Unifying Synergies between Self-supervised Learning and Dynamic Computation

Tarun Krishna[1,3]
tarun.krishna2@mail.dcu.ie
Ayush K. Rai[1,2]
ayush.rai3@mail.dcu.ie
Alexandru Drimbarean[4]
Alexandru.Drimbarean@xperi.com
Eric Arazo[1,3,5]
eric.arazo@insight-centre.org
Paul Albert[1,3,5]
paul.albert@insight-centre.org
Alan F. Smeaton[1,2]
alan.smeaton@dcu.ie
Kevin McGuinness[1,3]
kevin.mcguinness@dcu.ie
Noel E. O'Connor[1,3]
noel.oconnor@dcu.ie

[1] Insight Centre for Data Analytics, Dublin City University, Dublin, Ireland
[2] School of Computing, Dublin City University, Dublin, Ireland
[3] School of Electronic Engineering, Dublin City University, Dublin, Ireland
[4] Xperi Corporation, Galway, Ireland
[5] Equal Contribution

Abstract

Computationally expensive training strategies make self-supervised learning (SSL) impractical for resource-constrained industrial settings. Techniques like *knowledge distillation* (KD), *dynamic computation* (DC), and *pruning* are often used to obtain a lightweight model, which usually involves multiple epochs of fine-tuning (or distilling steps) of a large pre-trained model, making it more computationally challenging. In this work we present a novel perspective on the interplay between the SSL and DC paradigms. In particular, we show that it is feasible to simultaneously learn a *dense* and *gated* sub-network from scratch in an SSL setting without any additional fine-tuning or pruning steps. The co-evolution during pre-training of both dense and gated encoder offers a good accuracy-efficiency trade-off and therefore yields a generic and multi-purpose architecture for application-specific industrial settings. Extensive experiments on several image classification benchmarks including CIFAR-10/100, STL-10 and ImageNet-100, demonstrate that the proposed training strategy provides a *dense* and corresponding *gated* sub-network that achieves on-par performance compared with the vanilla self-supervised setting, but at a significant reduction in computation in terms of FLOPs, under a range of target budgets (t_d).

1 Introduction

Motivation. Self-supervised representation learning methods [4, 7, 11, 12, 14] are the standard approach for training large scale deep neural networks (DNNs). One of the main reasons for their popularity is their capability to leverage the inherent structure of data from a vast unlabeled corpus during pre-training, which makes them highly suitable for transfer learning [28]. However, this comes at the cost of substantially larger model size, computationally expensive training strategies (larger training times, large batch-sizes, etc.) [13, 28] and subsequently more expensive inference times. Though such strategies are effective for achieving state-of-the-art results in computer vision, they may not be practical in resource-constrained industrial settings that require lightweight models to be deployed on edge devices.

To lessen the computational burden, it is common to extract (or learn) a *lightweight* network from an off-the-shelf pre-trained model. This has been successfully achieved through techniques such as knowledge distillation (KD) [35], pruning [24], dynamic computation (DC) [58], etc. KD methods follow a standard two-step procedure of pre-training and distilling knowledge into a *student* network using self-supervised (SS) objective [1, 21, 51] or by together incorporating supervised and SS objectives [54], while pruning based approaches heavily rely on multiple steps of pre-train $\rightarrow$ prune $\leftrightarrow$ finetune to get a lightweight network irrespective of the objective, whereas methods based on dynamic/conditional computation [34, 58] again rely on a pre-trained model to obtain a *lightweight* network while keeping the network topology intact via a *gating* mechanism. These approaches are effective but using fine-tuning to obtain a sub-network from large pre-trained models (such as Large Language Models) can be computationally expensive and cumbersome. Also, since downstream tasks are diverse and vary widely, any change in the task requires repeating the entire procedure multiple times, making it inefficient and less transferable.

Research Questions. These limitations motivate us to ask the following question: *"Can we unify the learning of a lightweight sub-network along with a dense network from scratch and in a completely self-supervised fashion?"* A straightforward way to achieve this is via an *online* KD (with self-supervised objective) [6, 54] learning paradigm which involves training teacher (f_θ) and student (g_ϕ) networks simultaneously during a pre-training stage. Recognising that this adds to the computational burden during pre-training (extra g_ϕ), we adopt a different route to attain the same goal but with a simpler, efficient pre-training objective and faster inference than online KD-based methods.

This enables us to reformulate the research question to: "*Can we learn a single encoder (function) that could serve the dual purpose of being used as a dense and lightweight network with minimal additional overhead?*"

Our objective is to simultaneously learn a *dense* and a *lightweight* model through a unified pre-training procedure to maintain high performance on the downstream task. We achieve this by exploiting the Siamese setting (a common setting for SSL [3]) combined with a gating mechanism for dynamic channel selection (DCS) [47, 58]. We opted for dynamic channel selection over KD/pruning for two main reasons: first, the gating mechanism preserves the networks topology adding enough flexibility to the approach; second, these gating modules are computationally inexpensive. For the self-supervised objective we choose VICReg [4] due to its symmetric nature and its ability to regularise each branch independently as dense and sparse branches will have different statistics. Figure 1 (*left*) demonstrates this dual setting of obtaining a dense and a lightweight network (derived from the dense one).

