\boldsymbol{\epsilon}^\top)] \\
&= -2\alpha^2\boldsymbol{W}^t\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon}}[\boldsymbol{M}\boldsymbol{z}\boldsymbol{z}^\top\boldsymbol{M}^\top+\boldsymbol{\epsilon}\boldsymbol{z}^\top\boldsymbol{M}^\top+\boldsymbol{M}\boldsymbol{z}\boldsymbol{\epsilon}^\top+\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top] \\
&= -2\alpha^2\boldsymbol{W}^t\left(\boldsymbol{M}\mathbb{E}_{\boldsymbol{z}}[\boldsymbol{z}\boldsymbol{z}^\top]\boldsymbol{M}^\top+\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon}}[\boldsymbol{\epsilon}\boldsymbol{z}^\top\boldsymbol{M}^\top]+\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon}}[\boldsymbol{M}\boldsymbol{z}\boldsymbol{\epsilon}^\top]+\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top]\right)
\end{aligned}$$

Since the data generating model specifies

$$z_i = \begin{cases} 1, & \text{with probability } p_z \\ -1, & \text{with probability } p_z \\ 0, & \text{with probability } 1-2p_z \end{cases}$$

and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma_0^2\mathbf{I}_p)$, we get

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{z}}[\boldsymbol{z}\boldsymbol{z}^\top] &= 2p_z\boldsymbol{I} \\
\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon}}[\boldsymbol{\epsilon}\boldsymbol{z}^\top\boldsymbol{M}^\top] &= \mathbf{0} \\
\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon}}[\boldsymbol{M}\boldsymbol{z}\boldsymbol{\epsilon}^\top] &= \mathbf{0} \\
\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top] &= \sigma_0^2\mathbf{I}_p
\end{aligned}$$

Therefore

$$\begin{aligned}
\nabla_{\boldsymbol{W}^o} L_{\text{non-contrastive}}(\boldsymbol{W}^o, \boldsymbol{W}^t) &= -2\alpha^2\boldsymbol{W}^t\left(2p_z\boldsymbol{M}\boldsymbol{M}^\top+\sigma_0^2\mathbf{I}_p\right) \\
&= -2\alpha^2(2p_z\boldsymbol{W}^t\boldsymbol{M}\boldsymbol{M}^\top+\sigma_0^2\boldsymbol{W}^t)
\end{aligned}$$

Let $\eta_t \ll 1$ be the learning rate, $(\boldsymbol{W}_0^o, \boldsymbol{W}_0^t)$ be the initial values of $(\boldsymbol{W}^o, \boldsymbol{W}^t)$, and $(\boldsymbol{W}_t^o, \boldsymbol{W}_t^t)$ be their values after t gradient descent updates. Then

$$\begin{aligned}
\boldsymbol{W}_{t+1}^o &= \lambda_t\boldsymbol{W}_t^o - \eta_t\nabla_{\boldsymbol{W}_t^o} L_{\text{non-contrastive}}(\boldsymbol{W}_t^o, \boldsymbol{W}_t^t) \\
&= \lambda_t\boldsymbol{W}_t^o - \eta_t\left(-2\alpha^2(2p_z\boldsymbol{W}_t^t\boldsymbol{M}\boldsymbol{M}^\top+\sigma_0^2\boldsymbol{W}_t^t)\right) \\
&= \lambda_t\boldsymbol{W}_t^o + 2\eta_t\alpha^2(2p_z\boldsymbol{W}_t^t\boldsymbol{M}\boldsymbol{M}^\top+\sigma_0^2\boldsymbol{W}_t^t)
\end{aligned}$$

Therefore

$$\begin{aligned}
\boldsymbol{W}_{t+1}^o\boldsymbol{M} &= \lambda_t\boldsymbol{W}_t^o\boldsymbol{M} + 2\eta_t\alpha^2(2p_z\boldsymbol{W}_t^t\boldsymbol{M}\boldsymbol{M}^\top\boldsymbol{M}+\sigma_0^2\boldsymbol{W}_t^t\boldsymbol{M}) \\
&= \lambda_t\boldsymbol{W}_t^o\boldsymbol{M} + 2\eta_t\alpha^2(2p_z\boldsymbol{W}_t^t\boldsymbol{M}+\sigma_0^2\boldsymbol{W}_t^t\boldsymbol{M}) \\
&= \lambda_t\boldsymbol{W}_t^o\boldsymbol{M} + 2\eta_t\alpha^2(2p_z+\sigma_0^2)\boldsymbol{W}_t^t\boldsymbol{M}
\end{aligned}$$

Since $L_{\text{non-contrastive}}(\boldsymbol{W}^o, \boldsymbol{W}^t)$ is symmetric in $\boldsymbol{W}^o$ and $\boldsymbol{W}^t$, we also have

$$\boldsymbol{W}_{t+1}^t\boldsymbol{M} = \lambda_t\boldsymbol{W}_t^t\boldsymbol{M} + 2\eta_t\alpha^2(2p_z+\sigma_0^2)\boldsymbol{W}_t^o\boldsymbol{M}$$

Adding the above two equations gives

$$\begin{aligned}
\boldsymbol{W}_{t+1}^o\boldsymbol{M} + \boldsymbol{W}_{t+1}^t\boldsymbol{M} &= \lambda_t(\boldsymbol{W}_t^o\boldsymbol{M} + \boldsymbol{W}_t^t\boldsymbol{M}) + 2\eta_t\alpha^2(2p_z+\sigma_0^2)(\boldsymbol{W}_t^o\boldsymbol{M} + \boldsymbol{W}_t^t\boldsymbol{M}) \\
&= \left(\lambda_t + 2\eta_t\alpha^2(2p_z+\sigma_0^2)\right)(\boldsymbol{W}_t^o\boldsymbol{M} + \boldsymbol{W}_t^t\boldsymbol{M})
\end{aligned}$$

