The Effects of Nonlinear Signal on Expression-Based Prediction Performance

This manuscript (permalink) was automatically generated from greenelab/linear_models_manuscript@8b5bb49 on March 15, 2022.

Authors

- **Benjamin Heil**
  0000-0002-2811-1031 · ben-heil · autobencoder
  Genomics and Computational Biology Graduate Group, Perelman School of Medicine, University of Pennsylvania · Funded by Grant XXXXXXXX

Abstract

Introduction

Transcriptomic data contains a wealth of information about a person's biology, so predicting phenotypes from RNA-seq data is a promising field of research. Gene expression-based models are already being used to subtype cancer [1], predict transplant rejections [2], and uncover biases in public data [3]. In fact, both the capability of machine learning models [**arxiv:2202.05924?**] and the amount of transcriptomic data available [4,5] are increasing rapidly. It makes sense, then, that neural networks are frequently being used in the transcriptomic prediction space [6,7,8].

However, there are two conflicting ideas in the literature regarding the utility of nonlinear models. One theory is based on the prior biological understanding: the paths linking gene expression to phenotypes are complex [9,10], and nonlinear models like neural networks should be more capable of learning that complexity. Unlike purely linear models such as logistic regression, nonlinear models should be learn more sophisticated representations of the relationships between expression and phenotype. Accordingly, many have used nonlinear models to learn representations useful for making predictions of phenotypes from gene expression [11,12,13].

The other theory disagrees with the first hypothesis: when using expression to make predictions about phenotypes, linear models seem to do as well as or better than nonlinear ones in many cases[14]. While papers of this sort are harder to come by — scientists don't tend to write papers about how their deep learning model was worse than logistic regression — other complex biological problems have also seen linear models prove equivalent to nonlinear ones [15].

In this paper we demonstrate that both theories have merit. There is nonlinear signal relating the phenotypes to genotypes, but it doesn't always lead nonlinear models to provide better prediction accuracy. We construct a system of binary and multi-class classification problems on the Recount3 compendium[16] that allows us to show that linear and nonlinear models have similar accuracy on several (but not all) prediction tasks. We then remove the linear signals relating the phenotype to gene expression and show that there is in fact nonlinear signal in the data even when the linear models outperform the nonlinear ones. Finally, we validate the results by testing the same problems on a dataset from GTEx[17], running controls on simulated data, and examining different problem formulations such as samplewise splitting and pretraining.

In reconciling these two obstensibly conflicting theories, we assist future scientists by showing the importance of trying a linear baseline model before developing a complex nonlinear approach. While nonlinear models may outperform simpler models at the limit of infinite data, they don't necessarily do so even when trained on the largest datasets publicly available today.

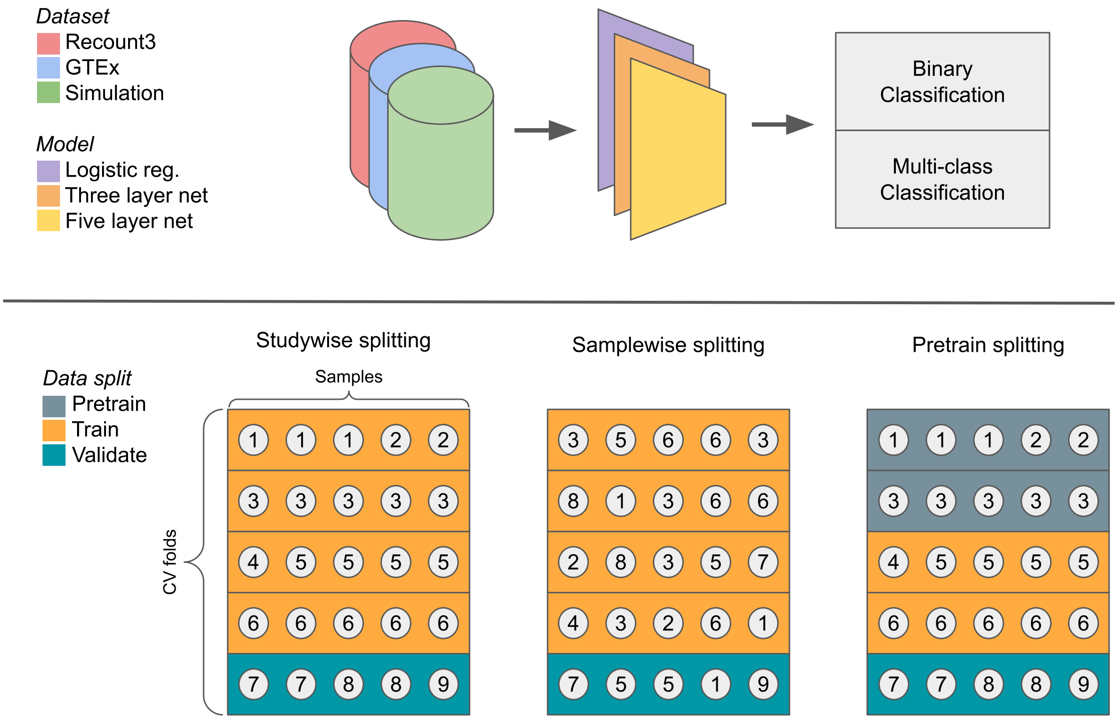
Results

Approach

We compared the performance of linear and nonlinear models across a number of datasets and on multiple tasks (fig. 1 top). Our datasets consisted of gene expression from Recount3 [16] with tissue labels from the recount3 metadata and sex labels from Flynn et al. [18], simulated data, and expression and tissue labels from GTEx[17]. Before use in model training, we removed scRNA samples, RPM normalized, and zero-one standardized the data (see Methods for more details).

We split our dataset via fivefold cross-validation to evaluate each model on multiple training and validation sets. In order to avoid leakage between folds, the studies were placed in their entirety into a fold instead of being split across folds (fig. 1 bottom). We then ran the models on increasingly large subsets of the training data to determine how model performance is affected by the amount of training data.

To ensure that artifacts specific to a single data split or model initialization don't drive the signal, we run each of our experiments with three different random seeds. As a result of these different dimensions of variation, each evaluation we perform reflects 150 trained instances of each model. The three models we selected were logistic regression, a three layer neural network, and a five layer neural network. Our three layer and five layer networks were chosen to be representative of a fairly shallow and moderately deep network respectively. Our logistic regression implementation was implemented and optimized as similarly to the neural nets as possible to allow comparisons unbiased by implementation details. For a comparison against a scikit-learn [19] implementation of logistic regression, see Supplementary Results section TODO.



Studywise splitting

Train	1	1	1	2	2
Train	3	3	3	3	3
Train	4	5	5	5	5
Train	6	6	6	6	6
Validate	7	7	8	8	9

Samplewise splitting

Train	3	5	6	6	3
Train	8	1	3	6	6
Train	2	8	3	5	7
Train	4	3	2	6	1
Validate	7	5	5	1	9

Pretrain splitting

Pretrain	1	1	1	2	2
Pretrain	3	3	3	3	3
Train	4	5	5	5	5
Train	6	6	6	6	6
Validate	7	7	8	8	9

Figure 1: Schematic of the model analysis workflow. We evaluate three models on multiple classification problems in three datasets (top). To do so, we use studywise splitting as a default, but also evaluate the effects of samplewise splitting and using a pretraining dataset.

