

Ten Quick Tips for Deep Learning in Biology

This manuscript ([permalink](#)) was automatically generated from [Benjamin-Lee/deep-rules@41b50f2](#) on October 25, 2020.

Authors

Please note the current author order is chronological and does not reflect the final order.

- **Benjamin D. Lee**

 [0000-0002-7133-8397](#) ·  [Benjamin-Lee](#)

Lab41, In-Q-Tel; School of Engineering and Applied Sciences, Harvard University; Department of Genetics, Harvard Medical School

- **Alexander J. Titus**

 [0000-0002-0145-9564](#) ·  [AlexanderTitus](#)

Titus Analytics

- **Kun-Hsing Yu**

 [0000-0001-9892-8218](#) ·  [khyu](#)

Department of Biomedical Informatics, Harvard Medical School

- **Marc G. Chevrette**

 [0000-0002-7209-0717](#) ·  [chevrm](#) ·  [wildtypeMC](#)

Wisconsin Institute for Discovery and Department of Plant Pathology, University of Wisconsin-Madison

- **Paul Allen Stewart**

 [0000-0003-0882-308X](#) ·  [pstew](#)

Biostatistics and Bioinformatics Shared Resource, Moffitt Cancer Center

- **Evan M. Cofer**

 [0000-0003-3877-0433](#) ·  [evancofer](#)

Lewis-Sigler Institute for Integrative Genomics, Princeton University; Graduate Program in Quantitative and Computational Biology, Princeton University

- **Sebastian Raschka**

 [0000-0001-6989-4493](#) ·  [rasbt](#)

Department of Statistics, University of Wisconsin-Madison

- **Finlay Maguire**

 [0000-0002-1203-9514](#) ·  [fmaguire](#)

Faculty of Computer Science, Dalhousie University

- **Benjamin J. Lengerich**

 [0000-0001-8690-9554](#) ·  [blengerich](#)

Computer Science Department, Carnegie Mellon University

- **Alexandr A. Kalinin**

 [0000-0003-4563-3226](#) ·  [alxndrkalinin](#)

Department of Computational Medicine and Bioinformatics, University of Michigan

- **Anthony Gitter**

 [0000-0002-5324-9833](#) ·  [agitter](#) ·  [anthonygitter](#)

Department of Biostatistics and Medical Informatics, University of Wisconsin-Madison; Morgridge Institute for Research

- **Casey S. Greene**

 [0000-0001-8713-9213](#) ·  [cgreene](#)

Department of Systems Pharmacology and Translational Therapeutics, Perelman School of Medicine, University of Pennsylvania

- **Simina M. Boca**

 [0000-0002-1400-3398](#) ·  [SiminaB](#)

Innovation Center for Biomedical Informatics, Georgetown University Medical Center; Department of Oncology, Georgetown University Medical Center; Department of Biostatistics, Bioinformatics and Biomathematics, Georgetown University Medical Center; Cancer Prevention and Control Program, Lombardi Comprehensive Cancer Center

- **Timothy J. Triche, Jr.**

 [0000-0001-5665-946X](#) ·  [ttriche](#)

Center for Epigenetics, Van Andel Research Institute; Department of Translational Genomics, Keck School of Medicine, University of Southern California

- **Thiago Britto-Borges**

 [0000-0002-6218-4429](#) ·  [tbrittoborges](#)

Section of Bioinformatics and Systems Cardiology, Department of Internal Medicine III and Klaus Tschira Institute for Integrative Computational Cardiology, University Hospital Heidelberg

- **Elana J. Fertig**

 [0000-0003-3204-342X](#) ·  [ejfertig](#)

Department of Oncology, Department of Biomedical Engineering, Department of Applied Mathematics and Statistics, Johns Hopkins University

- **Michael D. Kessler**

 [0000-0003-1258-5221](#) ·  [mdkessler](#)


Department of Oncology, Johns Hopkins University

- **Alexandra J. Lee**

 [0000-0002-0208-3730](#) ·  [ajlee21](#)

Genomics and Computational Biology Graduate Program, University of Pennsylvania; Department of Systems Pharmacology and Translational Therapeutics, University of Pennsylvania

- **Beth Signal**

·  [betsig](#)

Climate Change Cluster, University of Technology Sydney

Introduction

Machine learning is a branch of computational science organized around the learning by computers via algorithms of relationships between dataset variables. Traditionally this has referred to statistical methods such as linear and logistic regression, but as the scale and complexity of data have increased, many new ML methods have been developed to model and learn increasingly complex relationships. One powerful approach for learning complex (that is, both linear and non-linear) relationships between input and output variables is called an artificial neural network. These networks are modeled after the human brain, and they comprise artificial neurons arranged into layers. Each layer receive input from previous layers (the first of which represents the input data), and then “fires

off" its own weighted output that then serves as input into subsequent layers in the network. Thus, the process of "training" a neural network is in fact the tuning of these output weights such that a metric called a cost function is optimized. Deep learning utilizes artificial neural networks with many layers (hence the term "deep"), and given the computational advances of recent decades, is now capable of being applied to massive data sets and in enumerable contexts. In many circumstances, deep learning has the capacity to learn more complex relationships and make more accurate predictions than other methods. Therefore, deep learning has become its own subfield of machine learning, and in the context of biological research, has been increasingly used to derive novel insights from high-dimensional biological data [1]. For example, deep learning has been used to predict protein-drug binding kinetics [2], to identify the lab-of-origin of synthetic DNA [3], and to uncover the facial phenotypes of genetic disorders [4].

However, for researchers and students new to machine learning, and even for those experienced in using classical machine learning methods (for example, linear regression), using deep learning correctly can be daunting due to the highly complex nature of the method and the corresponding toolkits. General resources communicating best practices to the scientific community broadly and the biological community specifically are scarce, and any resources that do exist are prone to reaching obsolescence rapidly due to deep learning's active and specialized nature. In addition, the lack of established standards or concise recommendations for the application of deep learning to biological questions further limits newcomers from using state-of-the-art deep learning in their research.

To address this issue, we solicited input from a community of researchers with varied biological and deep learning interests. These individuals collaboratively contributed to the writing of this manuscript using the GitHub version control platform [5] and the Manubot manuscript generation toolset [6]. The goal was to articulate a practical, accessible, and concise set of guidelines and suggestions for biologically oriented researchers to follow when using deep learning.

In the course of our discussions, several themes became clear: the importance of understanding and applying machine learning fundamentals [7] as a baseline for utilizing deep learning, the necessity for extensive model comparisons with careful evaluation, and the need for critical thought in interpreting results generated by deep learning, among others. The major similarities between deep learning and traditional computational methods also became apparent. Although deep learning is a distinct subfield of machine learning, it is still a subfield. It is subject to the many limitations inherent to machine learning, and many best practices for machine learning also apply to deep learning. In addition, as with all computational methods, deep learning should be applied in a systematic manner that is reproducible and rigorously tested. Ultimately, the tips we collate range from high-level guidance to best practices for implementation, and it is our hope that they will provide actionable, deep learning-specific instruction for both new and experienced deep learning practitioners. By making deep learning more accessible for use in biological research, we aim to improve the overall usage and reporting quality of deep learning in the literature, and to enable increasing numbers of researchers to utilize these state-of-the-art techniques effectively and accurately.

Tip 1: Decide whether deep learning is appropriate for your problem

In recent years, the number of publications implementing deep learning in biology have risen tremendously. Given deep learning's usefulness across a range of scientific questions and data modalities, it may appear that it is a panacea for modeling problems. Indeed, neural networks are universal function approximators, meaning that they are in principle capable of learning any function [8,9]. If deep learning is so powerful and popular, why would one ever not choose to use it?

