Communication-efficient Federated Learning with Single-Step Synthetic Features Compressor for Faster Convergence

Yuhao Zhou\textsuperscript{1,2}, Mingjia Shi\textsuperscript{1,2}, Yuanxi Li\textsuperscript{3}, Yanan Sun\textsuperscript{1,2}, Qing Ye\textsuperscript{1,2,*}, Jiancheng Lv\textsuperscript{1,2,*}

\textsuperscript{1} Sichuan University
\textsuperscript{2} Engineering Research Center of Machine Learning and Industry Intelligence
\textsuperscript{3} University of Illinois at Urbana-Champaign

\{sooptg, 3101ihs\}@gmail.com, yuanxi3@illinois.edu, \{yeqing, ysun, lvjiancheng\}@scu.edu.cn

Abstract

Reducing communication overhead in federated learning (FL) is challenging but crucial for large-scale distributed privacy-preserving machine learning. While methods utilizing sparsification or other techniques can largely reduce the communication overhead, the convergence rate is also greatly compromised. In this paper, we propose a novel method named Single-Step Synthetic Features Compressor (3SFC) to achieve communication-efficient FL by directly constructing a tiny synthetic dataset containing synthetic features based on raw gradients. Therefore, 3SFC can achieve an extremely low compression rate when the constructed synthetic dataset contains only one data sample. Additionally, the compressing phase of 3SFC utilizes a similarity-based objective function so that it can be optimized with just one step, considerably improving its performance and robustness. To minimize the compressing error, error feedback (EF) is also incorporated into 3SFC. Experiments on multiple datasets and models suggest that 3SFC has significantly better convergence rates compared to competing methods with lower compression rates (i.e., up to 0.02%). Furthermore, ablation studies and visualizations show that 3SFC can carry more information than competing methods for every communication round, further validating its effectiveness.

1. Introduction

Until now, federated learning [22] (FL) is deemed as one of the most promising distributed techniques [8, 3] to tackle the isolated data island problem with privacy guarantees. However, the training process of FL involves frequent exchanging of model parameters between central servers and participating clients, which is becoming increasingly expensive, especially considering the rapid growth of model size today [6, 16, 10]. Moreover, participating clients of FL typically operate at unreliable and limited network connection rates compared to data centers [15], further hindering the large-scale deployments of FL. Consequently, communication is becoming the primary bottleneck for flexible FL at the scale [5].

To explore possible approaches for reducing communication overhead in FL, various methods have been proposed targeting different objectives. The work in [30, 21] applied top-\(k\) sparsification to the gradients so that only the most important information is transmitted at each epoch. Moreover, Wangni [33] reported that using top-\(k\) sparsification with error feedback (EF), the communication overhead of ResNet-50 [12] trained on ImageNet [24] could be reduced by 99.6\% while maintaining nearly the same model accuracy. On the other hand, The work in [2, 4] employed quantification to represent gradients by a lower precision data type with a considerably smaller size. Later Karimireddy [17] introduced error feedback to quantification as well, substantially improving the rate of convergence. In [11], instead of gradients, several data samples distilled from the full training dataset were transmitted as they are much smaller than the gradients, and they can produce similar gradients through back-propagation. More recently, Li [19] and Wu [34] proposed compressing and decompressing communication data using compressed sensing and knowledge distillation, respectively.

While the methods mentioned above have been proven useful in reducing communication overhead reduction, empirical evidence indicates that they both suffer from degraded model convergence rates. This means that the model is expected to converge much slower with a smaller compression rate, as demonstrated in Figure 1. Here, the compression rate is defined as Equation 1. In this paper, we propose a single-step synthetic features compressor (3SFC), to boost the convergence rate during training and carry more information under a limited communication budget. Instead
of transmitting raw gradients directly, 3SFC first constructs a tiny synthetic dataset for the FL model. Then a scaling coefficient is calculated to minimize the compression error. Finally, the constructed synthetic dataset and the scale coefficient are transmitted to the server. In addition, the error feedback [29] is also incorporated into 3SFC to further minimize the overall compression error.

\[
\text{Comp. Rate} = \frac{\text{Comp. Size}}{\text{Uncomp. Size}} = \frac{1}{\text{Comp. Ratio}}. \tag{1}
\]

Our contributions can be summarized as following:

1. Instead of transmitting gradients employed by most existing compression methods, 3SFC only transmits a tiny set of model inputs and labels, which is independent of the model architecture. Consequently, 3SFC can achieve an extremely low compression rate.