Linear and nonlinear models have similar performance in some tasks

To determine whether linear and nonlinear models performed similarly, we used them for a number of tissue prediction tasks. First we compared their abilities to differentiate between pairs of tissues (Fig. 2), and found that their performance was roughly equivalent. More specifically, given tissue pairs seemed to have maximum accuracy thresholds that models would achieve and then plateau.

In case the accuracy threshold effects were due to the relative easiness of the binary classification task, we selected a harder problem. Namely, we evaluated the models on their ability to predict which of the 21 most common tissues in the Recount3 dataset a sample belonged to. In this setting, we found that our five layer network and logistic regression performed roughly the same, while the three layer network had lower accuracy (fig. 3).

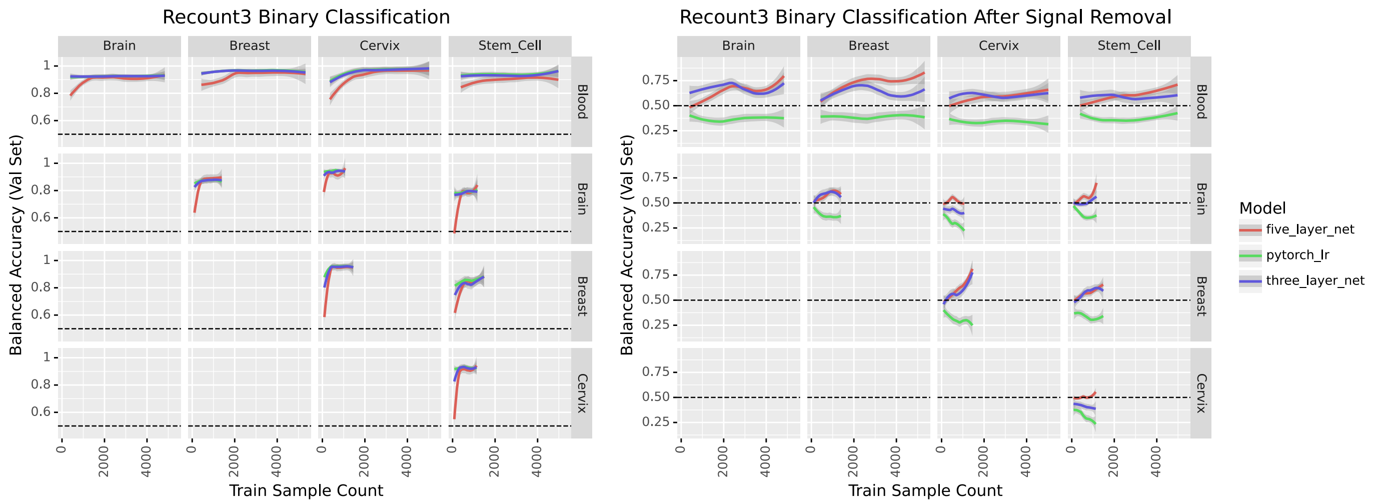


Figure 2: Comparison of models' binary classification performance before and after removing linear signal

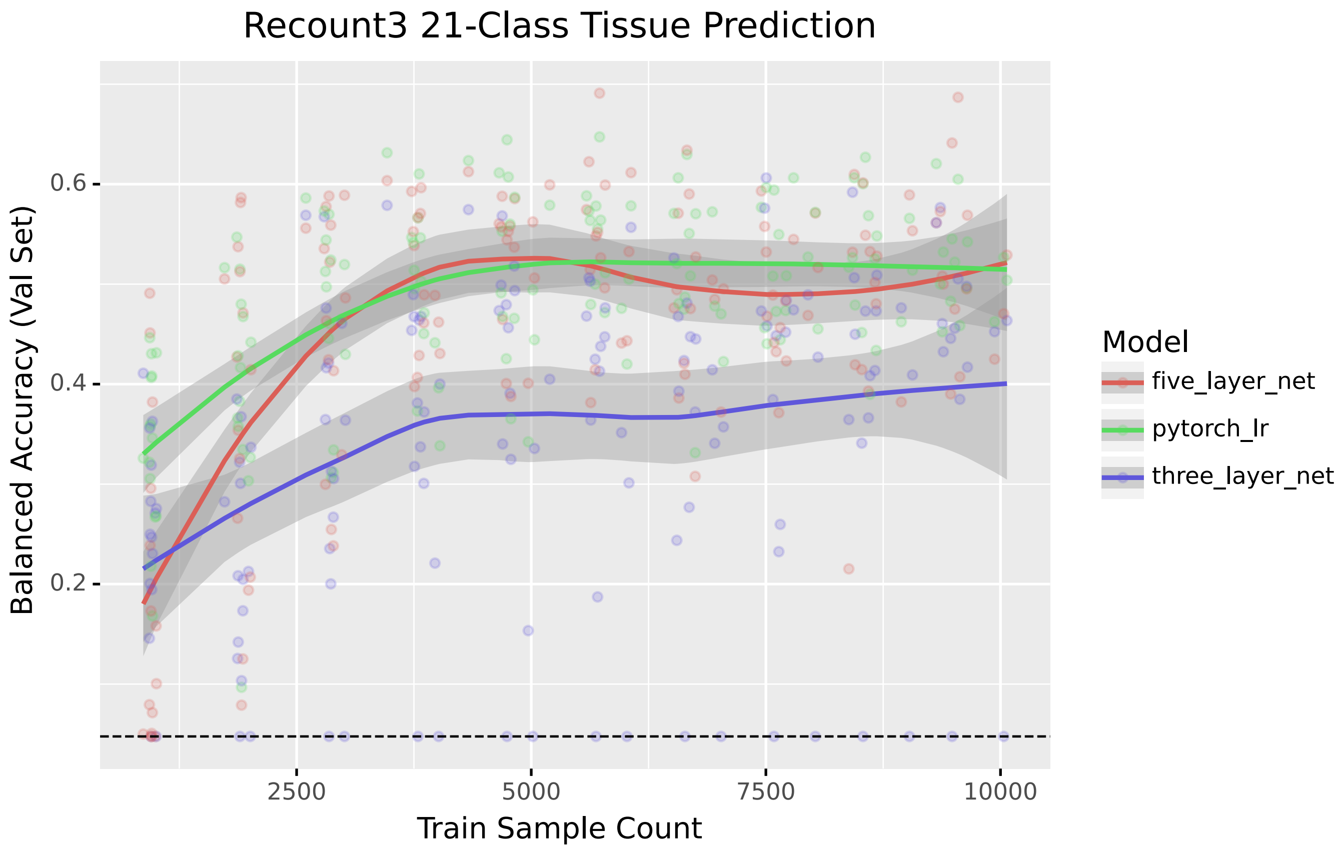


Figure 3: Graph of the Recount3 multiclass classification results. Each point represents the validation set balanced accuracy of a separate trained model. The color of the points and the trend lines shows their corresponding model class, while the dashed line represents the baseline accuracy of random predictions.

In spite of the similar performance of the models, there is nonlinear signal relevant to the problem

One can imagine a world where all the signal relevant to tissue prediction is linear. If that were the case, nonlinear models like neural networks would fail to give any advantage in a prediction problem. To determine whether there is nonlinear signal in our tissue prediction tasks learnable by our neural nets, we used Limma[20] to remove the linear signal associated with each tissue.

When we ran our models on the signal-removed data, we found that while the neural networks manage to perform better than the random classification baseline, the logistic regression models do worse than the baseline (Fig. 2). The anticorrelation between the amount of data used and the linear model performance is due to running the signal removal on the full dataset at once. Because there can be no predictive linear signal in the dataset, any linear model trained to greater than random performance on the training set will necessarily perform worse than random on the rest of the data. This artifact was selected as the lesser of two evils, as removing signal from the training and validation set poses its own problems (Supp results TODO).

