HHS Public Access
Author manuscript
Lect Notes Monogr Ser. Author manuscript; available in PMC 2021 September 15.

Published in final edited form as:
Lect Notes Monogr Ser. 2020 October ; 12444: . doi:10.1007/978-3-030-60548-3_17.

Federated Gradient Averaging for Multi-Site Training with Momentum-Based Optimizers

Samuel W. Remedios[1,2,3], **John A. Butman**[3], **Bennett A. Landman**[2], **Dzung L. Pham**[3,4]

[1]Johns Hopkins University, Baltimore MD 21218, USA

[2]Vanderbilt University, Nashville TN 37235, USA

[3]Clinical Center, National Institutes of Health, Bethesda MD 20814, USA

[4]Center for Neuroscience and Regenerative Medicine, Henry M. Jackson Foundation, Bethesda MD 20817, USA

Abstract

Multi-site training methods for artificial neural networks are of particular interest to the medical machine learning community primarily due to the difficulty of data sharing between institutions. However, contemporary multi-site techniques such as weight averaging and cyclic weight transfer make theoretical sacrifices to simplify implementation. In this paper, we implement federated gradient averaging (FGA), a variant of federated learning without data transfer that is mathematically equivalent to single site training with centralized data. We evaluate two scenarios: a simulated multi-site dataset for handwritten digit classification with MNIST and a real multi-site dataset with head CT hemorrhage segmentation. We compare federated gradient averaging to single site training, federated weight averaging (FWA), and cyclic weight transfer. In the MNIST task, we show that training with FGA results in a weight set equivalent to centralized single site training. In the hemorrhage segmentation task, we show that FGA achieves on average superior results to both FWA and cyclic weight transfer due to its ability to leverage momentum-based optimization.

Keywords

federated learning; multi-site; deep learning

1 Introduction

The quality of a machine learning model stems directly from its training data. Deep neural networks especially benefit from large datasets, utilizing thousands or millions of parameters to learn salient feature weights to account for many types of variation. The same is true when applying such models to medical imaging, except that site effects such as scanner manufacturer, acquisition parameters, and subject cohort greatly impact the generalization of a model trained to different sites. It is common to find models trained from a single

samuel.remedios@nih.gov .

site's data perform poorly on a new site [7]. This problem is further compounded by the difficulty of data sharing with medical images, which arises from ethical, logistical, and political concerns.

To address this problem, several multi-site approaches have been suggested. The term "federated learning" (FL) was first widely known from [12] in the application of learning from mobile devices. Here, the authors introduced a scenario in which an arbitrary number of clients solve individual optimization problems and aggregate their parameters as a central weighted average. This approach to FL has already seen implementation on real-world medical datasets such as brain tumor segmentation [14]. Similarly, asynchronous stochastic gradient descent [36] considered different GPU devices as clients, splitting training data among them and aggregating gradients before performing weight update to decrease training time. Related research in the field of continual learning [11, 20, 20] investigates methods to mitigate catastrophic forgetting [6], the phenomenon by which neural networks have no guarantee to retain useful weight values when training on new data. Bayesian neural networks, in which parameters are not point estimates but distributions of parameters, allow techniques such as distributed weight consolidation [18] to consider multi-site training as a continual learning problem. In transfer learning [22, 34], also known as pre-training or finetuning, model parameters are learnt first on some (usually large) dataset and subsequently trained for the problem of interest on a smaller dataset. Cyclic weight transfer (CWT) [4, 25] is a distributed method that leverages transfer learning to iteratively train a model at different institutes. While model generalization was reported to have improved, there is no mathematical guarantee that the CWT models will converge to an optimum of the combined datasets.

Although recent works [15] have shown the theoretical convergence for federated weight averaging (FWA, previously formulated as FedAVG [19]), in practice there is still a performance gap between centralized training at a single site and collaborative learning methods [28]. In contrast, federated gradient averaging (FGA) should offer equivalent performance to single-site learning, first formulated as FedSGD [19]. Gradient averaging has been a standard practice since the advent of deep learning [3] with regards to training a neural network using batches, as weight updates from a single sample are noisy and updates from the entire dataset are undesirable for empirical and theoretical reasons. Gradient averaging has also been used in asynchronous stochastic gradient descent [36] and in the TernGrad work [32], which considers the quantization of gradients to reduce network overhead.