The reason is simple: deep learning is not suited to every situation in reality. Training deep learning models requires a significant amount of data, computing power, and expertise. In some areas of biology where data collection is thoroughly automated, such as DNA sequencing, large amounts of

quality data may be available. For other areas which rely on manual data collection, there may not be enough data to effectively train models. Though there are methods to increase the amount of training data, such as data augmentation (in which existing data is slightly manipulated to yield “new” samples) and weak supervision (in which simple labeling heuristics are combined to produce noisy, probabilistic labels) [10], these methods cannot overcome a complete shortage of data. In the context of supervised classification, deep learning should be considered for datasets with at least one hundred samples per class [11] as a rule of thumb, though in all cases it is best suited to cases when datasets contain orders of magnitude more samples.

Furthermore, training deep learning models can be very demanding, often requiring extensive computing infrastructure and patience to achieve state-of-the-art performance [12]. In some deep learning contexts, such as generating human-like text, state-of-the-art models have over one hundred billion parameters [13]. Training such large models from scratch can be a costly and time-consuming undertaking [14]. Luckily, most deep learning research in biology will not require nearly as much computation, though it usually requires more than can be done feasibly on an individual consumer-grade device. Specialized hardware such as discrete graphics processing units (GPUs) or custom deep learning accelerators can dramatically reduce the time and cost required to train models, but this hardware is not universally accessible. Currently, both GPU- and deep learning-optimized accelerator-equipped servers can be rented from cloud providers, though working with these servers adds additional cost and complexity. As deep learning becomes more popular, these accelerators are likely to be more broadly available (for example, recent-generation iPhones already have such hardware). In contrast, traditional machine learning training can often be done on a laptop (or even a \$5 computer [15]) in seconds to minutes.

Beyond the necessity for greater data and computational capacity in deep learning, building and training deep learning models generally requires more expertise than traditional machine learning models. Currently, there are several competing programming frameworks for deep learning such as Tensorflow [16] and PyTorch [17]. These frameworks allow users to create and deploy entirely novel model architectures and are widely used in deep learning research as well as in industrial applications. This flexibility combined with the rapid development of the deep learning field has resulted in large, complex frameworks that can be daunting to new users. For readers new to software development but experienced in biology, gaining computational skills while interfacing with such complex industrial-grade tools can be a challenge. An advantage of machine learning over deep learning is that currently there are more tools capable of automating the model selection and training process. Automated machine learning (AutoML) tools such as TPOT [18], which is capable of using genetic programming to optimize machine learning pipelines, and Turi Create [19], a task-oriented machine learning and deep learning framework which automatically tests multiple machine learning models when training, allow users to achieve competitive performance with only a few lines of code. Luckily, there are efforts underway to reduce the expertise required to build and use deep learning models. Indeed, both TPOT and Turi Create, as well as other tools such as AutoKeras [20], are capable of abstracting away much of the programming required for “standard” deep learning tasks. Projects such as Keras [21], a high-level interface for TensorFlow, make it relatively straightforward to design and test custom deep learning architectures. In the future, projects such as these are likely to bring deep learning experimentation within reach to even more researchers.

There are some types of problems in which using deep learning is strongly indicated over machine learning. Assuming a sufficient quantity of quality data is available, applications such as computer vision and natural language processing are likely to benefit from deep learning. In fact, these areas were the first to see significant breakthroughs through the application of deep learning [22] during the recent deep learning revolution." For example, Ferreira et al. used deep learning to recognize individual birds from images [23]. This problem was historically difficult but, by combining automatic data collection using RFID tags with data augmentation and transfer learning (explained in [Tip 5](#)), the authors were able to use deep learning achieve 90% accuracy in several species. Other areas include

generative models, in which new samples are able to be created based on the training data, and reinforcement learning, in which agents are trained to interact with their environments. In general, before using deep learning, investigate whether similar problems (including analogous ones in other domains) have been solved successfully using deep learning.

Depending on the amount and the nature of the available data, as well as the task to be performed, deep learning may not always be able to outperform conventional methods. As an illustration, Rajkomar et al. [24] found that simpler baseline models achieved performance comparable with that of deep learning in a number of clinical prediction tasks using electronic health records, which may be a surprise to many. Another example is provided by Koutsoukas et al., who benchmarked several traditional machine learning approaches against deep neural networks for modeling bioactivity data on moderately sized datasets [25]. The researchers found that while well tuned deep learning approaches generally tend to outperform conventional classifiers, simple methods such as Naive Bayes classification tend to outperform deep learning as the noise in the dataset increases. Similarly, Chen et al. [26] tested deep learning and a variety of traditional machine learning methods such as logistic regression and random forests on five different clinical datasets, finding that the non deep learning methods matched or exceeded the accuracy of the deep learning model in all cases while requiring an order of magnitude less training time.

In conclusion, deep learning is a tool and, like any other tool, must be used after consideration of its strengths and weaknesses for the problem at hand. Once settled upon deep learning as a potential solution, practitioners should follow the scientific method and compare its performance to traditional methods, as we will see next.

Tip 2: Use traditional methods to establish performance baselines

Deep learning requires practitioners to consider a larger number and variety of tuning parameters (that is, algorithmic settings) than more traditional machine learning methods. These settings are often called hyperparameters, and their extensiveness can make it easy to fall into the trap of performing an unnecessarily convoluted analysis. Hence, before applying deep learning to a given problem, we highly recommend implementing a simpler model with fewer hyperparameters at the beginning of each study. Such models include logistic regression, random forests, k-nearest neighbors, naive Bayes, and support vector machines, and using them can help to establish baseline performance expectations. While performance baselines available from existing literature can also serve as helpful guides, an implementation of a simpler model that uses the same software framework as planned for deep learning can greatly help with assessing the correctness of data processing steps, performance evaluation pipelines, resource requirement estimates, and computational performance estimates. Furthermore, in some cases, it can even be useful to combine simpler baseline models with deep neural networks, as such hybrid models can improve generalization performance, model interpretability, and confidence estimation [27,28].

Another potential pitfall arises from comparing the performance of baseline conventional models trained with default settings with the performance of deep learning models that have undergone rigorous tuning and optimization. Since conventional off-the-shelf machine learning algorithms (for example, support vector machines and random forests) are also likely to benefit from hyperparameter tuning, such incongruity prevents the comparison of equally optimized models and can lead to false conclusions about model efficacy. Hu and Greene [29] discuss this under the umbrella of what they call the “Continental Breakfast Included” effect, and they describe how the unequal tuning of hyperparameters across different learning algorithms can especially skew evaluation when the performance of an algorithm varies substantially with modest changes to its hyperparameters. Therefore, practitioners should tune the settings of both traditional machine and deep learning-based methods before making claims about relative performance differences, as

performance comparisons among machine learning and deep learning models are only informative when the models are equally well optimized.

To sum this tip up, practitioners are encouraged to create and fully tune several traditional models and standard pipelines before implementing a deep learning model.

Tip 3: Understand the complexities of training deep neural networks

Correctly training deep neural networks is a non-trivial process, as there are many different options and potential pitfalls at every stage. To get good results, you must expect to train many networks across a range of different parameter and hyperparameter settings. Such training can be made more difficult by the demanding nature of these deep networks, which often require extensive computing infrastructure and optimization in order to achieve state-of-the-art performance [12]. Furthermore, this experimentation is often noisy, which necessitates increased repetition and exacerbates the organizational challenges inherent to deep learning. On the whole, all code, random seeds, parameters, and results must be carefully corralled using general good coding practices (e.g. version control [30], continuous integration etc.) in order to be effective and interpretable. This organization is also fundamental to the efficient sharing and reproducibility of research work [31,32], and to the ability to keep models up to date as new data becomes available.

One specific reproducibility pitfall that is often missed in the application of deep learning is the default use of non-deterministic algorithms by CUDA/CuDNN backends when using GPUs. That is, the CUDA/CuDNN architectures that facilitate the parallelized computing that power state-of-the-art DL often use algorithms by default that produce different outcomes from iteration to iteration. Therefore, achieving reproducibility in this context requires explicitly specifying the use of deterministic algorithms (which are typically available within your deep learning library), which is distinct from the setting of random seeds that typically achieve reproducibility by controlling pseudorandom deterministic procedures such as shuffling and initialization [33].

