MINI-BATCH OPTIMIZATION OF CONTRASTIVE LOSS

Kartik Sreenivasan[w], **Keon Lee**[k], **Jeong-Gwan Lee**[k], **Anna Lee**[k]
Jaewoong Cho[k], **Jy-yong Sohn**[y], **Dimitris Papailiopoulos**[w], **Kangwook Lee**[wk]

[k] KRAFTON, Inc. [y] Yonsei University [w] University of Wisconsin-Madison

ABSTRACT

In this paper, we study the effect of mini-batch selection on contrastive loss and propose new mini-batch selection methods to improve efficiency. Theoretically, we show that both the full-batch and mini-batch settings share the same solution, the simplex Equiangular Tight Frame (ETF), if all $\binom{N}{B}$ mini-batches are seen during training. However, when not all possible batches are seen, mini-batch training can lead to suboptimal solutions. To address this issue, we propose efficient mini-batch selection methods that compare favorably with existing methods. Our experimental results demonstrate the effectiveness of our proposed methods in finding a near-optimal solution with a reduced number of gradient steps and outperforming existing mini-batch selection methods.

1 INTRODUCTION

Contrastive learning has recently gained significant attention as a pre-training method for self-supervised learning, due to its ability to leverage the vast amount of freely available unlabeled data (Jaiswal et al., 2020). Contrastive loss is designed to ensure that embeddings of two samples are similar if they are considered a "positive" pair, in cases such as coming from the same class (Khosla et al., 2020), being an augmented version of one another (Chen et al., 2020), or being two different modalities of the same data (Radford et al., 2021). Conversely, if two samples form a "negative" pair, the contrastive loss encourages their embeddings to be dissimilar.

In practice, it is not feasible to consider all possible positive and negative pairs when implementing a contrastive learning algorithm due to the quadratic memory requirement ($\mathcal{O}(N^2)$) when working with N samples. To mitigate this issue, practitioners typically choose a set of mini-batches of size $B = \mathcal{O}(1)$, and consider only pairs within each mini-batch. This approach results in an overall memory requirement of $\mathcal{O}(|\mathcal{S}|B^2) = \mathcal{O}(|\mathcal{S}|)$ where $|\mathcal{S}|$ is the number of mini-batches. While this mini-batch technique has been observed to be effective in practice, there remains a lack of theoretical understanding as to whether this approach is optimal or if alternative methods may be more effective.

The primary research question that this paper aims to address is: *What is the most effective and principled approach to optimizing the contrastive loss when utilizing mini-batches?* We restrict our attention to a particular contrastive learning setting, where positive pairs consist of two different views of the same data. While this setting was chosen due to its analytical tractability, it precisely captures the multi-modal contrastive learning setup (CLIP (Radford et al., 2021)) and closely approximates the uni-modal case too (SimCLR (Chen et al., 2020)).

The primary contributions of this paper are twofold. First, we show that mini-batch and full-batch training are equivalent under some mild conditions. Specifically, we show that they are equivalent if and only if all $\binom{N}{B}$ mini-batches are selected.

From a computational complexity perspective, the identified equivalence condition may be seen as somewhat prohibitive, as it implies that all $\binom{N}{B} = \mathcal{O}(N^B)$ mini-batches must be considered. Our second contribution is to develop mini-batch selection algorithms that choose $\mathcal{O}(N)$ mini-batches in an adaptive fashion while training. The key idea is very simple borrowed ideas from recent literature in optimization theory (Kawaguchi & Lu, 2020), our proposed methods find mini-batches that contain the most informative pairs using simple greedy algorithms. We demonstrate via extensive experiments that our proposed greedy algorithms achieve superior performance compared to current mini-batch selection algorithms, and are able to find solutions that are very close to the full-batch solutions.

Summary of Main Theoretical Results. The original definition of ETF (Sustik et al., 2007) is for N vectors in a d-dimensional space where $d \leq N-1$. (See Def. 2 in Appendix B.) Papyan et al. (2020) consider the case when $d \geq N-1$ to characterize the phenomenon of neural collapse. In our work, we will use this definition of simplex ETFs which is stated below.

Definition 1 (Simplex Equiangular Tight Frame)**.** A set of N vectors $\{\boldsymbol{u}_i\}_{i=1}^N$ in the d-dimensional space forms a simplex Equiangular Tight Frame (ETF) if $\|\boldsymbol{u}_i\| = 1$ for every $i \in [N]$ and $\boldsymbol{u}_i^\intercal \boldsymbol{u}_j = -1/(N-1)$ for all $i \neq j$.

Main findings. For $d \geq N-1$, we prove that: (i) In the full-batch setting ($B = N$), the optimal solution of contrastive learning (formally defined in Sec. 2) is the simplex ETF, (ii) In the mini-batch setting ($B < N$), as long as we see all $\binom{N}{B}$ mini-batches, the optimal solution is the same as the full-batch solution and (iii) In the mini-batch setting ($B < N$), if we see fewer than $\binom{N}{B}$ mini-batches, there exist cases where the simplex ETF is no longer the optimal solution. Fig. 1 shows the illustration of these findings for a toy example.

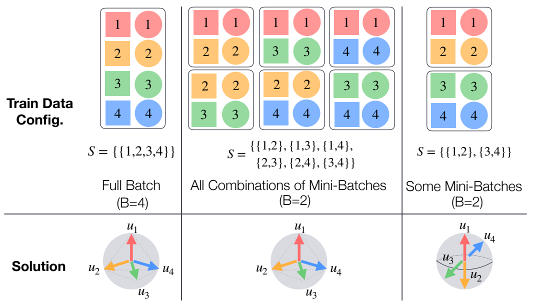

Train Data Config.
$S = \{\{1,2,3,4\}\}$
Full Batch (B=4)
$S = \{\{1,2\},\{1,3\},\{1,4\},\{2,3\},\{2,4\},\{3,4\}\}$
All Combinations of Mini-Batches (B=2)
$S = \{\{1,2\},\{3,4\}\}$
Some Mini-Batches (B=2)
Solution
u_1
u_2
u_3
u_4

Figure 1: Conceptual illustration on the effect of batch selection on contrastive learning. Consider optimizing $N = 4$ pairs of embedding vectors which are shown as colored squares and circles, respectively. The black rounded box represents a mini-batch. We compare three batch-selection methods: (i) full-batch, *i.e.*, $B = N = 4$, (ii) all $\binom{N}{B} = 6$ combinations, and (iii) some mini-batches. Here, S is the set of mini-batches. Our findings are: (i) when $B = N$, the solution forms the simplex ETF, and (ii) when $B < N$, minimizing all combinations of mini-batches results in the simplex ETF, while observing only a few mini-batches may give us a different solution.

2 Problem Setting

We consider the general contrastive learning setting, where the goal is to find $2N$ embedding vectors given N positive pairs. Given $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$, the goal is to find $\{(\boldsymbol{u}_i, \boldsymbol{v}_i)\}_{i=1}^N$ in $\mathbb{R}^d$. Note that this formulation captures both the multi-modal and the uni-modal setting as follows. For the former, one can view $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ as two different modalities of the same data while for the latter, $\boldsymbol{y}_i$ is one instantiation of a randomly augmented version of $\boldsymbol{x}_i$. We denote $\mathcal{U} = \{\boldsymbol{u}_i\}_{i=1}^N$ and $\mathcal{V} = \{\boldsymbol{v}_i\}_{i=1}^N$. Now, consider the problem of optimizing the embedding vectors for N pairs, which is given by

$$\min_{\mathcal{U},\mathcal{V}} \quad \mathcal{L}^{\text{con}}(\mathcal{U},\mathcal{V}) \quad s.t \quad \|\boldsymbol{u}_i\| = 1, \|\boldsymbol{v}_i\| = 1 \; \forall i \in [N], \tag{1}$$

where $\|\cdot\|$ denotes the ℓ_2 norm and the contrastive loss we consider is defined as

$$\mathcal{L}^{\text{con}}(\mathcal{U},\mathcal{V}) := \frac{1}{N}\sum_{i=1}^N -\log\left(\frac{e^{\boldsymbol{u}_i^\intercal \boldsymbol{v}_i}}{\sum_{j=1}^N e^{\boldsymbol{u}_i^\intercal \boldsymbol{v}_j}}\right) + \frac{1}{N}\sum_{i=1}^N -\log\left(\frac{e^{\boldsymbol{v}_i^\intercal \boldsymbol{u}_i}}{\sum_{j=1}^N e^{\boldsymbol{v}_i^\intercal \boldsymbol{u}_j}}\right).$$

Next, we consider the case of mini-batch contrastive loss which is typically considered in practice. There exist $\binom{N}{B}$ different mini-batches, each having B samples. For $k \in [\binom{N}{B}]$, let $\mathbb{B}_k$ be the k-th mini-batch so that $\mathcal{U}_{\mathbb{B}_k} := \{\boldsymbol{u}_i\}_{i \in \mathbb{B}_k}$ and $\mathcal{V}_{\mathbb{B}_k} := \{\boldsymbol{v}_i\}_{i \in \mathbb{B}_k}$. $\mathcal{S} \subseteq \left[\binom{N}{B}\right]$ denotes any subset of these mini-batches. Then the problem can be formulated as:

$$\min_{\mathcal{U},\mathcal{V}} \quad \frac{1}{|\mathcal{S}|}\sum_{i \in \mathcal{S}} \mathcal{L}^{\text{con}}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i}) \quad s.t \quad \|\boldsymbol{u}_i\| = 1, \|\boldsymbol{v}_i\| = 1 \; \forall i \in [N], \tag{2}$$

where the contrastive loss for k-th mini-batch is

$$\mathcal{L}^{\text{con}}(\mathcal{U}_{\mathbb{B}_k}, \mathcal{V}_{\mathbb{B}_k}) = \frac{1}{B}\sum_{i \in \mathbb{B}_k} -\log\left(\frac{e^{\boldsymbol{u}_i^\intercal \boldsymbol{v}_i}}{\sum_{j=1}^{\mathbb{B}_k} e^{\boldsymbol{u}_i^\intercal \boldsymbol{v}_j}}\right) + \frac{1}{B}\sum_{i \in \mathbb{B}_k} -\log\left(\frac{e^{\boldsymbol{v}_i^\intercal \boldsymbol{u}_i}}{\sum_{j=1}^{\mathbb{B}_k} e^{\boldsymbol{v}_i^\intercal \boldsymbol{u}_j}}\right).$$

3 THEORETICAL RESULTS

First, we show that the optimal solution for the problem in Eq. (1) is the simplex ETF (which follows almost directly from Lu & Steinerberger (2020))

Lemma 1 (Optimization with full-batch). *The optimal solution of the contrastive problem in Eq. (1) for full-batch satisfies $\boldsymbol{u}_i = \boldsymbol{v}_i$ for all $i \in [N]$, and $\{\boldsymbol{u}\}_{i=1}^N$ form a simplex ETF if $d \geq N-1$.*

Next, our main result explains why minimizing the batch-wise contrastive loss is meaningful. We show that the optimal solutions of the full-batch (Eq. (1)) and mini-batch (Eq. (2)) contrastive loss minimization problems are identical.

Theorem 1 (Optimization with all possible $\binom{N}{B}$ mini-batches). *Suppose $d \geq N-1$ and $B \geq 2$. The optimal solution of the mini-batch contrastive problem in Eq. (2) for $\mathcal{S} = \left[\binom{N}{B}\right]$ satisfies $\boldsymbol{u}_i = \boldsymbol{v}_i$ for all $i \in [N]$, and $\{\boldsymbol{u}_i\}_{i=1}^N$ form a simplex ETF. Therefore, the minimizer of this mini-batch problem is the same as that of the full-batch problem in Eq. (1).*

Now, we consider the cases when the solutions of mini-batch loss minimization and full-batch loss minimization *differ*. First, we show that when $B = 2$, minimizing the mini-batch loss over any strict subset of $\binom{N}{B}$ batches, is not equivalent to minimizing the full-batch loss in Eq. (1).

