Divide2Conquer (D2C): A Decentralized Approach Towards Overfitting Remediation in Deep Learning

Md. Saiful Bari Siddiqui
Department of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
saiful.bari@bracu.ac.bd

Md Mohaiminul Islam
Department of Computer Science and Engineering
United International University
Dhaka, Bangladesh
mohaiminul@cse.uiu.ac.bd

Md. Golam Rabiul Alam
Department of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
rabiul.alam@bracu.ac.bd

***Abstract*—Overfitting remains a persistent challenge in deep learning. It is primarily attributed to data outliers, noise, and limited training set sizes. This paper presents Divide2Conquer (D2C), a novel technique designed to address this issue. D2C proposes partitioning the training data into multiple subsets and training separate identical models on them. To avoid overfitting on any specific subset, the trained parameters from these models are aggregated and averaged periodically throughout the training phase, enabling the model to learn from the entire dataset while mitigating the impact of individual outliers or noise. Empirical evaluations on multiple benchmark datasets across various deep learning tasks demonstrate that D2C effectively improves generalization performance, particularly for larger datasets. This study verifies D2C's ability to achieve significant performance gains both as a standalone technique and when used in conjunction with other overfitting reduction methods through a series of experiments, including analysis of decision boundaries, loss curves, and other performance metrics. It also provides valuable insights into the implementation and hyperparameter tuning of D2C. Our codes are publicly available at: https://github.com/Saiful185/Divide2Conquer.**

***Index Terms*—Deep Learning, Hyperparameter, Image Classification, Overfitting, Text Classification.**

I. Introduction

The performance of deep learning models can often be characterized by the volume of available data and diversity in the said data, leaving the caveat of severely underperforming on unseen data from different distributions. This phenomenon is known as Overfitting. The situation worsens when imbalances and non-IIDness exist in the data landscape. Many studies [7] were conducted over the years to ameliorate the problem. Early Stopping [13] is an intuitive method that works well but can also limit the model's full learning potential [14]. Network reduction reduces variance by simplifying the model, yet it restricts learning of complex features [15]. Data Augmentation [16] is another popular method to combat overfitting, though selecting suitable techniques for specific datasets can be challenging, and acquiring additional training data often requires significant effort. Regularization methods, which add penalties to discourage over-reliance on training data, are highly effective. Dropout [4], a go-to regularization technique in deep learning, reduces overfitting by randomly dropping a fraction of neural connections during training. Despite employing these techniques, the model still has access to the entire training data, and the robust neural network structures find their ways to fit on the training set a bit too much. This is where our motivation comes from. Perhaps the neural network should not be allowed to train on the entire dataset. Maybe we should combine multiple models that train on different portions of the data.

We propose a novel method, Divide2Conquer (D2C) which proposes dividing the training data into multiple subsets and training a model(all having the same architecture & hyperparameters) on each subset. A weighted averaging of all the parameter weights from the instance model is performed after a certain number of epochs, and the averaged weights are shared back to each of the models (we denote them as **edge models** in this study). Then the whole process is repeated for several global epochs. D2C is loosely inspired by Federated Optimization [2], which involves training models on local devices and periodically aggregating their trained parameters on a central server. However, federated learning is primarily designed to preserve data privacy, not to enhance model generalization. Our experiments encompass multiple datasets across different domains, and each of the cases suggests that employing the D2C method can reduce overfitting significantly. The primary comparison to evaluate the method was between the performances of the base Neural Network architecture used for the **Edge Models** using the entire training data, and the performance of our models using the Divide2Conquer method. We summarize our contributions through this study below:

- We introduce a new method, D2C, that helps in reducing overfitting significantly while being conceptually simple and easy to implement.
- D2C can be applied on top of many other data augmentation and regularization techniques, and our experiments show that this results in a clear improvement to the model's generalization ability.
- We also extensively tune the hyperparameters introduced by this method and report the findings, providing important directions for future applications.

This work was supported by the Institute for Advanced Research (IAR), United International University under Grant UIU-IAR-02-2022-SE-06.

