Robustness of Vertically Federated Deep Learning

A Resiliency Analysis of Vertically Federated Deep Machine Learning Models with Non-Adversarial Noise

Matthew Linton Davis Brkfl Merritt

A final project presented for
Cognitive Modeling (Spring 2024)

Cognitive Science & Computer Science
Rensselaer Polytechnic Institute
United States
April 29th, 2024

Robustness of Vertically Federated Deep Learning (Cognitive Modeling Spring 2024)

Matthew Merritt [1 2]

Abstract

In order to apply the many skills I learned from the Cognitive Modeling Course, I chose a project topic that intersected my semester's research on Explainability in Federated Learning and the course's takeaways. In this paper, I present a thorough review of the concepts my project is based upon: an analysis of vertically federated deep machine learning models when faced with non-adversarial noise during training. By carrying out a principled and systematic analysis of this topic to ask cognitively motivated questions, I am able to propose and answer questions such as how the robustness of a model evolves in response to training with noise, what is the measurement/metrics of predictive performance/convergence in a non-centralized system (similar to how a brain is composed of distinct sections that collaborate for functionality), and how performance is impacted as the feature space is partitioned along different numbers of client parties.

1. Introduction

Machine learning research, developments, and applications have exploded in recent years. With applications ranging from Health and medicine ((Errounda & Liu, 2022), (Kim et al., 2021), (Lee et al., 2018)), to business ((Ou et al., 2020), (Yang et al., 2019)), and image recognition (Liu et al., 2020), to name a few. However, as the reliance on these systems develops, they require more data to continually improve.

As organizations holding valuable data wish to cooperate in ongoing efforts to train models, they are limited by both legal and principled obligations to keep this data private (e.g., GPDR is a regulation in EU law on data protection and privacy in the European Union and the European Economic Area (gdp). So, the dilemma remains of contribution while retaining privacy, and that's where vertical federated learning has found prominence.

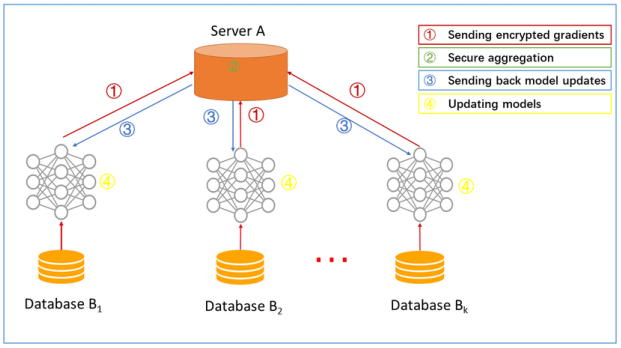

Figure 1. Architecture for a horizontal federated learning system

1.1. Vertical Federated Learning

1.1.1. Federated Learning

Federated learning was first introduced in 2016 by Google in an effort to take advantage of "the wealth of information of data suitable for learning models, which in turn can greatly improve the user experience" while respecting that the data is "often privacy sensitive, large in quantity, or both, which may preclude logging to the data center and training there using conventional approaches." (McMahan et al.). The proposed approach trains a series of deep machine-learning models locally on each device for the same task, with each then sharing their gradients with a global server. The global server aggregates these gradients (in the proposed approach by averaging) and then broadcasts updates to the client model, to which the client models update as seen in (fig 1).(Yang et al., 2019) The client models in this scenario have the same architecture and are trained over their local instances (e.g. identifying common spelling mistakes for spelling suggestion applications).