Let constant $c_t = 2\eta_t\alpha^2(2p_z + \sigma_0^2)$, then by recursion

$$\boldsymbol{W}_t^o\boldsymbol{M} + \boldsymbol{W}_t^t\boldsymbol{M} = \prod_{i=0}^{t-1}(\lambda_i + c_i)(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M})$$

Plugging into the expression for $\boldsymbol{W}_{t+1}^o\boldsymbol{M}$ above, we get

$$\begin{aligned}
\boldsymbol{W}_{t+1}^o\boldsymbol{M} &= \lambda_t\boldsymbol{W}_t^o\boldsymbol{M} + 2\eta_t\alpha^2(2p_z + \sigma_0^2)(\boldsymbol{W}_t^o\boldsymbol{M} + \boldsymbol{W}_t^t\boldsymbol{M} - \boldsymbol{W}_t^o\boldsymbol{M}) \\
&= \lambda_t\boldsymbol{W}_t^o\boldsymbol{M} + c_t\left(\prod_{i=0}^{t-1}(\lambda_i + c_i)(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M}) - \boldsymbol{W}_t^o\boldsymbol{M}\right) \\
&= (\lambda_t - c_t)\boldsymbol{W}_t^o\boldsymbol{M} + c_t\prod_{i=0}^{t-1}(\lambda_i + c_i)(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M})
\end{aligned}$$

Taking the recursion one step further, we get

$$\begin{aligned}
\boldsymbol{W}_{t+1}^o\boldsymbol{M} &= (\lambda_t - c_t)((\lambda_{t-1} - c_{t-1})\boldsymbol{W}_{t-1}^o\boldsymbol{M} + c_{t-1}\prod_{i=0}^{t-2}(\lambda_i + c_i)(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M})) \\
&\quad + c_t\prod_{i=0}^{t-1}(\lambda_i + c_i)(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M}) \\
&= (\lambda_t - c_t)(\lambda_{t-1} - c_{t-1})\boldsymbol{W}_{t-1}^o\boldsymbol{M} + c_{t-1}(\lambda_t - c_t)\prod_{i=0}^{t-2}(\lambda_i + c_i)(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M}) \\
&\quad + c_t\prod_{i=0}^{t-1}(\lambda_i + c_i)(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M}) \\
&= \cdots
\end{aligned}$$

Expanding this recursion all the way from t to 0,

$$\boldsymbol{W}_t^o\boldsymbol{M} = \prod_{i=0}^{t-1}(\lambda_i - c_i)\boldsymbol{W}_0^o\boldsymbol{M} + \sum_{j=0}^{t-1}\left(c_j\prod_{i=j}^{t-1}(\lambda_i - c_i)\prod_{i=0}^{j-1}(\lambda_i + c_i)\right)(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M})$$

Since the learning rate $\eta_t \ll 1$, masking Bernoulli parameter $\alpha \in (0,1)$, sparsity parameter $p_z \in (0, 0.5)$, and noise level $\sigma_0 = \mathcal{O}(1)$, these conditions result in $c_t = 2\eta_t\alpha^2(2p_z + \sigma_0^2) \in (0,1)$. $\eta_t \ll 1$ also ensures $c_t \leq \lambda_t, \forall t$. Therefore, denote $C_{1,t} := \prod_{i=0}^{t-1}(\lambda_i - c_i) \in (-1, 1)$.

On the other hand, the condition that $\boldsymbol{W}_t^o$ does not explode or vanish means the latter summation $C_{2,t} = \sum_{j=0}^{t-1}\left(c_i\prod_{i=j}^{t-1}(\lambda_i - c_i)\prod_{i=0}^{j-1}(\lambda_i + c_i)\right)(\boldsymbol{W}_0^o\boldsymbol{M} + \boldsymbol{W}_0^t\boldsymbol{M}) \in \mathbb{R}$ is non-degenerate. Indeed there are appropriate settings of η_t and λ_t that can ensure such non-degeneracy: each term in $C_{2,t}$, $c_i\prod_{i=j}^{t-1}(\lambda_i - c_i)\prod_{i=0}^{j-1}(\lambda_i + c_i)$, is an increasing function in each λ_i. By setting $\lambda_{t-1} = c_{t-1}$, we get $C_{2,t} = 0, \forall t$. In contrast, by setting $\eta_t = \eta_0, \forall t$ (i.e. a constant) and $\lambda_i = 1 - c_i/2, \forall i$, we get $c_t = c_0, \forall t$ and moreover,

$$\begin{aligned}
C_{2,t} &\geq c_{t-1}(\lambda_{t-1} - c_{t-1})\prod_{i=0}^{t-2}(\lambda_i + c_i) \quad \text{(by only keeping the term } j = t-1\text{)} \\
&= c_0(1 - 1.5c_0)\prod_{i=0}^{t-2}(1 + c_0/2) \\
&= c_0(1 - 1.5c_0)(1 + c_0/2)^{t-1} \\
&\to \infty
\end{aligned}$$

Hence, there exists appropriate settings of η_t and λ_t such that $0 < C_{2,t} < \infty$.

Hence we have showed that

$$\boldsymbol{W}_t^o \boldsymbol{M} = C_{1,t} \boldsymbol{W}_0^o \boldsymbol{M} + C_{2,t}(\boldsymbol{W}_0^o \boldsymbol{M} + \boldsymbol{W}_0^t \boldsymbol{M})$$

By symmetry,

$$\boldsymbol{W}_t^t \boldsymbol{M} = C_{1,t} \boldsymbol{W}_0^t \boldsymbol{M} + C_{2,t}(\boldsymbol{W}_0^o \boldsymbol{M} + \boldsymbol{W}_0^t \boldsymbol{M})$$

for some $C_{1,t} \in (0,1), C_{2,t} > 0$.