It should be noted that in this paper we do not follow the vocabulary of student-teacher

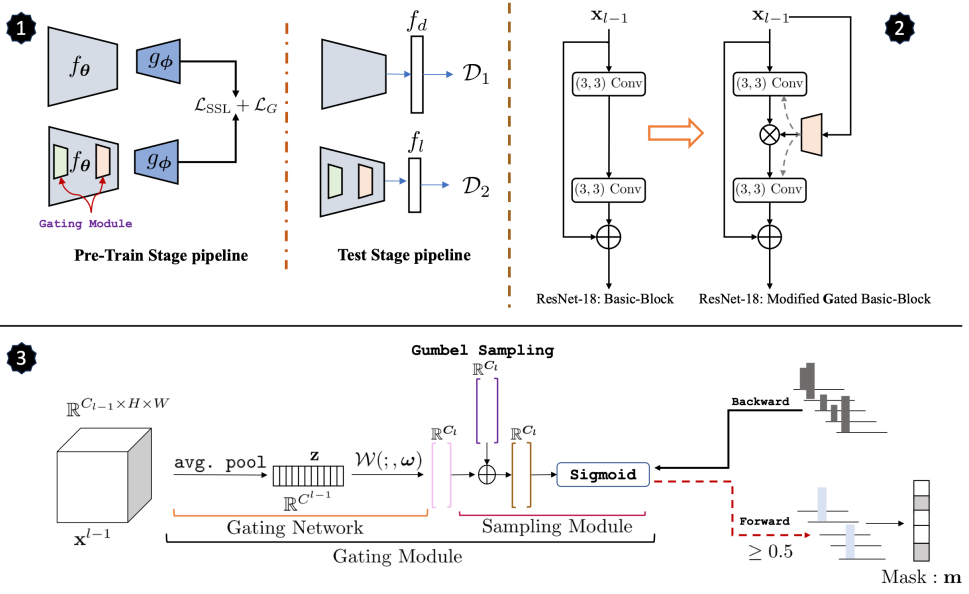

Figure 1: **1.** Illustrates the unification of SSL and DC during the pre-training and testing (inference) phase. f_d and f_l denotes respective linear layer for dense and gated network respectively. **Note:** dimensional size of f_d and f_l is the same, while in the figure this may look otherwise but is done to depict the fact that empirically, the dimension of f_l is less than f_d. **2.** Shows the modification of ResNet-18 basic block to accommodate the gating network during inference. **3.** Describes the gating module which comprises a *gating* network and *sampling* module.

networks. Instead we restrict the terminology to lightweight (gated)[1] and a dense network, where encoder and gates are randomly initialised and trained jointly from scratch, with the aim that they co-evolve during pre-training. It is, however, important to mention that in this work we are not proposing any new KD objective or KD-induced learning algorithm nor any new DC objective or any pruning-based learning. Instead we provide a novel perspective on exploiting the synergies that exist between self-supervised representation learning and dynamic computing. This approach is easily extendable to other symmetric-twins like Barlow-Twins [67], SimCLR [12] or W-MSE [67], while it may require some adjustments for non-symmetric methods like BYOL [29], MoCo [32], etc., which will be explored in future work. Our main contributions are:

- We present a novel perspective of unifying the learning of *dense* and *lightweight* networks by exploiting a symmetric joint embedding architecture of the SSL paradigm.
- We demonstrate that a single encoder can be exploited as a dense as well as a lightweight network; we show in Table 1 and Table 2 that a single base encoder can serve this dual purpose. This not only reduces computational overhead during training but also gives enough flexibility to use a single network and exploit it as per its requirement.
- We demonstrate exhaustively, through experiments that this unification preserves fea-

[1] we use lightweight and gated networks interchangeably.

ture quality across different experimental settings and gives on-par performance when compared with strict baselines (Section 4).

2 Background and Related Work

Self-supervised representation learning: SSL methods can be broadly divided into contrastive and non-contrastive techniques in the current scenario. At the core of these approaches is the concept of learning joint embedding representations realised via a Siamese [17] architecture through instance discrimination. Contrastive learning (CL) [12, 32, 33, 46, 52, 56, 61] based approaches intuitively try to bring similar instances closer while contrasting them with negative samples. Non-contrastive techniques include clustering methods [2, 7, 8], which alternate between cluster assignment and predicting clusters as pseudo labels. Approaches such as BYOL [29], OBoW [27], MoCo [32], and SimSiam [15] use a *teacher-student* approach to learn joint representations. While approaches like Barlow Twins [67], W-MSE [20], and VICReg [4] follow a more principled approach of information maximisation. We use VICReg for SSL because it can regulate each branch independently, which is more effective when each branch has different statistics.

Dynamic Computation: DC is a resource-efficient mechanism that reduces model complexity by skipping unimportant parts of the network while preserving the networks topology. Several authors including [23, 40, 48, 59, 60] have proposed adding decision branches to different layers of convolutional neural networks (CNN) for learning early exiting strategies leading to faster inference. BlockDrop [62] and SpotTune [30] learn a policy network to adaptively route the inference path through fine-tuned or pre-trained layers. ConvNet-AIG [58], [34] introduced a network that adaptively selects specific layers of importance to execute depending on the input image by specifying a target rate of each layer. GaterNet [16] introduced a network to generate input-dependent binary gates to select filters in the backbone network. DGNet [47] proposed a dual gating mechanism to induce sparsity along spatial and channel dimensions. Furthermore, dynamic channel pruning methods have also been devised such as feature boosting and suppression (FBS) [25] to dynamically amplify and suppress output channels computed by CNN layers. Other works learn sparsity through a three-stage pipeline: *pretrain-prune-finetune* as in [57] or use pre-trained models. See [36] for a more detailed explanation of sparsity, pruning and dynamic computing.