Theorem 2 (Optimization with fewer than $\binom{N}{B}$ mini-batches). *For $B = 2$, the minimizer of Eq. (2) for $\mathcal{S} \subsetneq \left[\binom{N}{2}\right]$ is not the minimizer of the full-batch loss in Eq. (1).*

We consider the general case of $B \geq 2$ under some mild assumptions on $\mathcal{S}$ in Appendix B where we also present the detailed proofs and auxiliary results.

4 DETAILS OF BATCH SELECTION ALGORITHMS

Since Thm. 1 shows that the solution of minimizing Eq. (2) with $\mathcal{S} = [\binom{N}{B}]$ is the same as that of minimizing Eq. (1), we can restrict our attention to the loss function over $\binom{N}{B}$ mini-batches. We now have the Stochastic Gradient Descent (SGD) setting of minimizing $\sum_{i \in \mathcal{S}} f_i(\boldsymbol{x})$ where each f_i is given by $\mathcal{L}^{\text{con}}$. Sampling the mini-batches uniformly at random, would require at least $\binom{N}{B}$ iterations to even ensure that we see each function. Since this is intractable, we must consider alternate batch selection algorithms. We classify these algorithms on the basis of two factors: (a) Partition versus Non-partition based on whether the mini-batches can have overlapping data points, (b) Adaptive versus non-adaptive based on whether the mini-batches are selected adaptively or just uniformly at random. Note that even in partition-based approaches, overlap does eventually occur across iterations. However, the same data point is not sampled again until all other points have been seen. Intuitively, this should make it more efficient which is reflected in their superior performance in our experiments. Now, we list the different kinds of batch selection algorithms.

NON-ADAPTIVE, NON-PARTITION-BASED. Uniform-random batch selection is a representative example of non-adaptive non-partition-based algorithms, in which a mini-batch is uniformly sampled from $\binom{N}{B}$ mini-batches independently for each iteration. In this paper, we do not consider this algorithm as a baseline due to the requirement of at least $\binom{N}{B}$ iterations to see all mini-batches.

NON-ADAPTIVE, PARTITION-BASED. We consider two baselines as non-adaptive partition-based algorithms, which we denote as FIXED-BATCH and SHUFFLED-BATCH. For FIXED-BATCH, the entire dataset of N samples is randomly permuted *once* and then assigned to N/B mini-batches in order. These batches remain unchanged throughout training. This is meant to replicate a scenario where data is split in a distributed environment where shuffling data across nodes is expensive. Clearly, only a specific set of N/B mini-batches is seen during training and as per Prop. 3, this is expected to lead to spurious solutions, atleast in when we directly optimize the embeddings. However, it is not clear whether this holds in the shared encoder setting. We note from the experiments in Sec. E.3 that this is indeed the case as seen by the consistently inferior performance of FIXED-BATCH. We also replicate this in simulations using a single-layer NN as a shared encoder (see Appendix E.2 for more details). The SHUFFLED-BATCH approach is designed to overcome this issue by ensuring that more of the $\binom{N}{B}$ mini-batches are seen as training proceeds. To this end, the data is randomly permuted at the beginning of every epoch with a *new* random seed before splitting it into mini-batches. Note that this is typically the batch selection method used in practice.

ADAPTIVE, NON-PARTITION-BASED. We consider ordered SGD (OSGD) proposed by Kawaguchi & Lu (2020) as a baseline for an adaptive non-partition-based algorithm. The baseline chooses top-1 mini-batch that maximizes the mini-batch loss per iteration.

ADAPTIVE, PARTITION-BASED. Here, we propose two partition-based, adaptive batch selection algorithms named BATCH CHOOSES SAMPLE (BcS) and SAMPLE CHOOSES BATCH (ScB). The proposed methods approximately find mini-batches that contain the most informative pairs using simple greedy algorithms. Inspired by ideas from recent literature in optimization theory (Kawaguchi & Lu, 2020; Lu et al., 2021; Loshchilov & Hutter, 2015), our proposed algorithms construct batches that have maximize the contrastive loss within each batch. One could view this as an extension of OSGD with the additional partition-based constraint which ensures that the same data point is not sampled again until all other points have been seen. We give an intuitive explanation of the algorithms here while leaving the pseudocode to Appendix 4.

BcS. BcS fills up N/B bins (which finally become our mini-batches) in the following manner. At each round $r = 1, \cdots, B$, we go over each bin $j = 1, \cdots, N/B$ and choose the sample index $i^\star$ that maximizes the contrastive loss $\mathcal{L}^{\text{con}}(\mathcal{U}_{\mathbb{B}_j \cup \{i\}}, \mathcal{V}_{\mathbb{B}_j \cup \{i\}})$ measured over bin j when i-th sample is included in the bin. In the first round (*i.e.*, $r = 1$), since the bins are empty, we choose sample i from $\mathcal{I}$ uniformly at random. After running the algorithm for r rounds, we end up with N/B bins, having B samples each. The set of sample indices stored in the j-th bin (mini-batch) is denoted by $\mathbb{B}_j$ for $j = 1, \cdots, N/B$. The pseudocode is provided in Algorithm 1. Note that the name of this algorithm comes from the fact that each mini-batch j chooses the optimal sample index $i^\star$. Since every remaining sample from $\mathcal{I}$ needs to be checked before being added to a new mini-batch, the time complexity of BcS is $\mathcal{O}(N^2 B)$.

ScB. ScB constructs B mini-batches (each with N/B samples) as below. For each round $r \in [N]$, we randomly choose a sample i and assign it to the mini-batch $\mathbb{B}_{j^\star}$ that maximizes the contrastive loss $\mathcal{L}^{\text{con}}(\mathcal{U}_{\mathbb{B}_j \cup \{i\}}, \mathcal{V}_{\mathbb{B}_j \cup \{i\}})$. Note that for the first N/B samples, we uniform-randomly choose the mini-batch index from $\mathcal{J}$ (the index set of empty mini-batches). The pseudo-code is provided in Algorithm 2. Note that the name of this algorithm comes from the fact that each sample i chooses the optimal mini-batch $j^\star$. Since the sample entering a new mini-batch is chosen at random each time, the time complexity of ScB is $\mathcal{O}(N^2)$ which makes it B times faster than BcS.

RAND variants In addition to BcS and ScB, we also consider a variant of our methods where we control the difficulty of each mini-batch by mixing in a fixed fraction of random samples. To this end, we first populate every mini-batch with $r\%$ randomly sampled data points and then fill up the rest using the specified algorithm. Note that when $r = 100\%$, this reduces to conventional random sampling. We apply this to both BcS and ScB which we hereby refer to as BcS $+$ Rand and ScB $+$ Rand respectively.

5 EXPERIMENTS

Datasets. We conduct experiments on synthetic data as well as two benchmark datasets: MS-COCO (Lin et al., 2014) and CC3M (Sharma et al., 2018). We present our experiments on synthetic data in the main paper and leave the remaining results to Appendix E.

Baselines. We compare the results from seven batching schemes: (i) Full-batch; (ii) $\binom{N}{B}$ mini-batches; (iii) FIXED-BATCH; (iv) SHUFFLED-BATCH; (v) OSGD; (vi)BcS and (vii) ScB. Detailed explanations can be found in Appendix 4.

Simulation on Synthetic Data. We consider the simpler setting of simulations where we generate embeddings by sampling N vectors from a d-dimensional Gaussian distribution and then normalize them so that $\|\boldsymbol{u}_i\| = \|\boldsymbol{v}_i\| = 1 \ \forall i$. Let $\mathcal{U} = \{\boldsymbol{u}_i\}_{i=1}^N$ and $\mathcal{V} = \{\boldsymbol{v}_i\}_{i=1}^N$. We then consider two regimes: (i) $d = 2N$, which satisfies the assumption $d \geq N - 1$ in our theoretical results, and (ii) $d = N/2$, which is more likely to occur in practice. For all experiments, we directly optimize embedding vectors $(\mathcal{U}, \mathcal{V})$ using SGD with the learning rate $\eta = 0.5$ with a projection step at every iteration to ensure that the embeddings remain unit norm. We set $B = 2$ for mini-batch schemes.

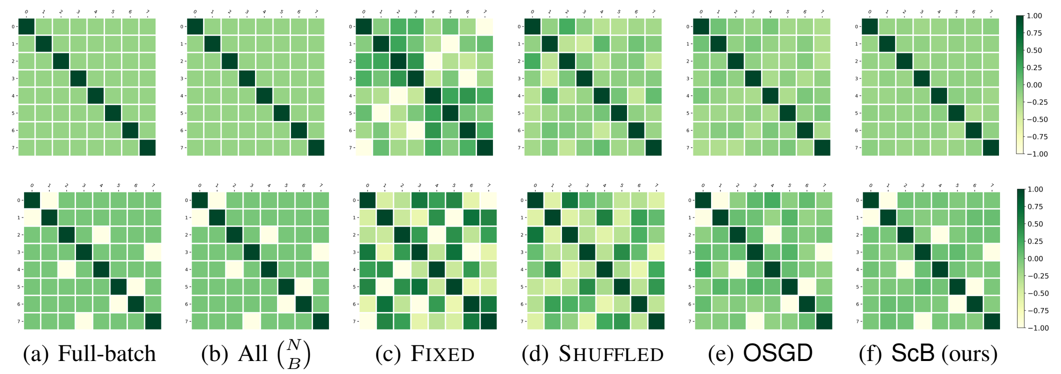


Figure 2: Heatmap of pairwise inner products for $N = 8, d = 16$ (Top row) and $N = 8, d = 4$ (Bottom row), when we optimize the N, d-dimensional embedding vectors using contrastive loss. The solution of minimizing all $\binom{N}{B}$ batches (b) is the same as that of full-batch minimization (a). Furthermore, the proposed algorithm (f) achieves the same solution in much fewer iterations.

Fig. 2 shows the heatmap plot of $N \times N$ gram matrix containing all the pairwise inner products $\boldsymbol{u}_i^\intercal \boldsymbol{v}_j$ of the embeddings, for $N = 8, d = 16$ (first row) and $N = 8, d = 4$ (second row). Broadly, we consider two settings: (a) full-batch contrastive learning, (b)-(f) mini-batch contrastive learning with different batch selection methods. As shown in Lemma 1, this results in the simplex ETF solution where $\boldsymbol{u}_i^\intercal \boldsymbol{v}_i$ is maximized for all i, and $\boldsymbol{u}_i^\intercal \boldsymbol{v}_j$ is minimized for $i \neq j$. For different batch selection methods we observe the following (especially in $d = 2N$ setting): (a) When we use full-batch *i.e.*, $B = N$, the trained embedding vectors converge to the simplex ETF solution with $\boldsymbol{u}_i^\intercal \boldsymbol{u}_i = 1$ for all i and $\boldsymbol{u}_i^\intercal \boldsymbol{u}_j = -1/(N-1)$ for all $i \neq j$; (b) When we use all $\binom{N}{B}$ mini-batches for training, the trained embedding vectors still reach the simplex ETF solution at the cost of extensive number of gradient steps; (c) Consider the case of having N/B non-overlapping mini-batches, when trained with this fixed mini-batch configuration, some pairs of trained embedding vectors have non-negative inner products, which is undesirable since it leads to higher loss; (d) When we follow the mini-batch configuration of (c) except that we shuffle the data at every epoch, the solution gets closer to simplex ETF; (e) When we select top-1 mini-batch that maximizes the mini-batch loss per iteration, the solution goes closer to (a); (f) Training with our method (in Algorithm 2) gives us the simplex ETF solution, with small memory/computation costs compared to (a) and (b). Note that this observation coincides with our theoretical results (Lem 1, Thm. 1, 2) and shows the superiority of our batch selection methods. When $d = N/2$, it shows a similar trend even though $d < N - 1$. As seen in the second row of Fig. 2, the solution of minimizing all $\binom{N}{B}$ batches (b) is the same as that of full-batch minimization (a). Furthermore, SCB (f) achieves the same solution in fewer iterations. See Appendix E.1 for more detailed experiments. Note that this still does not answer the question of generalization. Do the solutions identified by our batch-selection methods generalize to unseen data? To verify this, we also run experiments on “real” data (see Appendix E.3).