□

C.2 Assumptions for ReLU network with normalization

In this section, we provide details related to the assumptions made for Theorem 3 in Section 5.2.1. We also discuss the significance of important assumptions.

Assumption 1 (symmetric Bernoulli latent)**.** *Let the latent vector $\boldsymbol{z}$ be such that for each index $i \subset [d]$, 1 and $\forall l \notin K$, $z_l = 0$.*

$$z_i = \begin{cases} -1, & \text{with probability } \frac{\kappa}{2} \\ 1, & \text{with probability } \frac{\kappa}{2} \\ 0, & \text{with probability } 1-\kappa \end{cases}$$

Remark 3. *Our theoretical result for training dynamics uses the symmetric (i.e. $0, \pm 1$) instead of the binary (i.e. $0, 1$) latent, because the symmetric version is consistent with our experiments. Correspondingly, the ReLU activation is also the symmetric version (equation 14), with the bias term taken into consideration.*

Assumption 2 (warm start)**.** *We assume that $\boldsymbol{W}^o$ and $\boldsymbol{W}^t$ are both warm-started, i.e. let*

$$\boldsymbol{W}^o = \boldsymbol{M} + \boldsymbol{\Delta}^o = \boldsymbol{I} + \boldsymbol{\Delta}^o$$
$$\boldsymbol{W}^t = \boldsymbol{M} + \boldsymbol{\Delta}^t = \boldsymbol{I} + \boldsymbol{\Delta}^t$$

for some initial error $\boldsymbol{\Delta}^o, \boldsymbol{\Delta}^t$ in which $\forall i, j, \boldsymbol{\Delta}_{ij}^o, \boldsymbol{\Delta}_{ij}^t = o(\frac{1}{d})$.

Remark 4. *The $o(\frac{1}{d})$ requirement is such that in the Assumption 3 (bias) below, c_b^o and c_b^t are in $o(1)$. If we switch to a "with high probability (over $\boldsymbol{D}$ and $\boldsymbol{z}$)"-style statement, then the initial deviations $\boldsymbol{\Delta}_{ij}^o, \boldsymbol{\Delta}_{ij}^t$ can be larger.*

Assumption 3 (bias)**.** *Assume the bias $\boldsymbol{b}^o$ and $\boldsymbol{b}^t$ are fixed throughout the optimization process and satisfy the following requirements: for each set $i \in [d]$, let*

$$c_b^o = \sum_{l \neq i} \boldsymbol{\Delta}_{il}^o D_{1,ll} z_l + \sum_{j=1}^{d} (I_{ij} + \boldsymbol{\Delta}_{ij}^o) D_{1,jj} \boldsymbol{\epsilon}_j$$
$$c_b^t = \sum_{l \neq i} \boldsymbol{\Delta}_{il}^t D_{1,ll} z_l + \sum_{j=1}^{d} (I_{ij} + \boldsymbol{\Delta}_{ij}^t) D_{1,jj} \boldsymbol{\epsilon}_j$$

which are $o(1)$ since $\forall i, j, \boldsymbol{\Delta}_{ij}^o = o(\frac{1}{d})$ and $\boldsymbol{\epsilon}_j = o(1)$. Let b_i^o and b_i^t both be $o(1)$ satisfy:

$$\max\{-c_b^o, c_b^o, 0\} < b_i^o < \min\{1 + c_b^o, 1 - c_b^o\}$$
$$\max\{-c_b^t, c_b^t, 0\} < b_i^t < \min\{1 + c_b^t, 1 - c_b^t\}$$

Assumption 4 (row normalization)**.** *After each update on $\boldsymbol{W}^o$ and $\boldsymbol{W}^t$, they are row-normalized s.t. $\|\boldsymbol{W}_{i*}^o\|_2 = 1$ and $\|\boldsymbol{W}_{i*}^t\|_2 = 1$ for each i.*

C.3 Proof for ReLU network with normalization

Theorem (ReLU network with normalization, warm start, Theorem 3 restated)**.** *Under Assumptions 1, 2, 3, 4 in Appendix C.2, running:*

Repeat until both $\boldsymbol{W}^o$ *and* $\boldsymbol{W}^t$ *converges:*

- *Repeat until* $\boldsymbol{W}^o$ *converges:*

$$\boldsymbol{W}^o \leftarrow \textit{normalize}(\boldsymbol{W}^o - \eta \nabla_{\boldsymbol{W}^o} L(\boldsymbol{W}^o, \boldsymbol{W}^t))$$

- *Repeat until* $\boldsymbol{W}^t$ *converges:*

$$\boldsymbol{W}^t \leftarrow \textit{normalize}(\boldsymbol{W}^t - \eta \nabla_{\boldsymbol{W}^t} L(\boldsymbol{W}^o, \boldsymbol{W}^t))$$

will make $\boldsymbol{W}^o$ *and* $\boldsymbol{W}^t$ *both converge to* I

Proof. To start with, we simplify the expression of $L(\boldsymbol{W}^o, \boldsymbol{W}^t)$ under the above assumptions.