This nonlinear signal is real, and not an artifact of the signal removal method

We then simulated simple binary classification tasks to ensure that the results weren't due to an unknown signal in the data. Our initial simulated dataset consisted of two types of features: half of the features had a linear dividing line between the simulated classes while the other half had a nonlinear dividing line. After training to classify the simulated dataset, all models were able to effectively predict the simulated classes. After removing the linear signal from the dataset, nonlinear models were still able to easily predict the correct classes, but logistic regression was no better than random (fig 4 middle).

To ensure that the high performance of the nonlinear models wasn't due to nonlinear signal induced by the correction method, we generated another simulated dataset consiting solely of features with a linear dividing line between the classes. As before, all models were able to predict the different classes well. However, once the linear signal was removed all models had accuracy no better than random guessing, indicating that the signal removal method was not generating nonlinear signal (fig 4 left).

We also trained the models on a dataset where all features were gaussian noise as a negative control. As expected, the models all performed at baseline accuracy both before and after the signal removal process (fig. 4 right).

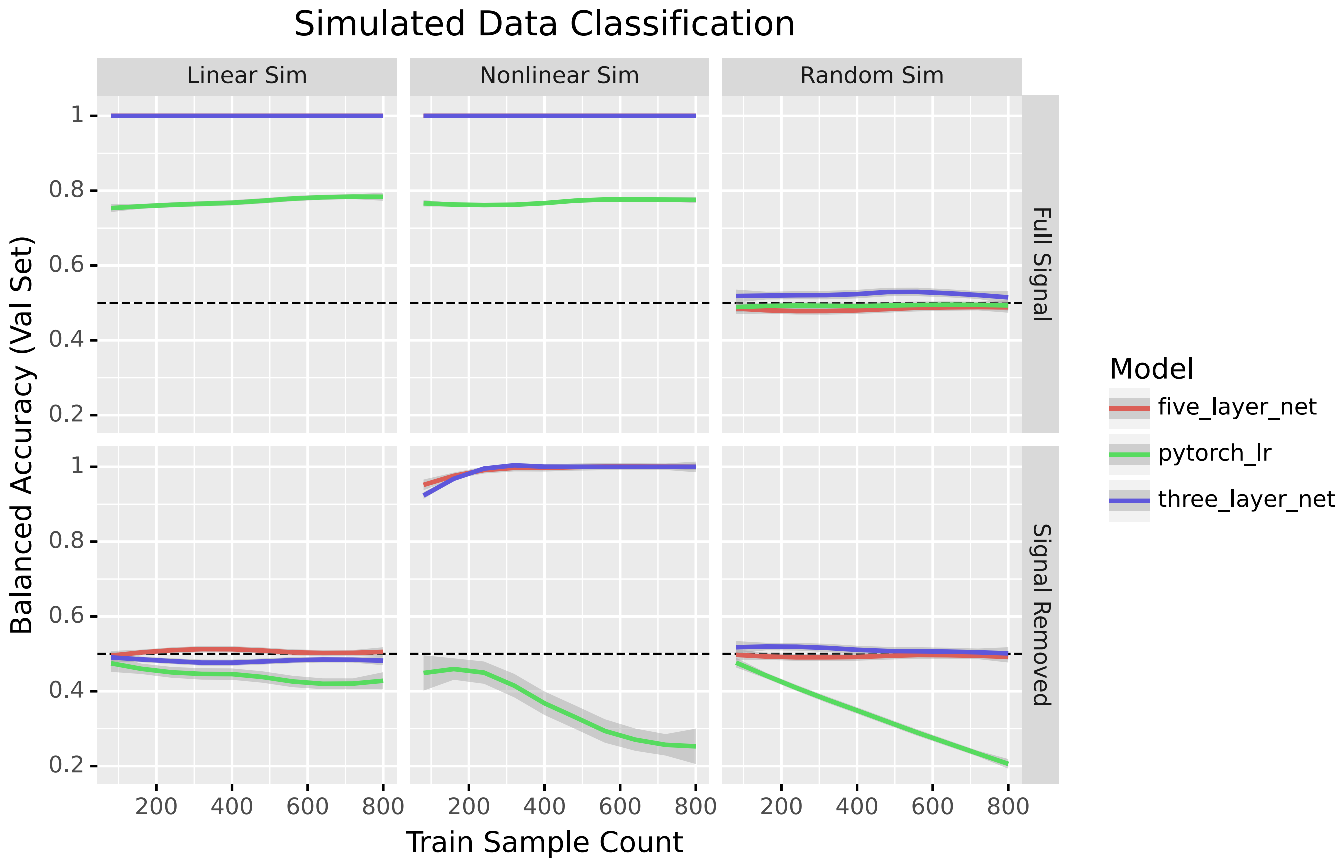


Figure 4: Performance of models in binary classification of simulated data before and after signal removal

Our results hold in a cleaner validation dataset

To validate our findings on a separate real dataset, we selected the expression data from GTEx [17]. Because it was generated by fewer labs with more consistent experimental design across samples, it is a less heterogeneous dataset than the Recount3 compendium. We trained our models to do binary classification on pairs of the five most common tissues in the dataset, then performed multiclass classification on all 31 tissues present in the dataset. Likely due to the cleaner nature of the GTEx data, all models were able to perform perfectly on the binary classification tasks (Fig. 5 bottom) The harder multitask classification problem showed logistic regression outperforming the five layer neural network, which in turn outperformed the three layer net (fig. 5 top).

The linear signal removal results on the binary classification problems were consistent with those from the Recount3 compendium. The neural networks performed less well in the low-data regime, indicating an increase in the difficulty of the problem, and the logistic regression implementation performed no better than random (Fig. 5 bottom). Similarly, the multiclass problem had the logistic regression model performing poorly, while the nonlinear models had performance that increased with an increase in data while remaining worse than before the linear signal was removed (Fig. 5 top).

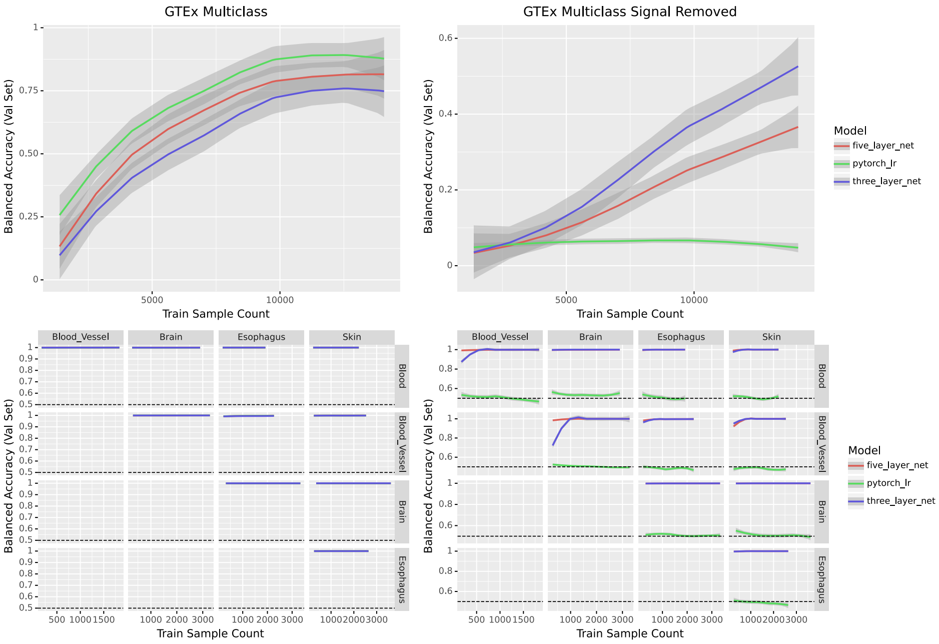


Figure 5: Performance of model on GTEx classification problems. The top figures show the difference in training models in the multiclass setting with and without signal removal, while the bottom figures show binary classification with and without signal removal.