6 CONCLUSION

In this paper, we analyze contrastive learning under different batch selection methods and design effective and principled algorithms to optimize contrastive loss using mini-batches. We first provide theoretical analysis that characterizes the optimal solution of contrastive learning based on the batch selection method. We show that mini-batch optimization is equivalent to full-batch optimization if and only if all $\binom{N}{B}$ mini-batches are selected. However, since this approach is intractable, we propose computationally-efficient batch selection methods which construct mini-batches based on simple greedy algorithms. Our experimental results on both synthetic and real datasets show that our batch selection methods outperform conventional methods. We note that our assumption that $N \leq d + 1$ is restrictive and leave extending our theoretical results to the $N > d + 1$ regime as an interesting open problem.

REFERENCES

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.

Kenji Kawaguchi and Haihao Lu. Ordered sgd: A new stochastic optimization framework for empirical risk minimization. In *International Conference on Artificial Intelligence and Statistics*, pp. 669–679. PMLR, 2020.

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

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural networks. *arXiv preprint arXiv:1511.06343*, 2015.

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

Yucheng Lu, Si Yi Meng, and Christopher De Sa. A general analysis of example-selection for stochastic gradient descent. In *International Conference on Learning Representations*, 2021.

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

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2556–2565, 2018.

Mátyás A Sustik, Joel A Tropp, Inderjit S Dhillon, and Robert W Heath Jr. On the existence of equiangular tight frames. *Linear Algebra and its applications*, 426(2-3):619–635, 2007.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

A Additional Definitions

Definition 2 (Sustik et al. (2007)). A set of N vectors $\{\boldsymbol{u}_i\}_{i=1}^N$ in the $\mathbb{R}^d$ form an equiangular tight frame (ETF) if (i) they are all unit norm: $\|\boldsymbol{u}_i\| = 1$ for every $i \in [N]$, (ii) they are equiangular: $|\boldsymbol{u}_i^\intercal \boldsymbol{v}_i| = \alpha$ for some $\alpha \geq 0$ and (iii) they form a tight frame: $\mathbf{U}\mathbf{U}^\intercal = (N/d)\mathbb{I}_d$ where $\mathbf{U}$ is $d \times N$ matrix whose columns are $\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_N$, and $\mathbb{I}_d$ is the $d \times d$ identity matrix.

B Main theoretical results

Recall that the contrastive loss is defined as:

$$\mathcal{L}^{\text{con}}(\mathcal{U}, \mathcal{V}) := \frac{1}{N}\sum_{i=1}^{N} -\log\left(\frac{e^{\boldsymbol{u}_i^\intercal \boldsymbol{v}_i}}{\sum_{j=1}^{N} e^{\boldsymbol{u}_i^\intercal \boldsymbol{v}_j}}\right) + \frac{1}{N}\sum_{i=1}^{N} -\log\left(\frac{e^{\boldsymbol{v}_i^\intercal \boldsymbol{u}_i}}{\sum_{j=1}^{N} e^{\boldsymbol{v}_i^\intercal \boldsymbol{u}_j}}\right).$$

We denote the first term as one-sided contrastive loss

$$\mathcal{L}(\mathcal{U}, \mathcal{V}) = \frac{1}{N}\sum_{i=1}^{N} -\log\left(\frac{e^{\boldsymbol{u}_i^\intercal \boldsymbol{v}_i}}{\sum_{j=1}^{N} e^{\boldsymbol{u}_i^\intercal \boldsymbol{v}_j}}\right). \tag{3}$$

which is the standard InfoNCE loss where the positive example of an anchor $\boldsymbol{u}_i$ is the corresponding embedding vector from the other view $\boldsymbol{v}_i$.

Then, the contrastive loss is given by the sum of the two one-sided contrastive losses:

$$\mathcal{L}^{\text{con}}(\mathcal{U}, \mathcal{V}) = \mathcal{L}(\mathcal{U}, \mathcal{V}) + \mathcal{L}(\mathcal{V}, \mathcal{U}). \tag{4}$$

Since $\mathcal{L}^{\text{con}}$ is symmetric in its arguments, results pertaining to the optimum of $\mathcal{L}(\mathcal{U}, \mathcal{V})$ readily extend to $\mathcal{L}^{\text{con}}$. This is a property that we will use to simplify our proofs below.

Lemma 1 (Optimization with full-batch). *The optimal solution of the contrastive problem in Eq. (1) for full-batch satisfies $\boldsymbol{u}_i = \boldsymbol{v}_i$ for all $i \in [N]$, and $\{\boldsymbol{u}\}_{i=1}^N$ form a simplex ETF if $d \geq N - 1$.*

Proof. First, consider the simpler problem of minimizing the one-sided contrastive loss from Eq (3) which reduces the problem to exactly the same setting as Lu & Steinerberger (2020):

$$\begin{aligned}
\mathcal{L}(\mathcal{U}, \mathcal{V}) &= \frac{1}{N}\sum_{i=1}^{N} -\log\left(\frac{e^{\boldsymbol{u}_i^\intercal \boldsymbol{v}_i}}{\sum_{j=1}^{N} e^{\boldsymbol{u}_i^\intercal \boldsymbol{v}_j}}\right) \\
&= \frac{1}{N}\sum_{i=1}^{N} \log\left(1 + \sum_{j=1, j\neq i}^{N} e^{(\boldsymbol{v}_j - \boldsymbol{v}_i)^\intercal \boldsymbol{u}_i}\right).
\end{aligned}$$

Note that, we have for any fixed $1 \leq i \leq N$

$$\begin{aligned}
\sum_{j=1, j\neq i}^{N} e^{(\boldsymbol{v}_j - \boldsymbol{v}_i)^\intercal \boldsymbol{u}_i} &= e^{-(\boldsymbol{v}_i^\intercal \boldsymbol{u}_i)} \sum_{j=1, j\neq i}^{N} e^{\boldsymbol{v}_j^\intercal \boldsymbol{u}_i} \\
&= (N-1)e^{-(\boldsymbol{v}_i^\intercal \boldsymbol{u}_i)}\left(\frac{1}{N-1}\right)\sum_{j=1, j\neq i}^{N} e^{\boldsymbol{v}_j^\intercal \boldsymbol{u}_i} \\
&\overset{(a)}{\geq} (N-1)e^{-(\boldsymbol{v}_i^\intercal \boldsymbol{u}_i)} \exp\left(\frac{1}{N-1}\sum_{j=1, j\neq i}^{N} \boldsymbol{v}_j^\intercal \boldsymbol{u}_i\right) \\
&\overset{(b)}{=} (N-1)e^{-(\boldsymbol{v}_i^\intercal \boldsymbol{u}_i)} \exp\left(\frac{\boldsymbol{v}^\intercal \boldsymbol{u}_i - \boldsymbol{v}_i^\intercal \boldsymbol{u}_i}{N-1}\right) \\
&= (N-1)\exp\left(\frac{\boldsymbol{v}^\intercal \boldsymbol{u}_i - N(\boldsymbol{v}_i^\intercal \boldsymbol{u}_i)}{N-1}\right)
\end{aligned} \tag{5}$$

where (a) follows by applying Jensen's inequality for e^t and (b) follows from $\boldsymbol{v} := \sum_{i=1}^N \boldsymbol{v}_i$. Since $\log(\cdot)$ is monotonic, we have that $x > y \Rightarrow \log(x) > \log(y)$ and therefore,

$$\begin{aligned}
\mathcal{L}(\mathcal{U},\mathcal{V}) &\geq \sum_{i=1}^N \log\left[1 + (N-1)\exp\left(\frac{\boldsymbol{v}^\intercal \boldsymbol{u}_i}{N-1} - \frac{N(\boldsymbol{v}_i^\intercal \boldsymbol{u}_i)}{N-1}\right)\right] \\
&\overset{(c)}{\geq} N \,\log\left[1 + (N-1)\exp\left(\frac{1}{N}\sum_{i=1}^N\left(\frac{\boldsymbol{v}^\intercal \boldsymbol{u}_i}{N-1} - \frac{N(\boldsymbol{v}_i^\intercal \boldsymbol{u}_i)}{N-1}\right)\right)\right] \\
&\overset{(d)}{=} N\log\left[1 + (N-1)\exp\left(\frac{1}{N}\left(\frac{\boldsymbol{v}^\intercal \boldsymbol{u}}{N-1} - \frac{N}{N-1}\sum_{i=1}^N(\boldsymbol{v}_i^\intercal \boldsymbol{u}_i)\right)\right)\right]
\end{aligned} \tag{6}$$

where (c) follows by applying Jensen's inequality to the convex function $\phi(t) = \log(1 + ae^{bt})$ for $a, b > 0$, and (d) follow from $\boldsymbol{u} := \sum_{i=1}^N \boldsymbol{u}_i$.

Note that for equalities to hold in Eq. (5) and (6), we need constants c_i, c such that

$$\boldsymbol{v}_j^\intercal \boldsymbol{u}_i = c_i \quad \forall j \neq i, \tag{7}$$

$$\frac{\boldsymbol{v}^\intercal \boldsymbol{u}_i}{N-1} - \frac{N(\boldsymbol{v}_i^\intercal \boldsymbol{u}_i)}{N-1} = c \quad \forall i \in [N]. \tag{8}$$

Since $\log(\cdot)$ and $\exp(\cdot)$ are both monotonic, minimizing the lower bound in Eq. (5) is equivalent to

$$\begin{aligned}
&\min \quad \frac{\boldsymbol{v}^\intercal \boldsymbol{u}}{N-1} - \frac{N}{N-1}\sum_{i=1}^N \boldsymbol{v}_i^\intercal \boldsymbol{u}_i \\
\Leftrightarrow &\max \quad N\sum_{i=1}^N \boldsymbol{v}_i^\intercal \boldsymbol{u}_i - \left\langle \sum_{i=1}^N \boldsymbol{v}_i, \sum_{i=1}^N \boldsymbol{u}_i \right\rangle.
\end{aligned} \tag{9}$$