Self-supervised dynamic computation and beyond: Most of the works on dynamic computation have been confined to supervised learning. Recently, [44] used SimSiam [14] as a self-supervised objective combined with a dynamic channel gating (DGNet) [47] mechanism and showed that comparable performance can be achieved under channel budget constraints. Likewise [50] used a channel gating-based dynamic pruning (CGNet) [39] augmented with CL to achieve inference speed-ups without substantial loss of performance. In a similar line of work, [10] used iterative magnitude pruning (IMP) to obtain a winning ticket [24] for a pre-trained task (self-supervised objective) and evaluated its performance on various downstream tasks. [53] extended the work done in [9] in a MoCo (pre-trained) setting augmented with ADMM [68] for systematic pruning. A self-supervised *loss* objective can serve as a tool for KD [35] and model compression (MC). [55] used a contrastive objective (along with a supervised loss for *task specific distillation*) to train a student network from a pre-trained network. Similar to [55] but in a completely self-supervised setting [1, 22] minimises the *KL*-divergence between the distribution of similarities for the teacher (pre-trained) and student networks, while SimReg [51] minimises the regression loss. The authors in [63] used

a two-step strategy to train a teacher (with labels and then using an SSL head with a fixed backbone) followed by training a student using a KD loss. However, we follow a more simplistic approach through the unification of SSL (VICReg) and DCS, where DCS maintains the network topology, making fine-tuning easier on different downstream tasks, unlike other methods that make network structure irreversible.

3 Preliminaries and Setup

1. VICReg as SSL objective: VICReg [4] learns a joint embedding space governed by a loss objective, which consists of *invariance* (s) (mean squared error (MSE)), *variance* (v) and *co-variance* (c), depicted Equation 1. Let us consider some image dataset $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{\mathcal{D}}$ and a set of transformations $\mathcal{T}$ (refer to supplementary material for details). An anchor image $\mathbf{x}_i \in \mathbb{R}^{H\times W\times 3}$ is augmented through transformations $t_1, t_2 \sim \mathcal{T}$ to get $\mathbf{x}_i^1 = t_1(\mathbf{x}_i)$ and $\mathbf{x}_i^2 = t_2(\mathbf{x}_i)$ respectively. Augmented views are encoded through f_θ (ResNet-18 [31] (R18) in this study) to get feature representations. Furthermore, these representations are mapped to an *embedding* space via *expander* (g_ϕ) where the final VICReg loss is applied between the embedding vectors $\mathbf{z}_i^1 = g_\phi(f_\theta(\mathbf{x}_i^1))$ and $\mathbf{z}_i^2 = g_\phi(f_\theta(\mathbf{x}_i^2))$. Formally the loss is defined on a batch of embedding vectors $\mathbf{Z^1} = [\mathbf{z}_1^1, \cdots, \mathbf{z}_{|\mathcal{B}|}^1]$ and $\mathbf{Z^2} = [\mathbf{z}_1^2, \cdots, \mathbf{z}_{|\mathcal{B}|}^2]$ as:

$$\mathcal{L}_{\text{VICReg}}(\mathbf{Z}^1, \mathbf{Z}^2) = \overbrace{\underbrace{\mu[v(\mathbf{Z}^1) + v(\mathbf{Z}^2)]}_{\textbf{Variance}} + \underbrace{\nu[c(\mathbf{Z}^1) + c(\mathbf{Z}^2)]}_{\textbf{Co-Variance}}}^{\text{Regularisation Term}} + \underbrace{\eta s(\mathbf{Z}^1, \mathbf{Z}^2)}_{\textbf{Invariance}}, \tag{1}$$

where $\mu = 25$, $\nu = 25$ and $\eta = 1.0$. For detailed description of Equation 1 refer to [4].

2. Gating for channel selection. The gating module comprises of a *gating network* [5, 47] and a *sampling module* [41] (Figure 1). The *gating network* can be thought of as a lightweight network that decides the *relevance* of channels referred to as *importance* vector. To enable a lightweight design of the gating network ($\mathcal{W}$) we follow the squeeze and excitation block design [37], similar to [34, 58]. This usually requires obtaining a context vector $\mathbf{z} \in \mathbb{R}^{C_{l-1}}$ via global average pooling to accumulate spatial information. This context vector $\mathbf{z}$ is processed through $\mathcal{W}$ to get relevance scores for each channel:

$$\mathcal{W}(\mathbf{z}, \boldsymbol{\omega}) = \mathbf{w}_2 * (\texttt{BatchNorm}(\mathbf{w}_1 * \mathbf{z}))_{\text{ReLU}}, \quad \{\mathbf{w}_1, \mathbf{w}_2\} \in \boldsymbol{\omega} \tag{2}$$

where $*$ denotes convolution, $\mathbf{w}_1 \in \mathbb{R}^{\frac{C_{l-1}}{r}\times C_l\times 1\times 1}, \mathbf{w}_2 \in \mathbb{R}^{C_l\times \frac{C_{l-1}}{r}\times 1\times 1}$ and r is defined as reduction rate (set to 4).

Finally, to make a selection over a subset of a channels, we need to map the output of $\mathcal{W}$ to a binary vector (or mask $\mathbf{m} \in \mathbb{R}^{C_l}$). This discrete selection works perfectly during inference but breaks the computational graph during training. To make training possible, the *sampling module* utilises the Gumbel-softmax reparameterisation trick [41] to make this discrete selection without breaking the computational graph. Figure 1 (*right 2*) shows the modification of ResNet18 *basic block* (during inference).

In this work we follow the setting of DGNet [47] for channel selection where sparsity is induced by setting a global target budget (t_d) to optimise a loss objective :

$$\mathcal{L}_G = \lambda \left(\frac{\sum_{l=1}^{L} F_l^R}{\sum_{l=1}^{L} F_l^O} - t_d \right)^2 \tag{3}$$

where F_l^R is the average FLOPs over the batch along with FLOPs contribution from the gating network $\mathcal{W}$ (which is fixed for each layer), while F_l^O is the original FLOPs without a gating module, $\lambda = 5$ [47] across all datasets and training regimes. Only blocks with gating modules take part in FLOP computation as they contribute to the sparsification of the network. We refer our approach as VICReg-Dual-Gating (**VDG**).