2. A similarity-based objective function is employed to construct synthetic inputs and labels, which drastically lowers the time and space complexity and improves the performance and robustness of 3SFC. Moreover, the error feedback is incorporated into 3SFC to minimize the overall compressing error and therefore boost the convergence rate. These design choices make 3SFC an effective solution for achieving communication-efficient FL while maintaining model accuracy.

3. 3SFC can achieve a significantly better convergence rate compared to competing methods under the same and even lower communication budget (i.e., up to a compression ratio of 3600×). Ablation study and other visualizations further validate the efficiency of 3SFC against other state-of-the-art works. The code is open-sourced for reproduction 1.

---

1https://github.com/Soptq/iccv23-3sfc

2. Related Work

**Sparsification:** Methods in [1, 21, 33] utilized sparsi-fiers to filter and send partial gradients to greatly reduce the communication overhead. Typical sparsifiers include random-\(k\), top-\(k\), etc. DGC [21] and STC [26] are considered as current state-of-the-arts. Recently, Sahu [25] formally showed that top-\(k\) is the communication-optimal sparsifier under a limited communication budget.

**Quantification:** Methods in [4, 2, 27] replaced the default 32-bit data type in Machine Learning (ML) training with the 8-bit or even 1-bit data type before communicating, and thereby reducing communication overhead. Compared to sparsification methods that can achieve a compression rate of 1/100 or even lower, quantification can at most achieve a compression rate of 1/32.

**Data distillation (Synthetic dataset) for FL:** Recent work [11, 13] have proposed using several synthetic data samples to represent gradients in FL. Since local models in FL are optimized for multiple steps locally, The synthesis process will first generate a synthetic dataset, then simulate optimizing its local model for multiple steps using the synthetic dataset, and finally, utilize the minimization of the \(\ell_2\) distance between simulated and real model weights as its objective function to optimize the synthetic dataset. While the multiple steps of simulation is intuitive, empirical results suggest that it leads to not only a high level of time and space complexity for synthesizing (i.e., calculate gradients and store intermediate model weights multiple times), but also great instability and possible collapse, especially for relatively large models and datasets. Figure 2 demonstrates such a collapse. Moreover, gradients of trainable parameters before the collapse are visualized in Figure 3, indicating that a phenomenon similar to the gradient explosion.
Figure 3: Before the collapse of the method in [11] in Figure 2, the gradients of its trainable parameters exhibit a phenomenon similar to the gradient explosion, where the magnitude of gradients increases as they backpropagate from the 128-th to the first group of parameters. This could be a possible reason for the collapse.

had occurred due to the multiple steps of simulation. Consequently, since previous work hardly converges under our experimental settings involving extremely low compression rate and relatively large models and datasets as Section 5 demonstrated, they will not be compared in our formal experiments.

To alleviate the above-mentioned problems for data distillation to achieve both computation and communication efficient gradient compression, 3SFC employs a similarity-based objective function, instead of \( L_2 \)-based objective function, to optimize the synthetic dataset. Empowered by the new objective function, 3SFC optimizes the synthetic dataset only once compared to dozens and hundreds by previous work, substantially enhancing the performance and stability under low compression rates and large models and datasets. Moreover, error feedback is introduced to the data distillation realm for the first time, further helping the optimization converge.

Others: Li [19] proposed utilizing compressed sensing for communication data compressing and decompressing. Wu [34] used knowledge distillation to distill the learned knowledge from the local model to a smaller model before communication.

3. Problem Formulation

Assume there are \( N \) clients participating in a FL training process, where the \( i \)-th client has a local dataset \( D_i \) that obeys distribution \( P_i \), and a loss function \( F_i(D_i, w_i) \) where \( w_i \) is the weight of its model \( M_i \). Note that in vanilla FL, all clients and servers share the same model architecture, i.e., \( M_1 = M_2 = ... = M_N = M \). The objective of FL is to solve Equation 2:

\[
\min_{w \in \mathbb{R}^d} G(F_1(D_1, w), F_2(D_2, w), ..., F_N(D_N, w)),
\]

where \( G(\cdot) \) is the linear aggregation function satisfying the sum of aggregation weights equals 1. Typical aggregation functions include arithmetic mean and weighted average based on \( |D_i| \) [22] where \( | \cdot | \) is the size of the \( \cdot \). The global model \( w \) at the \( t \)-th communication round is updated by Equation 3:

\[
w^{t+1} = w^t - G(g_1^t, g_2^t, ..., g_N^t)
\]

where \( g_i^t = w^t - w_i^t \) denotes the model weight differences after locally training for \( K \) rounds, and can be seen as accumulated gradients. Generally, to reduce communication overhead, a compressor \( C \) is applied to each client’s \( g_i^t \), so that the global model \( w \) can be updated by Equation 4:

\[
w^{t+1} = w^t - G(C(g_1^t), C(g_2^t), ..., C(g_N^t)).
\]