II. Related Works

Several authors have addressed the issue of overfitting while performing various tasks. M. Cogswell et al. proposed a new regularizer called DeCov, which helps reduce overfitting in deep neural networks by Decorrelating Representations [5]. Dropout [4] was proposed by Srivastava et al. back in 2014, and since then, this technique has been extensively used in very complex neural network architectures successfully. It was indeed an outstanding contribution, specifically for deep learning-based models. Batch Normalization [6] proposed by Ioffe and Sergei primarily focuses on better convergence and somewhat contributes to reducing overfitting. J. Kolluri et al. came up with L¼ regularization to solve the problems faced by L1 and L2 regularization techniques [9].

Ensembles are also often used to improve generalization. The ensemble-based ELVD model [3] managed to outperform the traditional VGGNet and DropoutNet models in terms of reducing overfitting. Min-Gu Kim et al. also proposed parallel ensemble networks to reduce overfitting in ECG data and prevent the degradation of generalization performance as the training progresses [1].

We implemented a method based on federated optimization preliminarily for facial expression recognition using FedNet [10]. This model achieves excellent results in terms of generalization on both CK+ and FER-2013 datasets. Overfitting remains a prevalent issue in supervised machine learning despite methods like Early Stopping, Network Reduction, Training Set Expansion, Regularization, and Dropout being effective [7]. Even dropout can't always counter overfitting and sometimes degrades the performance of a model [11]. The D2C method tries to build an approach that focuses on achieving even better generalization, and which can be implemented on top of other overfitting-reducing techniques.

III. Theoretical Framework

Overfitting happens when a model fits too well on the data that it trains on. It effectively captures even the random characteristics from the training data, randomness that would be insignificant in real-world applications. The presence of outliers and noisy data is a fundamental reason behind overfitting. In this study, we investigate a method to minimize the effect of these outliers and noisy data.

The contribution of outliers/noise can be minimized by dividing the training data into multiple shards and training each shard separately. That way, each data point will only occur once in one of the several data shards. The representative samples would be close to each other in the feature space and would be present in each of the data shards more or less uniformly. However, the individual outliers/noise would only be able to impact one of the training processes. Aggregation and averaging can be done periodically to combine all the models. In the best-case scenario, due to averaging, the effect of the individual outliers/noise would be reduced by a factor of N, where N is the number of data shards. The key factors behind our hypothesis and considerations regarding this method being able to address overfitting are discussed as follows:

1. Weighted averaging of parameters combines knowledge from different edge models while mitigating extreme updates caused by noise or outliers. If a model adjusts excessively to fit a noisy sample, this influence is minimized when its parameters are averaged with those of other models. By training on data subsets, any overfitting due to outliers is diluted across the models, resulting in averaged parameters that better represent the underlying data distribution and reduce noise impact.

2. Averaging the trainable parameters combats overfitting by regularizing model parameters. This approach moderates extreme updates from noise or outliers in individual models, resulting in more stable, generalizable parameters. By averaging parameter weights, the central model should converge to a shared solution that reflects the collective knowledge learned from different perspectives. This consensus learning approach encourages models to learn generalizable patterns and reduces reliance on individual model predictions that may be prone to overfitting.

3. Averaging parameter weights should smooth out the decision boundary learned by individual models. Extreme or noisy parameter updates that result in sharp or jagged decision boundaries in individual models are likely to be moderated and smoothed when combined through averaging, consequently helping in generalizing well to unseen data and reducing the risk of overfitting.

4. The optimal hyperparameters, such as the number of subsets and epochs before each round of central averaging vary by dataset and domain. The appropriate subset count depends on dataset size and class distribution. Suppose the edge model is complex or the subset count is too high. In that case, the overly reduced size and number of samples per class in each subset can result in significant overfitting in edge models, which in turn can degrade overall performance even after central averaging. This is similar to what happens in federated learning when the client data is minimal, as pointed out by Zhang et al. [17]. When sufficient data exists per class, more subsets can be beneficial, as noise or outliers are diluted by a factor of **N** (number of training subsets).

IV. Proposed Methodology

A. Creating Training Subsets

To train multiple neural networks in parallel with different data, the training set is first randomly shuffled and then divided into multiple subsets, maintaining consistent class distribution. Each subset is then fed into a separate model with identical neural network architecture and trainable parameters to enable parameter averaging.