3.1 Experimental Setup and Implementation Details

1. Pre-training: We closely follow the implementation of VICReg [4] (as our self-supervised objective) suited with our computational constraints using solo-learn library [19], while for dynamic gating we follow DGNet [47] for inducing channel sparsity via gating mechanism with ResNet18. The unified framework is depicted in Figure 1: the two branches have a *separate batch normalisation* layer following [66]. The encoder and gating networks are *randomly* initialised and trained with SGD for 500 epochs with a batch size of 512 on 2 Nvidia 2080Ti GPUs, with a warmup start of 10 epochs following a cosine decay with a base learning rate of 0.3 using the LARS optimizer [65]. Since we are using a very lightweight model as our gating network, there is no significant computational overhead during training, to be precise there is slight increase in computation which amounts to 2.11% of extra model parameters (0.013% of FLOPs computation). We pre-train for a target budget, $t_d = \{10\%, 30\%, 50\%\}$ for each of the datasets except for ImageNet-100 where t_d is restricted to $\{30\%, 50\%\}$ due to computational constraints. We report the inference speedup in terms of a hardware-independent theoretical metric of FLOPs and not wall-clock time as we do not avail any hardware accelerators to utilise sparsity during training. Any scaling parameter in the loss term is derived from the respective paper. The code is available at https://github.com/KrishnaTarun/Unification.

2. Evaluation: Pre-training and evaluation is carried on the train and validation data of CIFAR-10/100 [45], ImageNet100 [2] and STL-10 [18]. For pre-training with STL-10 we considered only the *un-labelled* set. We follow the standard practice of evaluating the trained encoder by freezing its weights and training a linear classifier on top of it. We trained a single linear layer for 100 epochs with a batch size of 512 on a single NVIDIA 2080Ti with a learning rate of 0.3 following step decay of 0.1 at 60^{th} and 80^{th} epochs. We report top-1 accuracy averaged over 5 runs.

3. Baselines: To exhaustively compare the performance of the dense and gated models we consider VICReg [4] as a SSL *dense* baseline while VICReg augmented with sparsity loss $\mathcal{L}_G$ (following Krishna *et al.* [44]) serves as a *gated* baseline, here goal is to train a lightweight gated encoder from scratch with a self-supervised objective.

4. Comparison with self-supervised KD: In order to make a fair assessment of this unification, we compare the gated network's performance with KD based methods specifically SEED [22] and SimReg [51] where the former was proposed to distill representational knowledge into a smaller network while the latter showed that a simple regression objective can serve as an effective tool for knowledge transfer. For SimReg: During pre-training (SSL) (500 epoch) we used VICReg [4] with a ResNet-18 (R18: Teacher) as base encoder trained on CIFAR-100 and ImageNet-100 ($\mathcal{D}_{\text{Pretrain}}$), while distillation is performed using the SimReg objective on $\mathcal{D}_{\text{Target}}$ ($= \mathcal{D}_{\text{Pretrain}}$) by further training it for another 130 epochs. For SEED: We use a MoCo-v2[3] pre-trained ResNet50 (R50) encoder trained on ImageNet-

[2] https://www.kaggle.com/datasets/ambityga/imagenet100
[3] https://github.com/facebookresearch/moco

Table 1: Linear Evaluation: ↑/↓ in orange font is comparison with *Baseline-1*, while blue font is comparison with *Baseline-2*. FLOPs R. denotes FLOP reduction. We report Top-1 accuracy averaged over 5 runs. Best viewed in color.

	VICReg			VICReg-Gating		VICReg-Dual-Gating		
	Baseline-1			Baseline-2		*this work*		
	Bardes *et al.* [🇫🇷]			Krishna *et al.* [🇨🇦]				
Dataset	Dense	FLOPs	t_d(%)	Gated	FLOPs R.	Dense ↑	Gated ↑	FLOPs R. ↑
CIFAR-10	91.11 ± 0.03	7.03E8	10%	87.75 ± 0.03	85.92%	88.99 ± 0.04 (↓ 2.12)	88.94 ± 0.06 (↓ 2.17) (↑ 1.19)	81.49% (↓ 4.43)
			30%	89.49 ± 0.04	69.27%	90.38 ± 0.04 (↓ 0.73)	90.27 ± 0.03 (↓ 0.84) (↑ 0.78)	66.43% (↓ 2.84)
			50%	90.70 ± 0.04	51.62%	90.20 ± 0.02 (↓ 0.91)	90.40 ± 0.06 (↓ 0.71) (↓ 0.30)	49.02% (↓ 2.60)
STL-10	86.15 ± 0.10	3.33E8	10%	82.48 ± 0.15	82.85%	84.29 ± 0.21 (↓ 1.86)	83.29 ± 0.05 (↓ 2.86) (↑ 0.81)	78.34% (↓ 4.51)
			30%	84.16 ± 0.11	68.38%	84.90 ± 0.05 (↓ 1.25)	84.85 ± 0.04 (↓ 1.30) (↑ 0.69)	65.24% (↓ 3.14)
			50%	85.40 ± 0.20	49.93%	85.75 ± 0.02 (↓ 0.40)	85.72 ± 0.02 (↓ 0.43) (↑ 0.32)	48.41% (↓ 1.52)
CIFAR-100	65.86 ± 0.10	7.03E8	10%	63.12 ± 0.09	84.82%	65.21 ± 0.06 (↓ 0.65)	64.31 ± 0.08 (↓ 1.55) (↑ 1.19)	81.71% (↓ 3.11)
			30%	65.41 ± 0.09	68.68%	65.90 ± 0.10 (↑ 0.04)	65.64 ± 0.00 (↓ 0.22) (↑ 0.23)	66.83% (↓ 1.85)
			50%	65.75 ± 0.12	50.04%	66.41 ± 0.05 (↑ 0.55)	66.40 ± 0.14 (↑ 0.54) (↑ 0.65)	49.06% (↓ 0.98)
ImageNet-100	77.74 ± 0.12	1.81E9	30%	74.04 ± 0.09	67.95%	75.12 ± 0.07 (↓ 2.62)	75.04 ± 0.10 (↓ 2.17) (↑ 1.00)	64.98% (↓ 2.97)
			50%	75.83 ± 0.07	50.11%	76.42 ± 0.26 (↓ 1.32)	76.24 ± 0.12 (↓ 1.51) (↑ 0.41)	47.69% (↓ 2.42)

1K ($\mathcal{D}_{\text{Pretrain}}$), while distillation using the SEED objective is performed for 200 epochs on $\mathcal{D}_{\text{Target}}$ (same as SimReg). Student networks are derived by sampling from R18's subspace with the number of filters for each *basic block* derived from our gating network i.e, channels are selected following policy learned by our gating module (see Figure (2,3) and Section 2 in supplementary) for fair comparison. The representation from the last average pooling layer is l_2 normalised and is evaluated using kNN as the evaluation criterion with $k = 1$ to report top-1 accuracy.