Similar to the suggestions in [Tip 2](#) about starting with simpler models, try to start with a relatively small network and then increase the size and complexity as needed. This can help to prevent practitioners from wasting significant time and resources on running highly complex models that feature numerous unresolved problems. Again, beware of the choices that are being made implicitly (that is, by default settings) by your framework of choice (for example, choice of optimization algorithm), as these seemingly trivial specifics can actually have significant effects on model performance. For example, adaptive methods often lead to faster convergence during training, but may lead to worse generalization performance on independent datasets [34]). These nuanced elements are easy to overlook, but it is critical to carefully consider them and to evaluate their potential impact (see [Tip 6](#)).

In short, use smaller and simpler networks to enable faster prototyping and follow general software development best practices to maximize reproducibility.

Tip 4: Know your data and your question

Having a well defined scientific question and a clear analysis plan is crucial for carrying out a successful deep learning project. Just like it would be inadvisable to set foot in a laboratory and begin experiments without having a defined endpoint, a deep learning project should not be undertaken without defined goals. Foremost, it is important to assess if a dataset exists that can answer the biological question of interest using a deep learning-based approach. If so, obtaining this data (and associated metadata), and reviewing the study protocol, should be pursued as early on in the project as possible. This can help to ensure that data is as expected and can prevent the wasted time and

effort that occur when issues are discovered later on in the analytic process. For example, a publication or resource might purportedly offer an appropriate dataset that is found to be inadequate upon acquisition. The data may be unstructured when it is supposed to be structured, crucial metadata such as sample stratification might be missing, or the usable sample size may be different than expected. Any of these data issues might limit a researcher's ability to use DL to address the biological question at hand, or might otherwise require adjustment before DL can be used. Data collection should also be carefully documented, or a data collection protocol should be created and specified in the project documentation. Information about the resources used, download dates, and dataset versions are critical to preserve. Doing so will help to minimize operational confusion and will increase the reproducibility of the analysis.

Once the dataset is obtained, it is important to learn why and how the data were collected before beginning analysis. The standardized metadata that exist in many fields can help with this (for example, see [\[35\]](#)), but if at all possible, seek out a subject matter expert who has experience with the type of data you are using. Doing so will minimize guesswork and is likely to increase the success rate of a deep learning project. For example, one might presume that data collected to test the impact of an intervention derives from a randomized controlled trial. However, this is not always the case, as ethical or practical concerns often necessitate an observational study design that features prospectively or retrospectively collected data. In order to ensure similar distributions of important characteristics across study groups in the absence of randomization, such a study may have selected individuals in a fashion that best matches attributes such as age, gender, or weight. Passively collected datasets can have their own peculiarities, and other study designs can include samples that originate from the same study site, the oversampling of ethnic groups or zip codes, or sample processing differences. Such information is critical to accurate data analysis, and so it is imperative that practitioners learn about study design assumptions and data specificities prior to performing modeling. Other study design considerations that should not be overlooked include knowing whether a study involves biological or technical replicates or both. For example, the existence in a dataset of samples collected from the same individuals at different time points can have significant effects on analyses that make assumptions about sample size and independence (that is, non-independence can lower the effective sample size). Another potential issue is the existence of systematic biases, which can be induced by confounding variables and can lead to artifacts or so-called "batch effects." As a consequence, models may learn to rely on the correlations that these systematic biases underpin, even though they are irrelevant to the scientific context of the study. This can lead to misguided predictions and misleading conclusions [\[36\]](#). As described in [Tip 1](#), unsupervised learning and other exploratory analyses can help to identify such biases in these datasets prior to applying a deep learning model.

Overall, practitioners should make sure to thoroughly study their data and understand its context and peculiarities *before* moving on to performing deep learning.

Tip 5: Choose an appropriate data representation and neural network architecture

While certain best practices have been established by the research community [\[37\]](#), architecture design choices remain largely problem-specific and are vastly empirical efforts requiring extensive experimentation. Furthermore, as deep learning is a quickly evolving field, many recommendations are often short-lived, and are frequently replaced by newer insights supported by recent empirical results. This is further complicated by the fact that many recommendations do not generalize well across different problems and datasets. Therefore, unfortunately, choosing how to represent your data and design your architecture is closer to an art than a science. That said, there are some general principles that are useful to follow when experimenting.

First and foremost, use your knowledge of the available data and your question (see [Tip 4](#)) to inform your data representation and architectural design choices. For example, if your dataset is an array of measurements with no natural ordering of inputs (such as gene expression data), multilayer perceptrons (MLPs) may be effective. These are the most basic type of neural network, and they are able to learn complex non-linear relationships across the input data despite their relative simplicity. Similarly, if your dataset is comprised of images, convolutional neural networks (CNNs) are a good choice because they emphasize local structures and adjacency within the data. CNNs may also be a good choice for learning on sequences, as recent empirical evidence suggests that they can outperform canonical sequence learning techniques such as recurrent neural networks (RNNs) and the closely related long short-term memory (LSTM) networks [\[38\]](#).

Deep learning models typically benefit from increasing the amount of labeled data with which to train on. Large amounts of data help to avoid overfitting (see [Tip 7](#)), and increase the likelihood of achieving top performance on a given task. In the event that there is not enough data available to train your model, consider using transfer learning. In transfer learning, a model whose weights were generated by training on another dataset is used as the starting point for training [\[39\]](#). Transfer learning is most useful when the pre-training and target datasets are of similar nature [\[39\]](#). For this reason, it is important to search for similar datasets that are already available. These can potentially be used to increase the size of the training set or for pre-training and subsequent fine-tuning on the target data. However, even when this assumption does not hold, transferring features still can still improve model performance compared with random feature initialization. For example Rojkomar et al. showed advantages of ImageNet-pretraining [\[40\]](#) for a model that is applied to grayscale medical image classification [\[41\]](#). In addition, or as an alternative to pre-training models on larger datasets for transfer learning yourself, you may also be able to obtain pre-trained models from public repositories, such as Kipoi [\[42\]](#) for genomics models. Moreover, learned features can be helpful even when a pre-training task is different from a target task [\[43\]](#). Another related approach is multi-task learning, which consists of simultaneously training a network for multiple separate tasks that share features. In fact, multi-task learning can be used separately or even in combination with transfer learning [\[44\]](#).

This tip can be distilled into two main action points: first, base your network's architecture on your knowledge of the problem and, second, take advantage of similar existing data or pre-trained deep learning models.

Tip 6: Tune your hyperparameters extensively and systematically

Given at least one hidden layer, a non-linear activation function, and a large number of hidden units [\[9\]](#), multi-layer neural networks can approximate arbitrary continuous functions that relate input and output variables. Deeper architectures that feature additional hidden layers and an increasing number of overall hidden units and learnable weight parameters (the so-called increasing “capacity” of neural networks) allow for solving increasingly complex problems. However, this increased capacity results in many more parameters to tune, which can pose additional challenges during model training. In general, one should expect to systematically evaluate the impact of numerous hyperparameters when applying deep neural networks to new data or challenges. Hyperparameters typically manifest as choices of optimization algorithms, learning rate, activation functions, number of hidden layers and hidden units, size of the training batches, weight initialization schemes, and seeds for pseudo-random number generators used for dataset shuffling and weight initialization. Moreover, additional hyperparameters are introduced by common techniques that facilitate the training of deeper architectures. These include norm penalties (typically in the form of L^2 regularization), dropout [\[45\]](#), and batch normalization [\[46\]](#), which can reduce the effect of the so-called vanishing or exploding gradient problem when working with deep neural networks.

This wide array of potential parameters can make it difficult to evaluate the extent to which neural network methods are well suited to solving a task, as it can be unclear to practitioners whether

previous successful applications were the result of interactions between unique data attributes and specific parameter sets. Similar to the Continental Breakfast Included effect that we discussed in [Tip 2](#), a lack of clarity on how extensive arrays of hyperparameters were tested and/or chosen can affect methods developers as they attempt to compare techniques. This effect also has implications for those seeking to use existing deep learning methods, as performance estimates from deep neural networks are often provided after tuning. The implication of this effect on users of deep neural networks is that attaining performance numbers that match those reported in publications is likely to require significant effort towards temporally expensive hyperparameter optimization.