B. Training & Averaging of Trained Parameters

Instead of one training loop, Our method requires two loops: an inner loop for subset training which is nested within an outer loop that performs central averaging. A central model with the same architecture as the edge models is first initialized. Within the inner loop, each edge model is iteratively trained on its subset for a specified number of local epochs. Then, the edge models' parameters are scaled based

Algorithm 1 Divide2Conquer

Input: Training dataset D, Number of subsets N, Local epochs E, Global epochs E_{global}, Batch size B, Learning rate lr
Output: Central model M_c with aggregated weights

function Divide2Conquer(D, N, E, E_{global}, B, lr)
Shuffle the dataset D
Divide D into N subsets: $\{D_1, D_2, \ldots, D_N\}$, maintaining class distributions
Initialize central model M_c with parameter vector θ_c
for $i = 1$ **to** N **do**
Initialize edge model M_i with $\theta_i = \theta_c$
Set scaling factor $s_i = \frac{|D_i|}{|D|}$
end for
for $g = 1$ **to** E_{global} **do**
Initialize weight accumulator $W_{central} = 0$
for $i = 1$ **to** N **do**
Set $\theta_i = \theta_c$
for $e = 1$ **to** E **do**
Train M_i on D_i using batch size B and learning rate lr
end for
Obtain updated weights θ_i from M_i
Scale weights: $\theta_i = s_i \cdot \theta_i$
Accumulate scaled weights into $W_{central}$: $W_{central} = W_{central} + \theta_i$
end for
Compute averaged central weights:

$$\theta_c = \frac{1}{\sum_{i=1}^{N} s_i} W_{central}$$

Update central model M_c with new weights θ_c
end for
return central model M_c
end function

on their data fraction and added to a local weight list. In the outer loop, these scaled parameters are averaged, updating the central model's parameters. Each edge model's weights are then reinitialized to match the central model, completing a global training epoch. This process repeats over several global epochs, with weighted parameter averaging to account for the data fraction of each subset. The whole process is summarized in the pseudo-code to implement D2C in Algorithm 1.

C. Hyperparameter Tuning

Tuning the overall model consists of two stages. Firstly, the edge model and its hyperparameters should be tuned using a subset/entire training set and the validation set to find a suitable model to train each training subset. Next, that base model is utilized to tune the Central model. The new **global hyperparameters** that need tuning, in this case, are the **Number of Subsets of the Training Set** and the **Number of Epochs before each round of Global Averaging**. Testing for different values for these hyperparameters, the implementation and the evaluation of our method are done using the best combination. The appropriate number of training subsets can be determined first by varying the number of training subsets and observing the performance metrics like test/validation accuracy, F1 score, log loss, and also the validation loss curve. The appropriate number of training epochs in each subset before the averaging is done each time is likely to be a small number $(1 \sim 2)$ since the edge model can quickly overfit the training subset with a reduced size. So, it makes more sense to set a small number of epochs initially and tune the number of training subsets. Once the number of subsets is determined, the same process can be repeated by varying the number of training epochs and determining the appropriate value of it.

V. Experimental Setup

A. Datasets

We used the Fashion MNIST [8] dataset for image classification, which consists of 70,000 grayscale images spread across ten fashion item categories. It is a benchmark dataset and its complexity and class diversity introduce variability and noise, making it suitable for our investigation into minimizing overfitting and improving generalization. For text classification, we used the AG NEWS [12] dataset, a benchmark dataset of approximately 127,600 news articles from four categories: World, Sports, Business, and Science/Technology. Its large size and balanced distribution make it ideal for evaluating generalization performance in Big Data contexts.

B. Experimental Details

For each dataset, We used a common Neural network architecture and identical hyperparameters to create the central model and the edge models, which is essential for weight aggregation and averaging. For **Image Classification**, we reshaped, normalized, and one-hot encoded images, then split the data 90-10 for training and validation using Scikit-Learn. The training set was further divided into subsets, each converted into tensors and batched. Each edge model (and hence the central model) had 4 convolutional blocks (32 to 256 kernels, 2x2 MaxPooling) and one fully connected layer with 128 neurons followed by the output layer. For **Text Classification**, we created a corpus, built word embeddings with 100-dimensional GloVe, and padded sequences to a uniform length. The edge model comprised 3 bidirectional LSTM layers (128, 64, 32 units) and two dense layers (128, 64 neurons) with an output layer. In both tasks, we used a 90-10 Training–Validation split, Adam optimizer, and applied dropout and batch normalization to reduce overfitting, supporting our method's compatibility with other overfitting prevention techniques. We used the predefined test sets for testing.