1.1.2. Vertical Federated Learning

This form of federated learning noted above is now termed horizontal federated learning, as the client models have overlapping feature spaces, but they differ in the sample space. Vertical federated learning, however, tackles the problem of client models possibly having differing feature spaces and an overlapping sample space - see (fig 2) (Hu

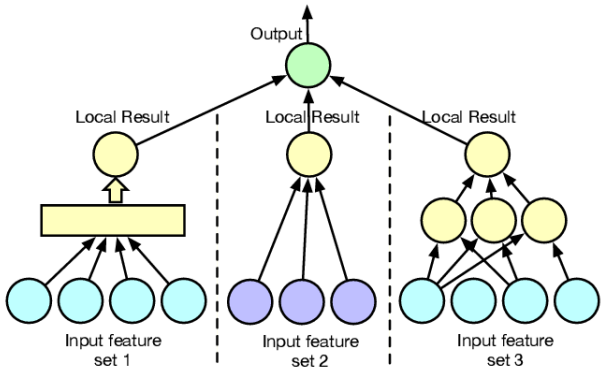

Figure 2. Architecture for a vertically federated learning system

et al.).

A common approach for vertical federated learning is to have each model transform their sample space of an instance into a latent space(an embedding) and then send the embedding to a global model. The global model aggregates the embeddings and generates an inference, which is then compared to the label, for instance, and the loss function (often Binary-Cross-Entropy for classification and Mean Squared Error for regression) is used to calculate the gradients for the server model and client models from which updates to the respective models take place.

Research on Federated Learning has blossomed since its formal introduction in the 2016 paper (McMahan et al.). Although extensive research has been conducted on the adversarial robustness (defined in the Robustness section) of vertically federated, there exists a research gap in non-adversarial data augmentation for vertically federated deep learning models, which this paper focuses on.

1.2. Deep Learning

Deep learning, specifically deep neural networks (DNN), is a form of machine learning that aims to internally form patterns for understanding to generate predictions (often termed as inferences). DNNs are able to generate impressively accurate predictions and have found major relevance within industry and research (Deng, 2014). To formally define a deep neural network for our purposes, we can borrow the definition from (Drenkow et al., 2021); "In the most general form, we consider a deep neural network as a composition of functions or computational layers which map from input space in the image(sample) domain to a prediction."

There are fascinating similarities between how the human brain works and how neural networks operate (Schyns et al.). This similarity allows us to ask meaningful, cognitively motivated questions such as:

1. How does the network's behavior/performance change as a function of an exogenous factor?
   (a) Changes to the number of layers in the client models.
   (b) Changes to the number of vertical data partitions (e.g., three vertical data partitions → three client models).
   (c) changes to the amount of noise applied to the training and/or testing data.
2. How can we modify the training algorithm to increase the network's robustness?

1.3. Robustness

Robustness and resilience regarding neural networks is a very overloaded term as found in a recent survey of the field (Drenkow et al., 2021). The leading definition(s) are often in relation to a model's ability to generalize/make correct predictions in the face of data that is new/unseen. There are commonly two types of sub-classification types used to further specify the type of robustness: adversarial and non-adversarial.

1.3.1. Adversarial

Frequently associated with ill-intent, "Adversarial changes to the original data are usually undetectable to the human eye but are disruptive enough to cause AI models to misclassify samples." (Ghaffari Laleh et al.). Adversarial robustness is the ability to retain accuracy over instances that have been adversarially changed.

1.3.2. Non-Adversarial

Non-Adversarial robustness often refers to the ability to retain accuracy over instances that may have corruptions/perturbations that may come "naturally" or low probability of occurrence instances. Examples include blurs in images for image recognition tasks, background noise in recordings used in speech recognition tasks, etc (Hendrycks & Dietterich) (Drenkow et al., 2021).

There is a surprising "disproportionate focus on adversarial research relative to non-adversarial conditions" (Drenkow et al., 2021) when it comes to deep learning, and even more so for vertically federated deep learning. Therefore prompting an interesting direction of research that this paper looks into.

2. Methods

2.1. Data

The dataset used for this analysis was the Adult Dataset (Becker & Kohavi, 1996). This dataset (also known as "Census Income" dataset) is commonly used for the classification

task to predict whether income exceeds $50K/yr.

To analyze potential label biases that may arise from the data, I confirmed that although there is not an even label distribution (as shown in the figure above), the distribution is not unbalanced enough to distract from the cognitive questions at hand (though an uneven label distribution may pose more interesting cognitive questions).

