Ten Quick Tips for Deep Learning in Biology

This manuscript (permalink) was automatically generated from Benjamin-Lee/deep-rules@ae758e8 on October 2, 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

Introduction

Deep learning (DL) is a subfield of machine learning (ML) focusing on artificial neural networks with many layers, which are increasingly used for the analysis of biological data [1]. In many cases, novel biological insights have been revealed through careful evaluation of DL methods ranging from predicting protein-drug binding kinetics [2] to identifying the lab-of-origin of synthetic DNA [3]. However, for researchers and students entirely new to this area and those experienced in using classical ML methods (*e.g.* linear regression), using DL correctly can be a daunting task. The lack of concise recommendations for biological applications of DL poses further a challenge for newcomers wishing to apply state-of-the-art DL in their research. As DL is an active and specialized research area, detailed resources are rapidly rendered obsolete, and only few resources articulate general DL best practices to the scientific community broadly and the biological community specifically. To address this issue, we solicited input from a community of researchers with varied biological and deep learning interests, who wrote this manuscript collaboratively using the GitHub version control platform [4] and Manubot [5].

In the course of our discussions, several themes became clear: the importance of understanding and applying ML fundamentals [6] as a baseline for utilizing DL, the necessity for extensive model comparisons with careful evaluation, and the need for critical thought in interpreting results generated by means of DL, among others. Ultimately, the tips we collate range from high-level guidance to the implementation of best practices, and it is our hope that they will provide actionable, DL-specific advice for both new and experienced DL practitioners alike who would like to employ DL in biological research. By increasing the accessibility of DL for applications in biological research, we aim

to improve the overall quality and reporting of DL in the literature, enabling more researchers to utilize these state-of-the art modeling techniques.

Tip 1: Concepts that apply to machine learning also apply to deep learning

Deep learning is a distinct subfield of machine learning, but it is still a subfield. DL has proven to be an extremely powerful paradigm capable of outperforming "traditional" machine learning approaches in certain contexts, but it is not immune to the many limitations inherent to machine learning. Many best practices for machine learning also apply to deep learning. Like all computational methods, deep learning should be applied in a systematic manner that is reproducible and rigorously tested.

Those developing deep learning models should select datasets to train and test model performance that are relevant to the problem at hand; non-salient data can hamper performance or lead to spurious conclusions. For example, supervised deep learning for phenotype prediction should be applied to datasets that contain large numbers of representative samples from all phenotypes to be predicted. Biases in testing data can also unduly influence measures of model performance, and it may be difficult to directly identify confounders from the model. Investigators should consider the extent to which the outcome of interest is likely to be predictable from the input data and begin by thoroughly inspecting the input data. Suppose that there are robust heritability estimates for a phenotype that suggest that the genetic contribution is modest but a deep learning model predicts the phenotype with very high accuracy. The model may be capturing signal unrelated to genetic mechanisms underlying the phenotype. In this case, a possible explanation is that people with similar genetic markers may have shared exposures. This is something that researchers should probe before reporting unrealistic accuracy measures. A similar situation can arise with tasks for which inter-rater reliability is modest but deep learning models produce very high accuracies. When coupled with imprudence, datasets that are confounded, biased, skewed, or of low quality will produce models of dubious performance and limited generalizability. Data exploration with unsupervised learning and data visualization can reveal the biases and technical artifacts in these datasets, providing a critical first step to assessing data quality before any deep learning model is applied. In some cases, these analyses can identify biases from known technical artifacts or sample processing which can be corrected through preprocessing techniques to support more accurate application of deep leaning models for subsequent prediction or feature identifaction problems from those datasets.

Using a test set more than once will lead to biased estimates of the generalization performance [7,8]. Deep supervised learning models should be trained, tuned, and tested on non-overlapping datasets. The data used for testing should be locked and only used one-time for evaluating the final model after all tuning steps are completed. Also, 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 [9]. Model performance should be evaluated with a carefully picked panel of relevant metrics that make minimal assumptions about the composition of the testing data [10], with particular consideration given to metrics that are most directly applicable to the task at hand.

In summary, if you are not familiar with machine learning, review a general machine learning guide such as [6] before diving right into deep learning.

Tip 2: Use traditional methods to establish performance baselines

Since deep learning requires practitioners to consider a larger number and variety of tuning parameters or algorithm settings (so-called hyperparameters) compared to more traditional methods, it is 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 simple model at the beginning of each study to establish adequate performance baselines. While performance

baselines available from existing literature also serve as a helpful guidance and should be taken into account, an implementation of a simple model (for example, linear or logistic regression) using the same software framework planned for DL is additionally helpful for assessing the correctness of data processing and performance evaluation pipelines. Examples of simple models include logistic regression, random forests, k-nearest neighbors, naive Bayes, and support vector machines. Beyond serving as a predictive performance baseline, an implementation of a simple model can also provide guidance for estimating computational performance and and resource requirements. Furthermore, in some cases, it can also be useful to combine simple baseline model with deep neural networks. Such hybrid models that combine DL and simpler models can improve generalization performance, model interpretability, and confidence estimation [11,12]. In addition, be sure to tune and compare current state-of-the-art tools (*e.g.* bioinformatics pipelines or image analysis workflows), regardless of whether they use ML, in order to gauge the relative effectiveness of your baseline and DL models.