VI. Results and Discussions

We used loss and accuracy curves, Test Accuracy, F1 Score, and log loss to analyze the results and evaluate the efficacy of our method. Another goal was to optimize the two new

global hyperparameters discussed in the previous section: the **Number of Subsets of the Training Set** and the **Number of Epochs before each round of Global Averaging**, referred to as N and E respectively throughout the paper.

A. Visualizing Decision Boundary

Firstly, we visualize the change in a model's decision boundary when D2C is applied. It is well known that when a model is overfitting, the decision boundary often becomes overly complex and jagged due to high variance. So, an effective overfitting reduction method should result in a smoother decision boundary. We used a simple binary classification dataset to simulate the impact of D2C on a simple decision boundary. This synthetic dataset consists of 240 data points and we created it using *Scikit-Learn*. The model architecture was comprised of three hidden layers each having 100 neurons. No regularization or data augmentation was used (only for decision boundary visualization), to observe the effect of D2C explicitly. At first, we followed the traditional approach and fed the entire training dataset to the sole model. Following the conclusion of training, we got the decision boundary shown in Figure 1a. Then, we applied the D2C method, i.e., divided our training set into multiple subsets. Then each of the subsets was trained parallelly and after a few epochs, the trained weights from each subset were averaged to get the final weights. In this case, we divided the training set into 3 subsets, resulting in the decision boundary shown in Figure 1b. Figure 1 shows us the change in the decision boundary brought by D2C.

The decision boundary for the traditional approach is highly complex and non-linear. The model creates intricate regions to distinguish between the two classes. The boundary closely follows individual points, resulting in a jagged, convoluted shape. The complexity of the decision boundary suggests that the model is overfitting to the training data. It is trying too hard to separate every point, including noise, rather than focusing on general patterns in the data. Evidence of overfitting is seen in how the model creates small pockets of blue in the red region and vice versa. This suggests the model is likely to produce very good results on training data, but may struggle with new, unseen data due to its sensitivity to specific details in the training set. The highly irregular boundary reflects poor generalization ability.

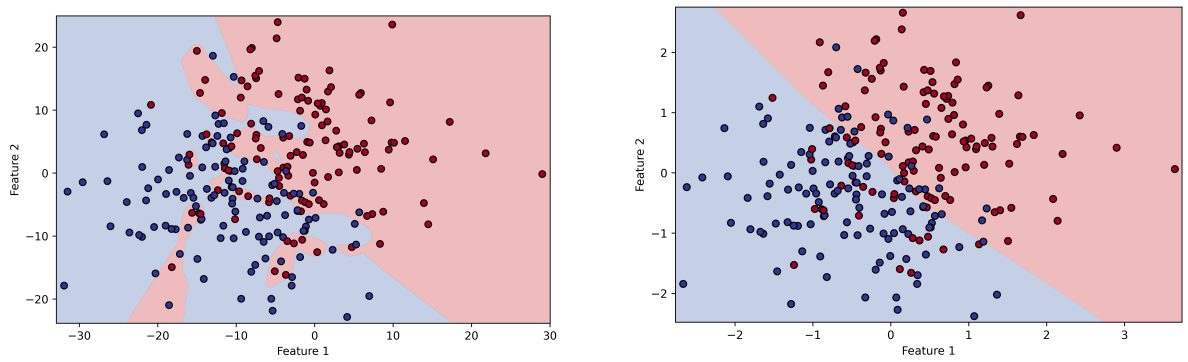
The decision boundary after applying D2C is much simpler, dividing the feature space almost diagonally. This suggests a less flexible, more generalized model. The decision boundary is smooth and does not follow the exact positions of individual data points as closely. The model is more generalized, focusing on the overall trend in the data rather than trying to accommodate every individual point, exhibiting much less overfitting compared to what we saw in the traditional approach. This clear improvement in generalization ability stems from the fact that the noisy samples are divvied up among the subsets. So, irregularity around the individual noisy sample does not affect all the subsets, but rather only one of them. Hence, the impact of the individual outlier is minimized after averaging. This is a clear sign of our proposed D2C being an effective overfitting-reducing method.