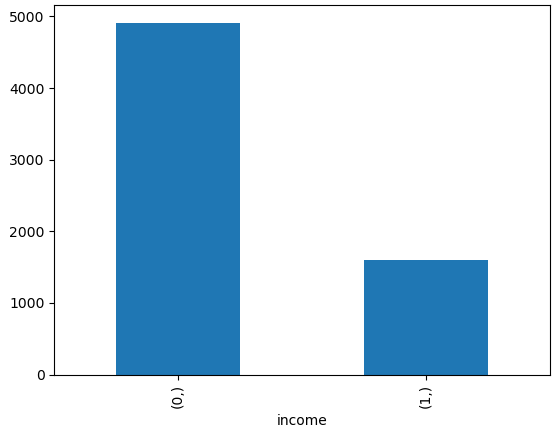

Figure 3. Label distribution for the adult dataset.
∗The y-axis represents the number of instances per dependent label

There were 32564 instances after the data-augmentation stage which removed rows with special/missing values.

2.1.1. Data Augmentation

To provide a strong analysis of this space, parsing the original dataset is a must. To provide data that would be easy to work with, I created a custom parser that would provide the following capabilities:

1. control over how many parties to vertically partition the data
2. normalization of the data
3. control of the % of data should be used for testing (and inversely training $(1 - \%)$
   (a) A test/train split of $\%20$, resulting in 6515 testing instances and 26049 training instances.
4. ability to set special values/missing values to NaN
5. ability to remove rows with special values (overrides the previous option)
6. selection of the output directory

By doing so, I gained the ability to quickly and cleanly create training and testing datasets vertically partitioned, which I could immediately use for experimenting.

2.1.2. Noise

To add non-adversarial noise, I used Pytorch's torch.rand_like function in the following way (tor):

```
noisy_tensor = concated_tensor +
↪  noise_scale *
↪  torch.randn_like(concated_tensor)
```

2.2. The Models

2.2.1. Parameters and Architecture

For a thorough analysis, the following key model parameters became of high interest:

1. Number of clients
2. Number of layers
3. Noise scales
   (a) This would be the amount of noise applied to the training and testing datasets for each client party as described above.
   (b) Whether a model trained on a noisy dataset or not
4. The Learning rate
   (a) Of the client models
   (b) Of the fusion model
   (c) Provided an interesting variable for speculation of prior sensitivity analysis

The design of the models was based on several assumptions and was parameterized in a way that the key model parameters noted above could be easily augmented for rapid experimentation. The key assumptions of our models were:

1. **No Multicollinearity** within the data
   (a) Though L2 (Ridge) normalization is often used to combat the side effects of multicollinearity between features, this was omitted due to this assumption of the independent variables.
2. **Normality** regarding the dataset's independent and dependent variables
3. **Exchangeability** of instances during the training phase regarding inference results (as long as the appropriate instance information is fed to the model)
4. **Consistency** for inference generation, post model training

5. **Positivity** as this is required for any meaningful causal inference

6. **No Linearity** in order to capture the assumed complex relationship of the data and as non-linear activation functions are essential in deep learning or else the models become tractable to a perception.

2.2.2. Metrics and Methods

To evaluate predictive performance, I used the AUPRC (Area under the precision-recall graph Curve) metric. As the task at hand is binary classification and there is a class imbalance, the following literature suggests using AUPRC compared to AUROC (Area under the receiver operating characteristic curve) ((McDermott et al., 2024), (Hancock et al., 2023)) and the Binary Cross Entropy loss function provided by Pytorch (noa, a).

2.3. Software and Libraries

Programmed in python3, the included libraries are detailed in the [Libraries used for the project] listing/figure.

```
1
2  # Libraries for Data Handling
3  import numpy as np
4  import pandas as pd
5  from torch.utils.data import DataLoader,
   ↪  TensorDataset
6  from pathlib import Path
7
8  # Libraries for Algorithms
9  import torch
10 import torch.nn as nn
11 import torch.optim as optim
12 from sklearn.metrics import
   ↪  average_precision_score, accuracy_score
13
14 # Libraries for Data Visulation Tools
15 import matplotlib.pyplot as plt
16 import seaborn as sns
17 from sklearn.metrics import
   ↪  confusion_matrix
```

Listing 1: Libraries used for the project.