While linear models seem to be equivalent to nonlinear ones in most problems, that isn't true for all problems

To rule out the possibility that our findings were specific to tissue prediction tasks, we examined models' ability to predict metadata-derived sex (Fig. 6). We used the same experimental setup as in our other Recount3 binary prediction tasks to train the models, but rather than using tissue labels we used metadata-derived sex labels. In this setting we found that, at least in the 5000-15000 sample range, the nonlinear models outperformed logistic regression. This result demonstrates that despite the compelling accuracy of linear models, there are still problem settings where nonlinear models perform better.

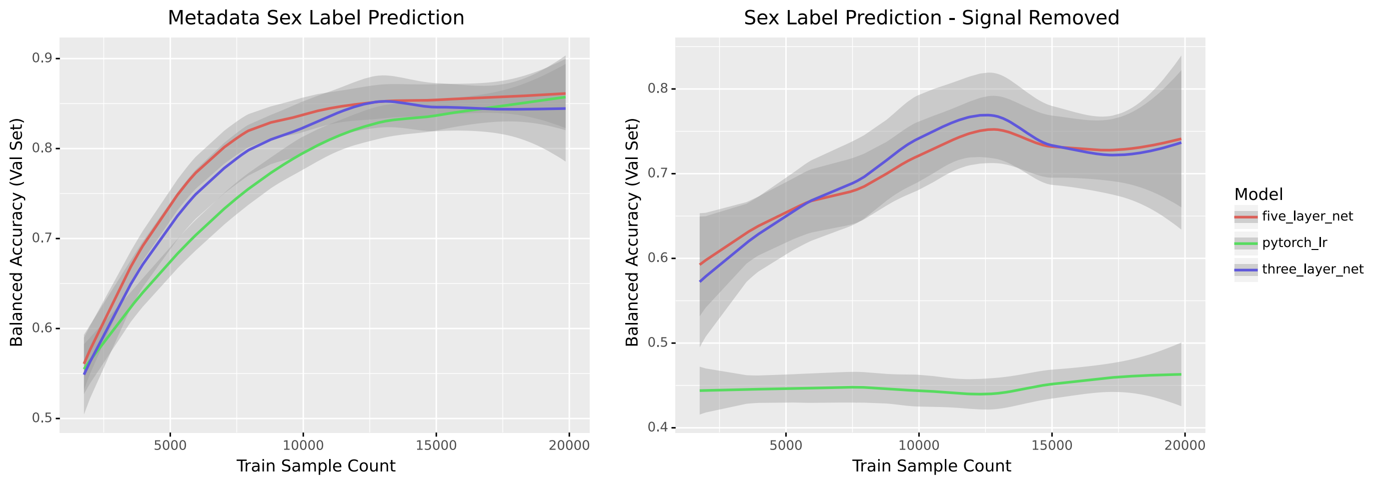


Figure 6: Metadata sex prediction

The results so far aren't invalidated by a jump in accuracy caused by pretraining

A common usage pattern in machine learning is to train models on a general dataset then fine-tune them on a dataset of interest. To ensure that our results weren't made irrelevant by different behavior in the pretraining context, we examined the performance of the models with and without pretraining (Supp. fig TODO). We split our data into three sets: pretraining, training, and validation (Fig. 1 bottom), then trained two identically initialized copies of each model. One copy was trained solely on the training data, while the other was trained on the pretraining data then fine-tuned on the training data.

The pretrained models showed high performance even when trained with small amounts of data from the training set. However, the nonlinear models did not have a greater performance gain from pretraining than logistic regression, and the balanced accuracy was similar across models. In fact, all models showed lower performance than when using the full training data, as models forget information from previous runs during fine-tuning [21].

Our results aren't caused by using a more sophisticated form of data splitting

We considered it possible that our results were an artifact of our method of dataset splitting, and set out to test it. There is a common method of data splitting we refer to as samplewise splitting (see Methods) that leaks information between the train and validation sets when used in transcriptomic tasks. To avoid this data leakage, we split the dataset at the study level in our Recount3 analyses. We found that there is in fact a large degree of performance inflation evident when comparing the sample-split results to the study-split results in the Recount3 multiclass setting (Supp Fig. 10). While this supports our decision to use study-level splitting, the relative performance of each model stays the same regardless of data splitting technique.

Methods

Data

Recount3

Our first dataset consisted of bulk RNA-seq data downloaded from the recount3 compendium [22] on TODO date. Before filtering, the dataset contained 317,258 samples, each containing 63,856 genes.

To filter out single-cell data, we removed all samples with a sparsity greater than 75 percent. We also removed all samples marked 'scrna-seq' by Recount3's pattern matching method (stored in the metadata as 'recount_pred.pattern.predict.type').

To ensure the samples were comparable, we converted the data to transcripts per kilobase million using gene lengths from BioMart [23]. To ensure the genes' magnitudes were comparable, we performed standardization to scale each gene's range from zero to one. We kept the 5,000 most variable genes within the dataset.

Samples were labeled with their corresponding tissues using the 'recount_pred.curated.tissue' field in the Recount3 metadata. These labels were based on manual curation by the Recount3 authors. A total of 20324 samples in the dataset had corresponding tissue labels.

Samples were also labeled with their corresponding sex using labels from Flynn et al. [3]. These labels were derived using pattern matching on metadata from the European Nucleotide Archive [24]. A total of 23,525 samples in our dataset had sex labels.

GTEx

We downloaded 17,382 TPM-normalized samples of bulk RNA-seq expression data from version 8 of GTEx to validate our results. We then zero-one standardized the data and kept the 5000 most variable genes. The tissue labels we used for the GTEx dataset were derived from the 'SMTS' column of the sample metadata file.

Simulated data

We generated three simulated datasets to ensure the signal removal process was working as expected. The first dataset contained 1000 samples of 5000 features corresponding to two classes. 2500 of those features contained linear signal. That is to say that the feature values corresponding to one class were drawn from a standard normal distribution, while the feature values corresponding to the other were drawn from a Gaussian with a mean of 6 and unit variance.

The nonlinear features were generated similarly. The values for the nonlinear features were drawn from a standard normal distribution for one class, while the second class had values drawn from either a mean 6 or mean -6 Gaussian with equal probability. These features are referred to as "nonlinear" because a linear classifier is unable to draw the two dividing lines necessary to correctly classify such data.

The second dataset was similar to the first dataset, but it consisted solely of 2500 linear features. The final dataset consisted solely of values drawn from a standard normal distribution regardless of class label.

Model architectures

We use three representative models to demonstrate the performance profiles of different model classes. Each model was a implemented in Pytorch [**arxiv:1912.01703?**], used the same optimizer, and was trained for at most 50 epochs.

The nonlinear models were fully connected neural networks. The first was a three layer network with hidden layers of size 2500 and 1250. Our second was a five layer network, with hidden layers of size 2500, 2500, 2500, and 1250. Both models used ReLU nonlinearities [25].