Ultimately, to get the best performance of your model, be sure to systematically optimize your hyperparameters on your training dataset, as introduced in the next section.

Tip 7: Address deep neural networks' increased tendency to overfit the dataset

Overfitting is a challenge inherent to machine learning in general, and is one of the most significant challenges you'll face when applying deep learning specifically. Overfitting occurs when a model fits patterns in the training data so closely that it is including non-generalizable noise or non-scientifically relevant perturbations in the relationships it is learning. In other words, the model fits patterns that are overly specific to the data it is training on rather than learning general relationships that hold across similar datasets. This subtle distinction is made clearer by seeing what happens when a model is tested on data to which it was not exposed during training: just as a student who memorizes exam materials struggles to correctly answer questions for which they have not studied, a machine learning model that has overfit to its training data will perform poorly on unseen test data. Deep learning models are particularly susceptible to overfitting due to their relatively large number of parameters and associated representational capacity. Just as some students may have greater potential for memorization, deep learning models seem more prone to overfitting than machine learning models with fewer parameters.

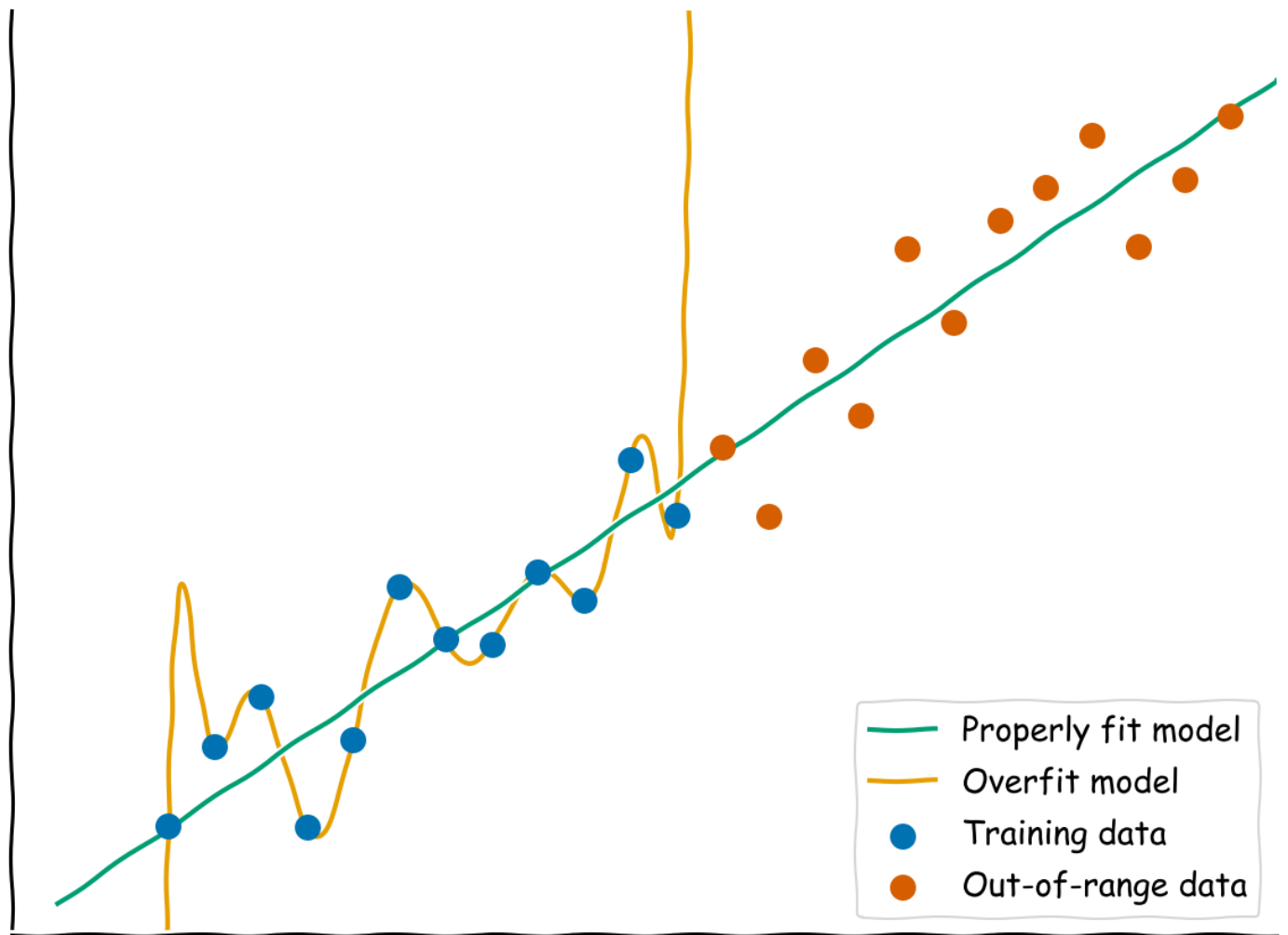


Figure 1: A visual example of overfitting and failure to generalize. While a high-degree polynomial achieves high accuracy on its training data, it performs poorly on data with specificities that have not been seen before. That is, the model has learned the training dataset specifically rather than learning a generalizable pattern that represents data of this type. In contrast, a simple linear regression works well on both datasets. The greater representational capacity of the polynomial is analogous to using a larger or deeper neural network.

In general, one of the most effective ways to combat overfitting is to detect it in the first place. One way to do this is to split the main dataset being worked on into three independent parts: a training set, a tuning set (also commonly called a validation set in the machine learning literature), and a test set. These three partitions allow us to optimize models by iterating between model learning on the training set and hyperparameter evaluation on the tuning set without affecting the final model assessment on the test set. That is, the data used for testing should be “locked away” and used only once to evaluate the final model after all training and tuning steps are completed. A researcher can then use the model’s performance on the independent test data as a measure of how overfit (i.e. non-generalizable) the model is. This type of approach is necessary for evaluating the generalizability of models without the biases that can arise from learning and testing on the same data [47,48]. While a slight drop in performance from the training set to the test set is normal, a significant drop is a clear sign of overfitting (see Figure 1 for a visual demonstration of an overfit model that performs poorly on test data).

If overfitting is an issue, there are a variety of techniques to reduce overfitting, including data augmentation and various regularization techniques [49,50]. Another way to reduce overfitting, as described by Chuang and Keiser, is to identify the baseline level of memorization that is occurring by training on data that has its labels randomly shuffled. By comparing the model performance with the shuffled data to that achieved with the actual data [51], a practitioner can identify overfitting as a model that performs no better on real data, as this suggests that any predictive capacity is not due to data-driven signal. One important caveat when working with partitioned data is the need to apply

transformation and normalization procedures equally to all datasets. The parameters required for such procedures (for example, quantile normalization, a common standardization method when analyzing gene-expression data) should only be derived from the training data, and not from the tuning or test data. Additionally, many conventional metrics for classification (e.g. area under the receiver operating characteristic curve or AUROC) have limited utility in cases of extreme class imbalance [52]. Therefore, model performance should be evaluated with a carefully picked panel of relevant metrics that make minimal assumptions about the composition of the testing data [53].

When working with biological and medical data, one must also carefully consider potential sources of bias and/or non-independence when defining training and test sets. For example, a deep learning model for pneumonia detection in chest X-rays appeared to performed well within the hospitals providing the training data, but then failed to generalize to other hospitals [54]. This resulted from the deep learning model picking up on signal related to which hospital the images were from, and represents a type of artifact or “batch effect” that practitioners must be vigilant towards. When dealing with sequence data, holding out test data that are evolutionarily related or that share structural homology to the training data can result in overfitting that is hard to detect due to the inherent relatedness of the partitioned data (cite?). In such situations, simply holding out test data selected from a random partition of the training data can be insufficient. Again, the best remedy for identifying confounding variables is to [know your data](#) and to test models on truly independent data.