3. Results

3.1. Diagnostics

The majority of the models run, seemed to reach convergence (excluding those trained with high amounts of noise). Convergence was assessed via training/validation loss trajectories and occasional diagnostic confusion matrices over the test set. As for each epoch, the loss over the epoch's training and evaluation dataset was calculated and printed to the command line along with the respective AUPRC. It should be noted that several epochs were required to reach convergence, as shown by the difference in max AURPC reached overtime for the 3-way partitioned approach in the following figure. This amount of epoches required until the model reached convergence seemed to increase to around 8 epochs as noise increased until the noise factor increased beyond 1.0, leading to less amounts of rounds required until convergence.

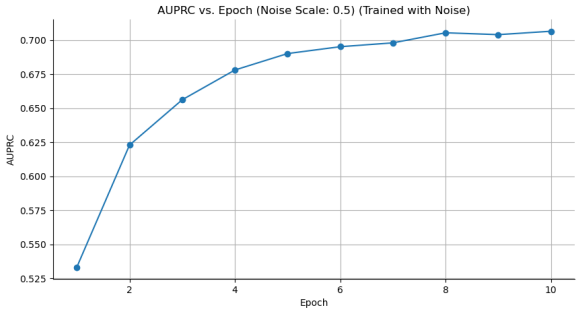

Figure 5. 3-way partitioned vfl model trained with a 0.5 noise factor

3.2. Interpretation, Validation, and Sensitivity

Experiments where models were trained with their training dataset augmented by the noise scale vs those that trained with the original dataset were shown to have marginal increases with small amounts of noise and significant decreases with large amounts of noise, as shown in the following figures.

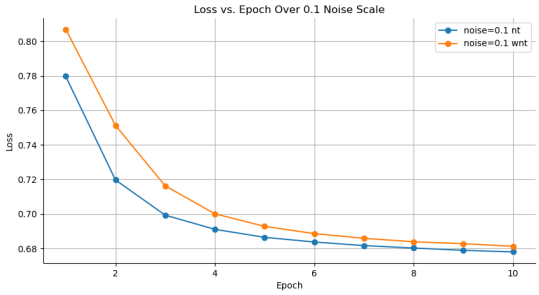

Figure 6. 3-way partitioned vfl model trained with a 0.1 noise factor
nt = trained model with noise
wnt = did not train model with noise