B. Analyzing Loss and Accuracy Curves

The loss curve is a definitive indicator of overfitting. When a model maintains a lower validation loss during training, it indicates that the model can generalize well to unseen data. Theoretically, overfitting occurs when a model becomes overly complex relative to the data it is learning, allowing it to capture not only meaningful patterns but also specific noise or minor irregularities unique to the training set. This focus on noise causes the model to perform poorly on new data, which is reflected in a higher validation loss or an accelerated rise in validation loss. In contrast, a model that keeps validation loss lower while training likely has a structure that is well-suited to the data's true underlying patterns without becoming excessively sensitive to its unique characteristics. That's why we monitor the validation loss curves from the different experiments we performed to compare and analyze if D2C improves the generalizing performance, in the form of minimizing the validation loss as the training progresses. For now, we keep the number of epochs before each round of central averaging, $\boldsymbol{E}$, equal to 1. It will ensure a proper comparison between the traditional method and our proposed method while varying the number of subsets, $\boldsymbol{N}$. In our experiments, how much we varied N depended on the performance (Loss curve, Accuracy) trend as we increased N, the number of subsets. In the case of all the datasets, we stopped varying N whenever we observed a definitive performance drop with increasing N.

Image Classification using Fashion MNIST: The loss and accuracy curves from Figure 2a and 2b show proof of what we inferred before. The validation loss starts increasing quite abruptly (the red line) after a few epochs in the case of the traditional approach. The overfitting in this case is very much apparent. The rate of increase in validation loss as the training progresses goes down very rapidly as we apply D2C and increase the number of subsets. The validation accuracy, however, doesn't come down as the loss increases (hence early stopping is not an option). Rather counterintuitively, the validation accuracy maintains an increasing trend almost throughout the 100 epochs. We also notice a slight improvement in the peak stable accuracy with our proposed method over the traditional approach. So, our method improves both validation accuracy and loss in this case during the training process, depicting better generalization.

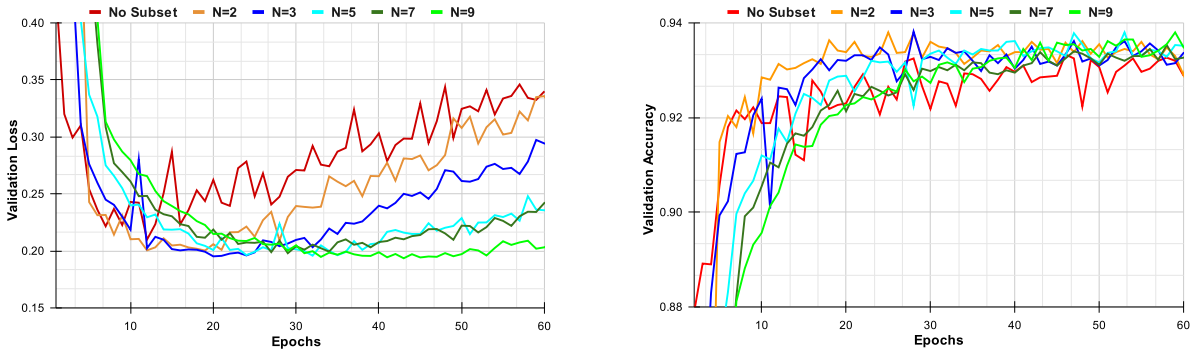
Text Classification using AG News: The loss and accuracy curves generated for AG News depict the clear improvements brought by D2C over a traditional monolithic approach. Figure 2c shows that our decentralized approach delays and minimizes the increase in validation loss significantly as the training progresses. The decreasing trend in validation loss persists longer when using D2C on AG News as opposed to the normal approach where it starts increasing after a short period. The convergence though, slows down a bit in this case too. More importantly, with no subsets, the validation loss shoots up sharply after around 8 epochs, which indicates



(a) No Subsets (Traditional)