In essence, practitioners should split data into training, tuning, and single-use testing sets to assess the performance of the model on data that can provide a reliable estimate of its generalization performance. Furthermore, be cognizant of the danger of skewed or biased data artificially inflating accuracy.

Tip 8: Deep learning models can be made more transparent

While model interpretability is a broad concept, in much of the machine learning literature (including in our guidelines), it refers to the ability to identify the discriminative features that influence or sway the predictions. In certain cases, the goal behind interpretation is to understand the underlying data generating processes while in other cases the goal is to understand why a model made the prediction that it did for a specific example or set of examples. Machine learning models vary widely in terms of interpretability: some are fully transparent while others are considered to be “black-boxes” that make predictions with little ability to examine why. Logistic regression and decision tree models are generally considered interpretable, while deep neural networks are often considered among the most difficult to interpret because they can have many parameters and non-linear relationships.

Knowing which of the input variables are influencing the outputted predictions, and potentially in what ways, can help with the application or extrapolation of machine learning models. This is particularly important in biomedicine, where subsequent decision making often requires human input, and where models are employed with the hope of better understanding why relationships exist in the first place. Furthermore, while prediction rules can be derived from high-throughput molecular datasets, most affordable clinical tests still rely on lower dimensional measurements of a limited number of biomarkers. Therefore, it is often still unclear how to translate the predictive capacity of deep learning models that encompassing non-linear relationships between countless input variables into clinically digestible terms. As a result, selecting which biomarkers to use for decision making remains an important modeling and interpretation challenge. In fact, many authors attribute a lower uptake of deep learning tools in healthcare to interpretability challenges [55,56]. Nonetheless, strategies to interpret both machine learning and deep learning models are rapidly emerging, and the literature on the topic is growing at an exponential rate [57]. Instead of recommending specific methods for either deep learning-specific or general-purpose model interpretation, we suggest consulting [58], which is freely available and continually updated.

While active research into model interpretability is enabling increased interpretation of models with many parameters and non-linear relationships, simpler traditional machine learning models often remain substantially easier to interpret. When deciding on a machine learning approach and model architecture, consider an interpretability versus accuracy tradeoff. A challenge in considering this tradeoff is that the extent to which one trades interpretability for accuracy depends on the problem itself. When the features provided to the model are already highly relevant to the task at hand, a simpler and more interpretable model that gives up only a little performance is often more useful. On the other hand, if features must be combined in complex ways to be meaningful for the task, the performance difference of a model capable of capturing that structure may outweigh the interpretability costs. An appropriate choice can only be made after careful consideration, which often includes estimating the performance of a simple linear model that serves as a [baseline](#). In cases where models are learned from high-throughput datasets, a small subset of features in the dataset may be strongly correlated with the complex combination of the larger feature set defined from the deep learning model. In this case, this more limited number of features can themselves be used in the subsequent simplified model to further enhance interpretability of the model. This feature reduction can be essential when defining biomarker panels for use in clinical applications.

Tip 9: Don't over-interpret predictions

Once we have trained an accurate deep learning model, we often want to use it to deduce relationships and inform scientific findings. However, in doing this, we need to be careful to correctly interpret the model's predictions. Given that deep learning models can be difficult to interpret intuitively, there is often a temptation to overinterpret the predictions in indulgent and/or inaccurate ways. As the classic statistical saying "correlation doesn't mean causation" implies, predictions by deep learning models don't necessarily speak to certain causal relationships. While we generally know this, and understand that accurately predicting an outcome doesn't imply the learning of any causal mechanism, it can be easy to forget this lesson when the predictions are extremely accurate. A poignant example of this lesson is from work where authors evaluated the capacities of several models to predict the probability of death for patients with pneumonia admitted to an intensive care unit [59,60]. Unsurprisingly, the neural network model achieved the best predictive accuracy. However, after fitting a rule-based model in order to better understand the relationships inherent to their data, the authors discovered that the hospital data implied the rule `HasAsthma(x) => LowerRisk(x)`. This rule contradicts medical understanding, as having asthma doesn't make pneumonia better! Nonetheless, this rule was supported by the data, as pneumonia patients with a history of asthma tended to receive more aggressive care. The neural network had therefore also learned to make predictions according to this rule despite the fact that it has nothing to do with causality or mechanism. Guiding treatment decisions according to the predictions of the neural network would have been disastrous, even though the neural network had high predictive accuracy.

To trust deep learning models, we must combine knowledge of the training data ([Tip 4](#)) with inspection of the model ([Tip 8](#)). To move beyond fitting predictive models and towards the building of an understanding that can inform scientific deduction, we suggest working to disentangle a model's internal logic by comparing data domains where models succeeds to those in which they fail. By doing so, we can avoid overinterpreting models and view them for what they are: complex statistical models trained on high dimensional data.

Tip 10: Don't share models trained on sensitive data

Practitioners may encounter datasets that cannot be shared, such as ones for which there would be significant ethical or legal issues associated with release [61]. Examples of such data include classified or confidential data, biological data related to trade secrets, and medical records or other personally identifiable information [62]. While deep learning models can capture information-rich abstractions of

multiple features of the data during the training process (which represents one of its great strengths), these features may be more prone to leak the data that they were trained over if the model is shared or allowed to be queried with arbitrary inputs [63,64]. In other words, the complex relationships learned about the input data can potentially be used to infer characteristics about the original dataset. This means that the strengths that imbue deep learnings with its great predictive capacity also raise the level of risk surrounding data privacy. Therefore, while there is tremendous promise for deep learning techniques to extract information that cannot readily be captured by traditional methods [65], it is imperative not to share models trained on sensitive data. This also holds true for certain traditional machine learning methods that learn by capturing specific details of the full training data (for example, k -nearest neighbors models).

Techniques to train deep neural networks without sharing unencrypted access to data are being advanced through implementations of homomorphic encryption, which serves to enable equivalent prediction on data that is encrypted end to end [66,67]. Privacy preserving techniques [68], such as differential privacy [69,70,71], can help to mitigate risks as long as the assumptions underlying these techniques are met. These methods provide a path towards a future where trained models and their predictions can be shared, but more software development and theoretical advances will be required to make these techniques easy to apply correctly in many settings. Unless you use these techniques, don't share the weights or arbitrary access to the predictions of models trained on sensitive data.

Conclusion

Deep learning techniques have the potential for wide use in biology and the capacity to meet or exceed the performance of both humans and the current state-of-the art algorithms across a wide range of tasks. Beyond simply achieving good predictive performance, deep learning has the potential to inform novel biological insights to fundamentally drive high value research. To realize this potential, the use of deep learning as a research tool must be approached as any other tool would be: scientifically and thoughtfully. We hope that our tips for applying deep learning to the biological sciences will serve as a starting point for discussion and not as an ending point.

Acknowledgements

The authors would like to thank Daniel Himmelstein and the developers of Manubot for creating the software that enabled the collaborative composition of this manuscript. We would also like to thank **[TODO: insert the names of the contributors who don't meet the standards for authorship]** for their contributions to the discussions that comprised the initial stage of the drafting process.