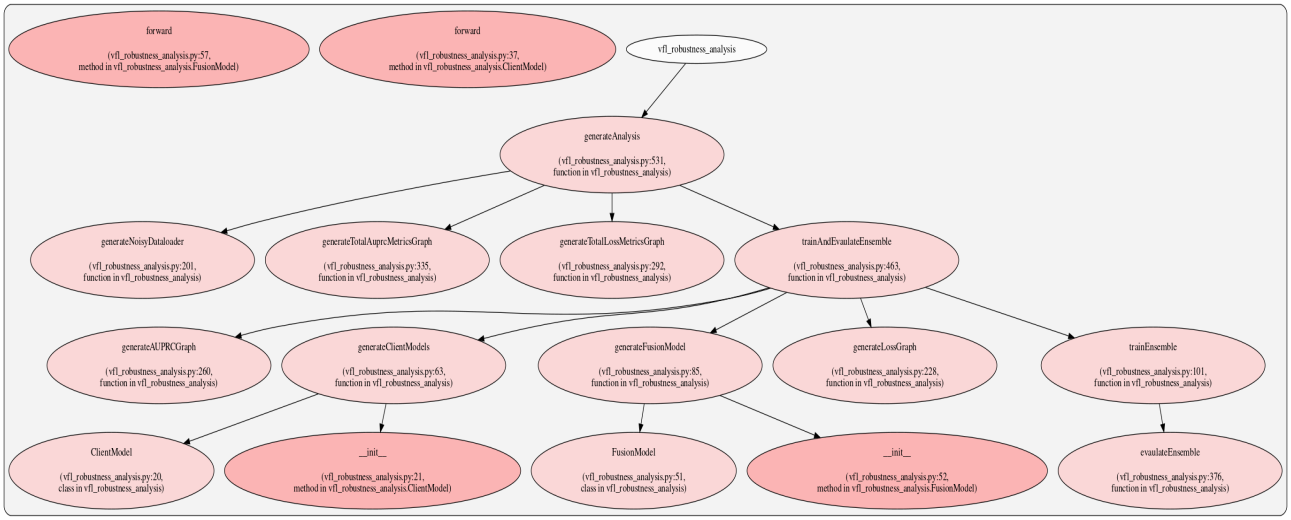

Figure 4. Design diagram

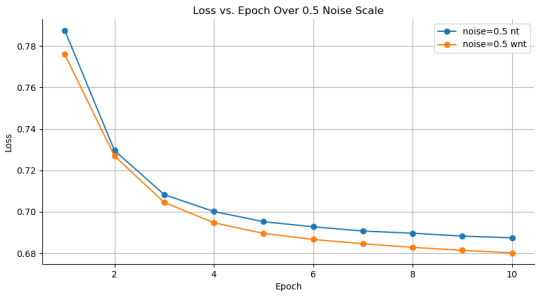

Figure 7. 3-way partitioned vfl model trained with a 0.5 noise factor
nt = trained model with noise
wnt = did not train model with noise

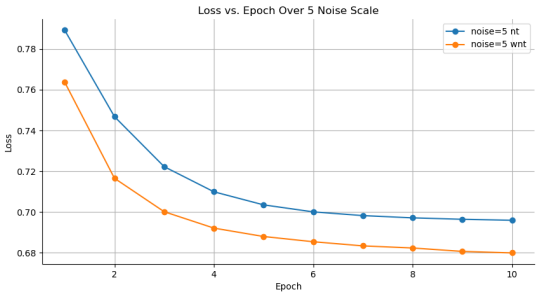

Figure 8. 3-way partitioned vfl model trained with a 5.0 noise factor
nt = trained model with noise
wnt = did not train model with noise

I was able to answer each question proposed in the abstract by modifying my experimental settings. To visit the questions one at a time; **1. How the robustness of a model evolves in response to training with noise**

As noted above, the VFL model's robustness improves marginally when trained with a low amount of noise(noise factor 0.1), but significantly decreases when the noise factor grows (a noise factor $>$ 0.1). This is supported by the graphs present below (figs 9−14)

2. What is the measurement/metrics of predictive performance/convergence in a non-centralized system I conducted research and answered this in section 2.2.2.

3. How performance is impacted as the feature space is partitioned along different numbers of client parties

Surprisingly, there is no noticeable difference in the performance of the models when the data is federated among different amounts of clients; the resulting AUPRC scores are approximately the same.

3.2.1. Predictive Performance

Utilizing the metrics defined in section 2.2.2, observing the following graphs, and using the ((McDermott et al., 2024), (Hancock et al., 2023)), an AUPRC would be considered near standard for a deep learning architecture (Becker & Kohavi, 1996). Training with no noise or small amounts of noise seems to achieve a high AUPRC as can be seen in the following graphs.

4. Discussion and Conclusion

The analysis conduction revealed several interesting conclusions:

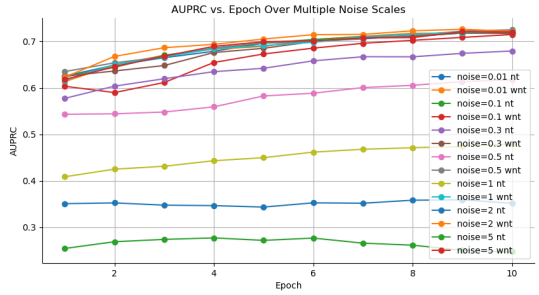

Figure 9. 3-way partitioned vfl model trained with multiple noise factors
nt = trained model with noise
wnt = did not train model with noise