$$L(\boldsymbol{W}^o, \boldsymbol{W}^t) = 2 - 2\mathbb{E}_{\boldsymbol{x}, \boldsymbol{D}_1, \boldsymbol{D}_2}[\sum_{i=1}^{d} \text{SReLU}_{\boldsymbol{b}^o}(\boldsymbol{W}^o_{i*}\boldsymbol{D}_1\boldsymbol{x}) \cdot \text{SReLU}_{\boldsymbol{b}^t}(\boldsymbol{W}^t_{i*}\boldsymbol{D}_2\boldsymbol{x})] \tag{23}$$

$$= 2 - 2\mathbb{E}_{\boldsymbol{x}, \boldsymbol{D}_1, \boldsymbol{D}_2}[\sum_{i=1}^{d} \text{SReLU}_{b^o_i}(\boldsymbol{W}^o_{i*}\boldsymbol{D}_1(\boldsymbol{M}\boldsymbol{z} + \boldsymbol{\epsilon})) \cdot \text{SReLU}_{b^t_i}(\boldsymbol{W}^t_{i*}\boldsymbol{D}_2(\boldsymbol{M}\boldsymbol{z} + \boldsymbol{\epsilon}))] \tag{24}$$

Since $\boldsymbol{M} = \boldsymbol{I}$, the SReLU term becomes

$$\begin{aligned} &\text{SReLU}_{b^o_i}(\boldsymbol{W}^o_{i*}\boldsymbol{D}_1(\boldsymbol{M}\boldsymbol{z} + \boldsymbol{\epsilon})) \\ &= \text{SReLU}_{b^o_i}(\boldsymbol{W}^o_{i*}\boldsymbol{D}_1(\boldsymbol{z} + \boldsymbol{\epsilon})) \\ &= \text{SReLU}_{b^o_i}(\sum_{j=1}^{d} W^o_{ij} D_{1,jj}(z_j + \boldsymbol{\epsilon}_j)) \end{aligned}$$

By Assumption 2 (warm-start), the above becomes

$$\begin{aligned} &\text{SReLU}_{b^o_i}(\sum_{j=1}^{d} (I_{ij} + \boldsymbol{\Delta}^o_{ij}) D_{1,jj}(z_j + \boldsymbol{\epsilon}_j)) \\ &= \text{SReLU}_{b^o_i}((1 + \boldsymbol{\Delta}^o_{ii}) D_{1,ii}(z_i + \boldsymbol{\epsilon}_i) + \sum_{j \neq i} \boldsymbol{\Delta}^o_{ij} D_{1,jj}(z_j + \boldsymbol{\epsilon}_j)) \end{aligned}$$

By Assumption 3 on the bias,

$$(1 + \boldsymbol{\Delta}^o_{ii}) D_{1,ii}(z_i + \boldsymbol{\epsilon}_i) + \sum_{j \neq i} \boldsymbol{\Delta}^o_{ij} D_{1,jj}(z_j + \boldsymbol{\epsilon}_j) \text{ is } \begin{cases} > b^o_i, & \text{if } D_{1,ii} = 1 \text{ and } z_i = 1 \\ < -b^o_i, & \text{if } D_{1,ii} = 1 \text{ and } z_i = -1 \\ \in (-b^o_i, b^o_i), & \text{if } D_{1,ii} = 0 \text{ or } z_i = 0 \end{cases}$$

Therefore the SReLU term can be simplified to

$$\begin{aligned} &\begin{cases} (1 + \boldsymbol{\Delta}^o_{ii})(1 + \boldsymbol{\epsilon}_i) + \sum_{j \neq i} \boldsymbol{\Delta}^o_{ij} D_{1,jj}(z_j + \boldsymbol{\epsilon}_j) - b^o_i, & \text{if } D_{1,ii} = 1 \text{ and } z_i = 1 \\ (1 + \boldsymbol{\Delta}^o_{ii})(-1 + \boldsymbol{\epsilon}_i) + \sum_{j \neq i} \boldsymbol{\Delta}^o_{ij} D_{1,jj}(z_j + \boldsymbol{\epsilon}_j) + b^o_i, & \text{if } D_{1,ii} = 1 \text{ and } z_i = -1 \\ 0, & \text{if } D_{1,ii} = 0 \text{ or } z_i = 0 \end{cases} \\ &= D_{1,ii}(\mathbf{1}_{z_i=1}((1 + \boldsymbol{\Delta}^o_{ii})(1 + \boldsymbol{\epsilon}_i) + \sum_{j \neq i} \boldsymbol{\Delta}^o_{ij} D_{1,jj}(z_j + \boldsymbol{\epsilon}_j) - b^o_i) \\ &\quad + \mathbf{1}_{z_i=-1}((1 + \boldsymbol{\Delta}^o_{ii})(-1 + \boldsymbol{\epsilon}_i) + \sum_{j \neq i} \boldsymbol{\Delta}^o_{ij} D_{1,jj}(z_j + \boldsymbol{\epsilon}_j) + b^o_i)) \\ &= D_{1,ii}(\mathbf{1}_{z_i=1}(1 + \boldsymbol{\Delta}^o_{ii} + \sum_{j \neq i} \boldsymbol{\Delta}^o_{ij} D_{1,jj} z_j - b^o_i + (\cdots)\boldsymbol{\epsilon}) + \mathbf{1}_{z_i=-1}(-1 - \boldsymbol{\Delta}^o_{ii} + \sum_{j \neq i} \boldsymbol{\Delta}^o_{ij} D_{1,jj} z_j + b^o_i + (\cdots)\boldsymbol{\epsilon})) \end{aligned}$$