4 Results

1. Quantitative assessment. Table 1 compares the performance of VDG with the other two baselines for dense and gated. The lightweight gated network achieves improved performance across all datasets and target budgets (t_d) as compared to *Baseline-2*, with a negligible drop at $t_d = 50\%$ for CIFAR-10 only. The improved gated performance can be attributed to the fact that both dense and lightweight gated models co-learn during pre-training via weight sharing. However, the performance gain is compensated by a slightly smaller reduction in FLOPs as compared to *Baseline-2*. Improved performance of the gated network further closes the gap with a completely self-supervised dense model (*Baseline-1*). In comparison to *Baseline-1*, we observe minor drop in performance of VDG e.g., at $t_d = 10\%$ across CIFAR-10 (↓ 2.17%), STL-10 (↓ 2.86%), CIFAR-100 (↓ 1.55%), ImageNet-100 (↓ 2.7%) but with a significant reduction in the number of FLOPs. This illustrates that even under severe budget constraints our model achieves comparable performance to *Baseline1*. This drop further decreases on increasing t_d to 50%.

Another important aspect of our learning method is the performance of the *dense* (f_θ) model. Ideally our aim is to achieve fewer fluctuations with varying t_d with a performance equivalent to dense model as in *Baseline-1*. However, we find that the performance of the dense network (*this work*) is slightly below the performance of the dense (*Baseline-1*) for CIFAR-10/STL-10/ImageNet-100 while for CIFAR-100 the performance is better than the self-supervised dense module. This is interesting because what we achieve from this single pre-training of 500 epochs, is a single base encoder (dense encoder) and gates (via gating modules) and their combination gives a gated lightweight network.

Table 2: Transfer Performance: dense and gated under VICReg-Dual-Gating is compared with the common dense *baseline* of VICReg. ↑ / ↓ represents increment/decrement in performance. We report Top-1 linear evaluation accuracy averaged over 5 runs.

Dataset	**VICReg**	**VICReg-Dual-Gating**					
	Bardes *et al.* [9]	10%		30%		50%	
From → To	Dense	Dense	Gated	Dense	Gated	Dense	Gated
CIFAR-100 → STL-10	70.37	64.69 ↓	64.29 ↓	65.63 ↓	65.93 ↓	66.52 ↓	67.30 ↓
CIFAR-100 → CIFAR-10	80.08	80.56 ↑	79.92 ↓	80.23 ↑	80.24 ↑	80.16 ↑	80.09 ↑
CIFAR-100 → ImageNet-100	39.45	35.88 ↓	36.31 ↓	38.49 ↓	38.20 ↓	39.41 ↓	40.68 ↑
ImageNet-100 → STL-10	72.80	-	-	65.86 ↓	65.01 ↓	67.74 ↓	67.15 ↓
ImageNet-100 → CIFAR-10	55.12	-	-	53.38 ↓	49.41 ↓	54.01 ↓	49.80 ↓
ImageNet-100 → CIFAR-100	31.42	-	-	28.81 ↓	25.55 ↓	28.75 ↓	25.14 ↓

Table 3: Comparison of KD methods *students* performance with our *gated* network.

Method	$\mathcal{D}_{\text{Pretrain}}$	$\mathcal{D}_{\text{Target}}$	SSL-Pre Method (Teacher)$_{\text{epoch}}$	Student	1-NN
SimReg [51]	CIFAR-100	CIFAR-100	VICReg (R18)$_{500}$	R18 (10%)	37.32%
	CIFAR-100	CIFAR-100	VICReg (R18)$_{500}$	R18 (30%)	45.71%
	CIFAR-100	CIFAR-100	VICReg (R18)$_{500}$	R18 (50%)	48.99%
	ImageNet-100	ImageNet-100	VICReg (R18)$_{500}$	R18 (30%)	63.80%
	ImageNet-100	ImageNet-100	VICReg (R18)$_{500}$	R18 (50%)	65.78%
SEED [22]	ImageNet-1K	CIFAR-100	MoCO-v2 (R50)$_{800}$	R18 (10%)	31.54%
	ImageNet-1K	CIFAR-100	MoCO-v2 (R50)$_{800}$	R18 (30%)	35.29%
	ImageNet-1K	CIFAR-100	MoCO-v2 (R50)$_{800}$	R18 (50%)	38.22%
	ImageNet-1K	ImageNet-100	MoCO-v2 (R50)$_{800}$	R18 (30%)	64.38%
	ImageNet-1K	ImageNet-100	MoCO-v2 (R50)$_{800}$	R18 (50%)	67.50%
Ours (Gated)	CIFAR-100	-	VICReg (R18)$_{500}$	R18 (10%)	**56.57%**
	CIFAR-100	-	VICReg (R18)$_{500}$	R18 (30%)	**58.49%**
	CIFAR-100	-	VICReg (R18)$_{500}$	R18 (50%)	**59.83%**
	ImageNet-100	-	VICReg (R18)$_{500}$	R18 (30%)	**65.54%**
	ImageNet-100	-	VICReg (R18)$_{500}$	R18 (50%)	**67.72%**

2. Transfer Learning: Table 2 compares the transfer performance of VDG (dense and gated) with VICReg. This experiment gives further insights into the quality of the learned representation in this joint setting. In general, there is a drop in performance for VICReg-Dual for both *dense* and *gated*, although the difference is not significant. However, for CIFAR-100 → CIFAR-10 *dense* and *gated* outperforms only *dense* in VICReg at a very low target budget. Even in the case when the model is pre-trained on ImageNet-100, performance is comparable. This is encouraging as this new perspective still maintains good generalisation and transferability.