The final model was an implementation of logistic regression, a linear model. It was designed to be trained as similarly to the neural nets as possible to allow for a fair comparison.

Model training

Optimization

Our models minimized the cross-entropy loss using an Adam [**arxiv:1412.6980?**] optimizer on minibatches of data. They also used inverse frequency weighting to avoid giving more weight to more common classes.

Regularization

The models used early stopping and gradient clipping to regularize their training. Both neural nets used dropout [26] with a probability of 0.5. The deeper network used batch normalization [27] to mitigate the vanishing gradient problem.

Hyperparameters

The hyperparameters for each model can be found in their corresponding config file at https://github.com/greenelab/saged/tree/master/model_configs/supervised.

Determinism

Model trainining was made deterministic by setting the Python, NumPy, and PyTorch random seeds for each run, as well as setting the PyTorch backends to deterministic and disabling the benchmark mode.

Logging

Model training progress was tracked and recorded using Neptune [28].

Signal removal

We used Limma[20] to remove linear signal associated with tissues in the data. More precisely, we ran the 'removeBatchEffect' funcion from Limma on the full dataset, using the tissue labels as batch labels.

Model Evaluation

In our analyses we use five-fold cross-validation with two types of data splitting. The first type is samplewise splitting. In the samplewise paradigm, gene expression samples are split into cross-validation folds at random without respect to which studies they belong to. In the stratified paradigm, samples are added to folds in chunks. For example, in a studywise split, the studies are randomly assigned to folds such that all samples in a given study end up in a single fold.

While samplewise splitting is common in the machine learning and computational biology literature, it is ill-suited to gene expression data. There are study-specific signals in the data, and having samples from the same study in the training and validation sets causes information leakage. As a result, samplewise splitting inflates the estimated performance of the models. Studywise splitting avoids leakage by ensuring all the study-specific signals stay within either the training or the validation sets.

Hardware

All analyses were performed on an Ubuntu 18.04 machine with 64 GB of RAM. The CPU used was an AMD Ryzen 7 3800xt processor with 16 cores, and the GPU used was an Nvidia RTX 3090. The pipeline can be run on a computer with lower specs, but would have to run fewer elements in parallel. From initiating data download to finishing all analyses and generating all figures, the full Snakemake [29] pipeline takes about TODO days to run.

Recount3 tissue prediction

In the Recount3 setting the multitissue classification analyses were trained on the 21 tissues (see Supp. Methods) that had at least 10 studies in the dataset. Each model was trained to determine which of the 21 tissues a given expression sample corresponded to. The models' performance was then measured based on the balanced accuracy across all classes.

The binary classification setting was similar. The five tissues with the most studies (brain, blood, breast, stem cell, and cervix) were compared against each other pairwise. The expression used in this setting was the set of samples labled as one of the two tissues being compared.

The data for both settings were split in a stratified manner based on study.

GTEx classification

The multitissue classification analysis for GTEx used all 31 tissues. Both the multiclass and binary settings were formulated and evaluated in the same way as in the recount data. However, rather than being split studywise, the data were stratified according to donor.

Simulated data classification/sex prediction

The sex prediction and simulated data classification tasks were solely binary. Both settings used balanced accuracy, as in the Recount3 and GTEx problems.

Pretraining

In order to test the effects of pretraining on the different model types, we split the data into three sets. Approximately forty percent of the data went into the pretraining set, forty percent went into the training set, and twenty percent went into the validation set. The data was split such that each study's samples were in only one of the three sets, to simulate the real-world scenario where a model is trained on a publicly available data then fine-tuned on a dataset of interest.

To evaluate the models, we made two copies of each model with the same weight initialization. The first copy was trained solely on the training data, while the second was trained on the pretraining data, then the training data. Both models were then evaluated on the validation set. This process was then repeated four more times with different studies assigned to the pretraining, training, and validation sets.

Conclusion

In this paper, we performed a series of analyses determining the relative performance of linear and nonlinear models in multiple domains. We found that, consistent with previous papers [14,15], linear and nonlinear models performed roughly equivalently in a number of tasks. That is to say that there are some tasks where linear models perform better, some tasks where nonlinear models have better performance, and some tasks where both model types are equivalent.

To determine what led to the performance of the two model classes, we removed all linear signal in the data and found that even in situations where both model types had the same performance there was residual signal that only our nonlinear models were capable of learning. This implies that the results that we observed were not driven by a lack of nonlinear signal. We then simulated data to ensure that the signal removal method was not inducing nonlinear signal. We continued by showing that these results held in slightly altered problem settings, such as using a pretraining dataset before the training dataset and using samplewise data splitting instead of studywise splitting. Finally, we validated our results on different datasets and domains by running the same analyses on GTEx data and predicting sex labels from expression.

We were able to show that there is both linear and nonlinear signal in our datasets, but that the existence of nonlinear signal does not necessarily lead nonlinear models to make higher-accuracy predictions. Given that there is nonlinear signal that relates expression to tissue types, why is it that such signal doesn't allow models to make better predictions? We believe that it is because the nonlinear signal is either redundant with the linear signal, or unreliable enough that nonlinear models choose to learn the linear signal instead.

One limitation of our study is that the results likely do not hold in an infinite data setting. Deep learning models have been shown to solve hard problems in biology and tend to greatly outperform linear models when given enough data. However, we do not yet live in a world with huge amounts of gene expression data and accompanying uniform metadata. Our results are generated on some of the largest labeled expression datasets in existence (Recount3 and GTEx), but our tens of thousands of samples are far from the millions or billions used in deep learning research.

We are also unable to make claims about all problem domains. There are many potential transcriptomic prediction tasks, and many datasets to perform them on. While we show that nonlinear signal is not always helpful in tissue prediction, and others have shown the same for various disease prediction tasks, there are also problems such as sex metadata prediction where the nonlinear signal seems to be important.

Ultimately, our results show that the existence of task-relevant nonlinear signal in the data does not necessarily lead nonlinear models to outperform linear ones. Determining what causes this disconnect is an exciting avenue of future research. Additionally we demonstrate that while there are problems where complicated models are useful, scientists making predictions from expression data should always include simple linear baseline models to determine whether more complex models are warranted.

Supplementary Materials

Methods

Recount3 tissues used

The tissues used from Recount3 were blood, breast, stem cell, cervix, brain, kidney, umbilical cord, lung, epithelium, prostate, liver, heart, skin, colon, bone marrow, muscle, tonsil, blood vessel, spinal cord, testis, and placenta.

Data exploration

To determine whether our results were driven by an artifact in the data, we performed exploratory data analysis. First, we looked for whether anything stood out when comparing per-sample performance between models. Upon doing so, we found that TODO describe after running on all-tissue When looking at the results, we noticed that some samples were consistently misclassified across models. We suspected it might be due to label imbalance, but a confusion matrix showed that not to be the case. We examined the metadata for attributes that might be correlated with sample prediction hardness, and found that these samples tend to have a lower read quality than other samples.

Results

Signal removal

As mentioned in the main results section, training a more accurate linear model on signal-removed training data leads to a less accurate model on the validation data due to removing signal on the entire dataset at once. However, the alternative can lead to even worse artifacts, as seen in fig. sup. 7, where the the linear and nonlinear models have random average performance via different methods. We suspect the swings in model performance in the nonlinear data are due to colinearity in the features. That is to say that given the option of a number of possible corrections that can be made to remove the linear signal in the data, there is no guarantee that the same one is selected in the train and validation sets. For that reason, it's possible to end up with results where the model performance varies wildly in different runs of the signal removal method based on the input data.