All that remains is to show that the solution that maximizes Eq 9 also satisfies the conditions in Eq. (7) and (8). To see this, first note that the maximization problem can be written as

$$\max \quad \boldsymbol{v}_{\text{stack}}^\intercal((N\mathbb{I}_N - \mathbf{1}_N\mathbf{1}_N^\intercal) \otimes \mathbb{I}_d)\boldsymbol{u}_{\text{stack}}$$

where $\boldsymbol{v}_{\text{stack}} = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n)$ is a vector in $\mathbb{R}^{Nd}$ formed by stacking the vectors $\boldsymbol{v}_i$ together. $\boldsymbol{u}_{\text{stack}}$ is similarly defined. Define $[n] := \{1, \cdots, n\}$ for a positive integer $n > 0$. $\mathbb{I}_N$ denotes the $N \times N$ identity matrix, $\mathbf{1}_N$ denotes the all-one vector in $\mathbb{R}^n$, and $\otimes$ denotes the Kronecker product. It is easy to see that $\|\boldsymbol{u}_{\text{stack}}\| = \|\boldsymbol{v}_{\text{stack}}\| = \sqrt{N}$ since each $\|\boldsymbol{u}_i\| = \|\boldsymbol{v}_i\| = 1$. Since the eigenvalues of $A \otimes B$ are the product of the eigenvalues of A and B, in order to analyze the spectrum of the middle term in the above maximization problem, it suffices to just consider the eigenvalues of $(N\mathbb{I}_N - \mathbf{1}_N\mathbf{1}_N^\intercal)$. As shown by the elegant analysis in Lu & Steinerberger (2020), $(N\mathbb{I}_N - \mathbf{1}_N\mathbf{1}_N^\intercal)\boldsymbol{p} = N\boldsymbol{p}$ for any $\boldsymbol{p} \in \mathbb{R}^N$ such that $\sum_{i=1}^N \boldsymbol{p}_i = 0$ and $(N\mathbb{I}_N - \mathbf{1}_N\mathbf{1}_N^\intercal)\boldsymbol{q} = 0$ for any $\boldsymbol{q} \in \mathbb{R}^N$ such that $\boldsymbol{q} = k\mathbf{1}_N$ for some $k \in \mathbb{R}$. Therefore it follows that its eigenvalues are N with multiplicity $(N-1)$ and 0. Since its largest eigenvalue is N and since $\|\boldsymbol{u}_{\text{stack}}\| = \|\boldsymbol{v}_{\text{stack}}\| = \sqrt{N}$, applying cauchy schwarz inequality, we have that

$$\begin{aligned}
&\max \quad \boldsymbol{v}_{\text{stack}}^\intercal(N\mathbb{I}_N - \mathbf{1}_N\mathbf{1}_N^\intercal) \otimes \mathbb{I}_d)\boldsymbol{u}_{\text{stack}}^\intercal \\
&= \|\boldsymbol{v}_{\text{stack}}\| \cdot \|(N\mathbb{I}_n - \mathbf{1}_n\mathbf{1}_n^\intercal) \otimes \mathbb{I}_d)\| \cdot \|\boldsymbol{u}_{\text{stack}}\| \\
&= \sqrt{N}(N)\sqrt{N} \\
&= N^2.
\end{aligned}$$

Moreover, we see that setting $\boldsymbol{u}_i = \boldsymbol{v}_i$ and setting $\{\boldsymbol{u}_i\}_{i=1}^N$ to be the simplex ETF attains the maximum above while also satisfying the conditions in Eq. (7) and (8) with $c_i = -1/(N-1)$ and $c = -N/(N-1)$. Therefore, the inequalities in Eq. (5) and (6) are actually equalities for $\boldsymbol{u}_i = \boldsymbol{v}_i$ when they are chosen to be the simplex ETF in $\mathbb{R}^d$ which is attainable since $d \geq N-1$. Therefore, we have shown that if $\mathcal{U}^* = \{\boldsymbol{u}_i^*\}_i = 1^N$ is the simplex ETF and $\boldsymbol{u}_i^* = \boldsymbol{v}_i^*$ $\forall i \in [N]$, then $\mathcal{U}^*, \mathcal{V}^* = arg\min_{\mathcal{U},\mathcal{V}} \mathcal{L}(\mathcal{U},\mathcal{V})$ over the unit sphere in $\mathbb{R}^n$. All that remains is to show that this is also the minimizer for $\mathcal{L}^{\text{con}}$.

First note that $\mathcal{U}^*, \mathcal{V}^*$ is also the minimizer for $\mathcal{L}(\mathcal{V}, \mathcal{U})$ through symmetry. One can repeat the proof exactly by simply exchanging $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ to see that this is indeed true. Now recalling Eq. (4), we have

$$\begin{aligned}\min \mathcal{L}^{\text{con}} &= \min\left(\mathcal{L}(\mathcal{U}, \mathcal{V}) + \mathcal{L}(\mathcal{U}, \mathcal{V})\right) \\ &\geq \min\left(\mathcal{L}(\mathcal{U}, \mathcal{V})\right) + \min\left(\mathcal{L}(\mathcal{U}, \mathcal{V})\right) \\ &= \mathcal{L}(\mathcal{U}^*, \mathcal{V}^*) + \mathcal{L}(\mathcal{V}^*, \mathcal{U}^*).\end{aligned} \tag{10}$$

However, since the minimizer of both terms in Eq. (10) is the same, the inequality becomes an equality giving us $\min \mathcal{L}^{\text{con}} = 2\mathcal{L}(\mathcal{U}^*, \mathcal{V}^*)$. Therefore, we have shown that $(\mathcal{U}^*, \mathcal{V}^*)$ is the minimizer of $\mathcal{L}^{\text{con}}$ completing the proof.

Remark 1. In the proof of the lemma, we only show that the simplex ETF attains the minimum of problem in Eq. (1), but not that it is the only minimizer. The proof of Lu & Steinerberger (2020) can be extended to show that this is indeed true as well. We omit it here for ease of exposition.

□

Proposition 1. *For any $B \geq 2$, there exist $\widetilde{\mathcal{U}}, \widetilde{\mathcal{V}}$ such that*

$$\frac{1}{\binom{N}{B}} \sum_{i=1}^{\binom{N}{B}} \mathcal{L}(\widetilde{\mathcal{U}}_{\mathbb{B}_i}, \widetilde{\mathcal{V}}_{\mathbb{B}_i}) \neq \mathcal{L}(\widetilde{\mathcal{U}}, \widetilde{\mathcal{V}}). \tag{11}$$

Moreover, for fixed B, there exists no constant C such that

$$\frac{1}{\binom{N}{B}} \sum_{i=1}^{\binom{N}{B}} \mathcal{L}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i}) = C \cdot \mathcal{L}(\mathcal{U}, \mathcal{V}) \quad \forall\, \mathcal{U}, \mathcal{V}. \tag{12}$$

Proof. Consider $\widetilde{\mathcal{U}}, \widetilde{\mathcal{V}}$ defined such that $\tilde{\boldsymbol{u}}_i = \tilde{\boldsymbol{v}}_i = \boldsymbol{e}_i \ \forall i \in [N]$. First note that $\tilde{\boldsymbol{u}}_i^\intercal \tilde{\boldsymbol{v}}_i = 1 \ \forall i \in [N]$ and $\tilde{\boldsymbol{u}}_i^\intercal \tilde{\boldsymbol{v}}_j = 0 \ \forall i \neq j$. Then,

$$\begin{aligned}\mathcal{L}(\widetilde{\mathcal{U}}, \widetilde{\mathcal{V}}) &= \frac{1}{N} \sum_{i=1}^{N} -\log\left(\frac{e^1}{e^1 + \sum_{j=1}^{N-1} e^0}\right) \\ &= \log(e + N - 1) - 1,\end{aligned} \tag{13}$$

$$\begin{aligned}\frac{1}{\binom{N}{B}} \sum_{i=1}^{\binom{N}{B}} \mathcal{L}(\widetilde{\mathcal{U}}_{\mathbb{B}_i}, \widetilde{\mathcal{V}}_{\mathbb{B}_i}) &= \frac{1}{\binom{N}{B}} \sum_{i=1}^{\binom{N}{B}} \frac{1}{B} \sum_{j \in B_i} -\log\left(\frac{e^1}{e^1 + \sum_{k=1}^{B-1} e^0}\right) \\ &= \log(e + B - 1) - 1.\end{aligned} \tag{14}$$

We now consider the second part of the statement. For contradiction, assume that there exists some $C \in \mathbb{R}$ such that Eq. (12) holds. Let $\widehat{\mathcal{U}}, \widehat{\mathcal{V}}$ be defined such that $\hat{\boldsymbol{u}}_i = \hat{\boldsymbol{v}}_i = \boldsymbol{e}_1 \ \forall i \in [N]$. Note that $\hat{\boldsymbol{u}}_i^\intercal \hat{\boldsymbol{v}}_j = 1 \ \forall i, j \in [N]$. Then,

$$\begin{aligned}\mathcal{L}(\widehat{\mathcal{U}}, \widehat{\mathcal{V}}) &= \frac{1}{N} \sum_{i=1}^{N} -\log\left(\frac{e^1}{e^1 + \sum_{j=1}^{N-1} e^1}\right) \\ &= \log(N),\end{aligned} \tag{15}$$

$$\begin{aligned}\frac{1}{\binom{N}{B}} \sum_{i=1}^{\binom{N}{B}} \mathcal{L}(\widehat{\mathcal{U}}_{\mathbb{B}_i}, \widehat{\mathcal{V}}_{\mathbb{B}_i}) &= \frac{1}{\binom{N}{B}} \sum_{i=1}^{\binom{N}{B}} \frac{1}{B} \sum_{j \in B_i} -\log\left(\frac{e^1}{e^1 + \sum_{k=1}^{B-1} e^1}\right) \\ &= \log(B).\end{aligned} \tag{16}$$

From Eq. (13) and 14, we have that $C = \frac{\log(e+B-1)-1}{\log(e+N-1)-1}$. Whereas from Eq. (15) and (16), we have that $C = \frac{\log(B)}{\log(N)}$ which is a contradiction. Therefore, there exists no $C \in \mathbb{R}$ such that Eq. (12) holds. This completes the proof. □

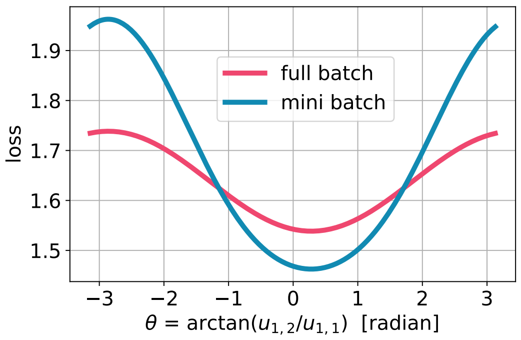


Figure 3: A comparison of mini-batch loss (LHS of Eq. (11)) and full-batch loss (RHS of Eq. (11)).

We illustrate the above proposition by visualizing the two loss functions in Fig. 3 when $N = 10, B = 2, d = 2$. Since these are still 40-dimensional functions, we visualize it along a single embedding vector $\boldsymbol{u}_1$ by freezing all other embeddings at the optimal solution and varying $\boldsymbol{u}_1 = [u_{1,1}, u_{1,2}]$ as $[\cos(\theta), \sin(\theta)]$ for $\theta \in [-\pi, \pi]$. One can confirm that two losses are not identical (even up to scaling), which corroborates the result of Prop. 1.