A summand in equation 24 becomes

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon},\boldsymbol{D}_1,\boldsymbol{D}_2}\left[\mathrm{SReLU}_{b_i^o}(\boldsymbol{W}_{i*}^o\boldsymbol{D}_1(\boldsymbol{M}\boldsymbol{z}+\boldsymbol{\epsilon}))\cdot\mathrm{SReLU}_{b_i^t}(\boldsymbol{W}_{i*}^t\boldsymbol{D}_2(\boldsymbol{M}\boldsymbol{z}+\boldsymbol{\epsilon}))\right]\\
&=\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon},\boldsymbol{D}_1,\boldsymbol{D}_2}[D_{1,ii}(\mathbf{1}_{z_i=1}(1+\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o D_{1,jj}z_j-b_i^o+(\cdots)\boldsymbol{\epsilon})\\
&\quad\mathbf{1}_{z_i=-1}(-1-\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o D_{1,jj}z_j+b_i^o+(\cdots)\boldsymbol{\epsilon}))\\
&\quad\cdot D_{2,ii}(\mathbf{1}_{z_i=1}(1+\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t D_{2,jj}z_j-b_i^t+(\cdots)\boldsymbol{\epsilon})+\mathbf{1}_{z_i=-1}(-1-\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t D_{2,jj}z_j+b_i^t+(\cdots)\boldsymbol{\epsilon}))]\\
&=\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon},\boldsymbol{D}_1,\boldsymbol{D}_2}[D_{1,ii}D_{2,ii}\cdot(\mathbf{1}_{z_i=1}(1+\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o D_{1,jj}z_j-b_i^o+(\cdots)\boldsymbol{\epsilon})(1+\boldsymbol{\Delta}_{ii}^t\\
&\quad\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t D_{2,jj}z_j-b_i^t+(\cdots)\boldsymbol{\epsilon})\\
&\quad+\mathbf{1}_{z_i=-1}(-1-\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o D_{1,jj}z_j+b_i^o+(\cdots)\boldsymbol{\epsilon})(-1-\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t D_{2,jj}z_j+b_i^t+(\cdots)\boldsymbol{\epsilon})]\\
&=\mathbb{E}_{\boldsymbol{z},\boldsymbol{\epsilon},\boldsymbol{D}_1,\boldsymbol{D}_2}[D_{1,ii}D_{2,ii}\cdot(\mathbf{1}_{z_i=1}(1+\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o D_{1,jj}z_j-b_i^o)(1+\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t D_{2,jj}z_j-b_i^t)\\
&\quad+\mathbf{1}_{z_i=-1}(-1-\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o D_{1,jj}z_j+b_i^o)(-1-\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t D_{2,jj}z_j+b_i^t)\\
&\quad+(\cdots)\boldsymbol{\epsilon}+\mathcal{O}(\|\boldsymbol{\epsilon}\|_2^2)]
\end{aligned}
$$

in which note that $\boldsymbol{\epsilon}_j=o(1)$, so we denote any terms containing 2-nd order $\boldsymbol{\epsilon}_i\boldsymbol{\epsilon}_j$ as $\mathcal{O}(\|\boldsymbol{\epsilon}\|_2^2)$. Also, the 1st-order $\boldsymbol{\epsilon}_i$ is removed after taking the expectation over $\boldsymbol{\epsilon}$. So the above simplifies to

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{z},\boldsymbol{D}_1,\boldsymbol{D}_2}[D_{1,ii}D_{2,ii}\cdot(\mathbf{1}_{z_i=1}(1+\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o D_{1,jj}z_j-b_i^o)(1+\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t D_{2,jj}z_j-b_i^t)\\
&\quad+\mathbf{1}_{z_i=-1}(-1-\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o D_{1,jj}z_j+b_i^o)(-1-\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t D_{2,jj}z_j+b_i^t))]+\mathbb{E}_{\boldsymbol{\epsilon}}[\mathcal{O}(\|\boldsymbol{\epsilon}\|_2^2)]
\end{aligned}
$$

For simplicity, we ignore the $\mathcal{O}(\|\boldsymbol{\epsilon}\|_2^2)$ terms in the following calculation. Moreover, note that $D_{1,ii}, D_{2,ii}, \mathbf{1}_{z_i=1}, \mathbf{1}_{z_i=-1}$ each appears only once in the expectation, so by independence, we can further simplify the above to

$$
\begin{aligned}
&\alpha^2(\mathbb{E}_{\boldsymbol{z}}[\mathbf{1}_{z_i=1}]\mathbb{E}_{\boldsymbol{z}}[(1+\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o\mathbb{E}_{\boldsymbol{D}_1}[D_{1,jj}]z_j-b_i^o)(1+\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t\mathbb{E}_{\boldsymbol{D}_2}[D_{2,jj}]z_j-b_i^t)]\\
&\quad+\mathbb{E}_{\boldsymbol{z}}[\mathbf{1}_{z_i=-1}]\mathbb{E}_{\boldsymbol{z}}[(-1-\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o\mathbb{E}_{\boldsymbol{D}_1}[D_{1,jj}]z_j+b_i^o)(-1-\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t\mathbb{E}_{\boldsymbol{D}_2}[D_{2,jj}]z_j+b_i^t)])\\
&=\alpha^2\frac{\kappa}{2}\mathbb{E}_{\boldsymbol{z}}[(1+\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o\alpha z_j-b_i^o)(1+\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t\alpha z_j-b_i^t)\\
&\quad+(-1-\boldsymbol{\Delta}_{ii}^o+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o\alpha z_j+b_i^o)(-1-\boldsymbol{\Delta}_{ii}^t+\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t\alpha z_j+b_i^t)]\\
&=\alpha^2\frac{\kappa}{2}\mathbb{E}_{\boldsymbol{z}}[(1+\boldsymbol{\Delta}_{ii}^o-b_i^o)(1+\boldsymbol{\Delta}_{ii}^t-b_i^t)+(1+\boldsymbol{\Delta}_{ii}^o-b_i^o)\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t\alpha z_j+(1+\boldsymbol{\Delta}_{ii}^t-b_i^t)\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o\alpha z_j\\
&+(\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o\alpha z_j)(\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t\alpha z_j)+(-1-\boldsymbol{\Delta}_{ii}^o+b_i^o)(-1-\boldsymbol{\Delta}_{ii}^t+b_i^t)\\
&+(-1-\boldsymbol{\Delta}_{ii}^o+b_i^o)\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t\alpha z_j+(-1-\boldsymbol{\Delta}_{ii}^t+b_i^t)\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o\alpha z_j+(\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^o\alpha z_j)(\sum_{j\neq i}\boldsymbol{\Delta}_{ij}^t\alpha z_j)]
\end{aligned}
$$