As a result, the objective of communication compressing can be modeled as Equation 5:

\[
C^* = \arg \min ||C(g_i^t) - g_i^t||^2 \text{ s.t. } ||C(g_i^t)||_0 \leq B,
\]

where \( B \) is the communication budget, constraining the maximum size of communication data at each communication round, and \( || \cdot || \) measures the distances of \( \cdot \). Moreover, letting \( \epsilon_i^t = ||C(g_i^t) - g_i^t|| \) denotes the compression error at time \( t \), then the error feedback can be utilized to optimize this error term by adding it to the \( g_i^{t+1} \). Thus, with error feedback, the global model can be updated as Equation 6:

\[
\begin{align*}
w^{t+1} &= w^t - G(C(g_1^t + \epsilon_1^t), C(g_2^t + \epsilon_2^t), ..., C(g_N^t + \epsilon_N^t)),
\epsilon_i^{t+1} &= g_i^t + \epsilon_i^t - C(g_i^t + \epsilon_i^t).
\end{align*}
\]

4. Our Approach

The general architecture of 3SFC is illustrated in Figure 4. At each epoch, the \( i \)-th client first trains its local model using its local dataset. After training, accumulated gradients can be obtained by subtracting the global model weights from the latest local model weights. Then, the \( i \)-th client will utilize the encoder to compress the averaged gradients into a synthetic dataset \( D_{syn,i}^t \) that fits the communication budget. When the compressed data is received by the server, the server will first decode the compressed data into accumulated gradients, and then it will aggregate the gradients and update the global model. As seen from Figure 4, in 3SFC, the compressor \( C \) consists of an encoder and a decoder, where the encoder is located on the clients and the decoder is placed on the servers.

4.1. Encoder with error feedback

The encoder in 3SFC is responsible for compressing \( g_i^t \) into a synthetic dataset \( D_{syn,i}^t \) and a scaling coefficient \( s_i^t \).
The objective of the encoder can be described by Equation 7:

\[
\min_{D_{\text{syn},i}^t, s_i^t} ||s_i^t \nabla_{w^t} F_i(D_{\text{syn},i}^t, w^t) - g_i^t - \epsilon_i^t||^2 + \lambda D_{\text{syn},i}^t 2
\]

s.t. \( D_{\text{syn},i}^t ||0 + 1 \leq B, \)

where \( g_i^t \) denotes the differences between the global model \( w^t \) and its local model, i.e., \( g_i^t = w^t - w_{i^t}^t \) for clients. \( \lambda D_{\text{syn},i}^t 2 \) is an \( \ell_2 \) regularization term to constrain the solution of \( D_{\text{syn},i}^t \) for better stability. Note that here the global model \( w^t \) is passed into \( F_i(\cdot) \) instead of \( w_{i^t}^t \), because \( w^t \) is the initial weight of every client’s local optimization process at each epoch. Since \( g_i^t + \epsilon_i^t \) is fixed, \( s_i^t \) can be derived from \( \nabla_{w^t} F_i(D_{\text{syn},i}^t, w^t) \) as shown in Equation 8:

\[
s_i^t = \frac{||g_i^t + \epsilon_i^t||}{||\nabla_{w^t} F_i(D_{\text{syn},i}^t, w^t)||} \cos(\theta)
\]

\[
= \frac{(g_i^t + \epsilon_i^t) \cdot \nabla_{w^t} F_i(D_{\text{syn},i}^t, w^t)}{||\nabla_{w^t} F_i(D_{\text{syn},i}^t, w^t)||^2},
\]

where \( \theta \) is the angle between two \( g_i^t + \epsilon_i^t \) and \( \nabla_{w^t} F_i(D_{\text{syn},i}^t, w^t) \). Consequently, the objective described in Equation 7 is equivalent to the following optimization problem:

\[
\min_{D_{\text{syn},i}^t} \begin{cases} 
1 - |\nabla_{w^t} F_i(D_{\text{syn},i}^t, w^t) \cdot (g_i^t + \epsilon_i^t)| + \lambda D_{\text{syn},i}^t 2 \\
\text{s.t. } ||D_{\text{syn},i}^t||_0 + 1 \leq B.
\end{cases}
\]

Namely, the objective is to find a synthetic dataset \( D_{\text{syn},i}^t \) that produces gradients that are most similar to \( g_i^t \) in terms of the direction. After solving Equation 9, \( s_i^t \) can be thus calculated by Equation 8 and the compression error \( \epsilon_i^t \) can be updated by Equation 6. Finally, \( D_{\text{syn},i}^t \) and \( s_i^t \) will be uploaded to others to represent the local gradients of client \( i \).