Theorem 1 (Optimization with all possible $\binom{N}{B}$ mini-batches)**.** *Suppose $d \geq N-1$ and $B \geq 2$. The optimal solution of the mini-batch contrastive problem in Eq. (2) for $\mathcal{S} = \left[\binom{N}{B}\right]$ satisfies $\boldsymbol{u}_i = \boldsymbol{v}_i$ for all $i \in [N]$, and $\{\boldsymbol{u}_i\}_{i=1}^N$ form a simplex ETF. Therefore, the minimizer of this mini-batch problem is the same as that of the full-batch problem in Eq. (1).*

Proof. For simplicity, first consider just one of the two terms in the two-sided loss. Therefore, the optimization problem becomes

$$\begin{aligned} \min_{\mathcal{U},\mathcal{V}} \quad & \frac{1}{\binom{N}{B}} \sum_{i=1}^{\binom{N}{B}} \{\mathcal{L}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i})\} \\ s.t. \quad & \|\boldsymbol{u}_i\| = 1, \|\boldsymbol{v}_i\| = 1 \ \forall i \in [N]. \end{aligned}$$

Similar to the proof of Lem. 1, we have that

$$\begin{aligned} \sum_{i=1}^{\binom{N}{B}} \mathcal{L}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i}) &= \sum_{i=1}^{\binom{N}{B}} \sum_{j \in \mathbb{B}_i} \log \left(1 + \sum_{\substack{k \in \mathbb{B}_i \\ k \neq j}} e^{\boldsymbol{u}_j{}^\intercal (\boldsymbol{v}_k - \boldsymbol{v}_j)} \right) \\ &\overset{(a)}{\geq} \sum_{i=1}^{\binom{N}{B}} \sum_{j \in \mathbb{B}_i} \log \left(1 + (B-1) \exp \left(\frac{\sum_{k \in \mathbb{B}_i, k \neq j} \boldsymbol{u}_j^\intercal (\boldsymbol{v}_k - \boldsymbol{v}_j)}{B-1} \right) \right) \\ &= \sum_{i=1}^{\binom{N}{B}} \sum_{j \in \mathbb{B}_i} \log \left(1 + (B-1) \exp \left(\frac{\sum_{k \in \mathbb{B}_i} (\boldsymbol{u}_j^\intercal \boldsymbol{v}_k - B \boldsymbol{u}_j^\intercal \boldsymbol{v}_j)}{B-1} \right) \right) \\ &\overset{(b)}{\geq} \binom{N}{B} \cdot B \log \left(1 + (B-1) \exp \left(\frac{\sum_{i=1}^{\binom{N}{B}} \sum_{j \in \mathbb{B}_i} \sum_{k \in \mathbb{B}_i} \boldsymbol{u}_j^\intercal \boldsymbol{v}_k - \sum_{i=1}^{\binom{N}{B}} \sum_{j \in \mathbb{B}_i} B \boldsymbol{u}_j^\intercal \boldsymbol{v}_j}{\binom{N}{B} \cdot B \cdot (B-1)} \right) \right) \end{aligned}$$

where (a) and (b) follows by applying Jensen's inequality to e^t and $\log(1 + ae^{bt})$ for $a, b > 0$, respectively. Note that for equalities to hold in Jensen's inequalities, we need constants c_j, c such that

$$\boldsymbol{u}_j^\intercal \boldsymbol{v}_k = c_j \quad \forall k \neq j, \tag{17}$$

$$\frac{\boldsymbol{u}^\intercal \boldsymbol{v}_i}{N-1} - \frac{N(\boldsymbol{u}_i^\intercal \boldsymbol{v}_i)}{N-1} = c \quad \forall i \in [N]. \tag{18}$$

Now, we carefully consider the two terms in the numerator:

$$A_1 := \sum_{i=1}^{\binom{N}{B}} \sum_{j \in \mathbb{B}_i} \sum_{k \in \mathbb{B}_i} \boldsymbol{u}_j^\intercal \boldsymbol{v}_k,$$
$$A_2 := \sum_{i=1}^{\binom{N}{B}} \sum_{j \in \mathbb{B}_i} B \boldsymbol{u}_j^\intercal \boldsymbol{v}_j.$$

To simplify A_1, first note that for any fixed $l, m \in [N]$ such that $l \neq m$, there are $\binom{N-2}{B-2}$ batches that contain l and m. And for $l = m$, there are $\binom{N-1}{B-1}$ batches that contain that pair. Since these terms all occur in A_1, we have that

$$\begin{aligned} A_1 &= \binom{N-2}{B-2} \sum_{l=1}^N \sum_{m=1}^N \boldsymbol{u}_l^\intercal \boldsymbol{v}_m + \left[\binom{N-1}{B-1} - \binom{N-2}{B-2}\right] \sum_{l=1}^N \boldsymbol{u}_l^\intercal \boldsymbol{v}_l \\ &= \binom{N-2}{B-2} \sum_{l=1}^N \sum_{m=1}^N \boldsymbol{u}_l^\intercal \boldsymbol{v}_m + \binom{N-2}{B-2} \left(\frac{N-B}{B-1}\right) \sum_{l=1}^N \boldsymbol{u}_l^\intercal \boldsymbol{v}_l \end{aligned}$$

where the final equality follows from the simple result that

$$\left[\binom{N-1}{B-1} - \binom{N-2}{B-2}\right] = \frac{(N-1)(N-2)!}{(N-B)!(B-1)(B-2)!} - \binom{N-2}{B-2} = \left(\frac{N-1}{B-1} - 1\right)\binom{N-2}{B-2}.$$

Similarly, we have that

$$A_2 = \binom{N-1}{B-1} B \sum_{l=1}^N \boldsymbol{u}_l^\intercal \boldsymbol{v}_l.$$

Plugging these back into the above inequality, we have that

$$\sum_{i=1}^{\binom{N}{B}} \mathcal{L}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i}) \geq \binom{N}{B} B \log\left(1 + (B-1)\exp\left(\frac{\binom{N-2}{B-2}\sum_{l=1}^N \sum_{m=1}^N \boldsymbol{u}_l^\intercal \boldsymbol{v}_m + \binom{N-2}{B-2}\left(\frac{N-B}{B-1}\right)\sum_{l=1}^N \boldsymbol{u}_l^\intercal \boldsymbol{v}_l - \binom{N-1}{B-1} B \sum_{l=1}^N \boldsymbol{u}_l^\intercal \boldsymbol{v}_l}{\binom{N}{B} B(B-1)}\right)\right).$$

Note that

$$\begin{aligned} \binom{N-2}{B-2}\left(\frac{N-B}{B-1}\right) - \binom{N-1}{B-1} B &= \binom{N-2}{B-2}\left(\frac{N-B}{B-1}\right) - \binom{N-2}{B-2}\left(\frac{N-1}{B-1}\right) B \\ &= \binom{N-2}{B-2}\left(\frac{N-B}{B-1} - \frac{NB-B}{B-1}\right) \\ &= -\binom{N-2}{B-2} N, \end{aligned}$$

and

$$\frac{\binom{N-2}{B-2}}{\binom{N}{B} B(B-1)} = \frac{1}{N(N-1)}.$$

Putting these together, we have that

$$\begin{aligned} \sum_{i=1}^{\binom{N}{B}} \mathcal{L}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i}) &\geq \binom{N}{B} B \log\left(1 + (B-1)\exp\left(\frac{\sum_{l=1}^N \sum_{m=1}^N \boldsymbol{u}_l^\intercal \boldsymbol{v}_m - N\sum_{l=1}^N \boldsymbol{u}_l^\intercal \boldsymbol{v}_l}{N(N-1)}\right)\right) \\ &= \binom{N}{B} B \log\left(1 + (B-1)\exp\left(\frac{\boldsymbol{u}^\intercal \boldsymbol{v} - N\sum_{i=1}^N \boldsymbol{u}_i^\intercal \boldsymbol{v}_i}{N(N-1)}\right)\right). \end{aligned}$$

Observe that the term inside the exponential is identical to Eq. (6) and therefore, we can reuse the same spectral analysis argument to show that the simplex ETF also minimizes $\sum_{i=1}^{\binom{N}{B}} \mathcal{L}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i})$. Once again, since the proof is symmetric the simplex ETF also minimizes $\sum_{i=1}^{\binom{N}{B}} \mathcal{L}(\mathcal{V}_{\mathbb{B}_i}, \mathcal{U}_{\mathbb{B}_i})$. $\square$

Theorem 2 (Optimization with fewer than $\binom{N}{B}$ mini-batches)**.** *For $B = 2$, the minimizer of Eq. (2) for $\mathcal{S} \subsetneq \left[\binom{N}{2}\right]$ is not the minimizer of the full-batch loss in Eq. (1).*

Proof. Consider a set of batches $\mathcal{S} \subset \left[\binom{N}{2}\right]$ with the batch size $B = 2$. Without loss of generality, assume that $(1, 2) \notin \bigcup_{i \in \mathcal{S}} \{\mathbb{B}_i\}$. For contradiction assume that the simplex ETF - $(\mathcal{U}^*, \mathcal{V}^*)$ is indeed the optimal solution of the loss over these $\mathcal{S}$ batches. Then, by definition, we have that for any $(\mathcal{U}, \mathcal{V}) \neq (\mathcal{U}^*, \mathcal{V}^*)$

$$\begin{aligned}
&\frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \mathcal{L}(\mathcal{U}^*_{\mathbb{B}_i}, \mathcal{V}^*_{\mathbb{B}_i}) \leq \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \mathcal{L}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i}) \\
&\Rightarrow \sum_{i \in \mathcal{S}} \mathcal{L}(\mathcal{U}^*_{\mathbb{B}_i}, \mathcal{V}^*_{\mathbb{B}_i}) \leq \sum_{i \in \mathcal{S}} \mathcal{L}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i}) \qquad (19)
\end{aligned}$$

where $(\mathcal{U}^*, \mathcal{V}^*)$ is defined such that $\boldsymbol{u}_i^* = \boldsymbol{v}_i^*$ for all $i \in [N]$ and $\boldsymbol{u}_i^{*\top} \boldsymbol{v}_j^* = -1/(N-1)$ for all $i \neq j$. Also recall that $\|\boldsymbol{u}_i\| = \|\boldsymbol{v}_i\| = 1$ for all $i \in [N]$. Therefore, we also have

$$\begin{aligned}
\sum_{i \in \mathcal{S}} \mathcal{L}(\mathcal{U}^*_{\mathbb{B}_i}, \mathcal{V}^*_{\mathbb{B}_i}) &= \sum_{i \in \mathcal{S}} \sum_{j \in \mathbb{B}_i} \log \left(1 + \sum_{k \in \mathbb{B}_i, k \neq j} \exp \left(\boldsymbol{u}_j^{*\top} (\boldsymbol{v}_k^* - \boldsymbol{v}_j^*) \right) \right) \\
&= \sum_{i \in \mathcal{S}} \sum_{j \in \mathbb{B}_i} \log \left(1 + \sum_{k \in \mathbb{B}_i, k \neq j} \exp \left(-\frac{1}{N-1} - 1 \right) \right) \\
&= \sum_{i \in \mathcal{J}} \sum_{j \in \mathbb{B}_i} \log \left(1 + \exp \left(-\frac{1}{N-1} - 1 \right) \right) \qquad (20)
\end{aligned}$$

where the last equality is due to the fact that $|\mathbb{B}_i| = 2$.