By Assumption 1 (symmetric Bernoulli latent), $\mathbb{E}_{\boldsymbol{z}}[z_j] = 0$ and $\mathbb{E}_{\boldsymbol{z}}[z_i z_j] = \begin{cases} 0, & \text{if } i \neq j \\ \kappa, & \text{if } i = j \end{cases}$

So the above simplifies to

$$\alpha^2 \frac{\kappa}{2}[(1 + \boldsymbol{\Delta}^o_{ii} - b^o_i)(1 + \boldsymbol{\Delta}^t_{ii} - b^t_i) + \alpha^2\kappa \sum_{j\neq i} \boldsymbol{\Delta}^o_{ij}\boldsymbol{\Delta}^t_{ij} + (-1 - \boldsymbol{\Delta}^o_{ii} + b^o_i)(-1 - \boldsymbol{\Delta}^t_{ii} + b^t_i) + \alpha^2\kappa \sum_{j\neq i} \boldsymbol{\Delta}^o_{ij}\boldsymbol{\Delta}^t_{ij}]$$
$$= \alpha^2\kappa[(1 + \boldsymbol{\Delta}^o_{ii} - b^o_i)(1 + \boldsymbol{\Delta}^t_{ii} - b^t_i) + \alpha^2\kappa \sum_{j\neq i} \boldsymbol{\Delta}^o_{ij}\boldsymbol{\Delta}^t_{ij}]$$

Plugging into equation 24,

$$L(\boldsymbol{W}^o, \boldsymbol{W}^t) = 2 - 2\alpha^2\kappa \sum_{i=1}^{d} (1 + \boldsymbol{\Delta}^o_{ii} - b^o_i)(1 + \boldsymbol{\Delta}^t_{ii} - b^t_i) + \alpha^2\kappa \sum_{j\neq i} \boldsymbol{\Delta}^o_{ij}\boldsymbol{\Delta}^t_{ij}$$

Hence

$$\nabla_{\boldsymbol{W}^o_{ii}} L(\boldsymbol{W}^o, \boldsymbol{W}^t) = -2\alpha^2\kappa(1 + \boldsymbol{\Delta}^t_{ii} - b^t_i)$$
$$\nabla_{\boldsymbol{W}^o_{ij}} L(\boldsymbol{W}^o, \boldsymbol{W}^t) = -2\alpha^2\kappa \cdot \alpha^2\kappa \boldsymbol{\Delta}^t_{ij}$$

Note that $b^t_i = o(1)$ by Assumption 3 on the bias, and that $\boldsymbol{\Delta}^t_{ii}, \boldsymbol{\Delta}^t_{ij} = o(\frac{1}{d})$ by Assumption 2 (warm start), so the above gradient updates satisfy

$$\nabla_{\boldsymbol{W}^o_{ii}} L(\boldsymbol{W}^o, \boldsymbol{W}^t) = -2\alpha^2\kappa(1 + o(1))$$
$$\nabla_{\boldsymbol{W}^o_{ij}} L(\boldsymbol{W}^o, \boldsymbol{W}^t) = -2\alpha^2\kappa \cdot \alpha^2\kappa \cdot o(\frac{1}{d})$$

After each update, with row-normalization, $\boldsymbol{W}^o$ still satisfies the warm-start condition. In the following, we characterize the convergence point.

By Lemma 1, since

$$\|\nabla_{\boldsymbol{W}^o_{ij}} L(\boldsymbol{W}^o, \boldsymbol{W}^t)\|_2 = (1 + o(1))\alpha^2\kappa|\boldsymbol{\Delta}^t_{ij}| \cdot \|\nabla_{\boldsymbol{W}^o_{ii}} L(\boldsymbol{W}^o, \boldsymbol{W}^t)\|_2 \quad \forall j \neq i$$

the convergence points $\boldsymbol{W}^{o*}$ satisfy

$$|\boldsymbol{W}^{o*}_{ij}| = (1 + o(1))\alpha^2\kappa|\boldsymbol{\Delta}^t_{ij}|\boldsymbol{W}^{o*}_{ii} \quad \forall j \neq i$$

and consequently

$$|\boldsymbol{W}^{o*}_{ij}| \leq (1 + o(1))\alpha^2\kappa|\boldsymbol{\Delta}^t_{ij}| \quad \forall j \neq i$$

Next, while updating $\boldsymbol{W}^t$ to convergence $\boldsymbol{W}^{t*}$, the same argument implies

$$|\boldsymbol{W}^{t*}_{ij}| = (1 + o(1))\alpha^2\kappa|\boldsymbol{W}^{o*}_{ij}|\boldsymbol{W}^{t*}_{ii} \quad \forall j \neq i$$

and consequently

$$|\boldsymbol{W}^{t*}_{ij}| \leq (1 + o(1))\alpha^2\kappa|\boldsymbol{W}^{o*}_{ij}| \leq ((1 + o(1))\alpha^2\kappa)^2|\boldsymbol{\Delta}^t_{ij}|$$