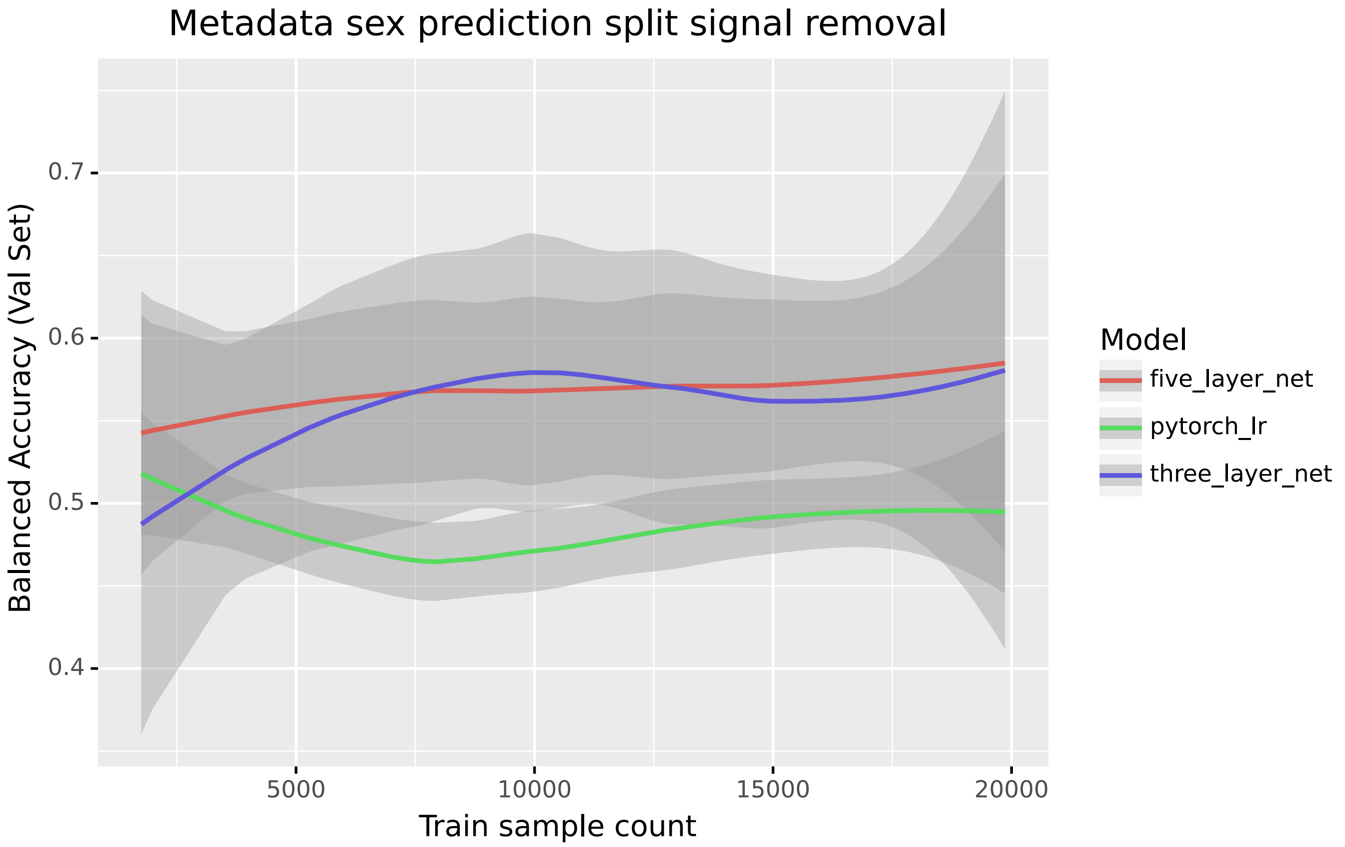


Figure 7:
Sex prediction results when removing signal from training fold and validation fold separately.

Scikit-learn logistic regression

The Pytorch logistic regression implementation we used was designed to be as close to the neural network implementations as possible to ensure the models were comparable. Accordingly, we were optimizing for similarity of implementation instead of maximal performance. Scikit-learn, on the other hand, optimizes their models to have the best out-of-box performance they can manage. To understand the magnitude of the difference, we compared the sklearn logistic regression model to our pytorch models. We found that it generally outperformed the other models (supp. fig. 8).