3. SSL-KD vs SSL-Gating: Table 3 compares the performance of SSL-KD methods with our SSL-Gating framework. To avoid confusion, we would like to reiterate we don't follow student-teacher paradigm, "Student" in Table 3 for "Ours (Gated)" is basically a R18 (base-encoder) with gates (see Figure 1) and $\mathcal{D}_{\text{Pretrain}} = \mathcal{D}_{\text{Target}}$. Results in Table 3 are very promising as we outperform both the KD methods by a substantial margin across all budgets. This result suggests that combining gating could serve as a general recipe to obtain a lightweight network along with a *dense* network during pre-training.

4. Qualitative assessment: Figure 2 shows uniform manifold approximation and projection (UMAP) [49] embeddings of the learned representations ($f_\theta(\mathbf{x}) \in \mathbb{R}^{512}$) trained using the dual-setting on the STL-10 dataset and compares it with VICReg [9]. The learned structure is similar to dense (VICReg) at a very low budget. Furthermore, the classes appear to be visually distinct, similar to the VICReg setting, and this is observed for both the dense and gated networks of VICReg-Dual.

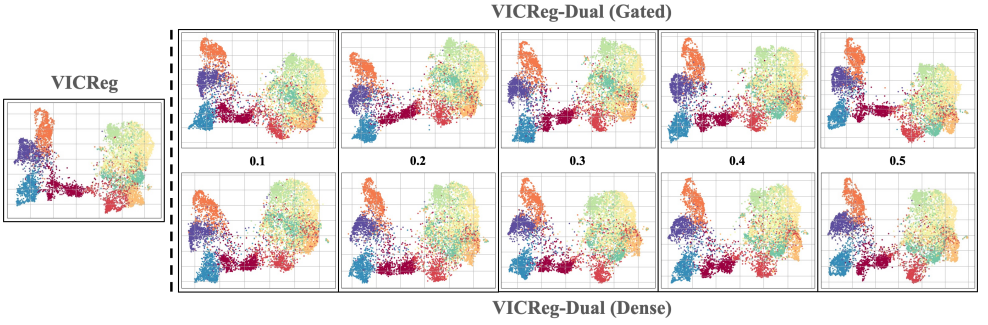

Figure 2: **Qualitative analysis:** UMAP embeddings of the learned representations: *lightweight* gated network (*top* row), while dense network (*bottom*) row over different target budgets t_d. This is compared with embeddings of VICReg (dense) trained without any sort of sparsity. Best viewed in color.

5 Additional Insights

Table 4: BT vs VICReg in dual setting.

Dataset	t_d	BT-Dual Dense	BT-Dual Gated	BT-Dual FLOPs R.	VICReg-Dual Dense	VICReg-Dual Gated	VICReg-Dual FLOPs R.
CIFAR-100	10%	53.64	53.01	80.60%	**54.92**	54.66	81.71%
	30%	55.03	54.89	66.09%	**56.34**	55.53	66.83%
	50%	55.63	55.39	47.64%	**56.83**	57.25	49.06%
STL-10	10%	73.45	73.49	76.74%	**76.61**	76.45	78.27%
	30%	75.51	76.02	64.12%	**78.55**	78.80	65.17%
	50%	76.99	77.10	48.32%	**79.69**	79.95	48.39%

For all experimental settings and studies discussed hereafter; models (and settings) were pre-trained for 500 epochs on CIFAR-100 and STL-10 datasets. Instead of using linear evaluation we use kNN as the evaluation criterion with $k = 1$ to report *top-1* accuracy. In all tables, **bold** and underline are the best performing results for the dense and gated module, respectively. Representations are not l_2 normalised.

1. Comparison with the symmetric Barlow Twins (BT) architecture. VICReg is build upon the findings of BT [67] and it is straightforward to apply the dual setting to BT because it minimises the cross-correlation (regularisation term) to identity $\mathcal{I}$ although the loss function is entirely mutual unlike in VICReg. In Table 4 we compare the performance of BT augmented with *our* setting. We observe that 1-NN performance of BT is low as compared to VICReg-Dual. The drop in performance could be attributed to the fact that VICReg applies independent regularisation which are later matched through the invariance loss. This further validates the hypothesis of choosing VICReg as our objective.

Table 5: Alternative base encoders.

Dataset	t_d(%)	VICReg-Dual (1×ResNet-18) Dense	Gated	FLOPs R.	VICReg-Dual (2×ResNet-18) Dense	Gated	FLOPs R.
CIFAR-100	10%	**54.92**	54.66	81.71%	53.85	52.44	85.77%
	30%	**56.34**	55.53	66.83%	55.65	55.52	70.07%
	50%	**56.83**	57.25	49.06%	55.64	55.71	52.60%
STL-10	10%	**76.61**	76.45	78.27%	74.93	74.89	82.48%
	30%	**78.55**	78.80	65.17%	76.83	76.71	69.26%
	50%	**79.69**	79.95	48.39%	77.74	78.00	51.10%

Table 6: Investigating the role of MSE.