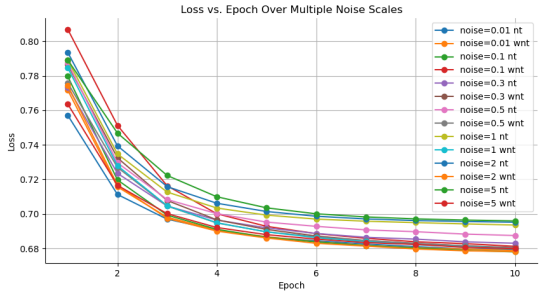

Figure 12. 3-way partitioned vfl model trained with a multiple noise factors
nt = trained model with noise
wnt = did not train model with noise

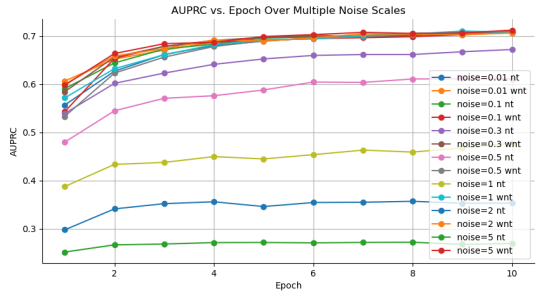

Figure 10. 7-way partitioned vfl model trained with a multiple noise factors
nt = trained model with noise
wnt = did not train model with noise

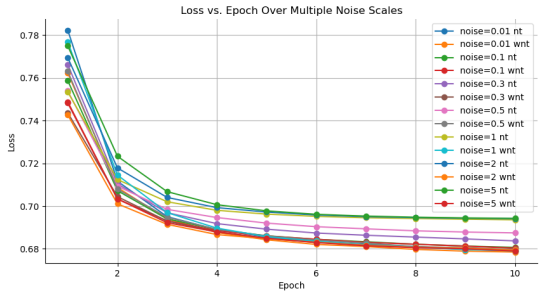

Figure 13. 7-way partitioned vfl model trained with a multiple noise factor
nt = trained model with noise
wnt = did not train model with noise

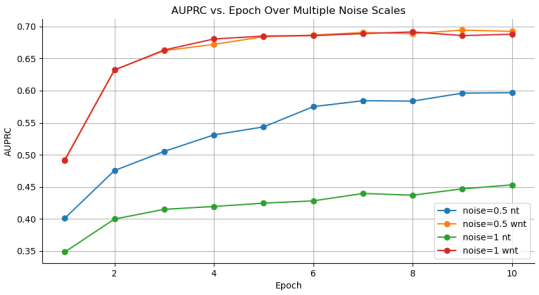

Figure 11. 10-way partitioned vfl model trained with a multiple noise factors
nt = trained model with noise
wnt = did not train model with noise

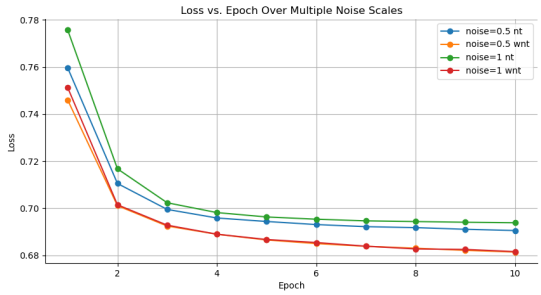

Figure 14. 10-way partitioned vfl model trained with a multiple noise factor
nt = trained model with noise
wnt = did not train model with noise

1. Small amounts of non-adversarial noise added to a training set increase performance on deep learning vertically federated model predictive accuracy and robustness.

2. The number of client models when the independent variables are vertically partitioned among them (without overlap) does not seem to affect performance or robustness.

In the course of research, experimentation, and analysis of this project, I have learned a lot and continually found new directions from which this research could expand, such as a thorough analysis of the effect of various learning rates, layer depths, and different types of noises that can be applied. I learned a lot relating to the robustness of deep neural networks, robustness, and conducting research where I am in control of all computational steps. Mordantly, the biggest challenge is not being able to do further research - and as for the future outlooks, I hope to rectify this by delving further into research in this area. This research is but a first step, and hopefully, by using the documented, parameterized, and public code I developed here (pro), others and I can further investigate these directions and more.