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. [13] found that simpler baseline models achieved performance comparable with that of DL 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 [14]. 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.

It is worth noting that conventional off-the-shelf machine learning algorithms (e.g., support vector machines and random forests) are also likely to benefit from hyperparameter tuning. It can be tempting to train baseline models with these conventional methods using default settings, which may provide acceptable but not stellar performance, but then tune the settings for DL algorithms to further optimize performance. Hu and Greene [15] discuss a “Continental Breakfast Included” effect by which unequal hyperparameter tuning for different learning algorithms skews the evaluation of these methods, especially when the performance of an algorithm varies substantially with modest changes to its hyperparameters. Those wishing to compare different learning algorithms should tune the settings of both traditional and DL-based methods to optimize performance before making claims about relative performance differences. The performance comparison among DL models and many other ML approaches is informative only when the models are similarly well tuned.

To sum this tip up, create and fully tune several traditional models and standard pipelines before implementing a DL model.

Tip 3: Understand the complexities of training deep neural networks

Correctly training deep neural networks is a non-trivial process. There are many different options and potential pitfalls at every stage. To get good results, you must expect to train many networks with a range of different parameter and hyperparameter settings. Deep learning can be very demanding, often requiring extensive computing infrastructure and patience to achieve state-of-the-art performance [16]. The experimentation inherent to DL is often noisy (requiring repetition) and represents a significant organizational challenge. All code, random seeds, parameters, and results must be carefully corralled using general good coding practices (for example, version control [17], continuous integration etc.) in order to be effective and interpretable. This organization is also key to being able to efficiently share and reproduce your work [18,19] as well as to update your model as new data becomes available.

One specific reproducibility pitfall that is often missed in deep learning applications is the default use of non-deterministic algorithms by CUDA/CuDNN backends when using GPUs. Making this process

reproducible is distinct from setting random seeds, which will primarily affect pseudorandom deterministic procedures such as shuffling and initialization, and requires explicitly specifying the use of deterministic algorithms in your DL library [20].

Similar to Tip 4, try to start with a relatively small network and increase the size and complexity as needed to prevent wasting time and resources. Beware of the seemingly trivial choices that are being made implicitly by default settings in your framework of choice e.g. choice of optimization algorithm (adaptive methods often lead to faster convergence during training but may lead to worse generalization performance on independent datasets [21]). These need to be carefully considered and their impacts evaluated (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 step foot in a laboratory and begin experiments without having a defined endpoint, a deep learning project should not be undertaken without preparation. Foremost, it is important to assess if a dataset exists that can answer the biological question of interest for the given deep learning model; obtaining said data and associated metadata and reviewing the study protocol should be pursued as early on in the project as possible. A publication or resource might purportedly offer a dataset that seems to be a good fit to test your hypothesis, but the act of obtaining it can reveal numerous problems. It may be unstructured when it is supposed to be structured, crucial metadata such as sample stratification are missing, or the usable sample size is different than what is reported. Data collection should be documented or a data collection protocol should be created and specified in the project documentation. Information such as the resource used, the date downloaded, and the version of the dataset, if any, will help minimize operational confusion and will allow for transparency during the publication process.

Once the dataset is obtained, it is easy to begin analysis without a good understanding of the study design, namely why the data was collected and how. Metadata has been standardized in many fields and can help with this (for example, see [22]), but if at all possible, seek out a subject matter expert who has experience with this type of data. Receiving first-hand knowledge of the “gotchas” of a dataset will minimize the amount of guesswork and increase the success rate of a deep learning project. For example, if the main reason why the data was collected was to test the impact of an intervention, then it may be the case that a randomized controlled trial was performed. However, it is not always possible to perform a randomized trial for ethical or practical reasons. Therefore, an observational study design is often considered, with the data either prospectively or retrospectively collected. In order to ensure similar distributions of important characteristics across study groups in the absence of randomization, individuals may be matched based on age, gender, or weight. Study designs will often have different assumptions and caveats, and these cannot be ignored during a data analysis. Many datasets are now passively collected or do not have a specific design, but even in this case it is important to know how individuals or samples were treated. Samples originating from the same study site, oversampling of ethnic groups or zip codes, and sample processing differences are all sources of variation that need to be accounted for.

Systematic biases, which can be induced by confounding variables, for example, can lead to artifacts or so-called “batch effects.” As a consequence, models may learn to rely on correlations that are irrelevant in the scientific context of the study and may result in misguided predictions and misleading conclusions [23]. Other study design considerations that should not be overlooked include knowing whether a study involves biological or technical replicates or both. For example, are some

samples collected from the same individuals at different time points? Are those time points before and after some treatment? If one assumes that all the samples are independent but that is in fact not the case, a variety of issues may arise, including having a lower effective sample size than expected. As described in Tip 1, unsupervised learning and other exploratory analyses can be identify such biases in these datasets prior to applying the deep learning model.