Dataset	t_d(%)	VICReg-Dual w/ Invariance Dense	Gated	FLOPs R.	VICReg-Dual w/o Invariance Dense	Gated	FLOPs R.
CIFAR-100	10%	**54.92**	54.66	81.71%	4.06	6.31	93.77%
	30%	**56.34**	55.53	66.83%	4.84	4.89	82.72%
	50%	**56.83**	57.25	49.06%	2.79	4.15	49.05%
STL-10	10%	**76.61**	76.45	78.27%	11.79	18.02	90.71%
	30%	**78.55**	78.80	65.17%	12.25	17.89	74.96%
	50%	**79.69**	79.95	48.39%	12.81	15.72	51.91%

2. Training with a different base encoder. In this setting we train a model with two base encoders, one w/ gate (gated) and other w/o gate (dense) i.e, (2 × ResNet-18) (results in Table 5). An interesting observation is that VICReg-Dual with a single base encoder outperforms a more powerful setting with two base encoders (Table 5) although the FLOPs reduction (FLOPs R.) is higher (↑) in the setting of two different encoders. This is due to the fact that the sparsity loss ($\mathcal{L}_G$) operates solely on the un-shared branch so there is no trade-off involved, as in case of single

base encoder which simultaneously tries to enforce sparsity and visual invariance.

3. Role of *mean squared error* in co-evolving. It's a well known fact in SSL that these methods suffer from dimensional collapse [38, 42]. Training without any regularisation term or any trick [15, 29] would lead to dimensional collapse. Also, the authors in [43] showed that MSE serves as a better option for exact logit matching as compared to Kullback-Leibler (KL) divergence. So, in order to understand the role of MSE we trained a model w/o the *invariance* loss. As Table 6 shows, we found that there is large performance drop if we remove the invariance term. This implies that the invariance term plays a crucial role and seem to be an important factor, not only for co-evolving, but for self-supervision.

6 Discussion and Conclusion

In this work we presented a novel perspective on unifying synergies between SSL and DC. We exploit DC to induce sparsity into symmetric branches of self-supervised models enabling both branches to co-evolve with each other during training. In addition, this approach also allows simultaneous training of a dense and gated (sparse) sub-network from scratch with a target budget t_d under a self-supervised training objective with minimal computational overhead via weight sharing, thereby offering a good accuracy-efficiency trade-off for a given downstream application. As a result, our single base encoder offers enough flexibility to serve a dual purpose to reduce excessive computational overhead, which we validated through exhaustive experimentation (Tables (1, 2, 3)). However, there are limitations with this work. First, the dense model performance degrades and its performance further fluctuates with varying t_d. We experimented with RotNet [26] as a extra proxy loss for the dense branch but it did not yield good performance. Second, we have not imposed any constraint in the training objective that enforces a uniform distribution of channel activations, i.e. preservation of channel diversity during inference (which could also be a solution to the first limitation). Third, in future we will extend this setting to contrastive and non-symmetric architectures.

The work in this paper is an initial attempt to draw parallels and make an inter-connection between both of these fields. However, more research is needed to build a better intuition and insight into these models and also help us understand their other attributes, such as generalisation and ability to transfer to other downstream tasks.

7 Acknowledgments

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant number SFI/12/RC/2289_P2, co-funded by the European Regional Development Fund and Xperi FotoNation. This work is dedicated to the memory of Dr. Kevin McGuinness, whose contributions will always be remembered. This work stands as a testament to his enduring influence and our deep appreciation for his invaluable contributions.

References

[1] Soroush Abbasi Koohpayegani, Ajinkya Tejankar, and Hamed Pirsiavash. Compress: Self-supervised learning by compressing representations. *Advances in Neural Informa-*

tion Processing Systems, 33:12980–12992, 2020.

[2] Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. *arXiv preprint arXiv:1911.05371*, 2019.

[3] Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tian, et al. A cookbook of self-supervised learning. *arXiv preprint arXiv:2304.12210*, 2023.

[4] Adrien Bardes, Jean Ponce, and Yann Lecun. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. Technical report.

[5] Babak Ehteshami Bejnordi, Tijmen Blankevoort, and Max Welling. Batch-shaping for learning conditional channel gated networks. In *International Conference on Learning Representations*.

[6] Prashant Bhat, Elahe Arani, and Bahram Zonooz. Distill on the go: online knowledge distillation in self-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2678–2687, 2021.

[7] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. Technical report. URL https://github.com/facebookresearch/swav.

[8] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pages 132–149, 2018.

[9] Mathilde Caron, Ari Morcos, Piotr Bojanowski, Julien Mairal, and Armand Joulin. Pruning convolutional neural networks with self-supervision. *arXiv preprint arXiv:2001.03554*, 2020.

[10] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Michael Carbin, and Zhangyang Wang. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16306–16316, 2021.

[11] Ting Chen, Google Research, Calvin Luo, and Lala Li. Intriguing Properties of Contrastive Losses.

[12] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. 2020. URL http://arxiv.org/abs/2002.05709.

[13] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

[14] Xinlei Chen and Kaiming He. Exploring Simple Siamese Representation Learning. Technical report.

[15] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

[16] Zhourong Chen, Yang Li, Samy Bengio, and Si Si. You look twice: Gaternet for dynamic filter selection in cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9172–9180, 2019.

[17] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, 2005. doi: 10.1109/CVPR.2005.202.

[18] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[19] Victor Guilherme Turrisi da Costa, Enrico Fini, Moin Nabi, Nicu Sebe, and Elisa Ricci. solo-learn: A library of self-supervised methods for visual representation learning. *J. Mach. Learn. Res.*, 23:56–1, 2022.

[20] Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for Self-Supervised Representation Learning. Technical report.

[21] Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. SEED: SELF-SUPERVISED DISTILLATION FOR VISUAL REPRESENTATION. Technical report.

[22] Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. Seed: Self-supervised distillation for visual representation. *arXiv preprint arXiv:2101.04731*, 2021.

[23] Michael Figurnov, Maxwell D Collins, Yukun Zhu, Li Zhang, Jonathan Huang, Dmitry Vetrov, and Ruslan Salakhutdinov. Spatially adaptive computation time for residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1039–1048, 2017.