References

Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation) (text with eea relevance). URL https://gdpr.eu/.

BCELoss — PyTorch 2.3 documentation, a. URL https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.html.

What is deep learning? | IBM, b. URL https://www.ibm.com/topics/deep-learning.

Matthew linton davis brkfl merritt, cognitive modeling spring 2023 project. URL https://github.com/MattLMerritt/Cognative-Modeling-2024/tree/main.

torch.rand_like — PyTorch 2.3 documentation. URL https://pytorch.org/docs/stable/generated/torch.rand_like.html.

Becker, B. and Kohavi, R. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

Deng, L. A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing*, 3:e2, 2014. doi: 10.1017/atsip.2013.9.

Drenkow, N. G., Sani, N., Shpitser, I., and Unberath, M. A systematic review of robustness in deep learning for computer vision: Mind the gap? 2021. URL https://api.semanticscholar.org/CorpusID:244773637.

Errounda, F. Z. and Liu, Y. A mobility forecasting framework with vertical federated learning. *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 301–310, 2022. URL https://api.semanticscholar.org/CorpusID:251473620.

Freiesleben, T. Artificial neural nets and the representation of human concepts. doi: 10.48550/ARXIV.2312.05337. URL https://arxiv.org/abs/2312.05337. Publisher: [object Object] Version Number: 2.

Freiesleben, T. and Grote, T. Beyond generalization: a theory of robustness in machine learning. 202(4):109. ISSN 1573-0964. doi: 10.1007/s11229-023-04334-9. URL https://link.springer.com/10.1007/s11229-023-04334-9.

Ghaffari Laleh, N., Truhn, D., Veldhuizen, G. P., Han, T., van Treeck, M., Buelow, R. D., Langer, R., Dislich, B., Boor, P., Schulz, V., and Kather, J. N. Adversarial attacks and adversarial robustness in computational pathology. 13(1):5711. ISSN 2041-1723. doi: 10.1038/s41467-022-33266-0. URL https://www.nature.com/articles/s41467-022-33266-0. Publisher: Nature Publishing Group.

Hancock, J. T., Khoshgoftaar, T. M., and Johnson, J. M. Evaluating classifier performance with highly imbalanced big data. *Journal of Big Data*, 10(1): 42, Apr 2023. ISSN 2196-1115. doi: 10.1186/s40537-023-00724-5. URL https://doi.org/10.1186/s40537-023-00724-5.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. URL http://arxiv.org/abs/1903.12261.

Hu, Y., Niu, D., Yang, J., and Zhou, S. FDML: A collaborative machine learning framework for distributed features. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pp. 2232–2240. Association for Computing Machinery. ISBN 978-1-4503-6201-6. doi: 10.1145/3292500.3330765. URL https://dl.acm.org/doi/10.1145/3292500.3330765.

Kim, J., Li, W., Bath, T., Jiang, X., and Ohno-Machado, L. VERTIcal grid logistic regression with confidence intervals (VERTIGO-CI). *AMIA Jt Summits Transl Sci Proc*, 2021:355–364, May 2021.

Kleppe, A., Albregtsen, F., Vlatkovic, L., Pradhan, M., Nielsen, B., Hveem, T. S., Askautrud, H. A., Kristensen, G. B., Nesbakken, A., Trovik, J., Wæhre, H., Tomlinson, I., Shepherd, N. A., Novelli, M., Kerr, D. J., and Danielsen, H. E. Chromatin organisation and cancer prognosis: a pan-cancer study. 19 (3):356–369. ISSN 1470-2045, 1474-5488. doi: 10.1016/S1470-2045(17)30899-9. URL https://www.thelancet.com/journals/lanonc/article/PIIS1470-20451730899-9/fulltext. Publisher: Elsevier.

Lee, J., Sun, J., Wang, F., Wang, S., Jun, C.-H., and Jiang, X. Privacy-Preserving patient similarity learning in a federated environment: Development and analysis. *JMIR Med Inform*, 6(2):e20, April 2018.