Now, let us consider $(\widetilde{\mathcal{U}}, \widetilde{\mathcal{V}})$ defined such that $\tilde{\boldsymbol{u}}_i = \tilde{\boldsymbol{v}}_i$ for all $i \in [N]$, and $\tilde{\boldsymbol{u}}_i^\top \tilde{\boldsymbol{v}}_j = -1/(N-2)$ for all $i \neq j, (i, j) \notin \{(1, 2), (2, 1)\}$. Intuitively, this is equivalent to placing $\tilde{\boldsymbol{u}}_2, \ldots, \tilde{\boldsymbol{u}}_N$ on a simplex ETF of $N-1$ points and setting $\tilde{\boldsymbol{u}}_1 = \tilde{\boldsymbol{u}}_2$. This is clearly possible because $d > N-1 \Rightarrow d > N-2$ which is the condition required to place $N-1$ points on a simplex ETF in $\mathbb{R}^d$. Now,

$$\begin{aligned}
\sum_{i \in \mathcal{S}} \mathcal{L}(\widetilde{\mathcal{U}}_{\mathbb{B}_i}, \widetilde{\mathcal{V}}_{\mathbb{B}_i}) &= \sum_{i \in \mathcal{S}} \sum_{j \in \mathbb{B}_i} \log \left(1 + \sum_{k \in \mathbb{B}_i, k \neq j} \exp \left(\tilde{\boldsymbol{u}}_j^\top (\tilde{\boldsymbol{v}}_k - \tilde{\boldsymbol{v}}_j) \right) \right) \\
&= \sum_{i \in \mathcal{S}} \sum_{j \in \mathbb{B}_i} \log \left(1 + \sum_{k \in \mathbb{B}_i, k \neq j} \exp \left(-\frac{1}{N-2} - 1 \right) \right) \\
&= \sum_{i \in \mathcal{S}} \sum_{j \in \mathbb{B}_i} \log \left(1 + \exp \left(-\frac{1}{N-2} - 1 \right) \right) \qquad (21)
\end{aligned}$$

where the last equality follows since $(1, 2) \notin \bigcup_{i \in \mathcal{S}} \{\mathbb{B}_i\}$. It is easy to see from Eq. (20) and 21 that $\sum_{i \in \mathcal{S}} \mathcal{L}(\widetilde{\mathcal{U}}_{\mathbb{B}_i}, \widetilde{\mathcal{V}}_{\mathbb{B}_i}) < \sum_{i \in \mathcal{S}} \mathcal{L}(\mathcal{U}^*_{\mathbb{B}_i}, \mathcal{V}^*_{\mathbb{B}_i})$ which contradicts Eq. (19). Therefore, the optimal solution of the contrastive loss over any $\mathcal{S} \subset \left[\binom{N}{2}\right]$ batches is not the simplex ETF completing the proof. $\square$

Proposition 2. *Suppose $B \geq 2$, and let $\mathcal{S} \subseteq \left[\binom{N}{B}\right]$ be a set of mini-batch indices. If there exist two data points that never belong together in any mini-batch, i.e., $\exists i, j \in [N]$ s.t. $\{i, j\} \not\subset \mathbb{B}_k$ for any $k \in \mathcal{S}$, then the optimal solution of this problem is not the minimizer of the full-batch problem in Eq. (1).*

Proof. The proof follows in a fairly similar manner to that of Thm. 2. Consider a set of batches of size $B \geq 2$, $\mathcal{S} \subset [\binom{N}{B}]$. Without loss of generality, assume that $\{1, 2\} \not\subset \mathbb{B}_k$ for any $k \in \mathcal{S}$. For contradiction, assume that the simplex ETF - $(\mathcal{U}^*, \mathcal{V}^*)$ is the optimal solution of the loss over these $\mathcal{S}$ batches. Then, by definition, we have that for any $(\mathcal{U}, \mathcal{V}) \neq (\mathcal{U}^*, \mathcal{V}^*)$

Once again, for contradiction assume that the simplex ETF - $(\mathcal{U}^*, \mathcal{V}^*)$ is indeed the optimal solution of the loss over these $\mathcal{S}$ batches. Then, by definition for any $(\mathcal{U}, \mathcal{V}) \neq (\mathcal{U}^*, \mathcal{V}^*)$

$$\begin{aligned}
&\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}\mathcal{L}(\mathcal{U}^*_{\mathbb{B}_i}, \mathcal{V}^*_{\mathbb{B}_i}) \leq \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}\mathcal{L}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i}) \\
&\Rightarrow \sum_{i\in\mathcal{S}}\mathcal{L}(\mathcal{U}^*_{\mathbb{B}_i}, \mathcal{V}^*_{\mathbb{B}_i}) \leq \sum_{i\in\mathcal{S}}\mathcal{L}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i})
\end{aligned} \tag{22}$$

where $(\mathcal{U}^*, \mathcal{V}^*)$ is defined such that $\boldsymbol{u}_i^* = \boldsymbol{v}_i^*$ for all $i \in [N]$ and $\boldsymbol{u}_i^{*\mathsf{T}}\boldsymbol{v}_j^* = -1/(N-1)$ for all $i \neq j$. Also recall that $\|\boldsymbol{u}_i\| = \|\boldsymbol{v}_i\| = 1$ for all $i \in [N]$. Therefore, we also have

$$\begin{aligned}
\sum_{i\in\mathcal{S}}\mathcal{L}(\mathcal{U}^*_{\mathbb{B}_i}, \mathcal{V}^*_{\mathbb{B}_i}) &= \sum_{i\in\mathcal{S}}\sum_{j\in\mathbb{B}_i}\log\left(1 + \sum_{k\in\mathbb{B}_i, k\neq j}\exp\left(\boldsymbol{u}_j^{*\mathsf{T}}(\boldsymbol{v}_k^* - \boldsymbol{v}_j^*)\right)\right) \\
&= \sum_{i\in\mathcal{S}}\sum_{j\in\mathbb{B}_i}\log\left(1 + \sum_{k\in\mathbb{B}_i, k\neq j}\exp\left(-\frac{1}{N-1} - 1\right)\right) \\
&= \sum_{i\in\mathcal{J}}\sum_{j\in\mathbb{B}_i}\log\left(1 + (B-1)\exp\left(-\frac{1}{N-1} - 1\right)\right)
\end{aligned} \tag{23}$$

Now, let us consider $(\widetilde{\mathcal{U}}, \widetilde{\mathcal{V}})$ defined such that $\tilde{\boldsymbol{u}}_i = \tilde{\boldsymbol{v}}_i$ for all $i \in [N]$, $\tilde{\boldsymbol{u}}_2 = \tilde{\boldsymbol{v}}_2$ and $\tilde{\boldsymbol{u}}_i^{\mathsf{T}}\tilde{\boldsymbol{v}}_j = -1/(N-2)$ for all $i \neq j, (i,j) \notin \{(1,2),(2,1)\}$. Once again, note that this is equivalent to placing $\tilde{\boldsymbol{u}}_2, \ldots, \tilde{\boldsymbol{u}}_N$ on a simplex ETF of $N-1$ points and setting $\tilde{\boldsymbol{u}}_1 = \tilde{\boldsymbol{u}}_2$. Now,

$$\begin{aligned}
\sum_{i\in\mathcal{S}}\mathcal{L}(\widetilde{\mathcal{U}}_{\mathbb{B}_i}, \widetilde{\mathcal{V}}_{\mathbb{B}_i}) &= \sum_{i\in\mathcal{S}}\sum_{j\in\mathbb{B}_i}\log\left(1 + \sum_{k\in\mathbb{B}_i, k\neq j}\exp\left(\tilde{\boldsymbol{u}}_j^{\mathsf{T}}(\tilde{\boldsymbol{v}}_k - \tilde{\boldsymbol{v}}_j)\right)\right) \\
&= \sum_{i\in\mathcal{S}}\sum_{j\in\mathbb{B}_i}\log\left(1 + \sum_{k\in\mathbb{B}_i, k\neq j}\exp\left(-\frac{1}{N-2} - 1\right)\right) \\
&= \sum_{i\in\mathcal{S}}\sum_{j\in\mathbb{B}_i}\log\left(1 + (B-1)\exp\left(-\frac{1}{N-2} - 1\right)\right)
\end{aligned} \tag{24}$$

where for the final equality note that following. The only pair for which $\tilde{\boldsymbol{u}}_j^{\mathsf{T}}\tilde{\boldsymbol{v}}_k \neq -1/(N-2)$ is $(j,k) = (1,2)$. Since there is no $i \in \mathcal{S}$ such that $\{1,2\} \in \mathbb{B}_i$, this term never appears in our loss. From Eq. (23) and Eq. (24), we have that $\sum_{i\in\mathcal{S}}\mathcal{L}(\widetilde{\mathcal{U}}_{\mathbb{B}_i}, \widetilde{\mathcal{V}}_{\mathbb{B}_i}) < \sum_{i\in\mathcal{S}}\mathcal{L}(\mathcal{U}^*_{\mathbb{B}_i}, \mathcal{V}^*_{\mathbb{B}_i})$ which contradicts Eq. (22). Therefore, the optimal solution of the contrastive loss over any such $\mathcal{S} \subset [\binom{N}{B}]$ batches is not the simplex ETF completing the proof.

$\square$

Proposition 3. *Suppose $B \geq 2$, and let $\mathcal{S} \subseteq \left[\binom{N}{B}\right]$ be a set of mini-batch inidices satisfying $\mathbb{B}_i \bigcap \mathbb{B}_j = \varnothing, \forall i,j \in \mathcal{S}$ and $\bigcup_{i\in\mathcal{S}}\mathbb{B}_i = [N]$, i.e., $\{\mathbb{B}_i\}_{i\in\mathcal{S}}$ forms non-overlapping mini-batches that cover all data samples. Then, the minimizer of the mini-batch loss optimization problem in Eq. (2) is different from the minimizer of the full-batch loss optimization problem in Eq. (1).*

Proof. Applying Lem. 1 specifically to a single batch $\mathbb{B}_i$ gives us that the optimal solution for just the loss over this batch is the simplex ETF over B points. In the case of non-overlapping batches, the objective function can be separated across batches and therefore the optimal solution for the sum of the losses is equal to the solution of minimizing each term independently. More precisely, we have

$$\min_{\mathcal{U},\mathcal{V}} \sum_{i=1}^{N/B} \mathcal{L}^{\text{con}}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i}) = \sum_{i=1}^{N/B} \min_{\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i}} \mathcal{L}^{\text{con}}(\mathcal{U}_{\mathbb{B}_i}, \mathcal{V}_{\mathbb{B}_i})$$

where $\mathcal{U}_{\mathbb{B}_i} = \{u_j : j \in \mathbb{B}_i\}$ and $\mathcal{V}_{\mathbb{B}_i} = \{v_j : j \in \mathbb{B}_i\}$, respectively, and the equality follows from the fact that $\mathbb{B}_i$'s are disjoint.

□

C Algorithm Descriptions

Algorithm 1 BcS Algorithm

```
Input: the number of positive pairs $N$, batch size $B$, embedding vectors $\mathcal{U} = \{\boldsymbol{u}\}_{i=1}^N$, $\mathcal{V} = \{\boldsymbol{v}\}_{i=1}^N$
Output: selected batches $\{\mathbb{B}_j\}_{j=1}^{N/B}$
$\mathcal{I} \leftarrow [N]$   # Indices of samples
$\mathcal{J} \leftarrow [N/B]$   # Indices of batches
for $r = 1$ to $B$ do
  for $j = 1$ to $N/B$ do
    if $r = 1$ then
      $i^\star \leftarrow \text{Unif}(\mathcal{I})$
      $\mathbb{B}_j \leftarrow \{i^\star\}$   # Batch initialization
    else
      $i^\star \leftarrow \arg\max_{i \in \mathcal{I}} \mathcal{L}^{\text{con}}(\mathcal{U}_{\mathbb{B}_j \cup \{i\}}, \mathcal{V}_{\mathbb{B}_j \cup \{i\}})$
      $\mathbb{B}_j \leftarrow \mathbb{B}_j \cup \{i^\star\}$
    end if
    $\mathcal{I} \leftarrow \mathcal{I} \setminus \{i^\star\}$
  end for
end for
Return $\{\mathbb{B}_j\}_{j=1}^{N/B}$
```

D Experiment Details

In this section, we provide additional details of the configurations for the experiments shown in section 5. To make a fair comparison with the original CLIP model, we implement our algorithms based on the official CLIP repository[1]. All experiments are performed on local NVIDIA DGX A100 Tensor Core GPUs with 40GB of memory each. Our anonymized codebase can be found at `https://anonymous.4open.science/r/mini-batch-cl-0EC4/`.

D.1 Datasets

We conduct our evaluation on multiple benchmark datasets including MS-COCO (Lin et al., 2014) and CC3M (Sharma et al., 2018) comprised of image-caption pairs for multi-modal retrieval tasks. The fine-tuning performance of different batch selection strategies reported in Table 2, are measured on these datasets reshaped in 224x224 pixel resolution to enable the RN50 backbone to process the input.