### 4.2. Decoder

After receiving \( D_{\text{syn},i}^t \) and \( s_i^t \) from others, the decoder at server \( j \) will attempt to reconstruct the gradients for local model updating by the following equation:

\[
g_i^t + \epsilon_i^t = s_i^t \nabla_{w^t} F_j(D_{\text{syn},i}^t, w^t).
\]

Note that the success of the reconstruction depends on the assumption that the server \( j \) has access to the global model \( w^t \) and \( F_j(\cdot) = F_i(\cdot) \), which can be easily satisfied. Finally, for servers, following Equation 6, the global model of server \( j \) can be updated accordingly.

### 4.3. Algorithm and complexity analysis

The pseudocode of 3SFC is presented in Algorithm 1. In 3SFC, during the training process, clients will solve two op-
timization problems instead of one compared to the vanilla FL method FedAvg [22]: the empirical risk minimization problem on the local dataset (Line 4) and Equation 9 for compression (Line 8). The solvers to these two problems are not nested, meaning the time complexity of 3SFC equals $O(N E(K + S))$. In terms of the space complexity, 3SFC additionally stores the $w^t$, $D^t_{syn,i}$, $s^t_i$ and $\epsilon^t_i$, which are all fixed size parameters. Hence, 3SFC shares the same space complexity, $O(N)$, with FedAvg as well.

5. Experiments

Datasets: Following the conventions of the community [26, 36, 4], five datasets including MNIST [9], FMNIST [35], EMNIST [7], Cifar10 and Cifar100 [18] are used in the experiments. To simulate the Non-i.i.d. characteristic, all datasets are manually partitioned into multiple subsets based on the Dirichlet distribution, which is commonly used in the FL setting [31, 20]. Figure 5 illustrates our partitions. As can be seen, different clients own different datasets in terms of both quantity and category.

Models: To cover both simple and complicated learning problems, five models including Multi-Layer Perceptron (MLP), MnistNet, ConvNet, ResNet [12] and RegNet [23] are used in the experiments. Here, MnistNet has two convolutional layers and two linear layers, and ConvNet has four convolutional layers and one linear layer. Additionally, for ResNet and RegNet, all batch normalization layers [14] and dropout layers [28] are deleted from the model as their parameters are not trainable [32]. This simplification has also been used in previous studies [26, 36].

Table 1: Test accuracies of FedSynth in our preliminary experiments with 10 clients after 200 epochs of training. As can be seen, the model is barely optimized (as discussed in Section 2) with FedSynth with an extremely high compression ratio, while other methods like 3SFC and DGC achieve much higher performances as Table 2 illustrated. Consequently, FedSynth is not compared with 3SFC in the latter experiments. These results validate our observations described in Section 2.
We first compare the final accuracy of 3SFC with other methods: FedAvg [22], DGC [21], signSGD with EF [4] and STC [26]. Specifically, FedAvg is a traditional FL training method without any compression, DGC is a sparsification-based method that can have an extremely low compression rate. For quantification-based methods like signSGD and STC, we leave their compression rate to be 1/32, and will later do dedicated evaluations between them and 3SFC.

The comprehensive accuracy comparison results of 3SFC and other methods are shown in Table 2. It can be observed that under the same compression rate, 3SFC yields higher test accuracy consistently compared to DGC after training, suggesting that 3SFC brings a faster convergence rate to the model training when the communication budget is limited. On the other hand, 3SFC still achieves comparable or even better test accuracy compared to DGC after training, suggesting that 3SFC brings a faster convergence rate to the model training when the communication budget is limited. On the other hand, 3SFC still achieves comparable or even better test accuracy compared to DGC after training, suggesting that 3SFC brings a faster convergence rate to the model training when the communication budget is limited. On the other hand, 3SFC still achieves comparable or even better test accuracy compared to DGC after training, suggesting that 3SFC brings a faster convergence rate to the model training when the communication budget is limited. On the other hand, 3SFC still achieves comparable or even better test accuracy compared to DGC after training, suggesting that 3SFC brings a faster convergence rate to the model training when the communication budget is limited.