Lee, J.-G., Roh, Y., Song, H., and Whang, S. E. Machine learning robustness, fairness, and their convergence. pp. 4046–4047. doi: 10.1145/3447548.3470799. URL https://dl.acm.org/doi/10.1145/3447548.3470799. Conference Name: KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining ISBN: 9781450383325 Place: Virtual Event Singapore Publisher: ACM.

Liu, F., Wu, X., Ge, S., Fan, W., and Zou, Y. Federated learning for vision-and-language grounding problems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):11572–11579, Apr. 2020. doi: 10.1609/aaai.v34i07.6824. URL https://ojs.aaai.org/index.php/AAAI/article/view/6824.

Liu, J., Xie, C., Kenthapadi, K., Koyejo, O., and Li, B. RVFR: Robust vertical federated learning via feature subspace recovery.

Lyu, L., Yu, H., Ma, X., Chen, C., Sun, L., Zhao, J., Yang, Q., and Yu, P. S. Privacy and robustness in federated learning: Attacks and defenses. pp. 1–21. ISSN 2162-237X, 2162-2388. doi: 10.1109/TNNLS.2022.3216981. URL https://ieeexplore.ieee.org/document/9945997/.

McDermott, M. B. A., Hansen, L. H., Zhang, H., Angelotti, G., and Gallifant, J. A closer look at auroc and auprc under class imbalance, 2024.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y. Communication-efficient learning of deep networks from decentralized data. URL http://arxiv.org/abs/1602.05629.

Niazi, M. K. K., Parwani, A. V., and Gurcan, M. N. Digital pathology and artificial intelligence. 20 (5):e253–e261. ISSN 1470-2045, 1474-5488. doi: 10.1016/S1470-2045(19)30154-8. URL https://www.thelancet.com/journals/lanonc/article/PIIS1470-2045(19)30154-8/fulltext. Publisher: Elsevier.

Ou, W., Zeng, J., Guo, Z., Yan, W., Liu, D., and Fuentes, S. A homomorphic-encryption-based vertical federated learning scheme for rick management. *Computer Science and Information Systems*, 17:819–834, 01 2020. doi: 10.2298/CSIS190923022O.

Schyns, P. G., Snoek, L., and Daube, C. Degrees of algorithmic equivalence between the brain and its DNN models. 26(12):1090–1102. ISSN 1364-6613. doi: 10.1016/j.tics.2022.09.003. URL https://www.sciencedirect.com/science/article/pii/S1364661322002200.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013. URL https://api.semanticscholar.org/CorpusID:604334.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. *arXiv: Machine Learning*, 2018. URL https://api.semanticscholar.org/CorpusID:52962648.

Yang, L., Chai, D., Zhang, J., Jin, Y., Wang, L., Liu, H., Tian, H., Xu, Q., and Chen, K. A survey on vertical federated learning: From a layered perspective, a. URL http://arxiv.org/abs/2304.01829.

Yang, Q., Liu, Y., Cheng, Y., Kang, Y., Chen, T., and Yu, H. Horizontal federated learning. In Yang, Q., Liu, Y., Cheng, Y., Kang, Y., Chen, T., and Yu, H. (eds.), *Federated Learning*, pp. 49–67. Springer International Publishing, b. ISBN 978-3-031-01585-4. doi: 10.1007/978-3-031-01585-4_4. URL https://doi.org/10.1007/978-3-031-01585-4_4.

Yang, Q., Liu, Y., Chen, T., and Tong, Y. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.*, 10(2), jan 2019. ISSN 2157-6904. doi: 10.1145/3298981. URL https://doi.org/10.1145/3298981.

Yang, Y.-Y., Rashtchian, C., Zhang, H., Salakhutdinov, R., and Chaudhuri, K. A closer look at accuracy vs. robustness. *arXiv: Learning*, 2020. URL https://api.semanticscholar.org/CorpusID:220496204.

Yao, H., Wang, J., Dai, P., Bo, L., and Chen, Y. An efficient and robust system for vertically federated random forest. URL http://arxiv.org/abs/2201.10761.