References

1. Opportunities and obstacles for deep learning in biology and medicine

Travers Ching, Daniel S. Himmelstein, Brett K. Beaulieu-Jones, Alexandr A. Kalinin, Brian T. Do, Gregory P. Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M. Hoffman, ... Casey S. Greene

Journal of The Royal Society Interface (2018-04-04) <https://doi.org/gddkhn>

DOI: [10.1098/rsif.2017.0387](https://doi.org/10.1098/rsif.2017.0387) · PMID: [29618526](https://pubmed.ncbi.nlm.nih.gov/29618526/) · PMCID: [PMC5938574](https://pubmed.ncbi.nlm.nih.gov/PMC5938574/)

2. VAMPnets for deep learning of molecular kinetics

Andreas Mardt, Luca Pasquali, Hao Wu, Frank Noé

Nature Communications (2018-01-02) <https://doi.org/gcvf62>

DOI: [10.1038/s41467-017-02388-1](https://doi.org/10.1038/s41467-017-02388-1) · PMID: [29295994](https://pubmed.ncbi.nlm.nih.gov/29295994/) · PMCID: [PMC5750224](https://pubmed.ncbi.nlm.nih.gov/PMC5750224/)

3. Deep learning to predict the lab-of-origin of engineered DNA

Alec A. K. Nielsen, Christopher A. Voigt

Nature Communications (2018-08-07) <https://doi.org/gd27sw>

DOI: [10.1038/s41467-018-05378-z](https://doi.org/10.1038/s41467-018-05378-z) · PMID: [30087331](https://pubmed.ncbi.nlm.nih.gov/30087331/) · PMCID: [PMC6081423](https://pubmed.ncbi.nlm.nih.gov/PMC6081423/)

4. Identifying facial phenotypes of genetic disorders using deep learning

Yaron Gurovich, Yair Hanani, Omri Bar, Guy Nadav, Nicole Fleischer, Dekel Gelbman, Lina Basel-Salmon, Peter M. Krawitz, Susanne B. Kamphausen, Martin Zenker, ... Karen W. Gripp

Nature Medicine (2019-01-07) <https://doi.org/czdm>

DOI: [10.1038/s41591-018-0279-0](https://doi.org/10.1038/s41591-018-0279-0) · PMID: [30617323](https://pubmed.ncbi.nlm.nih.gov/30617323/)

5. Benjamin-Lee/deep-rules GitHub repository

Benjamin Lee

GitHub (2018) <https://github.com/Benjamin-Lee/deep-rules>

6. Open collaborative writing with Manubot

Daniel S. Himmelstein, Vincent Rubinetti, David R. Slochower, Dongbo Hu, Venkat S. Malladi, Casey S. Greene, Anthony Gitter

Manubot (2020-05-25) <https://greenelab.github.io/meta-review/>

7. Ten quick tips for machine learning in computational biology

Davide Chicco

BioData Mining (2017-12-08) <https://doi.org/gdb9wr>

DOI: [10.1186/s13040-017-0155-3](https://doi.org/10.1186/s13040-017-0155-3) · PMID: [29234465](https://pubmed.ncbi.nlm.nih.gov/29234465/) · PMCID: [PMC5721660](https://pubmed.ncbi.nlm.nih.gov/PMC5721660/)

8. Approximation by superpositions of a sigmoidal function

G. Cybenko

Mathematics of Control, Signals, and Systems (1989-12) <https://doi.org/dp3968>

DOI: [10.1007/bf02551274](https://doi.org/10.1007/bf02551274)

9. Approximation capabilities of multilayer feedforward networks

Kurt Hornik

Neural Networks (1991) <https://doi.org/dzwxkd>

DOI: [10.1016/0893-6080\(91\)90009-t](https://doi.org/10.1016/0893-6080(91)90009-t)

10. Data Programming: Creating Large Training Sets, Quickly

Alexander Ratner, Christopher De Sa, Sen Wu, Daniel Selsam, Christopher Ré

arXiv (2016-05-25) <https://arxiv.org/abs/1605.07723v3>

11. **How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?**
Junghwan Cho, Kyewook Lee, Ellie Shin, Garry Choy, Synho Do
arXiv (2016-01-11) <https://arxiv.org/abs/1511.06348>
12. **Efficient Processing of Deep Neural Networks: A Tutorial and Survey**
Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, Joel S. Emer
Proceedings of the IEEE (2017-12) <https://doi.org/gcnp38>
DOI: [10.1109/jproc.2017.2761740](https://doi.org/10.1109/jproc.2017.2761740)
13. **Language Models are Few-Shot Learners**
Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, ... Dario Amodei
arXiv (2020-07-24) <https://arxiv.org/abs/2005.14165>
14. **Energy and Policy Considerations for Deep Learning in NLP**
Emma Strubell, Ananya Ganesh, Andrew McCallum
arXiv (2019-06-07) <https://arxiv.org/abs/1906.02243>
15. **A Machine Learning Driven IoT Solution for Noise Classification in Smart Cities**
Yasser Alsouda, Sabri Pillana, Arianit Kurti
arXiv (2018-09-05) <https://arxiv.org/abs/1809.00238>
16. **TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems**
Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, ... Xiaoqiang Zheng
arXiv (2016-03-17) <https://arxiv.org/abs/1603.04467>
17. **PyTorch: An Imperative Style, High-Performance Deep Learning Library**
Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, ... Soumith Chintala
arXiv (2019-12-05) <https://arxiv.org/abs/1912.01703>
18. **Automating Biomedical Data Science Through Tree-Based Pipeline Optimization**
Randal S. Olson, Ryan J. Urbanowicz, Peter C. Andrews, Nicole A. Lavender, La Creis Kidd, Jason H. Moore
Lecture Notes in Computer Science (2016) <https://doi.org/ggfptv>
DOI: [10.1007/978-3-319-31204-0_9](https://doi.org/10.1007/978-3-319-31204-0_9)
19. **apple/turicreate**
Apple
(2020-10-24) <https://github.com/apple/turicreate>
20. **Auto-Keras: An Efficient Neural Architecture Search System**
Haifeng Jin, Qingquan Song, Xia Hu
arXiv (2019-03-27) <https://arxiv.org/abs/1806.10282>
21. **Keras: the Python deep learning API** <https://keras.io/>
22. **ImageNet classification with deep convolutional neural networks**
Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton
Communications of the ACM (2017-05-24) <https://doi.org/gbhxhs>
DOI: [10.1145/3065386](https://doi.org/10.1145/3065386)

23. **Deep learning-based methods for individual recognition in small birds**
André C. Ferreira, Liliana R. Silva, Francesco Renna, Hanja B. Brandl, Julien P. Renoult, Damien R. Farine, Rita Covas, Claire Doutrelant
Methods in Ecology and Evolution (2020-07-26) <https://doi.org/d438>
DOI: [10.1111/2041-210x.13436](https://doi.org/10.1111/2041-210x.13436)
24. **Scalable and accurate deep learning with electronic health records**
Alvin Rajkomar, Eyal Oren, Kai Chen, Andrew M. Dai, Nissan Hajaj, Michaela Hardt, Peter J. Liu, Xiaobing Liu, Jake Marcus, Mimi Sun, ... Jeffrey Dean
npj Digital Medicine (2018-05-08) <https://doi.org/gdqcc8>
DOI: [10.1038/s41746-018-0029-1](https://doi.org/10.1038/s41746-018-0029-1) · PMID: [31304302](https://pubmed.ncbi.nlm.nih.gov/31304302/) · PMCID: [PMC6550175](https://pubmed.ncbi.nlm.nih.gov/PMC6550175/)
25. **Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data**
Alexios Koutsoukas, Keith J. Monaghan, Xiaoli Li, Jun Huan
Journal of Cheminformatics (2017-06-28) <https://doi.org/gfwv4d>
DOI: [10.1186/s13321-017-0226-y](https://doi.org/10.1186/s13321-017-0226-y) · PMID: [29086090](https://pubmed.ncbi.nlm.nih.gov/29086090/) · PMCID: [PMC5489441](https://pubmed.ncbi.nlm.nih.gov/PMC5489441/)
26. **Deep learning and alternative learning strategies for retrospective real-world clinical data**
David Chen, Sijia Liu, Paul Kingsbury, Sunghwan Sohn, Curtis B. Storlie, Elizabeth B. Habermann, James M. Naessens, David W. Larson, Hongfang Liu
npj Digital Medicine (2019-05-30) <https://doi.org/ghfwvh>
DOI: [10.1038/s41746-019-0122-0](https://doi.org/10.1038/s41746-019-0122-0) · PMID: [31304389](https://pubmed.ncbi.nlm.nih.gov/31304389/) · PMCID: [PMC6550223](https://pubmed.ncbi.nlm.nih.gov/PMC6550223/)
27. **Deep k-Nearest Neighbors: Towards Confident, Interpretable and Robust Deep Learning**
Nicolas Papernot, Patrick McDaniel
arXiv (2018-03-14) <https://arxiv.org/abs/1803.04765>
28. **To Trust Or Not To Trust A Classifier**
Heinrich Jiang, Been Kim, Melody Y. Guan, Maya Gupta
arXiv (2018-10-30) <https://arxiv.org/abs/1805.11783>
29. **Parameter tuning is a key part of dimensionality reduction via deep variational autoencoders for single cell RNA transcriptomics**
Qiwen Hu, Casey S. Greene
World Scientific Pub Co Pte Lt (2018-11) <https://doi.org/gf5pc7>
DOI: [10.1142/9789813279827_0033](https://doi.org/10.1142/9789813279827_0033)
30. **Ten Simple Rules for Taking Advantage of Git and GitHub**
Yasset Perez-Riverol, Laurent Gatto, Rui Wang, Timo Sachsenberg, Julian Uszkoreit, Felipe da Veiga Leprevost, Christian Fufezan, Tobias Ternent, Stephen J. Eglén, Daniel S. Katz, ... Juan Antonio Vizcaíno
PLOS Computational Biology (2016-07-14) <https://doi.org/gbrb39>
DOI: [10.1371/journal.pcbi.1004947](https://doi.org/10.1371/journal.pcbi.1004947) · PMID: [27415786](https://pubmed.ncbi.nlm.nih.gov/27415786/) · PMCID: [PMC4945047](https://pubmed.ncbi.nlm.nih.gov/PMC4945047/)
31. **Ten Simple Rules for Reproducible Computational Research**
Geir Kjetil Sandve, Anton Nekrutenko, James Taylor, Eivind Hovig
PLoS Computational Biology (2013-10-24) <https://doi.org/pjbj>
DOI: [10.1371/journal.pcbi.1003285](https://doi.org/10.1371/journal.pcbi.1003285) · PMID: [24204232](https://pubmed.ncbi.nlm.nih.gov/24204232/) · PMCID: [PMC3812051](https://pubmed.ncbi.nlm.nih.gov/PMC3812051/)
32. **Ten Simple Rules for Reproducible Research in Jupyter Notebooks**
Adam Rule, Amanda Birmingham, Cristal Zuniga, Ilkay Altintas, Shih-Cheng Huang, Rob Knight,