MS-COCO. Microsoft Common Objects in COntext (MS-COCO) (Lin et al., 2014) is a high quality image dataset containing together with captions and labels. There are 5 captions and different number of labels *i.e.*, objects, corresponding to each image. The dataset we tested on is the 2017 version downloaded from the official web page[2], which consists of 118,757 training images and

[1]https://github.com/mlfoundations/open_clip
[2]https://cocodataset.org

Algorithm 2 ScB Algorithm

```
Input: the number of positive pairs $N$, batch size $B$, embedding vectors $\mathcal{U} = \{\boldsymbol{u}\}_{i=1}^{N}$, $\mathcal{V} = \{\boldsymbol{v}\}_{i=1}^{N}$
Output: selected batches $\{\mathbb{B}_j\}_{j=1}^{N/B}$
$\mathcal{I} \leftarrow [N]$  # Indices of samples
$\mathcal{J} \leftarrow [N/B]$  # Indices of batches
for $r = 1$ to $N$ do
  $i \leftarrow \text{Unif}(\mathcal{I})$
  if $r \leq N/B$ then
    $j^* \leftarrow \text{Unif}(\mathcal{J})$
    $\mathbb{B}_{j^*} \leftarrow \{i\}$  # Batch initialization
  else
    $j^\star \leftarrow \arg\max_{j \in \mathcal{J}} \mathcal{L}^{\text{con}}(\mathcal{U}_{\mathbb{B}_j \cup \{i\}}, \mathcal{V}_{\mathbb{B}_j \cup \{i\}})$
    $\mathbb{B}_{j^\star} \leftarrow \mathbb{B}_{j^\star} \cup \{i\}$
  end if
  $\mathcal{J} \leftarrow \mathcal{J} \setminus \{j^\star\}$
  $\mathcal{I} \leftarrow \mathcal{I} \setminus \{i\}$
  if $\mathcal{J} = \varnothing$ then
    $\mathcal{J} \leftarrow [N/B]$
  end if
end for
Return $\{\mathbb{B}_j\}_{j=1}^{N/B}$
```

5,000 images for validation. We sample the dataset to retain *one caption per image* pairs and limit the caption diversity of images in the dataset from consideration.

CC3M. Conceptual Captions 3M (Sharma et al., 2018) contains 3,318,333 image and description pairs for the training set. In constrast with MS-COCO, each image has one ground truth caption harvested from the web. Since Google currently provides the dataset as image URL and caption pairs[3], we developed a downloader ourselves to prepare the dataset. The successfully downloaded data size is 2,856,515 for train set and 13,137 for validation set respectively. Since we measure performance via fine-tuning, we randomly sample a 4.21% subset of the training set instead of using the entire data to ease the computational burden. The choice of the fraction 4.21% is to match the training set size of MS-COCO.

Since both datasets do not provide an annotated test split, we randomly split each training set in order to retain a separate validation set for hyper-parameter optimization. The original validation sets are instead used as test sets to report the performance on the unused data. In details, we make the 5,000 samples of MS-COCO and the 13,137 samples of CC3M unseen during the hyper-parameter selection process until the phase of evaluation. Therefore, it is important to note that validation set denotes a part of training set and test set denotes the original validation set throughout this paper. After we search through every combinations of hyper-parameters to find the most optimal values on this validation set, we evaluate our models on test sets.

D.2 Model Architecture

We use the modified ResNet-50 which is the smallest among 5 ResNet-based CLIP models provided by Radford et al. (2021) in all of our experiments. There are two main reasons for the selection: First, a small model is able to show relatively faster results with less training efforts, and secondly, the ultimate goal of this work is to demonstrate the effectiveness of our framework on a smaller scale compared to the original CLIP rather than to achieve state-of-the-art performance. This pre-trained CLIP model, denoted as RN50 is used as is for training without any modification in the architecture.

[3]https://github.com/google-research-datasets/conceptual-captions

D.3 Hyperparameters

In all experiments with different batch selection strategies, hyperparameter search is performed on a validation split of each dataset stated in D.1 to ensure optimal performance. Table D.3 shows the range of values specified for each hyperparameter.

Table 1: Hyperparameter search for all batch selection methods.

Hyperparameter	Search space
Batch size	[100, 200, 400]
Warmup step	[3600, 1800, 900]
Learning factor	[0.00005, 0.0001, 0.0005]
Weight decay	0.1
Optimizer	AdamW
Momentum	0.9
Random sampling	30%, 50%, 70%

We use AdamW optimizer with weight decay of 0.1, while the remaining parameters except learning rate are the same as the default[4]: $\beta_1 = 0.9, \beta_2 = 0.999$ and $\epsilon = 10^{-8}$. Inspired by Vaswani et al. (2017), we search among various learning rate factors using grid search on this formula.

$$\text{lrate} = \text{lr\_factor} \times \min\{\text{step\_num}^{-0.5}, \text{step\_num} \times \text{warmup\_step}^{-1.5}\} \tag{25}$$

The warmup step varies according to the batch size, for example, when the batch size is 200, the corresponding warmup step is 1800 as described in Table D.3. The initial temperature parameter τ from the RN50 model is approximately 4.6 and it changes as the training proceeds. The algorithms mostly converge within 10 epochs with any parameter combinations we make with the lists.

We add r that denotes a random sampling ratio as a hyper-parameter to carefully observe the effect of our batch selection algorithms by adjusting the level of difficulty of each batch. The initially given $r\%$ remains the same in every epoch until convergence. The ratio is tuned for each batch size and algorithm in the same manner as with other parameters and the best results are reported in Section E.3.

E Additional Experimental Results

E.1 Simulation Results

Below, we present the heatmap of pairwise inner products for different combinations of d and N. Fig. 4 shows the values of all pairwise inner products $\boldsymbol{u}_i^\intercal \boldsymbol{v}_j$ for different batching schemes via heat maps for $d = 2N$. (a) demonstrates that the result of minimizing the full-batch loss in Eq. (1) converges to the simplex ETF. Note that all diagonal elements are equal to 1 and all off-diagonal elements are equal to $-1/(N-1)$. (b) shows that the result of minimizing the contrastive loss w.r.t. all $\binom{N}{B}$ mini-batches converges to the same solution of minimizing the full-batch loss which agrees with Thm 1. We observe that the result of our proposed batch selection algorithms ((f), (g)) converges to the ETF solution while the results with FIXED BATCH and SHUFFLED BATCH do not ((c), (d)). OSGD performs better than the baselines but still appears to have converged to a solution different from the simplex ETF (e).

Fig. 5 represents the experimental results for $d = N/2$ which is the more practical setting. Happily, we still get similar results even though our assumption $d \geq N - 1$ is violated: 1) the solution minimizing the contrastive loss over all $\binom{N}{B}$ mini-batches converges to the solution of minimizing the full-batch loss, and 2) the solution of the proposed algorithm converges to that of minimizing the full-batch loss while the result with `fixed batch` and `shuffled batch` do not. However, there are two things to point out compared to the previous $d = 2N$ regime. First, convergence is

[4]https://pytorch.org/docs/stable/generated/torch.optim.AdamW

much slower, particularly for larger N. For example, the minimization over all $\binom{N}{B}$ mini-batches did not converge until 50,000 epochs, well after the other batch selection schemes. Next, we note that we are unable to recover the exact solution as that of full-batch minimization although they are mere permutations of each other. For example, the results from BcS and ScB for the last two rows ($N = 16, 20$), are different permutations of the solution to the full-batch minimization problem. It is important to note that these solutions are still equivalent in our case since the order of embeddings is artificial.

E.2 Simulation Results with shared encoder

To simulate with more practical settings, we train a tiny model considering it as shared encoder. For the model, we use a single fully connected layer without activation. We only test $N = \{4, 8, 12\}$ and omit $N = \{16, 20\}$ for the simplicity. We first sample data from a 4-dimensional Gaussian distribution. And then pass it to our model and normalize them like in Appendix E.1 during training. We use SGD with the learning rate $\eta = 0.1$ and train until $10,000$ epochs. The output dimension d of our model are $d = 2N$ and $d = N/2$. As in Fig. 6 and 7, we can see the similar trend to the experiments in the previous section. Interestingly, the heatmap of SHUFFLED-BATCH is more closer to that of full-batch (hence ours as well) than the previous section, which is somewhat aligned with our practical experiments in Sec. E.3.

E.3 Experiments on multi-modal real datasets

We test on two image captioning datasets: MS-COCO, and CC3M. For the image and text encoders, we use the pre-trained ResNet50 (RN50) and Transformer described in (Radford et al., 2021), respectively. We load the pre-trained weights from CLIP, and then train them on the target dataset. The embedding dimension d of both encoders is 1024. We compare six batching schemes: (i)

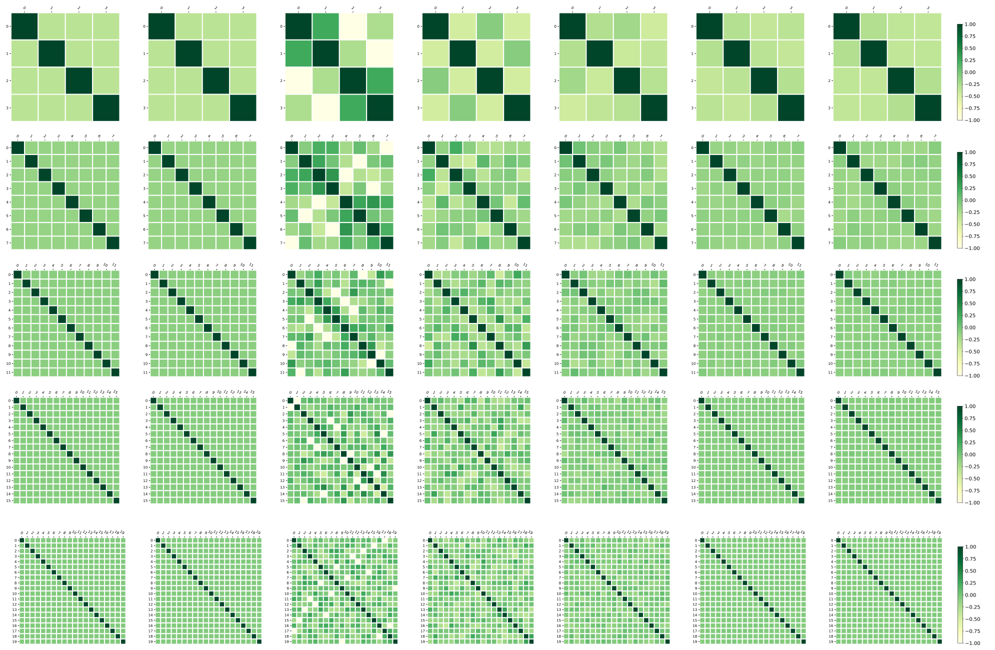

(a) Full-batch (b) All $\binom{N}{B}$ (c) FIXED (d) SHUFFLED (e) OSGD (f) BcS (g) ScB

Figure 4: Heatmap of $N \times N$ matrix representing pairwise inner products for the $d \geq N-1$ setting. We have optimization with (a) full-batch, (b) $\binom{N}{B}$ mini-batches, (c) FIXED-BATCH, (d) SHUFFLED-BATCH, (e) OSGD, (f) BcS, and ScB. The first row represents the case for $N = 4, d = 8$, the second row for $N = 8, d = 16$, the third row for $N = 12, d = 24$, the fourth row for $N = 16, d = 32$ and the last row for $N = 20, d = 40$.

