We have:

\[ f \text{ is zero-padded to } K \text{ size as } K \text{ size as } K \]

When padding size is \( K \), the input \( a \in \mathbb{R}^{B \times C' \times H \times W} \) is zero-padded to \( a' \in \mathbb{R}^{B \times C' \times (H+2K-2) \times (W+2K-2)} \). The output of this Conv layer is \( f_{\text{Conv}}(a) \in \mathbb{R}^{B' \times C' \times H' \times W'} \).

We have:

\[
\text{AvgPool}(f_{\text{Conv}}(a)) = \frac{1}{BH'W'} \sum_{b=1}^{B} \sum_{h=1}^{H'} \sum_{w=1}^{W'} f_{\text{Conv}}(a)_{b,:h,w}
\]

\[
= \frac{1}{BH'W'} \sum_{b=1}^{B} \sum_{h=1}^{H+K-1} \sum_{w=1}^{W+K-1} \sum_{c=1}^{C} \sum_{i=1}^{K} \sum_{j=1}^{K} W_{c,:i,j} a'_{b,c,h+i-1,w+j-1}
\]

\[
= \frac{1}{BH'W'} \sum_{c=1}^{C} \sum_{i=1}^{K} \sum_{j=1}^{K} \left( W_{c,:i,j} \sum_{b=1}^{B} \sum_{h=1}^{H+K-1} \sum_{w=1}^{W+K-1} a'_{b,c,h+i-1,w+j-1} \right)
\]

\[
= \frac{1}{BH'W'} \sum_{c=1}^{C} \sum_{i=1}^{K} \sum_{j=1}^{K} \left( W_{c,:i,j} \sum_{b=1}^{B} \sum_{h=1}^{H} \sum_{w=1}^{W} a_{b,c,h,w} \right)
\]

\[
= \frac{1}{BH'W'} \sum_{c=1}^{C} \sum_{i=1}^{K} \sum_{j=1}^{K} \left( W_{c,:i,j} \sum_{b=1}^{B} \sum_{h=1}^{H} \sum_{w=1}^{W} \text{AvgPool}_{DP}(a)_{b,c,h,w} \right)
\]

If \( stride \neq 1 \), we suppose \( stride = 2 \) as an instance. We can transform the convolution with \( stride = 2 \) into 4 convolutions with \( stride = 1 \), by breaking down both \( a \) and \( W \) as in Fig. 1. Then the proposition still holds for each of the 4 convolutions. Meanwhile, we need to store 4 pre-convolution means with respect to the 4 parts of \( a \). Similarly, if \( stride = m \), we just need to transform the convolution into \( m^2 \) convolutions with \( stride = 1 \), and store \( m^2 \) pre-convolution means.

As for the cases where \( padding \neq K - 1 \), we hypothesize the pixels at the edges of images are less informative background, which can be ignored securely. So the proposition still holds with negligible error.

A.2. Proof of Proposition 2

Proof. This proposition is an immediate deduction of Theorem 3.2 in [2] by simply regarding inputs of task.

---

1 We notice that the theorem IDs are different between the openreview camera-ready version and arxiv version. We are in line with the former.
\( \mathcal{T}_t \) as in-distribution data, and inputs of task \( \mathcal{T}_t \) as out-of-distribution data. A slight difference is that we assume the “out-of-distribution data” are in the deterministic worst case, so we do not have a second moment term \( \sqrt{\sigma_{\text{min}}(\Sigma)} \) which is derived from the stochasticity of the out-of-distribution data in Lemma A.7 of [2].

As for multi-head setting, we have

\[
\sqrt{\mathcal{L}'(B_t^*, v_t^*)} = \sqrt{\max_{\|x\| \leq 1} (v_t^T B_t^* x - v_t^T B_t^* x)^2} \\
= \|v_t^T B_t^* - v_t^T B_t^*\|_2 \\
\geq \|v_t^T B_t^* - v_t^T B_t^*\|_2^2 - \|v_t^T B_t^* - v_t^T B_t^*\|_2^2 \\
= \mathcal{L}(B_t^*, v_t^*) - \epsilon_{mh}
\]

in which \( \epsilon_{mh} = \|v_t^T B_t^* - v_t^T B_t^*\|_2 \) is the divergence between the two sub-networks corresponding to the two heads. We can regard \( \epsilon_{mh} \) as a relaxation provided by multi-head setting.