Repeating the above alternating optimization process, since $(1 + o(1))\alpha^2\kappa$ is a constant in $(0, 1)$, the exponential multiplier will make

$$|\boldsymbol{W}^{o*}_{ij}|, |\boldsymbol{W}^{t*}_{ij}| \to 0, \forall j \neq i$$

and

$$|\boldsymbol{W}^{o*}_{ii}|, |\boldsymbol{W}^{t*}_{ii}| \to 1, \forall i$$

Therefore both $\boldsymbol{W}^o$ and $\boldsymbol{W}^t$ converge to I. □

D TECHNICAL LEMMAS

Lemma 1 (convergence of proportional update with normalization)**.** *For any $\boldsymbol{v}^{(0)} \in \mathbb{R}^d$, consider the following update*

$$\begin{aligned}
\boldsymbol{v}_1^{(t+1)} &\leftarrow \boldsymbol{v}_1^{(t)} + c^{(t)} \\
\boldsymbol{v}_i^{(t+1)} &\leftarrow \boldsymbol{v}_i^{(t)} + ac^{(t)} \quad \forall i = 2..d \\
\boldsymbol{v}^{(t+1)} &\leftarrow l_2\text{-}normalize(\boldsymbol{v}^{(t+1)})
\end{aligned}$$

in which $a \in \mathbb{R}$ and $\exists c > 0$ s.t. $\forall t, c^{(t)} > c$. Moreover, the initial $\boldsymbol{v}^{(0)}$ satisfies

$$\begin{aligned}
\boldsymbol{v}_1^{(0)} + a\boldsymbol{v}_2^{(0)} + \cdots a\boldsymbol{v}_d^{(0)} > 0 \\
\sum_{i=1}^{d} (\boldsymbol{v}_i^{(0)})^2 = 1
\end{aligned}$$

Then, repeating the above update will cause $\boldsymbol{v}^{(t)}$ to converge to

$$\begin{aligned}
\boldsymbol{v}_1^{(*)} &\leftarrow \frac{1}{\sqrt{1 + (d-1)a^2}} \\
\boldsymbol{v}_i^{(*)} &\leftarrow \frac{a}{\sqrt{1 + (d-1)a^2}} \quad \forall i = 2..d
\end{aligned}$$

Proof. By the update rule,

$$\begin{aligned}
\boldsymbol{v}_1^{(t+1)} &= \frac{\boldsymbol{v}_1^{(t)} + c^{(t)}}{\sqrt{(\boldsymbol{v}_1^{(t)} + c^{(t)})^2 + \sum_{i=2}^{d} (\boldsymbol{v}_i^{(t)} + ac^{(t)})^2}} \\
\boldsymbol{v}_i^{(t+1)} &= \frac{\boldsymbol{v}_i^{(t)} + ac^{(t)}}{\sqrt{(\boldsymbol{v}_1^{(t)} + c^{(t)})^2 + \sum_{i=2}^{d} (\boldsymbol{v}_i^{(t)} + ac^{(t)})^2}} \quad \forall i = 2..d
\end{aligned}$$

First, we prove by induction that $\forall t$,

$$\boldsymbol{v}_1^{(t)} + a\boldsymbol{v}_2^{(t)} + \cdots a\boldsymbol{v}_d^{(t)} > 0 \tag{25}$$

It is given that equation 25 holds for $t = 0$. Suppose it holds for $t = t_0$, then by the update rule

$$\begin{aligned}
&\boldsymbol{v}_1^{(t_0+1)} + a\boldsymbol{v}_2^{(t_0+1)} + \cdots a\boldsymbol{v}_d^{(t_0+1)} \\
&= \frac{\boldsymbol{v}_1^{(t_0)} + c^{(t_0)}}{\sqrt{(\boldsymbol{v}_1^{(t_0)} + c^{(t_0)})^2 + \sum_{i=2}^{d} (\boldsymbol{v}_i^{(t_0)} + ac^{(t_0)})^2}} + a \sum_{i=2}^{d} \frac{\boldsymbol{v}_i^{(t_0)} + ac^{(t_0)}}{\sqrt{(\boldsymbol{v}_1^{(t_0)} + c^{(t_0)})^2 + \sum_{i=2}^{d} (\boldsymbol{v}_i^{(t_0)} + ac^{(t_0)})^2}} \\
&= \frac{\boldsymbol{v}_1^{(t_0)} + c^{(t_0)} + a\sum_{i=2}^{d} (\boldsymbol{v}_i^{(t_0)} + ac^{(t_0)})}{\sqrt{(\boldsymbol{v}_1^{(t_0)} + c^{(t_0)})^2 + \sum_{i=2}^{d} (\boldsymbol{v}_i^{(t_0)} + ac^{(t_0)})^2}} \\
&= \frac{\boldsymbol{v}_1^{(t_0)} + a\sum_{i=2}^{d} \boldsymbol{v}_i^{(t_0)} + (d-1)a^2c^{(t_0)}}{\sqrt{(\boldsymbol{v}_1^{(t_0)} + c^{(t_0)})^2 + \sum_{i=2}^{d} (\boldsymbol{v}_i^{(t_0)} + ac^{(t_0)})^2}} \\
&> \frac{(d-1)a^2c^{(t_0)}}{\sqrt{(\boldsymbol{v}_1^{(t_0)} + c^{(t_0)})^2 + \sum_{i=2}^{d} (\boldsymbol{v}_i^{(t_0)} + ac^{(t_0)})^2}} \quad \text{(since equation 25 holds for } t = t_0\text{)} \\
&\geq \frac{(d-1)a^2c}{\sqrt{(\boldsymbol{v}_1^{(t_0)} + c^{(t_0)})^2 + \sum_{i=2}^{d} (\boldsymbol{v}_i^{(t_0)} + ac^{(t_0)})^2}} \quad \text{(since it is given that } \forall t, c^{(t)} > c\text{)} \\
&\geq 0 \quad \text{(since it is given that } c > 0\text{)}
\end{aligned}$$

Hence, equation 25 also holds for $t = t_0 + 1$. Therefore, by induction, equation 25 also holds for all t.