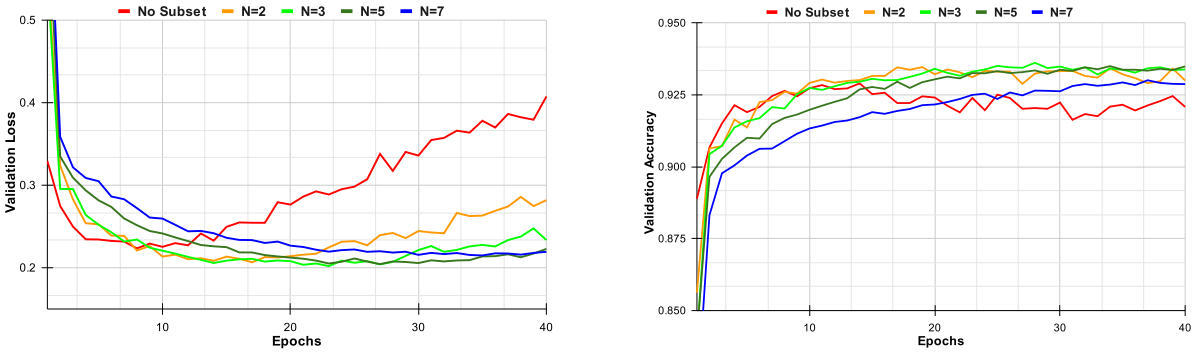
(b) N = 3 (D2C)

Fig. 1: Decision boundary Visualization with the Traditional approach vs. the Divide2Conquer method.



(a) Loss on Fashion MNIST

(b) Accuracy on Fashion MNIST



(c) Loss on AG News

(d) Accuracy on AG News

Fig. 2: Curves for the traditional approach vs. D2C for different numbers of subsets on Fashion MNIST and AG News.

severe overfitting and impacts the accuracy too as can be seen from Figure 2d. However in turn when using our approach, the same sharp increase in the validation loss is mitigated drastically. Even with only two subsets, the improvement is quite significant. When the number of subsets is increased further to more than two, the loss curve becomes quite stable, almost entirely eliminating the increasing trend and showing consistent improvement as the training progresses.

The experiments on Fashion MNIST and AG News show that D2C improves both true accuracy and true loss on unseen data during the training process compared to the traditional centralized training method. This further consolidates our findings across multiple domains.

C. Evaluating Accuracy, F1 Score, ROC_AUC & Log Loss

Image Classification: Table I summarizes the findings from the predefined test set of the Fashion MNIST based on our proposed method. We can observe that the best balance in accuracy, F1 score, and ROC_AUC were achieved using the Divide2Conquer method setting N and E were both set to **3**.

The best result in terms of accuracy came, however, when the hyperparameters N and E were set to **7** and **1** respectively. Our approach shows improvement in terms of all the metrics over the traditional approach with no subsets. Table I also shows that there is a trend of decreasing log loss as we increase the number of subsets. However, no such trend is visible consistently when we increase the number of epochs, E. The decrease in Log Loss is important since a lower log loss indicates that the model's predicted probabilities align better with the true class labels. This means the model is more confident and accurate in its predictions, even if the accuracies are similar. So, we can conclude that D2C clearly brings an improvement over the traditional approach, despite being used on top of other generalization techniques.

TABLE I: Performance Metrics with Traditional Approach vs. Different Settings Using the D2C Method on Fashion MNIST.

Model Settings	*Accuracy*	*F1-Score*	*Log Loss*	*AUC-ROC*
No Subsets	0.9314	0.9315	0.3467	0.99559
N=2, E=1	0.9323	0.9319	0.3460	0.99549
N=3, E=1	0.9330	0.9329	0.3179	0.99545
N=3, E=2	0.9319	0.9316	0.3303	0.99552
N=3, E=3	0.9337	**0.9337**	0.3225	0.99590
N=5, E=1	0.9322	0.9321	0.2692	0.99589
N=7, E=1	**0.9338**	0.9334	0.2467	0.99584
N=7, E=2	0.9305	0.9304	0.2575	**0.99600**
N=9, E=1	0.9323	0.9323	**0.2341**	**0.99600**

Text Classification: Table II summarizes the findings from our experiments with the AG News dataset. We can clearly observe that the best accuracy and F1 Score were achieved by D2C, for $N = 5$ and $E = 2$ respectively. Once again, a significant improvement over the traditional approach with no subsets can be perceived in terms of all the metrics. The best method in terms of ROC_AUC happens to be the model with the highest number of subsets, i.e., $N = 7$. The effect of using D2C is more apparent in this dataset compared to the previous one. The accuracy improved significantly compared to the traditional approach in every variation of our proposed method (for all values of N and E). AG News is the largest dataset we've used for our experiments. For each class, we had 30000 samples in this dataset. So, despite dividing the dataset into more and more subsets, each edge model had access to enough data to train on and at the same time take advantage of the regularizing effect that dividing the training set introduces. As a result, the performance of the models improved significantly, achieving better generalization.