In this work, we recast FedSGD [19] as FGA, show its equivalence to centralized training, and show FGA permits the use of not only SGD but also momentum-based optimizers such as Adam [10] without sacrificing the benefits of momentum via variable resets. We show the first use case of FGA for learning from unshared data housed at physically separate institutes and directly compare it to federated weight averaging (FWA) [14] and cyclic weight transfer (CWT) [25]. We first evaluate FGA in a simulated disjoint multi-site scenario with the handwritten digit classification dataset MNIST [13], then on a multi-site CT hemorrhage segmentation task.

2 Method

2.1 Background

The training of artificial neural networks can be interpreted as finding parameters Θ for a function f which minimize a loss function $\mathscr{L}$ over N input-output pairs $\{x_i, y_i\}$, $i = 1, 2, \ldots, N$:

$$\underset{\Theta}{\arg\min} \sum_{i}^{N} \mathscr{L}(f_{\Theta}(x_i), y_i) \tag{1}$$

These parameters are usually updated iteratively via gradient descent optimization:

$$\Theta_t \leftarrow \Theta_{t-1} + \eta \nabla_{\Theta_{t-1}} \frac{1}{N} \sum_{i}^{N} \mathscr{L}\left(f_{\Theta_{t-1}}(x_i), y_i\right) \tag{2}$$

In other words, the new weights Θ at time t are updated from the previous weights at time t−1 plus the gradient of the loss function scaled down by η, the learning rate (or step size). It is conventional to use Stochastic Gradient Descent (SGD)-based optimizers in practice, where the gradient is not calculated over all of the training samples but a randomly sampled subset (known as a batch).

To accelerate training, several momentum-based optimizers have been proposed [27]. The use of momentum has been shown to reduce sensitivity to noisy gradients and improve overall convergence. Adam [10] is a ubiquitous deep learning optimizer that updates parameters via momentum variables $\hat{m}$ and $\hat{v}$ (described in Algorithm 1 in [10]), each of which are functions of the gradients $\nabla \Theta_{t-1}$.

$$\Theta_t \leftarrow \Theta_{t-1} + \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{3}$$

2.2 Federated Learning

Multi-site learning approaches aim to find parameters Θ which minimize the loss L not over just N input-output pairs at a single site, but at all participating sites $s = 1, 2, \ldots$, S of S participating sites. The class of multi-site learning approaches that we will discuss make use of a central server c which administrates weight and gradient collection, aggregation, and distribution.

Because the number of samples calculated within a batch is independent of the gradient calculation, the gradient of the average is the average of the gradients. We initialize the model f_{Θ} at the central server c and copy the model to each local site s. The local sites sample a training batch and compute a gradient, then send the gradient back to the central server. The server averages the gradients across sites and sends the averaged gradient to each local site. Each local site computes its own momentum terms based on the averaged gradient and updates weights. In this way, the momentum across sites is guaranteed to be identical (within floating point errors) due to calculation based on the same gradient. The exact procedure is described in Algorithm 1, expanded from FedSGD [19] to work with momentum-based optimizers.

Algorithm 1 FGA implementation. As in [10], g_t^2 indicates the Hadamard (element-wise) product $g_t \odot g_t$, and β_1^t and β_2^t refer to raising β_1 and β_2 to the power t.