Niema Moshiri, Mai H. Nguyen, Sara Brin Rosenthal, Fernando Pérez, Peter W. Rose
arXiv (2018-10-19) <https://arxiv.org/abs/1810.08055>

33. Deep Learning SDK Documentation

NVIDIA

(2018-11-01) <https://docs.nvidia.com/deeplearning/sdk/cudnn-developer-guide/index.html#reproducibility>

34. The Marginal Value of Adaptive Gradient Methods in Machine Learning

Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, Benjamin Recht

Advances in Neural Information Processing Systems 30 (2017) <http://papers.nips.cc/paper/7003-the-marginal-value-of-adaptive-gradient-methods-in-machine-learning.pdf>

35. Minimum information about a microarray experiment (MIAME)—toward standards for microarray data

Alvis Brazma, Pascal Hingamp, John Quackenbush, Gavin Sherlock, Paul Spellman, Chris Stoeckert, John Aach, Wilhelm Ansorge, Catherine A. Ball, Helen C. Causton, ... Martin Vingron

Nature Genetics (2001-12) <https://doi.org/ck257n>

DOI: [10.1038/ng1201-365](https://doi.org/10.1038/ng1201-365) · PMID: [11726920](https://pubmed.ncbi.nlm.nih.gov/11726920/)

36. Tackling the widespread and critical impact of batch effects in high-throughput data

Jeffrey T. Leek, Robert B. Scharpf, Héctor Corrada Bravo, David Simcha, Benjamin Langmead, W. Evan Johnson, Donald Geman, Keith Baggerly, Rafael A. Irizarry

Nature Reviews Genetics (2010-09-14) <https://doi.org/cfr324>

DOI: [10.1038/nrg2825](https://doi.org/10.1038/nrg2825) · PMID: [20838408](https://pubmed.ncbi.nlm.nih.gov/20838408/) · PMCID: [PMC3880143](https://pubmed.ncbi.nlm.nih.gov/PMC3880143/)

37. Neural Networks: Tricks of the Trade

Lecture Notes in Computer Science

(2012) <https://doi.org/gfvtvt>

DOI: [10.1007/978-3-642-35289-8](https://doi.org/10.1007/978-3-642-35289-8)

38. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling

Shaojie Bai, J. Zico Kolter, Vladlen Koltun

arXiv (2018-04-20) <https://arxiv.org/abs/1803.01271>

39. How transferable are features in deep neural networks?

Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson

Advances in Neural Information Processing Systems 27 (2014) <http://papers.nips.cc/paper/5347-how-transferable-are-features-in-deep-neural-networks.pdf>

40. ImageNet Large Scale Visual Recognition Challenge

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, ... Li Fei-Fei

International Journal of Computer Vision (2015-04-11) <https://doi.org/gcgk7w>

DOI: [10.1007/s11263-015-0816-y](https://doi.org/10.1007/s11263-015-0816-y)

41. High-Throughput Classification of Radiographs Using Deep Convolutional Neural Networks

Alvin Rajkomar, Sneha Lingam, Andrew G. Taylor, Michael Blum, John Mongan

Journal of Digital Imaging (2016-10-11) <https://doi.org/gcgk7v>

DOI: [10.1007/s10278-016-9914-9](https://doi.org/10.1007/s10278-016-9914-9) · PMID: [27730417](https://pubmed.ncbi.nlm.nih.gov/27730417/) · PMCID: [PMC5267603](https://pubmed.ncbi.nlm.nih.gov/PMC5267603/)

42. **Kipoi: accelerating the community exchange and reuse of predictive models for genomics**
Žiga Avsec, Roman Kreuzhuber, Johnny Israeli, Nancy Xu, Jun Cheng, Avanti Shrikumar, Abhimanyu Banerjee, Daniel S. Kim, Lara Urban, Anshul Kundaje, ... Julien Gagneur
bioRxiv (2018-07-24) <https://doi.org/gd24sx>
DOI: [10.1101/375345](https://doi.org/10.1101/375345)
43. **CNN Features Off-the-Shelf: An Astounding Baseline for Recognition**
Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, Stefan Carlsson
Institute of Electrical and Electronics Engineers (IEEE) (2014-06) <https://doi.org/f3np4s>
DOI: [10.1109/cvprw.2014.131](https://doi.org/10.1109/cvprw.2014.131)
44. **Deep Model Based Transfer and Multi-Task Learning for Biological Image Analysis**
Wenlu Zhang, Rongjian Li, Tao Zeng, Qian Sun, Sudhir Kumar, Jieping Ye, Shuiwang Ji
IEEE Transactions on Big Data (2020-06-01) <https://doi.org/gfvs28>
DOI: [10.1109/tbdata.2016.2573280](https://doi.org/10.1109/tbdata.2016.2573280)
45. **Dropout: a simple way to prevent neural networks from overfitting**
Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov
The Journal of Machine Learning Research (2014-01-01)
46. **Batch normalization: accelerating deep network training by reducing internal covariate shift**
Sergey Ioffe, Christian Szegedy
Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37 (2015-07-06)
47. **Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning**
Sebastian Raschka
arXiv (2018-12-04) <https://arxiv.org/abs/1811.12808>
48. **Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms**
Thomas G. Dietterich
Neural Computation (1998-10) <https://doi.org/fqc9w5>
DOI: [10.1162/089976698300017197](https://doi.org/10.1162/089976698300017197) · PMID: [9744903](https://pubmed.ncbi.nlm.nih.gov/9744903/)
49. **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>
50. **A simple weight decay can improve generalization**
Anders Krogh, John A. Hertz
Proceedings of the 4th International Conference on Neural Information Processing Systems (1991-12-02)
ISBN: [9781558602229](https://doi.org/9781558602229)
51. **Adversarial Controls for Scientific Machine Learning**
Kangway V. Chuang, Michael J. Keiser
ACS Chemical Biology (2018-10-19) <https://doi.org/gfk9mh>
DOI: [10.1021/acscchembio.8b00881](https://doi.org/10.1021/acscchembio.8b00881) · PMID: [30336670](https://pubmed.ncbi.nlm.nih.gov/30336670/)
52. **The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets.**
Takaya Saito, Marc Rehmsmeier