VII. Conclusion

Our study demonstrates the proposed D2C method can treat overfitting and produce significant performance gains through better generalization. In this paper, we tested D2C across multiple datasets from different domains. This method outperformed the traditional approach in all cases in terms of key metrics and showed significantly better resistance against overfitting. D2C showed a consistent ability to minimize the validation loss during training and to produce smoother decision boundaries. Furthermore, D2C can complement other generalization techniques and performs better with larger datasets, highlighting its potential for real-world applications.

TABLE II: Performance Metrics with Traditional Approach vs. Different Settings Using D2C on AG News.

Model Settings	*Accuracy*	*F1 Score*	*Log Loss*	*ROC_AUC*
No Subsets	0.9243	0.9243	0.2611	0.98764
N=2, E=1	0.9291	0.9291	0.2556	0.98873
N=3, E=1	0.9288	0.9288	0.2520	0.98872
N=3, E=2	0.9275	0.9275	0.2567	0.98852
N=5, E=1	0.9293	0.9292	0.2412	0.98881
N=5, E=2	**0.9295**	**0.9296**	0.2451	0.98875
N=7, E=1	0.9271	0.9271	**0.2353**	**0.98916**

References

[1] Kim, M., Choi, C. & Pan, S. Ensemble networks for user recognition in various situations based on electrocardiogram. *IEEE Access*. **8** pp. 36527-36535 (2020)

[2] Konečnỳ, J., McMahan, B. & Ramage, D. Federated optimization: Distributed optimization beyond the datacenter. *ArXiv Preprint ArXiv:1511.03575*. (2015)

[3] Shanmugavel, A., Ellappan, V., Mahendran, A., Subramanian, M., Lakshmanan, R. & Mazzara, M. A novel ensemble based reduced overfitting model with convolutional neural network for traffic sign recognition system. *Electronics*. **12**, 926 (2023)

[4] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The Journal Of Machine Learning Research*. **15**, 1929-1958 (2014)

[5] Cogswell, M., Ahmed, F., Girshick, R., Zitnick, L. & Batra, D. Reducing overfitting in deep networks by decorrelating representations. *ArXiv Preprint ArXiv:1511.06068*. (2015)

[6] Ioffe, S. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ArXiv, 2015. ArXiv:1502.03167*.

[7] Ying, X. An overview of overfitting and its solutions. *Journal Of Physics: Conference Series*. **1168** pp. 022022 (2019)

[8] Xiao, H. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *ArXiv Preprint ArXiv:1708.07747*. (2017)

[9] Kolluri, J., Kotte, V., Phridviraj, M. & Razia, S. Reducing overfitting problem in machine learning using novel L1/4 regularization method. *2020 4th International Conference On Trends In Electronics And Informatics (ICOEI)(48184)*. pp. 934-938 (2020)

[10] Siddiqui, M., Shusmita, S., Sabreen, S. & Alam, M. FedNet: Federated Implementation of Neural Networks for Facial Expression Recognition. *2022 International Conference On Decision Aid Sciences And Applications (DASA)*. pp. 82-87 (2022)

[11] Li, X., Chen, S., Hu, X. & Yang, J. Understanding the disharmony between dropout and batch normalization by variance shift. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 2682-2690 (2019)

[12] Zhang, X., Zhao, J. & LeCun, Y. Character-level convolutional networks for text classification. *Advances In Neural Information Processing Systems*. **28** (2015)

[13] Morgan, N. & Bourlard, H. Generalization and parameter estimation in feedforward nets: Some experiments. *Advances In Neural Information Processing Systems*. **2** (1989)

[14] Prechelt, L. Early stopping-but when?. *Neural Networks: Tricks Of The Trade*. pp. 55-69 (2002)

[15] Fürnkranz, J. Pruning algorithms for rule learning. *Machine Learning*. **27** pp. 139-172 (1997)

[16] Sun, Y., Wang, X. & Tang, X. Deep learning face representation from predicting 10,000 classes. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*. pp. 1891-1898 (2014)

[17] Zhang, Xu, Yinchuan Li, Wenpeng Li, Kaiyang Guo, and Yunfeng Shao. "Personalized federated learning via variational Bayesian inference." In International Conference on Machine Learning, pp. 26293-26310. PMLR, 2022.