Require: local training data $\mathcal{D}_s = \{x_i, y_i\}_{i=1}^{N_s}$
Require: central learning rate η, local decay rates $\beta_{s,1}, \beta_{s,2}$, global small constant ϵ
Require: differentiable loss function $\mathscr{L}$ defined on training pairs (x, y)
Require: Central initialized model Θ_c
procedure GLOBAL_TRAINING
  for $s \leftarrow 1, 2, ..., \mathcal{S}$ **do**
    Initialize local site time-step: $t_s \leftarrow 0$
    Initialize local site momentum terms: $m_{s,t} \leftarrow 0, v_{s,t} \leftarrow 0$
    Initialize local site model: $\Theta_{s,t} \leftarrow \Theta_c$
  end for
  while Convergence criteria not met **do**
    for $s \leftarrow 1, 2, ..., \mathcal{S}$ **do**
      $t_s \leftarrow t_s + 1$
      Sample local site training batch: $\{x_{s,i}, y_{s,i}\} \in \mathcal{B}_s \sim \mathcal{D}_s$
      Compute local site batch gradient: $g_s \leftarrow \nabla_\Theta \mathscr{L}(f_{\Theta_{s,t}}(x_{s,i}), y_{s,i})$
      Send gradient to central server
    end for
    Central server averages gradients across sites: $g_c \leftarrow \frac{1}{S}\sum_{s=1}^{S} g_s$
    Send averaged gradient g_c to each local client.
    for $s \leftarrow 1, 2, ..., \mathcal{S}$ **do**
      Compute 1st moment: $m_{s,t} \leftarrow \beta_{s,1} \cdot m_{s,(t-1)} + (1 - \beta_{s,1}) \cdot g_c$
      Compute 2nd moment: $v_{s,t} \leftarrow \beta_{s,2} \cdot v_{s,(t-1)} + (1 - \beta_{s,2}) \cdot g_c^2$
      Compute bias-corrected 1st moment: $\hat{m}_{s,t} \leftarrow m_{s,t}/(1 - \beta_{s,1}^t)$
      Compute bias-corrected 2nd moment: $\hat{v}_{s,t} \leftarrow v_{s,t}/(1 - \beta_{s,2}^t)$
      Update local model: $\Theta_{s,t} \leftarrow \Theta_{s,(t-1)} - \eta \cdot \hat{m}_{s,t}/(\hat{v}_{s,t} + \epsilon)$
    end for
  end while
end procedure

2.3 Implementation

We have implemented FGA as a set of two main Python scripts: server and client. We enabled communication between server and client with secure shell tunneling. The server script is a Python Flask server which runs indefinitely and awaits incoming gradients. After gradients from all participating sites are received they are averaged element-wise and returned to each client. The client script is identical at each site and is a conventional neural network script with the additional step of sending gradients to the server and awaiting the averaged gradient before using an optimizer to update the local weights. Our source code is publicly available here [24]. FL was conducted between two different physical institutions: the NIH Clinical Center (A) and Vanderbilt University (B) . At A, we used TensorFlow 2.1, a Tesla V100-SXM3 GPU, CUDA version 10.1, and NVIDIA driver version 418.116.00. At B, we used TensorFlow 2.0 with the TensorFlow Determinism Patch [21], a GeForce RTX 2080 Ti GPU, CUDA version 10.2, and NVIDIA driver version 440.48.02.

3 Experiments

We evaluated our implementation of FGA in a two scenarios: the handwritten dataset MNIST [13] data with simulated multi-site separation and a real multi-site dataset for CT hemorrhage segmentation [25]. With MNIST, we compare FGA to single site centralized data training, single site training at A and B, FWA, and CWT. With CT hemorrhage segmentation, we make the same comparisons except for centralized data training due to privacy restrictions on data transfer. Because there are two training institutes in these experiments, our implementation of CWT results in two different models; one model begins at institute A and after 100 epochs ends at institute A and the other model begins at institute B and ends at institute B. Both CWT models are evaluated at all test sites.

3.1 MNIST

From the MNIST dataset we split the training and validation sets into two disjoint sets such that A has image-label pairs of digits 0, 1, 2, 3, 4 and B has image-label pairs of digits 5, 6, 7, 8, 9. Models at both sites have 10 final neurons and thus have the capacity to predict classes at the other site, but have no supporting training or validation data. The test set is entirely withheld and contains samples from all ten digits. An epoch was defined as the model training on every sample from the training set once.

A small convolutional neural network (CNN) architecture was used, consisting of four convolutional layers with 16 3×3 filters activated by ReLU alternating with max pooling layers and finally connected through global max pooling to a fully connected layer with neurons equal to the number of classes. Softmax activation was used for final classification probabilities. All input and output data as well as model weights were 64-bit floats for greater precision in comparing FGA and single-site centralized data training. Convergence was defined as completing 100 training epochs. We trained the CNN using the Adam optimizer with a learning rate of 10^{-4} and with a batch size of 4096 for all methods except FGA for which the client batch size was 2048 to compare against single-site centralized-data training. This is because FGA updates weights by gradients from all participating sites. As a result, only 6 batches were needed to complete an epoch with FGA. To facilitate comparison between FGA and single-site centralized-data training, training data were not shuffled between epochs.