### A.3. Proof of Proposition 3

**Proof.** For all task \( \mathcal{T}_t \) with \( t' \leq t \), we firstly assume that \( B_{t'-1}^* = B_0^* \). Then there exists \( v_0 \) such that \( \mathcal{L}_t(B_{t'-1}^*, v_0) = \mathcal{L}_t(B_t^*, v_t^*) \). By the proof of Proposition A.21 in [2] we have \( v_t^p = v_0 \). Since overparametrized model can achieve zero loss, we have \( \mathcal{L}(B_{t'-1}^*, v_t^p) = \mathcal{L}(B_0^*, v_0) = 0 \). So the gradient is also zero, and thus the feature extractor does not change at all, which means \( B_t^* = B_0^* \) as well.

When \( t' = 1 \), the assumption \( B_{t'-1}^* = B_0^* \) is naturally satisfied. Then inductively, for all task \( \mathcal{T}_t \) with \( t' \leq t \), we have \( B_t^* = B_0^* \), which leads to \( \mathcal{L}_t(B_t^*, v_t^*) = \mathcal{L}_t(B_0^*, v_t^*) \) as we desire.

### B. Experiment Details

#### B.1. Dataset Split

For CIFAR100 and CUB200, we use the official train/test split. For Caltech101 and Flowers102, we randomly choose 80% images as training set, and the others as test set. During hyperparameter search, we further randomly choose 10% images in training set for validation. The details of dataset statistics are in Tab. 1.

### B.2. Implementation

All experiments are implemented using Pytorch 1.10 with CUDA 10.2 on NVIDIA 1080Ti GPU. The datasets and dataloaders are built via Avalanche [3]. We use ResNet18 implemented by timm [6], and its pre-trained parameters are downloaded from torchvision. The baselines are reproduced on the basis of the official codebases of [1,4,5], in which we only modify the models and dataset interfaces.

#### B.3. Data Pre-processing

For each image, we pre-process it as follows (Pytorch style):

```python
Compose(
    Resize(size=256),
    CenterCrop(size=(224, 224)),
    ToTensor(),
    Normalize(
        mean=[0.4850, 0.4560, 0.4060],
        std=[0.2290, 0.2240, 0.2250]
    )
)
```

There is no extra data augmentation applied.

#### B.4. Hyperparameter Search

For each baseline, we extensively search the best hyperparameter, which is shown in Tab. 2. We train the model for a fixed number of epochs for each method, which depends on the complexity of the dataset. We randomly choose 10% images in training set for validation, exclude them from training, and use them to determine the best hyperparameters. After the hyperparameter search, we retrain the model with the whole training set and report the results on test set. Note that our ConFiT does not need hyperparameter search.
### C. Difference between IRS and Internal Covariate Shift

We notice that there is another concept called Internal Covariate Shift (ICS) about BN. To avoid confusion, we here clarify the difference between IRS and ICS.

Firstly, we would emphasize that ICS and IRS are totally different concepts. ICS describes the instability of intermediate representations’ distribution during the training stage, i.e., the inputs of intermediate layers are unstable in training, which impedes the learning of networks. One of the motivations behind BN is to address ICS, so as to make the network easier to train.

Whereas IRS is defined in the scenario of continual learning, which means the intermediate representations of data of previous tasks were shifted, because the network fitted the data of the newest task. The shifted representation will disrupt the function of BN in testing, since the running moments will no longer be representative of the true moments of intermediate representations.

Overall, BN can solve ICS, but will suffer from IRS in continual learning, which is what we have attempted to address.

### References


[3] Vincenzo Lomonaco, Lorenzo Pellegrini, Andrea Cossu, Antonio Carta, Gabriele Graffieti, Tyler L. Hayes, Matthias De Lange, Marc Masana, Jary Pomponi, Gido M. van de Ven, Martin Mundt, Qi She, Keiland Cooper, Jeremy Forest, Eden Belouadah, Simone Calderara, German Ignacio Parisi, Fabio Cuzzolin, Andreas S. Tolias, Simone Scardapane, Luca Antiga, Subhat Ahmad, Adrian Popescu, Christopher Kanan, Joost van de Weijer, Tinne Tuytelaars, Davide Bacciu, and Davide Maltoni. Avalanche: An end-to-end library for continual learning. In CVPR Workshops, 2021. 2