[24] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2018.

[25] X Gao, Yiren Zhao, L Dudziak, Robert Mullins, and X Cheng-Zhong. Dynamic channel pruning: Feature boosting and suppression. https://iclr. cc/Conferences/2019, 2019.

[26] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[27] Spyros Gidaris, Andrei Bursuc, Gilles Puy, Nikos Komodakis, Matthieu Cord, and Patrick Pérez. Online bag-of-visual-words generation for unsupervised representation learning. *arXiv preprint arXiv:2012.11552*, 2020.

[28] Priya Goyal, Mathilde Caron, Benjamin Lefaudeux, Min Xu, Pengchao Wang, Vivek Pai, Mannat Singh, Vitaliy Liptchinsky, Ishan Misra, Armand Joulin, et al. Self-supervised pretraining of visual features in the wild. *arXiv preprint arXiv:2103.01988*, 2021.

[29] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[30] Yunhui Guo, Honghui Shi, Abhishek Kumar, Kristen Grauman, Tajana Rosing, and Rogerio Feris. Spottune: transfer learning through adaptive fine-tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4805–4814, 2019.

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

[32] 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*, pages 9729–9738, 2020.

[33] Olivier J Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, S M Ali Eslami, and Aaron Van Den Oord. Data-Efficient Image Recognition with Contrastive Predictive Coding. Technical report, 2020.

[34] Charles Herrmann, Richard Strong Bowen, and Ramin Zabih. Channel selection using gumbel softmax. In *European Conference on Computer Vision*, pages 241–257. Springer, 2020.

[35] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[36] Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research*, 22(241):1–124, 2021.

[37] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[38] 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*, pages 9598–9608, 2021.

[39] Weizhe Hua, Yuan Zhou, Christopher De Sa, Zhiru Zhang, and G Edward Suh. Channel gating neural networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 1886–1896, 2019.

[40] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Weinberger. Multi-scale dense networks for resource efficient image classification. In *International Conference on Learning Representations*, 2018.

[41] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[42] Li Jing, Pascal Vincent, Yann Lecun, and Yuandong Tian. UNDERSTANDING DIMENSIONAL COLLAPSE IN CON-TRASTIVE SELF-SUPERVISED LEARNING.

[43] Taehyeon Kim, Jaehoon Oh, NakYil Kim, Sangwook Cho, and Se-Young Yun. Comparing kullback-leibler divergence and mean squared error loss in knowledge distillation. *arXiv preprint arXiv:2105.08919*, 2021.

[44] Tarun Krishna, Ayush K Rai, Yasser AD Djilali, Alan F Smeaton, Kevin McGuinness, and Noel E O'Connor. Dynamic channel selection in self-supervised learning. *arXiv e-prints*, pages arXiv–2207, 2022.

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

[46] Phuc H. Le-Khac, Graham Healy, and Alan F. Smeaton. Contrastive representation learning: A framework and review. *IEEE Access*, 8:193907–193934, 2020. doi: 10.1109/ACCESS.2020.3031549.

[47] Fanrong Li, Gang Li, Xiangyu He, and Jian Cheng. Dynamic dual gating neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5330–5339, October 2021.

[48] Mason McGill and Pietro Perona. Deciding how to decide: Dynamic routing in artificial neural networks. In *International Conference on Machine Learning*, pages 2363–2372. PMLR, 2017.

[49] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[50] Jian Meng, Li Yang, Jinwoo Shin, Deliang Fan, and Jae-Sun Seo. Contrastive dual gating: Learning sparse features with contrastive learning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12247–12255, 2022. doi: 10.1109/CVPR52688.2022.01194.

[51] KL Navaneet, Soroush Abbasi Koohpayegani, Ajinkya Tejankar, and Hamed Pirsiavash. Simreg: Regression as a simple yet effective tool for self-supervised knowledge distillation. 2021.

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

[53] Siyuan Pan, Yiming Qin, Tingyao Li, Xiaoshuang Li, and Liang Hou. Momentum contrastive pruning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2646–2655, 2022. doi: 10.1109/CVPRW56347.2022.00298.

[54] Yonglong Tian, Dilip Krishnan, Google Research, and Phillip Isola. CONTRASTIVE REPRESENTATION DISTILLATION. Technical report.

[55] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*, 2019.

[56] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? *Advances in Neural Information Processing Systems*, 33:6827–6839, 2020.

[57] Rishabh Tiwari, Udbhav Bamba, Arnav Chavan, and Deepak Gupta. Chipnet: Budget-aware pruning with heaviside continuous approximations. In *International Conference on Learning Representations*, 2020.

[58] Andreas Veit and Serge Belongie. Convolutional networks with adaptive inference graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–18, 2018.

[59] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 409–424, 2018.

[60] Wenhao Wu, Dongliang He, Xiao Tan, Shifeng Chen, Yi Yang, and Shilei Wen. Dynamic inference: A new approach toward efficient video action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 676–677, 2020.

[61] 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*, pages 3733–3742, 2018.

[62] Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S. Davis, Kristen Grauman, and Rogerio Feris. Blockdrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[63] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *European Conference on Computer Vision*, pages 588–604. Springer, 2020.

[64] Chuanguang Yang, Zhulin An, Helong Zhou, Fuzhen Zhuang, Yongjun Xu, and Qian Zhang. Online knowledge distillation via mutual contrastive learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[65] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

[66] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1gMCsAqY7.

[67] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow Twins: Self-Supervised Learning via Redundancy Reduction. 2021.

[68] Tianyun Zhang, Shaokai Ye, Kaiqi Zhang, Jian Tang, Wujie Wen, Makan Fardad, and Yanzhi Wang. A systematic dnn weight pruning framework using alternating direction method of multipliers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 184–199, 2018.