3.2 Hemorrhage Segmentation

A full description of the CT head imaging data is provided in [25]. Site A consisted of 34 training volumes and 34 testing volumes, and site B had 32 training volumes and 27 testing volumes. Additional test data (stored at Site A but acquired at different locations and using different scan protocols) were labeled Site C (11 volumes) and Site D (20 volumes). Since all selected multi-site methods are agnostic to architecture choice, a modified, reduced U-net [26] was chosen as the CNN architecture, replacing double convolution layers and max pooling layers with single convolution layers of kernel size 5 and stride 2 on the down-sampling arc and transpose convolution layers with the same kernel and stride settings to replace double convolution and up-sampling layers on the up-sampling arc. The exact architecture is publicly available [24]. These implementation choices were made to retain the same field-of-view but reduce memory consumption and processing time. This 2D reduced U-net was trained on randomly extracted 2D patches as described in [25] for 100 epochs with a batch size of 256 using the Adam optimizer with a learning rate of 10^{-4}. During training and before patch collection, 20% of the CT volumes are set aside for validation.

4 Results

4.1 MNIST

A comparison of all multi-site learning methods is shown in Table 1. Since our goal is to show the equivalence of FGA to centralized training, the final convergence of the model is secondary to the weight values. When inspecting the individual weight values of single site $A \cup B$ and FGA, they differed by no more than 10^{-12} due to floating point rounding due to gradient calculations. This is expected because FGA is mathematically equivalent to centralized training. Regarding performance across methods, as expected, Single Site A and B each only predict the classes for which training data was provided, resulting in about 50% accuracy. Since CWT methods each terminate at one site, their models are biased towards their most recent site.

4.2 Hemorrhage Segmentation

Qualitative results are shown in Fig. 1 and quantitative results are shown in Tab. 2. Considering the union of all test sets, FGA significantly ($p < 0.005$) outperforms the other multi-site training methods although the improvement within some test data sets was subtle. On dataset D, cyclic model B achieved greater performance, though FGA is still comparable and more stable, as it does not depend on CWT terminating at a site which may have a similar distribution to that test set. Qualitatively, all methods generated reasonable results, but in general FGA consistently suffered from fewer false positives. CWT performed as well or better than FWA in this scenario, as opposed to the MNIST scenario with disjoint training data.

5 Discussion

We have implemented federated gradient averaging (FGA) with the intent to enable momentum-based optimizers to be used in collaborative learning scenarios without sacrificing the benefits of momentum. We first validated FGA on a disjoint handwritten digit classification task and compared to FWA, CWT, and single-site training. Since FGA results in identical weights at each learning step to the single site scenario, we conclude that they are equivalent. We then applied FGA to truly multi-site data in a head CT hemorrhage segmentation task and showed its improved performance over other federated methods. Furthermore, we have shown that this approach is stable for application in real-world scenarios where different physically separate sites have different hardware, framework versions, driver versions, and CUDA library versions.

Regarding training time and communication overhead, FGA is about 2× slower than FWA and CWT, and FWA and cyclic weight transfer are about the same speed as the slowest machine in single site training. Regarding scalability, FGA suffers no time penalty as the number of participating sites increases but is hampered in speed by the slowest participating site. CWT does not scale well as the number of participants increases, as the model must complete an epoch at each site before “seeing” all the data. FWA enjoys the best of both worlds, scaling well with the number of participants without delaying learning from unseen sites between epochs, but does not achieve performance equal to centralized training [28, 19, 14]. However, in general, FGA suffers from much more communication overhead than FWA and CWT, which indeed was partially the motivation for FWA [19]. There are many other factors involved in determining convergence time across sites, including variation in traditional hyperparameter choice (batch, learning rate, number of training epochs), dataset size, hardware, framework and driver versions, and network connectivity. Beyond these, all methods proposed in this paper involve some form of synchronization. For FGA, every step is performed in lockstep, and thus all sites must wait for the slowest site to finish processing its batch before averaging is performed. For FWA, every epoch is performed in lockstep and all sites must therefore wait for the slowest site to complete its epoch before performing the average. Additionally, many methods have considered gradient averaging as incurring too much network overhead, which is one reason why weight averaging is preferred [19, 35]. There is also recent research to transform the gradients into communication-efficient variants [32, 33, 2, 16].