Next, we use the above fact to prove that $\exists \gamma \in (0,1)$ s.t. $\forall t, \forall i = 2..d$,

$$|\boldsymbol{v}_i^{(t+1)} - a\boldsymbol{v}_1^{(t+1)}| < \gamma |\boldsymbol{v}_i^{(t)} - a\boldsymbol{v}_1^{(t)}| \tag{26}$$

To show this, plugging in the update equation, we get $\forall t, \forall i = 2..d$,

$$\begin{aligned}
&|\boldsymbol{v}_i^{(t+1)} - a\boldsymbol{v}_1^{(t+1)}| \\
&= \left| \frac{\boldsymbol{v}_i^{(t)} + ac^{(t)}}{\sqrt{(\boldsymbol{v}_1^{(t)} + c^{(t)})^2 + \sum_{i=2}^d (\boldsymbol{v}_i^{(t)} + ac^{(t)})^2}} - a \frac{\boldsymbol{v}_1^{(t)} + c^{(t)}}{\sqrt{(\boldsymbol{v}_1^{(t)} + c^{(t)})^2 + \sum_{i=2}^d (\boldsymbol{v}_i^{(t)} + ac^{(t)})^2}} \right| \\
&= \left| \frac{\boldsymbol{v}_i^{(t)} + ac^{(t)} - a(\boldsymbol{v}_1^{(t)} + c^{(t)})}{\sqrt{(\boldsymbol{v}_1^{(t)} + c^{(t)})^2 + \sum_{i=2}^d (\boldsymbol{v}_i^{(t)} + ac^{(t)})^2}} \right| \\
&= \left| \frac{\boldsymbol{v}_i^{(t)} - a\boldsymbol{v}_1^{(t)}}{\sqrt{(\boldsymbol{v}_1^{(t)} + c^{(t)})^2 + \sum_{i=2}^d (\boldsymbol{v}_i^{(t)} + ac^{(t)})^2}} \right| \\
&= \left| \frac{\boldsymbol{v}_i^{(t)} - a\boldsymbol{v}_1^{(t)}}{\sqrt{\sum_{i=1}^d (\boldsymbol{v}_i^{(t)})^2 + 2c^{(t)}\boldsymbol{v}_1^{(t)} + (c^{(t)})^2 + \sum_{i=2}^d 2ac^{(t)}\boldsymbol{v}_i^{(t)} + (d-1)a^2(c^{(t)})^2}} \right| \\
&= \left| \frac{\boldsymbol{v}_i^{(t)} - a\boldsymbol{v}_1^{(t)}}{\sqrt{\sum_{i=1}^d (\boldsymbol{v}_i^{(t)})^2 + 2c^{(t)}(\boldsymbol{v}_1^{(t)} + a\sum_{i=2}^d \boldsymbol{v}_i^{(t)}) + (1 + (d-1)a^2)(c^{(t)})^2}} \right| \\
&= \left| \frac{\boldsymbol{v}_i^{(t)} - a\boldsymbol{v}_1^{(t)}}{\sqrt{1 + 2c^{(t)}(\boldsymbol{v}_1^{(t)} + a\sum_{i=2}^d \boldsymbol{v}_i^{(t)}) + (1 + (d-1)a^2)(c^{(t)})^2}} \right| \quad \text{(since each } \boldsymbol{v}^{(t)} \text{ is } l_2\text{-normalized)} \\
&\leq \left| \frac{\boldsymbol{v}_i^{(t)} - a\boldsymbol{v}_1^{(t)}}{\sqrt{1 + (1 + (d-1)a^2)(c^{(t)})^2}} \right| \quad \text{(since } c^{(t)} > c > 0 \text{ and equation 25)} \\
&< \left| \frac{\boldsymbol{v}_i^{(t)} - a\boldsymbol{v}_1^{(t)}}{\sqrt{1 + (1 + (d-1)a^2)c^2}} \right| \quad \text{(since } c^{(t)} > c > 0)
\end{aligned}$$

Hence, equation 26 holds with the constant

$$\gamma = \frac{1}{\sqrt{1 + (1 + (d-1)a^2)c^2}} \in (0,1)$$

Therefore,

$$\begin{aligned}
&\lim_{t\to\infty} |\boldsymbol{v}_i^{(t)} - a\boldsymbol{v}_1^{(t)}| \\
&\leq \lim_{t\to\infty} \gamma^t |\boldsymbol{v}_i^{(0)} - a\boldsymbol{v}_1^{(0)}| \\
&= 0 \quad (\text{since } \gamma \in (0,1))
\end{aligned}$$

Finally, since each $\boldsymbol{v}^{(t)}$ is l_2-normalized, the above relation for $t \to \infty$ leads to the following equations

$$\begin{aligned}
\lim_{t\to\infty} \sum_{i=1}^d (\boldsymbol{v}_i^{(t)})^2 &= 1 \\
\lim_{t\to\infty} \boldsymbol{v}_i^{(t)} &= a \lim_{t\to\infty} \boldsymbol{v}_1^{(t)} \quad \forall i = 2..d
\end{aligned}$$

whose unique solution is the one stated in the lemma. □