6.2. Further comparisons between 3SFC and STC

To further evaluate 3SFC compared to STC for fairness, we gradually increase the communication budget of 3SFC and compare both their compression ratio and test accuracy, which is shown in Table 3. As the table suggests, 3SFC can achieve comparable or even better test accuracy while saving a significant amount of communication traffic. For example, when training ResNet on Cifar10 with 10 clients, 3SFC reports a comparable final accuracy (0.3954 compared to 0.4009) while communicating 189.4× fewer data.

### 6.3. Compression efficiency

To study why 3SFC achieves a faster convergence rate, we restrain the compression rate of 3SFC and DGC to be extremely low compression rate. For quantification-based methods like signSGD and STC, we leave their compression rate to be 1/32, and will later do dedicated evaluations between them and 3SFC.

The comprehensive accuracy comparison results of 3SFC and other methods are shown in Table 2.
Figure 6: Test accuracy and training loss comparisons after 200 epochs of training. Compared to other methods, 3SFC owns the fastest convergence rate with respect to the amount of traffic communicated, with the highest compression ratio. (a) MLP trained on MNIST. (b) MLP trained on MNIST. (c) MLP trained on FMNIST. (d) MLP trained on FMNIST. (e) RegNet trained on Cifar100. (f) RegNet trained on Cifar100.

Table 3: Test accuracy and compression ratio comparisons of STC and 3SFC with different communication budgets. 3SFC mostly achieves higher test accuracy while having a higher compression ratio, suggesting 3SFC compresses and decompresses the communication data more efficiently.

Table 4 shows the ablation study of 3SFC in terms of EF, communication budget $B$ and local iteration $K$. As observed from Table 4, compared to 3SFC w/ EF, disabling EF in 3SFC drastically degrades the model performance after training in all experiments, validating the effectiveness of EF. Moreover, MLP trained on MNIST by 3SFC w/o EF ob-
In this paper, we propose a single-step synthetic features compressor (3SFC) for communication-efficient FL. 3SFC compresses the data using a similarity-based objective function in a single step, thus saving both compute and storage resources, and maintaining the robustness of the algorithm. Moreover, error feedback is employed to further minimize the compression error. Comparisons of test accuracy and compression ratio show that 3SFC achieves significantly faster convergence rates with lower compression rates. An ablation study demonstrates the role of different parameters, and visualizations of compression efficiency further validates the effectiveness of 3SFC.

7. Conclusion

In this paper, we propose a single-step synthetic features compressor (3SFC) for communication-efficient FL. 3SFC compresses the data using a similarity-based objective function in a single step, thus saving both compute and storage resources, and maintaining the robustness of the algorithm. Moreover, error feedback is employed to further minimize the compression error. Comparisons of test accuracy and compression ratio show that 3SFC achieves significantly faster convergence rates with lower compression rates. An ablation study demonstrates the role of different parameters, and visualizations of compression efficiency further validates the effectiveness of 3SFC.

Table 4: The ablation study with different parameters of 3SFC (i.e., with/without EF, communication budgets \( B \), local iteration \( K \)). The configuration for the Base is \( 1 \times B \) and \( K = 5 \). From the table, it is clear that enabling EF in 3SFC has an important role in helping models converge, validating its effectiveness. Moreover, Increasing \( B \) or \( K \) can both further boost the convergence rate of the training.