Finally, when deploying collaborative learning the protection of patient privacy is of utmost importance, especially when working with medical data [1, 5, 17, 31]. Previous works have demonstrated the dangers of sending raw data through the network [30] and to this end differentially private (DP) methods have been proposed [8, 29, 14]. With both FWA and FGA approaches, implementation of DP mitigates one class of privacy concerns, although FWA benefits particularly from obfuscation of batch gradients by sending weights only at the end of the epoch instead of at the end of the batch. In this sense, FGA relies even more on DP methods. However, the reconstruction of training data from the weights or gradients alone is nontrivial [30]. Despite difficulties to reconstruct data or determine whether a subject was a participant in a study, all collaborative multisite learning approaches are still vulnerable to malicious participant attacks [9]. Even DP models may leak training distribution information since they operate by tracking weight update changes over time. Best practices regarding safeguarding of patient data, model variants, and collaborative learning method selection are still under discussion by groups such as Project MONAI [23].

Acknowledgments

We would like to first thank Shuo Han, Blake Dewey, Aaron Carass, and Jacob Reinhold from Johns Hopkins University for insightful conversations about determinism in GPUs and floating point precision. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No.DGE-1746891. Support for this work included funding from the Intramural Research Program of the NIH Clinical Center and the Department of Defense in the Center for Neuroscience and Regenerative Medicine, the National Multiple Sclerosis Society RG-1507-05243 (Pham), and NIH grant 1R01EB017230-01A1 (Landman), as well as NSF 1452485 (Landman). This work received support from the Advanced Computing Center for Research and Education (ACCRE) at the Vanderbilt University, Nashville, TN, as well as in part by ViSE/VICTR VR3029. We also extend gratitude to NVIDIA for their support by means of the NVIDIA hardware grant.

References

1. Accountability Act: Health insurance portability and accountability act of 1996. Public law104, 191 (1996)
2. Alistarh D, Grubic D, Li J, Tomioka R, Vojnovic M: Qsgd: Communication-efficient sgd via gradient quantization and encoding. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R (eds.) Advances in Neural Information Processing Systems 30, pp. 1709–1720. Curran Associates, Inc. (2017)