Figure 8: TODO description, build figure

Figure 8:
TODO description, build figure

Recount3 Pretraining Figure

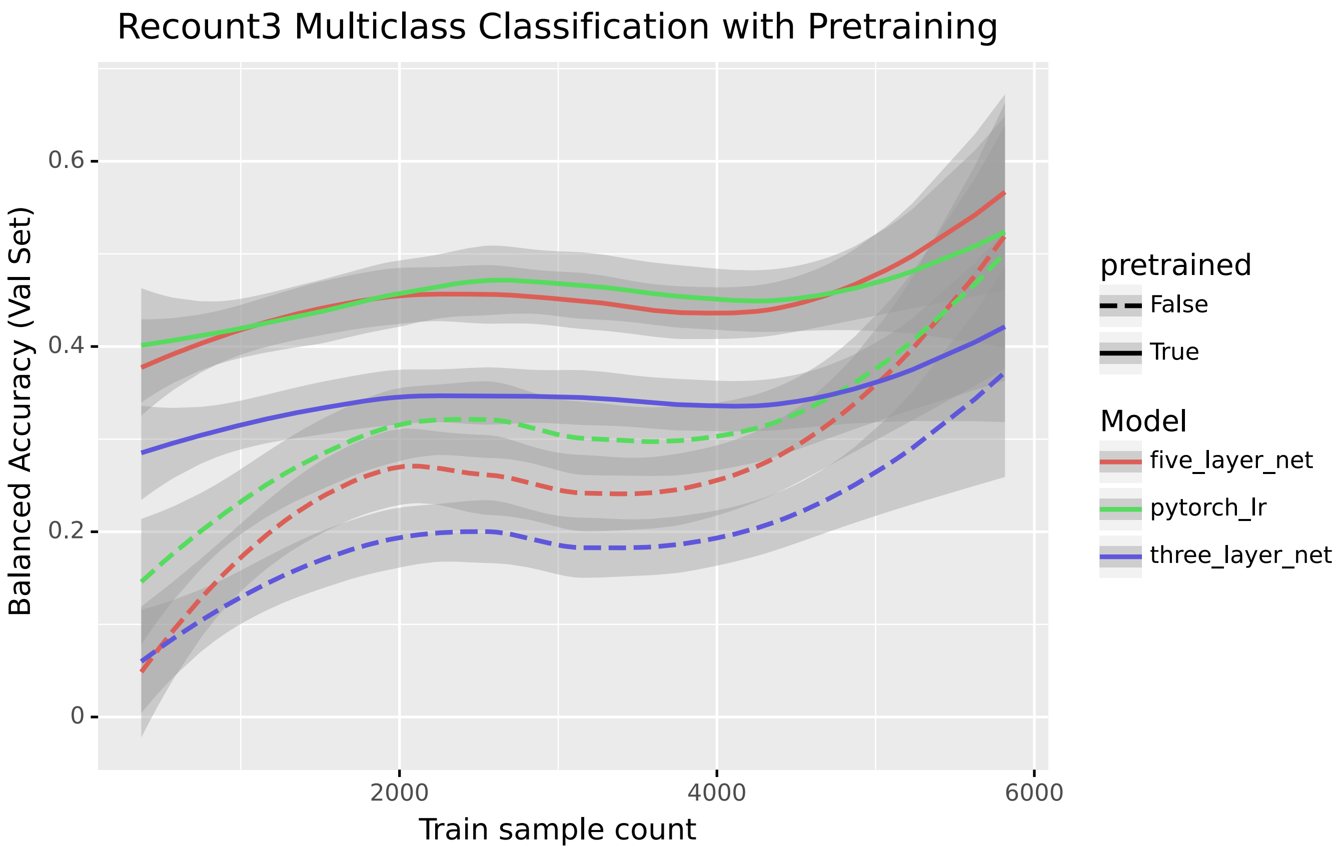


Figure 9:
Performance of Recount3 multiclass prediction with pretraining

Samplewise splitting

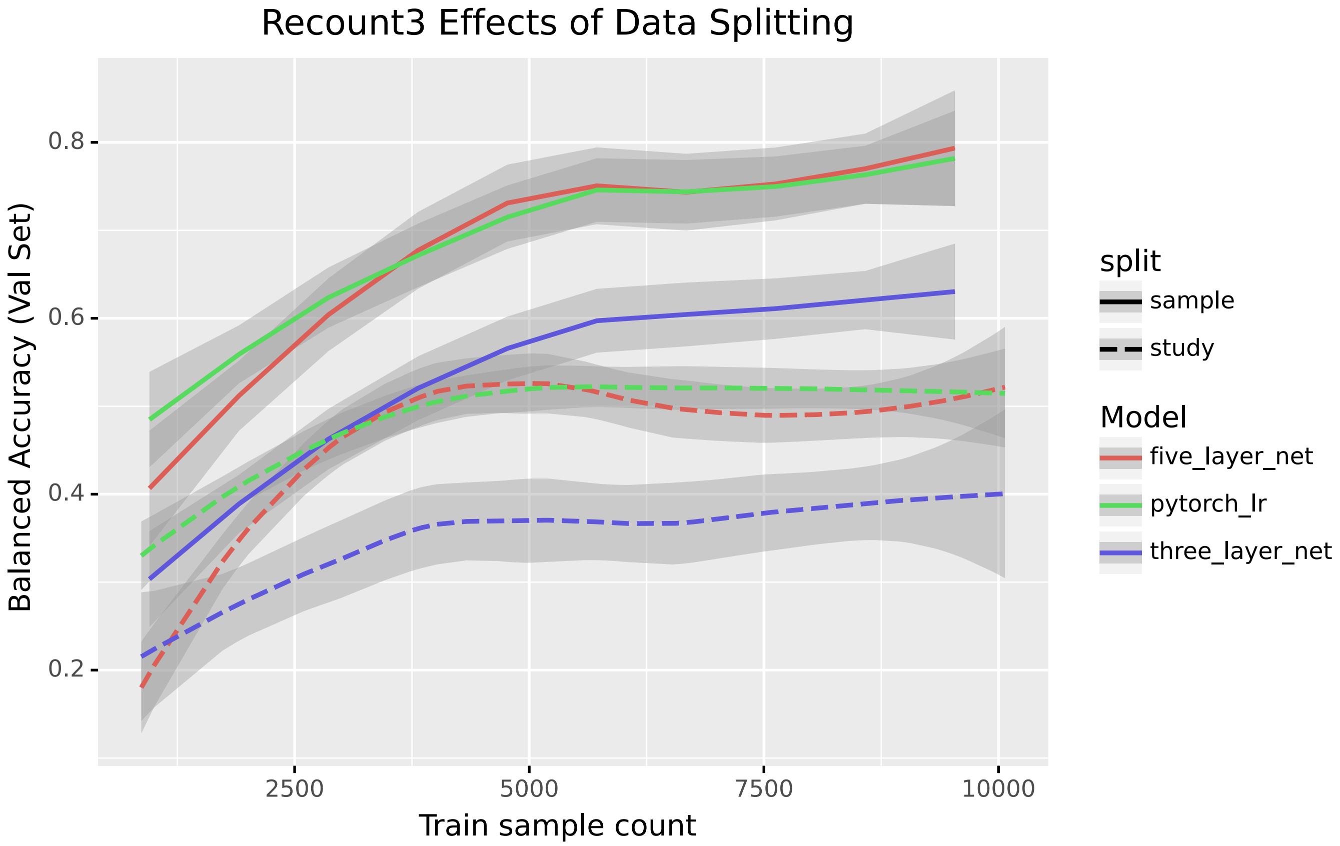


Figure 10:
Performance of models with different data splitting

References

1. **Supervised Risk Predictor of Breast Cancer Based on Intrinsic Subtypes**
Joel S Parker, Michael Mullins, Maggie CU Cheang, Samuel Leung, David Voduc, Tammi Vickery, Sherri Davies, Christiane Fauron, Xiaping He, Zhiyuan Hu, … Philip S Bernard
Journal of Clinical Oncology (2009-03-10) https://doi.org/c2688w
DOI: 10.1200/jco.2008.18.1370 · PMID: 19204204 · PMCID: PMC2667820

2. **Gene Expression Profiling for the Identification and Classification of Antibody-Mediated Heart Rejection**
Alexandre Loupy, Jean Paul Duong Van Huyen, Luis Hidalgo, Jeff Reeve, Maud Racapé, Olivier Aubert, Jeffery M Venner, Konrad Falmuski, Marie Cécile Bories, Thibaut Beuscart, … Philip F Halloran
Circulation (2017-03-07) https://doi.org/f9vfvw
DOI: 10.1161/circulationaha.116.022907 · PMID: 28148598

3. **Large-scale labeling and assessment of sex bias in publicly available expression data**
Emily Flynn, Annie Chang, Russ B Altman
BMC bioinformatics (2021-03-30) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8011224/
DOI: 10.1186/s12859-021-04070-2 · PMID: 33784977 · PMCID: PMC8011224

4. **Massive mining of publicly available RNA-seq data from human and mouse**
Alexander Lachmann, Denis Torre, Alexandra B Keenan, Kathleen M Jagodnik, Hoyjin J Lee, Lily Wang, Moshe C Silverstein, Avi Ma'ayan
Nature Communications (2018-04-10) https://doi.org/gc92dr
DOI: 10.1038/s41467-018-03751-6 · PMID: 29636450 · PMCID: PMC5893633

5. **A curated database reveals trends in single-cell transcriptomics**
Valentine Svensson, Eduardo da Veiga Beltrame, Lior Pachter
Database (2020) https://doi.org/gjnr3h
DOI: 10.1093/database/baaa073 · PMID: 33247933 · PMCID: PMC7698659

6. **DeePathology: Deep Multi-Task Learning for Inferring Molecular Pathology from Cancer Transcriptome**
Behrooz Azarkhalili, Ali Saberi, Hamidreza Chitsaz, Ali Sharifi-Zarchi
Scientific Reports (2019-11-11) https://doi.org/gpg7vc
DOI: 10.1038/s41598-019-52937-5 · PMID: 31712594 · PMCID: PMC6848155

7. **Bias-invariant RNA-sequencing metadata annotation**
Hannes Wartmann, Sven Heins, Karin Kloiber, Stefan Bonn
GigaScience (2021-09) https://doi.org/gph9xp
DOI: 10.1093/gigascience/giab064 · PMID: 34553213 · PMCID: PMC8559615