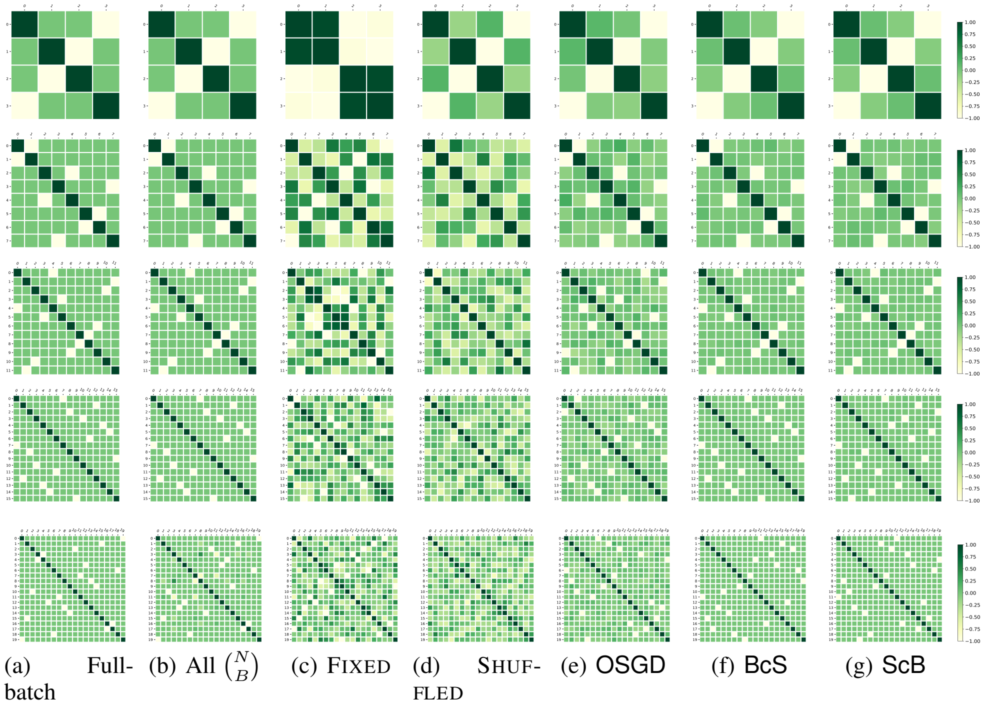


Figure 5: Heatmap of $N \times N$ matrix representing pairwise inner products for the $d < N - 1$ setting. We have optimization with (a) full-batch, (b) $\binom{N}{B}$ mini-batches, (c) FIXED-BATCH, (d) SHUFFLED-BATCH, (e) OSGD, (f) BcS, and ScB. The first row represents the case for $N = 4, d = 2$, the second row for $N = 8, d = 4$, the third row for $N = 12, d = 6$, the fourth row for $N = 16, d = 8$ and the last row for $N = 20, d = 10$.

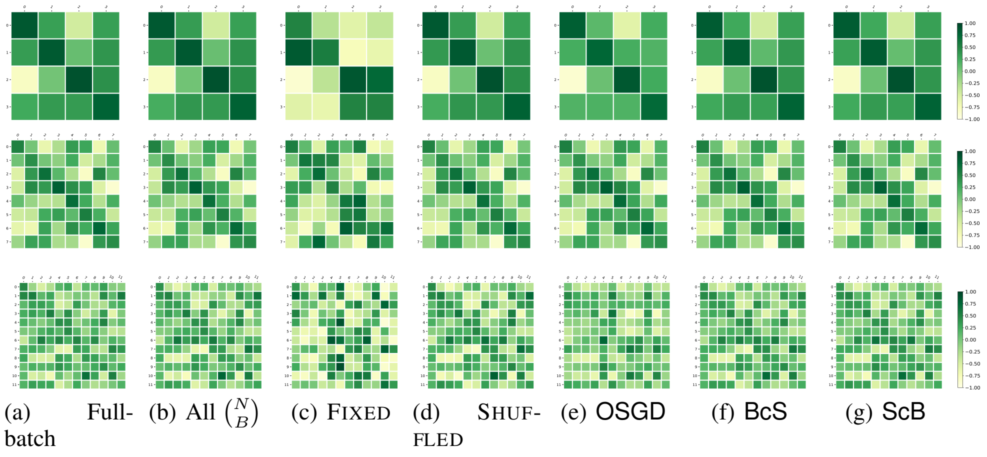


Figure 6: Heatmap of $N \times N$ matrix representing pairwise inner products for the $d \geq N - 1$ setting. We have optimization with (a) full-batch, (b) $\binom{N}{B}$ mini-batches, (c) FIXED-BATCH, (d) SHUFFLED-BATCH, (e) OSGD, (f) BcS, and ScB. The first row represents the case for $N = 4, d = 8$, the second row for $N = 8, d = 16$, and the last row for $N = 12, d = 24$.

FIXED-BATCH; (ii) SHUFFLED-BATCH; (iii) BcS; (iv) BcS + Rand; (v) ScB; and (vi) ScB + Rand. Due to the computational complexity issue, we could not run OSGD. We train models for 10 epochs and report top-1 recall (R@1) performances on the test dataset. We use the batch size $B \in \{100, 200, 400\}$, and AdamW optimizer with the standard learning rate scheduler defined

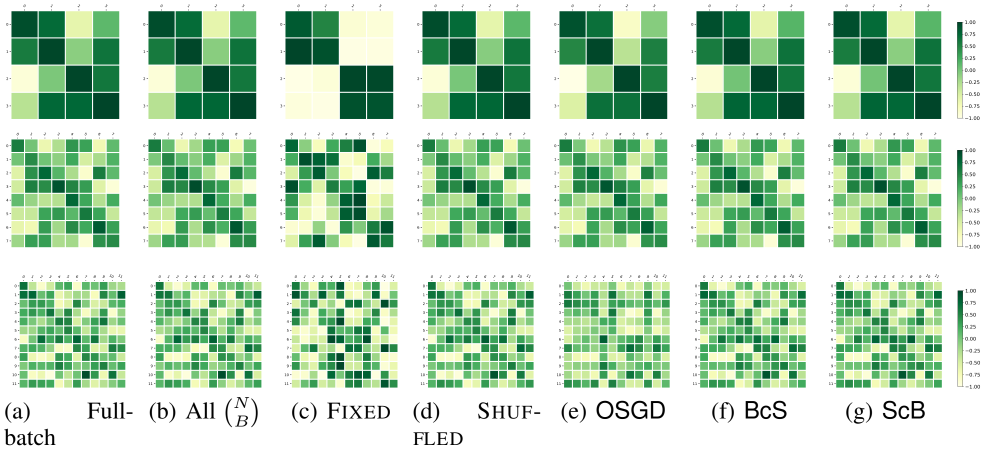

(a) Full-batch (b) All $\binom{N}{B}$ (c) FIXED (d) SHUFFLED (e) OSGD (f) BcS (g) ScB

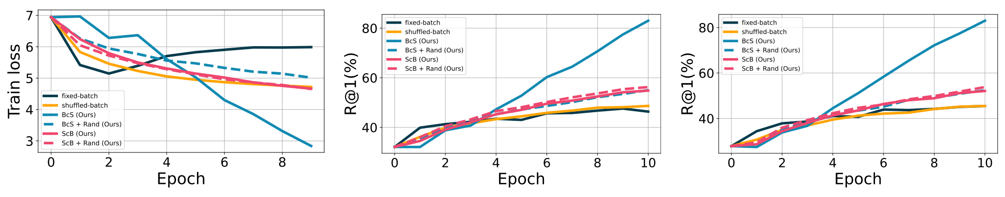
Figure 7: Heatmap of $N \times N$ matrix representing pairwise inner products for the $d < N-1$ setting. We have optimization with (a) full-batch, (b) $\binom{N}{B}$ mini-batches, (c) FIXED-BATCH, (d) SHUFFLED-BATCH, (e) OSGD, (f) BcS, and ScB. The first row represents the case for $N = 4, d = 2$, the second row for $N = 8, d = 4$, and the last row for $N = 12, d = 6$.

in (Vaswani et al., 2017), and we choose a scaling factor for the learning rate via grid search. Details of hyper-parameters is in Appendix D.

Results Table 2 shows the performances of various batch selection methods, for MS-COCO and CC3M. One can confirm that (i) SHUFFLED-BATCH has a better performance than FIXED-BATCH, and (ii) some of our batch selection methods perform better than SHUFFLED-BATCH for all batch sizes. The gap between our method and baselines is more visible in the small batch size regime. It is shown that compared with SHUFFLED-BATCH method, our proposed methods (ScB and ScB + Rand) attain higher R@1 score (approximately 1%) across all batch sizes. We also note that BcS + Rand performs similarly to SHUFFLED-BATCH. Note that despite the fact that our proposed batch selection methods do not guarantee generalization, they still outperform the baselines in terms of test accuracy. We note that occasionally, BcS's performance is worse than FIXED-BATCH. We expect that this can be resolved by more careful tuning of hyperparameters.

Fig. 8 shows results obtained from training with $B = 100$ on the MS-COCO dataset in order to observe whether the training curves of our proposed methods align with what we expect from the proposed algorithms defined in Sec. 4. As mentioned earlier, our greedy methods are designed to find mini-batches containing the most informative pairs, therefore we expect that it is able to learn more robust representations and achieve fast convergence than SHUFFLED-BATCH. Here, we report the full-batch loss and image-to-text, text-to-image retrieval (R@1) on the train set.

Fig 8(a) shows the full-batch train loss of overall methods, which is to calculate contrastive loss considering all the data points in the train set, not in the mini-batch. Therefore, it assures concrete trend of training rather than mini-batch train loss. The full-batch loss of BcS is especially lower than other methods, which means it fits on the train set well. However, as we stated in Sec. E.3, the generalization ability becomes lower than the others. Fig. 8(b) and 8(c) demonstrates how well each model can fit on the train set at each epoch. They also describe that BcS models fit faster than other methods on the train set.



(a) Full-batch loss (b) Image-to-text R@1 (c) Text-to-image R@1

Figure 8: (a) Full-batch train loss demonstrates that the exact training trends considering all the training data points. The training curve of BcS decreases more compared to other methods. The image-to-text R@1 on train set shows that how well each method cover the train set distribution. In (b), BcS shows higher R@1 score compared to other method at the same epoch. Other proposed method (BcS+ Rand, ScB and ScB + Rand) have a faster convergence speed on the train set compared with SHUFFLED-BATCH.

Table 2: Retrieval performances (R@1) on MS-COCO and CC3M datasets. The boldface indicates the best result.

	Image-to-text						Text-to-image					
Dataset	MS-COCO			CC3M			MS-COCO			CC3M		
Algorithm \ Batch size	100	200	400	100	200	400	100	200	400	100	200	400
FIXED-BATCH	39.10	40.54	40.72	37.53	38.07	39.04	35.96	36.30	37.72	34.64	35.76	36.29
SHUFFLED-BATCH	40.86	41.66	41.54	39.05	39.26	39.89	37.28	38.72	38.74	36.22	36.79	37.09
BcS (Ours)	34.50	36.38	37.58	33.92	34.16	36.43	31.10	31.84	34.44	31.44	31.18	33.06
BcS + Rand (Ours)	41.68	42.04	42.34	38.56	38.92	39.70	38.36	39.16	39.46	36.72	36.68	38.00
ScB (Ours)	40.46	**42.44**	42.24	39.89	**39.86**	**40.33**	38.26	39.06	**39.76**	36.75	**37.76**	**38.20**
ScB + Rand (Ours)	**42.18**	42.22	**42.74**	**40.19**	39.79	39.78	**38.90**	**39.54**	39.42	**37.75**	37.50	37.68