3. Bengio Y: Practical recommendations for gradient-based training of deep architectures. In: Neural networks: Tricks of the trade, pp. 437–478. Springer (2012)
4. Chang K, Balachandar N, Lam C, Yi D, Brown J, Beers A, Rosen B, Rubin DL, Kalpathy-Cramer J: Distributed deep learning networks among institutions for medical imaging. Journal of the American Medical Informatics Association (2018)
5. Fetzer DT, West OC: The hipaa privacy rule and protected health information: implications in research involving dicom image databases. Academic radiology15(3), 390–395 (2008) [PubMed: 18280936]
6. French RM: Catastrophic forgetting in connectionist networks. Trends in cognitive sciences3(4), 128–135 (1999) [PubMed: 10322466]
7. Gibson E, Hu Y, Ghavami N, Ahmed HU, Moore C, Emberton M, Huis-man HJ, Barratt DC: Inter-site variability in prostate segmentation accuracy using deep learning. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 506–514. Springer (2018)
8. Goodfellow I, Bengio Y, Courville A: Deep learning. MIT press (2016)
9. Hitaj B, Ateniese G, Perez-Cruz F: Deep models under the gan: information leakage from collaborative deep learning. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 603–618 (2017)
10. Kingma DP, Ba J: Adam: A method for stochastic optimization. arXiv preprintarXiv:1412.6980 (2014)
11. Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, Milan K, Quan J, Ramalho T, Grabska-Barwinska A, et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences114(13), 3521–3526 (2017)
12. Konecnỳ J, McMahan HB, Ramage D, Richtárik P: Federated optimization: Distributed machine learning for on-device intelligence. arXiv preprint arXiv:1610.02527 (2016)
13. LeCun Y, Cortes C, Burges C: Mnist handwritten digit database. ATT Labs[Online]. Available: http://yann.lecun.com/exdb/mnist2 (2010)
14. Li W, Milletarì F, Xu D, Rieke N, Hancox J, Zhu W, Baust M, Cheng Y, Ourselin S, Cardoso MJ, et al.: Privacy-preserving federated brain tumour segmentation. In: International Workshop on Machine Learning in Medical Imaging. pp. 133–141. Springer (2019)
15. Li X, Huang K, Yang W, Wang S, Zhang Z: On the convergence of fedavg on non-iid data. arXiv preprint arXiv:1907.02189 (2019)
16. Lin Y, Han S, Mao H, Wang Y, Dally WJ: Deep gradient compression: Reducing the communication bandwidth for distributed training. arXiv preprint arXiv:1712.01887 (2017)
17. Luxton DD, Kayl RA, Mishkind MC: mhealth data security: The need for hipaa-compliant standardization. Telemedicine and e-Health18(4), 284–288 (2012) [PubMed: 22400974]
18. McClure P, Zheng CY, Kaczmarzyk J, Rogers-Lee J, Ghosh S, Nielson D, Bandettini PA, Pereira F: Distributed weight consolidation: A brain segmentation case study. In: Advances in Neural Information Processing Systems. pp. 4093–4103 (2018) [PubMed: 34376963]
19. McMahan HB, Moore E, Ramage D, Hampson S, et al.: Communication-efficient learning of deep networks from decentralized data. arXiv preprint arXiv:1602.05629 (2016)
20. Nguyen CV, Li Y, Bui TD, Turner RE: Variational continual learning. arXiv preprint arXiv:1710.10628 (2017)
21. NVIDIA: Tensorflow determinism. https://github.com/NVIDIA/framework-determinism (2020), accessed: 2020-06-27
22. Pan SJ, Yang Q, et al.: A survey on transfer learning. IEEE Transactions onknowledge and data engineering22(10), 1345–1359 (2010)
23. MONAI. https://monai.io/ (2020), accessed: 2020-07-14
24. Remedios SW: Federated gradient averaging implementation. https://github.com/sremedios/federated_gradient_averaging (2020), accessed: 2020-06-27
25. Remedios SW, Roy S, Bermudez C, Patel MB, Butman JA, Landman BA, Pham DL: Distributed deep learning across multisite datasets for generalized CT hemorrhage segmentation. Medical physics47(1), 89–98 (2020) [PubMed: 31660621]

26. Ronneberger O, Fischer P, Brox T: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)

27. Ruder S: An overview of gradient descent optimization algorithms. arXiv preprintarXiv:1609.04747 (2016)

28. Sheller MJ, Edwards B, Reina GA, Martin J, Pati S, Kotrotsou A, Milchenko M, Xu W, Marcus D, Colen RR, et al.: Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. Scientific Reports10(1), 1–12 (2020) [PubMed: 31913322]

29. Shokri R, Shmatikov V: Privacy-preserving deep learning. In: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security. pp. 1310–1321 (2015)

30. Shokri R, Stronati M, Song C, Shmatikov V: Membership inference attacks against machine learning models. In: 2017 IEEE Symposium on Security and Privacy (SP). pp. 3–18. IEEE (2017)

31. Thompson LA, Black E, Duff WP, Black NP, Saliba H, Dawson K: Protected health information on social networking sites: ethical and legal considerations. Journal of medical Internet research13(1) (2011)

32. Wen W, Xu C, Yan F, Wu C, Wang Y, Chen Y, Li H: Terngrad: Ternary gradients to reduce communication in distributed deep learning. In: Advances in neural information processing systems. pp. 1509–1519 (2017)

33. Ye M, Abbe E: Communication-computation efficient gradient coding. arXiv preprint arXiv:1802.03475 (2018)

34. Yosinski J, Clune J, Bengio Y, Lipson H: How transferable are features in deep neural networks? In: Advances in neural information processing systems. pp. 3320–3328 (2014)

35. Yu H, Yang S, Zhu S: Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 5693–5700 (2019)

36. Zhang S, Zhang C, You Z, Zheng R, Xu B: Asynchronous stochastic gradient descent for dnn training. In: Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. pp. 6660–6663. IEEE (2013)