8. **Improved prediction of smoking status via isoform-aware RNA-seq deep learning models**
Zifeng Wang, Aria Masoomi, Zhonghui Xu, Adel Boueiz, Sool Lee, Tingting Zhao, Russell Bowler, Michael Cho, Edwin K Silverman, Craig Hersh, … Peter J Castaldi
PLOS Computational Biology (2021-10-11) https://doi.org/gph9xq
DOI: 10.1371/journal.pcbi.1009433 · PMID: 34634029 · PMCID: PMC8530282

9. **The evolution of gene expression and the transcriptome–phenotype relationship**
Peter W Harrison, Alison E Wright, Judith E Mank
Seminars in Cell & Developmental Biology (2012-04) https://doi.org/fxqd2g
DOI: 10.1016/j.semcdb.2011.12.004 · PMID: 22210502 · PMCID: PMC3378502

10. **Nonlinear Dynamics in Gene Regulation Promote Robustness and Evolvability of Gene Expression Levels**
Arno Steinacher, Declan G Bates, Ozgur E Akman, Orkun S Soyer
PLOS ONE (2016-04-15) https://doi.org/f8xrrq
DOI: 10.1371/journal.pone.0153295 · PMID: 27082741 · PMCID: PMC4833316

11. **ADAGE-Based Integration of Publicly Available Pseudomonas aeruginosa Gene Expression Data with Denoising Autoencoders Illuminates Microbe-Host Interactions**
Jie Tan, John H Hammond, Deborah A Hogan, Casey S Greene
mSystems (2016-02-23) https://doi.org/gcgmbq
DOI: 10.1128/msystems.00025-15 · PMID: 27822512 · PMCID: PMC5069748

12. **A semi-supervised deep learning method based on stacked sparse auto-encoder for cancer prediction using RNA-seq data**
Yawen Xiao, Jun Wu, Zongli Lin, Xiaodong Zhao
Computer Methods and Programs in Biomedicine (2018-11) https://doi.org/gfnm5c
DOI: 10.1016/j.cmpb.2018.10.004 · PMID: 30415723

13. **A biological network-based regularized artificial neural network model for robust phenotype prediction from gene expression data**
Tianyu Kang, Wei Ding, Luoyan Zhang, Daniel Ziemek, Kourosh Zarringhalam
BMC Bioinformatics (2017-12) https://doi.org/gf8cm6
DOI: 10.1186/s12859-017-1984-2 · PMID: 29258445 · PMCID: PMC5735940

14. **Standard machine learning approaches outperform deep representation learning on phenotype prediction from transcriptomics data**
Aaron M Smith, Jonathan R Walsh, John Long, Craig B Davis, Peter Henstock, Martin R Hodge, Mateusz Maciejewski, Xinmeng Jasmine Mu, Stephen Ra, Shanrong Zhao, … Charles K Fisher
BMC Bioinformatics (2020-03-20) https://doi.org/ggpc9d
DOI: 10.1186/s12859-020-3427-8 · PMID: 32197580 · PMCID: PMC7085143

15. **A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models**
Evangelia Christodoulou, Jie Ma, Gary S Collins, Ewout W Steyerberg, Jan Y Verbakel, Ben Van Calster
Journal of Clinical Epidemiology (2019-06) https://doi.org/gfzstd
DOI: 10.1016/j.jclinepi.2019.02.004 · PMID: 30763612

16. **recount3: summaries and queries for large-scale RNA-seq expression and splicing**
Christopher Wilks, Shijie C Zheng, Feng Yong Chen, Rone Charles, Brad Solomon, Jonathan P Ling, Eddie Luidy Imada, David Zhang, Lance Joseph, Jeffrey T Leek, … Ben Langmead
Genome Biology (2021-11-29) https://doi.org/gnm7zc
DOI: 10.1186/s13059-021-02533-6 · PMID: 34844637 · PMCID: PMC8628444

17. **The Genotype-Tissue Expression (GTEx) project**
John Lonsdale, Jeffrey Thomas, Mike Salvatore, Rebecca Phillips, Edmund Lo, Saboor Shad, Richard Hasz, Gary Walters, Fernando Garcia, Nancy Young, … Helen F Moore
Nature Genetics (2013-05-29) https://doi.org/gd5z68
DOI: 10.1038/ng.2653 · PMID: 23715323 · PMCID: PMC4010069

18. **Large-scale labeling and assessment of sex bias in publicly available expression data**
Emily Flynn, Annie Chang, Russ B Altman
BMC Bioinformatics (2021-03-30) https://doi.org/gpjt3n
DOI: 10.1186/s12859-021-04070-2 · PMID: 33784977 · PMCID: PMC8011224

19. **Scikit-learn: Machine Learning in Python**
Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, ... Édouard Duchesnay
Journal of Machine Learning Research (2011) http://jmlr.org/papers/v12/pedregosa11a.html

20. **limma powers differential expression analyses for RNA-sequencing and microarray studies**
Matthew E Ritchie, Belinda Phipson, Di Wu, Yifang Hu, Charity W Law, Wei Shi, Gordon K Smyth
Nucleic Acids Research (2015-01-20) https://doi.org/f7c4n5
DOI: 10.1093/nar/gkv007 · PMID: 25605792 · PMCID: PMC4402510

21. **Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem**
Michael McCloskey, Neal J Cohen
Psychology of Learning and Motivation (1989) https://doi.org/ckpftf
DOI: 10.1016/s0079-7421(08)60536-8

22. **Phosphorylation of ETS transcription factor ER81 in a complex with its coactivators CREB-binding protein and p300**
S Papoutsopoulou, R Janknecht
Molecular and cellular biology (2000-10) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC86284/
DOI: 10.1128/mcb.20.19.7300-7310.2000 · PMID: 10982847 · PMCID: PMC86284

23. **BioMart--biological queries made easy**
Damian Smedley, Syed Haider, Benoit Ballester, Richard Holland, Darin London, Gudmundur Thorisson, Arek Kasprzyk
BMC genomics (2009-01-14) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2649164/
DOI: 10.1186/1471-2164-10-22 · PMID: 19144180 · PMCID: PMC2649164

24. **The European Nucleotide Archive**
Rasko Leinonen, Ruth Akhtar, Ewan Birney, Lawrence Bower, Ana Cerdeno-Tárraga, Ying Cheng, Iain Cleland, Nadeem Faruque, Neil Goodgame, Richard Gibson, ... Guy Cochrane
Nucleic acids research (2011-01) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3013801/
DOI: 10.1093/nar/gkq967 · PMID: 20972220 · PMCID: PMC3013801

25. **Rectified linear units improve restricted boltzmann machines**
Vinod Nair, Geoffrey E Hinton
Proceedings of the 27th International Conference on International Conference on Machine Learning (2010-06-21) https://dl.acm.org/doi/10.5555/3104322.3104425
ISBN: 9781605589077

26. **Dropout: A Simple Way to Prevent Neural Networks from Overfitting**
Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov
Journal of Machine Learning Research (2014) http://jmlr.org/papers/v15/srivastava14a.html

27. **Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift**
Sergey Ioffe, Christian Szegedy
Proceedings of the 32nd International Conference on Machine Learning (2015-06-01) https://proceedings.mlr.press/v37/ioffe15.html

28. **Neptune: Experiment management and collaboration tool**
neptune.ai
(2020) https://neptune.ai

29. **Snakemake--a scalable bioinformatics workflow engine**

J Koster, S Rahmann
Bioinformatics (2012-08-20) https://doi.org/gd2xzq
DOI: 10.1093/bioinformatics/bts480 · PMID: 22908215