PloS one (2015-03-04) <https://www.ncbi.nlm.nih.gov/pubmed/25738806>
DOI: [10.1371/journal.pone.0118432](https://doi.org/10.1371/journal.pone.0118432) · PMID: [25738806](https://pubmed.ncbi.nlm.nih.gov/25738806/) · PMCID: [PMC4349800](https://pubmed.ncbi.nlm.nih.gov/PMC4349800/)

53. Comparison of Deep Learning With Multiple Machine Learning Methods and Metrics Using Diverse Drug Discovery Data Sets

Alexandru Korotcov, Valery Tkachenko, Daniel P. Russo, Sean Ekins

Molecular Pharmaceutics (2017-11-13) <https://doi.org/gcj4p2>

DOI: [10.1021/acs.molpharmaceut.7b00578](https://doi.org/10.1021/acs.molpharmaceut.7b00578) · PMID: [29096442](https://pubmed.ncbi.nlm.nih.gov/29096442/) · PMCID: [PMC5741413](https://pubmed.ncbi.nlm.nih.gov/PMC5741413/)

54. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study

John R. Zech, Marcus A. Badgeley, Manway Liu, Anthony B. Costa, Joseph J. Titano, Eric Karl Oermann

PLOS Medicine (2018-11-06) <https://doi.org/gfj53h>

DOI: [10.1371/journal.pmed.1002683](https://doi.org/10.1371/journal.pmed.1002683) · PMID: [30399157](https://pubmed.ncbi.nlm.nih.gov/30399157/) · PMCID: [PMC6219764](https://pubmed.ncbi.nlm.nih.gov/PMC6219764/)

55. Deep Learning for Health Informatics

Daniele Ravi, Charence Wong, Fani Deligianni, Melissa Berthelot, Javier Andreu-Perez, Benny Lo, Guang-Zhong Yang

IEEE Journal of Biomedical and Health Informatics (2017-01) <https://doi.org/gfgtzt>

DOI: [10.1109/jbhi.2016.2636665](https://doi.org/10.1109/jbhi.2016.2636665) · PMID: [28055930](https://pubmed.ncbi.nlm.nih.gov/28055930/)

56. Towards trustable machine learning

Nature Biomedical Engineering

(2018-10-10) <https://doi.org/gfw9cn>

DOI: [10.1038/s41551-018-0315-x](https://doi.org/10.1038/s41551-018-0315-x) · PMID: [31015650](https://pubmed.ncbi.nlm.nih.gov/31015650/)

57. On Interpretability of Artificial Neural Networks

Fenglei Fan, Jinjun Xiong, Ge Wang

arXiv (2020-01-09) <https://arxiv.org/abs/2001.02522>

58. Interpretable Machine Learning

Christoph Molnar

<https://christophm.github.io/interpretable-ml-book/>

59. An evaluation of machine-learning methods for predicting pneumonia mortality

Gregory F. Cooper, Constantin F. Aliferis, Richard Ambrosino, John Aronis, Bruce G. Buchanan, Richard Caruana, Michael J. Fine, Clark Glymour, Geoffrey Gordon, Barbara H. Hanusa, ... Peter Spirtes

Artificial Intelligence in Medicine (1997-02) <https://doi.org/b6vnmd>

DOI: [10.1016/s0933-3657\(96\)00367-3](https://doi.org/10.1016/s0933-3657(96)00367-3)

60. Intelligible Models for HealthCare

Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, Noemie Elhadad

Association for Computing Machinery (ACM) (2015) <https://doi.org/gftgxx>

DOI: [10.1145/2783258.2788613](https://doi.org/10.1145/2783258.2788613)

61. Ten simple rules for responsible big data research

Matthew Zook, Solon Barocas, danah boyd, Kate Crawford, Emily Keller, Seeta Peña Gangadharan, Alyssa Goodman, Rachelle Hollander, Barbara A. Koenig, Jacob Metcalf, ... Frank Pasquale

PLOS Computational Biology (2017-03-30) <https://doi.org/gdqfcx>

DOI: [10.1371/journal.pcbi.1005399](https://doi.org/10.1371/journal.pcbi.1005399) · PMID: [28358831](https://pubmed.ncbi.nlm.nih.gov/28358831/) · PMCID: [PMC5373508](https://pubmed.ncbi.nlm.nih.gov/PMC5373508/)

62. **Responsible, practical genomic data sharing that accelerates research**
James Brian Byrd, Anna C. Greene, Deepashree Venkatesh Prasad, Xiaoqian Jiang, Casey S. Greene
Nature Reviews Genetics (2020-07-21) <https://doi.org/gg7c57>
DOI: [10.1038/s41576-020-0257-5](https://doi.org/10.1038/s41576-020-0257-5) · PMID: [32694666](https://pubmed.ncbi.nlm.nih.gov/32694666/)
63. **Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures**
Matt Fredrikson, Somesh Jha, Thomas Ristenpart
Association for Computing Machinery (ACM) (2015) <https://doi.org/cwdm>
DOI: [10.1145/2810103.2813677](https://doi.org/10.1145/2810103.2813677)
64. **Membership Inference Attacks against Machine Learning Models**
Reza Shokri, Marco Stronati, Congzheng Song, Vitaly Shmatikov
arXiv (2017-04-04) <https://arxiv.org/abs/1610.05820>
65. **Convolutional Networks on Graphs for Learning Molecular Fingerprints**
David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, Ryan P. Adams
arXiv (2015-11-04) <https://arxiv.org/abs/1509.09292>
66. **SIG-DB: Leveraging homomorphic encryption to securely interrogate privately held genomic databases**
Alexander J. Titus, Audrey Flower, Patrick Hagerty, Paul Gamble, Charlie Lewis, Todd Stavish, Kevin P. O'Connell, Greg Shipley, Stephanie M. Rogers
PLOS Computational Biology (2018-09-04) <https://doi.org/gd6xd5>
DOI: [10.1371/journal.pcbi.1006454](https://doi.org/10.1371/journal.pcbi.1006454) · PMID: [30180163](https://pubmed.ncbi.nlm.nih.gov/30180163/) · PMCID: [PMC6138421](https://pubmed.ncbi.nlm.nih.gov/PMC6138421/)
67. **Towards the AlexNet Moment for Homomorphic Encryption: HCNN, the First Homomorphic CNN on Encrypted Data with GPUs**
Ahmad Al Badawi, Jin Chao, Jie Lin, Chan Fook Mun, Jun Jie Sim, Benjamin Hong Meng Tan, Xiao Nan, Khin Mi Mi Aung, Vijay Ramaseshan Chandrasekhar
arXiv (2020-08-20) <https://arxiv.org/abs/1811.00778>
68. **A generic framework for privacy preserving deep learning**
Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, Jonathan Passerat-Palmbach
arXiv (2018-11-14) <https://arxiv.org/abs/1811.04017>
69. **Deep Learning with Differential Privacy**
Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, Li Zhang
Association for Computing Machinery (ACM) (2016) <https://doi.org/gcrnp3>
DOI: [10.1145/2976749.2978318](https://doi.org/10.1145/2976749.2978318)
70. **Privacy-preserving generative deep neural networks support clinical data sharing**
Brett K. Beaulieu-Jones, Zhiwei Steven Wu, Chris Williams, Ran Lee, Sanjeev P. Bhavnani, James Brian Byrd, Casey S. Greene
bioRxiv (2018-12-20) <https://doi.org/gcnzrn>
DOI: [10.1101/159756](https://doi.org/10.1101/159756)
71. **Privacy-Preserving Distributed Deep Learning for Clinical Data**
Brett K. Beaulieu-Jones, William Yuan, Samuel G. Finlayson, Zhiwei Steven Wu
arXiv (2018-12-05) <https://arxiv.org/abs/1812.01484>