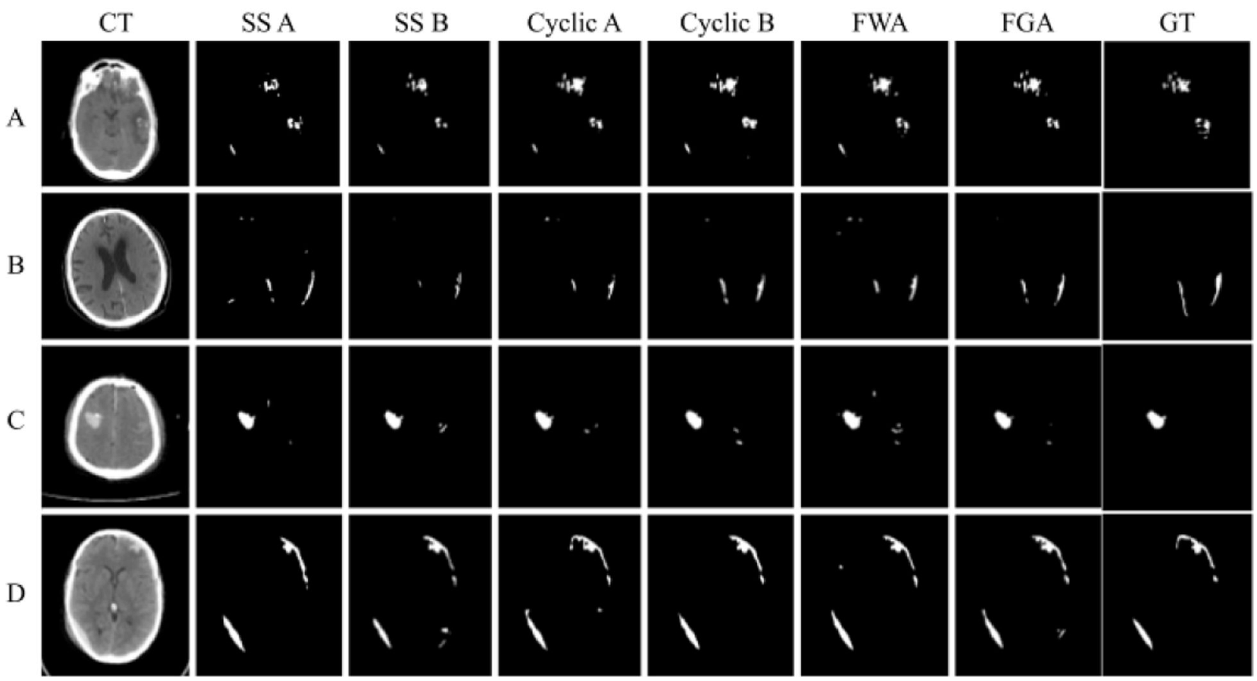

Fig.1:
Qualitative results of segmentation methods at each site.

Table 1:

MNIST experiment: balanced accuracy scores (BAS) for each training method and site for the test set. The test set consists of all 10 classes and is identical across sites and methods.

Method	**Single Site**			**Cyclic**		**FWA**	**FGA**
Train Site	*A*	*B*	*AUB*	*A*	*B*		
BAS	49.39%	48.26%	**93.74%**	79.34%	77.94%	88.37%	**93.74%**

Table 2:

Dice scores for each training method and site for test sets at each site. Statistical significance with $p < 0.005$ is determined by the paired t-test and is indicated by an asterisk, and best results are indicated by bold text.

Method		Site A	Site B	Site C	Site D	All Sites
Single Site	*A*	0.67 ± 0.06	0.69 ± 0.08	0.69 ± 0.07	0.63 ± 0.05	0.69 ± 0.08
	B	0.71 ± 0.06	0.69 ± 0.08	0.75 ± 0.07	0.64 ± 0.06	0.73 ± 0.07
Cyclic	*A*	0.72 ± 0.07	0.73 ± 0.07	0.74 ± 0.08	0.63 ± 0.06	0.73 ± 0.07
	B	0.73 ± 0.05	0.72 ± 0.07	0.73 ± 0.08	**0.66 ± 0.06**	0.72 ± 0.07
FWA		0.71 ± 0.06	0.72 ± 0.06	0.71 ± 0.08	0.63 ± 0.06	0.72 ± 0.06
FGA		**0.77 ± 0.07***	**0.76 ± 0.08***	**0.78 ± 0.07**	0.65 ± 0.07	**0.76 ± 0.08***