<table>
<thead>
<tr>
<th>Methods</th>
<th>MNIST</th>
<th>EMNIST</th>
<th>FMNIST</th>
<th>Cifar10</th>
<th>Cifar100</th>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MLP</td>
<td>MLP</td>
<td>MLP</td>
<td>ResNet</td>
<td>ResNet</td>
<td>Avg</td>
</tr>
<tr>
<td>10 Clients</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3SFC w/ EF</td>
<td>0.8876</td>
<td>0.5494</td>
<td>0.7881</td>
<td>0.8179</td>
<td>0.2667</td>
<td>0.3753</td>
</tr>
<tr>
<td>3SFC w/o EF</td>
<td>0.4580</td>
<td>0.2397</td>
<td>0.5746</td>
<td>0.7324</td>
<td>0.4495</td>
<td>0.2313</td>
</tr>
<tr>
<td>3SFC w/ EF (( 2 \times B ))</td>
<td>0.8961</td>
<td>0.5820</td>
<td>0.8031</td>
<td>0.8356</td>
<td>0.6308</td>
<td>0.3642</td>
</tr>
<tr>
<td>3SFC w/ EF (( 4 \times B ))</td>
<td>0.8958</td>
<td>0.5955</td>
<td>0.8063</td>
<td>0.8420</td>
<td>0.6241</td>
<td>0.3954</td>
</tr>
<tr>
<td>3SFC w/ EF (( K = 1 ))</td>
<td>0.6939</td>
<td>0.3152</td>
<td>0.6500</td>
<td>0.7807</td>
<td>0.5207</td>
<td>0.2001</td>
</tr>
<tr>
<td>3SFC w/ EF (( K = 10 ))</td>
<td>0.8961</td>
<td>0.6075</td>
<td>0.8212</td>
<td>0.8383</td>
<td>0.6333</td>
<td>0.3909</td>
</tr>
<tr>
<td>20 Clients</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3SFC w/ EF</td>
<td>0.8918</td>
<td>0.5556</td>
<td>0.8013</td>
<td>0.8217</td>
<td>0.6044</td>
<td>0.3049</td>
</tr>
<tr>
<td>3SFC w/o EF</td>
<td>0.7607</td>
<td>0.5170</td>
<td>0.6008</td>
<td>0.7450</td>
<td>0.4625</td>
<td>0.2373</td>
</tr>
<tr>
<td>3SFC w/ EF (( 2 \times B ))</td>
<td>0.8948</td>
<td>0.5832</td>
<td>0.8053</td>
<td>0.8343</td>
<td>0.6165</td>
<td>0.4350</td>
</tr>
<tr>
<td>3SFC w/ EF (( 4 \times B ))</td>
<td>0.8963</td>
<td>0.5961</td>
<td>0.8070</td>
<td>0.8372</td>
<td>0.6187</td>
<td>0.4364</td>
</tr>
<tr>
<td>3SFC w/ EF (( K = 1 ))</td>
<td>0.7504</td>
<td>0.5387</td>
<td>0.6441</td>
<td>0.7706</td>
<td>0.5249</td>
<td>0.2155</td>
</tr>
<tr>
<td>3SFC w/ EF (( K = 10 ))</td>
<td>0.9061</td>
<td>0.6127</td>
<td>0.8289</td>
<td>0.8294</td>
<td>0.6049</td>
<td>0.3095</td>
</tr>
<tr>
<td>40 Clients</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3SFC w/ EF</td>
<td>0.8886</td>
<td>0.5595</td>
<td>0.7945</td>
<td>0.8270</td>
<td>0.6145</td>
<td>0.2869</td>
</tr>
<tr>
<td>3SFC w/o EF</td>
<td>0.4830</td>
<td>0.2512</td>
<td>0.6349</td>
<td>0.7575</td>
<td>0.4742</td>
<td>0.2380</td>
</tr>
<tr>
<td>3SFC w/ EF (( 2 \times B ))</td>
<td>0.8932</td>
<td>0.5676</td>
<td>0.8027</td>
<td>0.8374</td>
<td>0.6132</td>
<td>0.3747</td>
</tr>
<tr>
<td>3SFC w/ EF (( 4 \times B ))</td>
<td>0.8949</td>
<td>0.5995</td>
<td>0.8073</td>
<td>0.8412</td>
<td>0.6115</td>
<td>0.3695</td>
</tr>
<tr>
<td>3SFC w/ EF (( K = 1 ))</td>
<td>0.6956</td>
<td>0.3129</td>
<td>0.6547</td>
<td>0.7752</td>
<td>0.5243</td>
<td>0.2254</td>
</tr>
<tr>
<td>3SFC w/ EF (( K = 10 ))</td>
<td>0.9072</td>
<td>0.6074</td>
<td>0.8277</td>
<td>0.8399</td>
<td>0.5875</td>
<td>0.3099</td>
</tr>
</tbody>
</table>

Figure 7: Compression efficiency comparisons. 3SFC owns significantly higher compression efficiency compared to DGC under the same compression rate, suggesting 3SFC is a much more efficient compressor compared to DGC.

(a) RegNet trained on Cifar10.  (b) RegNet trained on Cifar100.

(c) ResNet trained on Cifar10.  (d) ResNet trained on Cifar100.
8. Acknowledgments

This work is supported by the Key Program of National Science Foundation of China (Grant No. 61836006).

References