In general, deep learning has an increased tendency for overfitting, compared to classical methods, due to the large number of parameters being estimated, making issues of adequate sample size even more important (see Tip 7). For a large dataset, overfitting may not be a concern, but the modeling power of deep learning may lead to more spurious correlations and thus incorrect interpretation of results (see Tip 9). It is important to note that molecular or imaging datasets often require appropriate clinical or demographic data to support robust analyses; this must always be balanced with the need to protect patient privacy (see Tip 10). Looking at these annotations can also clarify the study design (for example, by seeing if all the individuals are adolescents or women) or at least help the analyst employing deep learning to know what questions to ask.

Data simulation is a powerful approach to develop an understanding of how data and analytical methods interact. In data simulation, a model is used to learn the true distribution of a training set for the purpose of creating new data points. Often, researchers may perform simulations under some assumptions about the data generating process to identify useful model architectures and hyperparameters. Simulated datasets can be used to verify the correctness of a model's implementation. To accurately test the performance of the model, it is important that simulated datasets be generated for a range of parameters. For example, varying the parameters to violate the model's assumptions can test the sensitivity of the model's performance. Parameter tuning the simulation can help researchers identify the key features that drive method performance. In other cases, neural networks can be used to simulate data to better understand how to structure analyses. For example, it is possible to study how analytical strategies cope with varying number of noise sources by using neural networks to simulate genome-wide data [24]. Simulating data from assumptions about the data generating distribution can help to debug or characterize deep learning models, and deep learning models can also simulate data in cases where it is hard to make reasonable assumptions from first principles.

Basically, thoroughly study your data and ensure that you understand its context and peculiarities *before* jumping into deep learning.

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

Unfortunately, choosing how to represent your data and design your architecture is closer to an art than a science. While certain best practices have been established by the research community [25], 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 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. With that being 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), which are the most basic type of neural network, may be effective. 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 they can outperform canonical sequence learning techniques such as recurrent neural networks (RNNs) and the closely related long short-term memory (LSTM) networks [26].

DL models can typically benefit from large amounts of labeled data to avoid overfitting (see Tip 7) and to achieve top performance on a task in hand. 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 [27]. Transfer learning is most useful when pre-training and target datasets are of similar nature [27]. For this reason, it is important to search for similar datasets that are already available and may 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 improve performance of the model compared to just random feature initialization. For example Rojkomar et al. showed advantages of ImageNet-pretraining [28] for the model that is applied to grayscale medical image classification [29]. 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 [30] for genomics models. Moreover, learned features can be helpful even when pre-training task was different from the target one [31]. Related to this property of transfer learning is multi-task learning, in which a network is trained jointly for multiple tasks simultaneously, sharing the same set of features across them. Multi-task learning can be used separately or even in combination with transfer learning [32].

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

Multi-layer neural networks can approximate arbitrary continuous functions, given at least one hidden layer, a non-linear activation function, and a large number of hidden units [33]. The same theory applies to deeper architectures, which require an exponentially smaller number of hidden units to approximate functions with the same complexity as neural networks with only one hidden layer. The flexibility of neural networks to approximate arbitrary, continuous functions as well as the overall trend towards deeper architectures with an increasing number of hidden units and learnable weight parameters (the so-called increasing "capacity" of neural networks) allows for solving more and more complex problems but also poses additional challenges during model training. You should expect to systematically evaluate the impact of numerous hyperparameters when you aim to apply deep neural networks to new data or challenges. Hyperparameters are typically manifested in the choice of optimization algorithms, learning rate, activation functions, number of hidden layers and hidden units, size of the training batches, weight initialization schemes, and also seeds for pseudo-random number generators used for dataset shuffling and weight initialization. Moreover, additional hyperparameters are introduced common techniques that facilitate the training of deeper architectures, such as norm penalties (typically in the form of L^2 regularization), Dropout [34], and Batch Normalization [35], which can reduce the effect of the so-called vanishing or exploding gradient problem when working with deep neural networks.

This flexibility also makes it difficult to evaluate the extent to which neural network methods are well suited to solving a task. We discussed how the Continental Breakfast Included effect could affect methods developers seeking to compare techniques in Tip 2. This effect also has implications for those seeking to use existing deep learning methods because 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 a relatively large input of human and computation time for hyperparameter optimization.

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

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

Overfitting is one of the most significant dangers you'll face in deep learning (and traditional machine learning). Put simply, overfitting occurs when a model fits patterns in the training data too closely, includes noise or non-scientifically relevant perturbations, or in the most extreme case, simply memorizes patterns in the training set. 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. To continue the student analogy, a smarter student has greater potential for memorization than average one and thus may be more inclined to memorize.

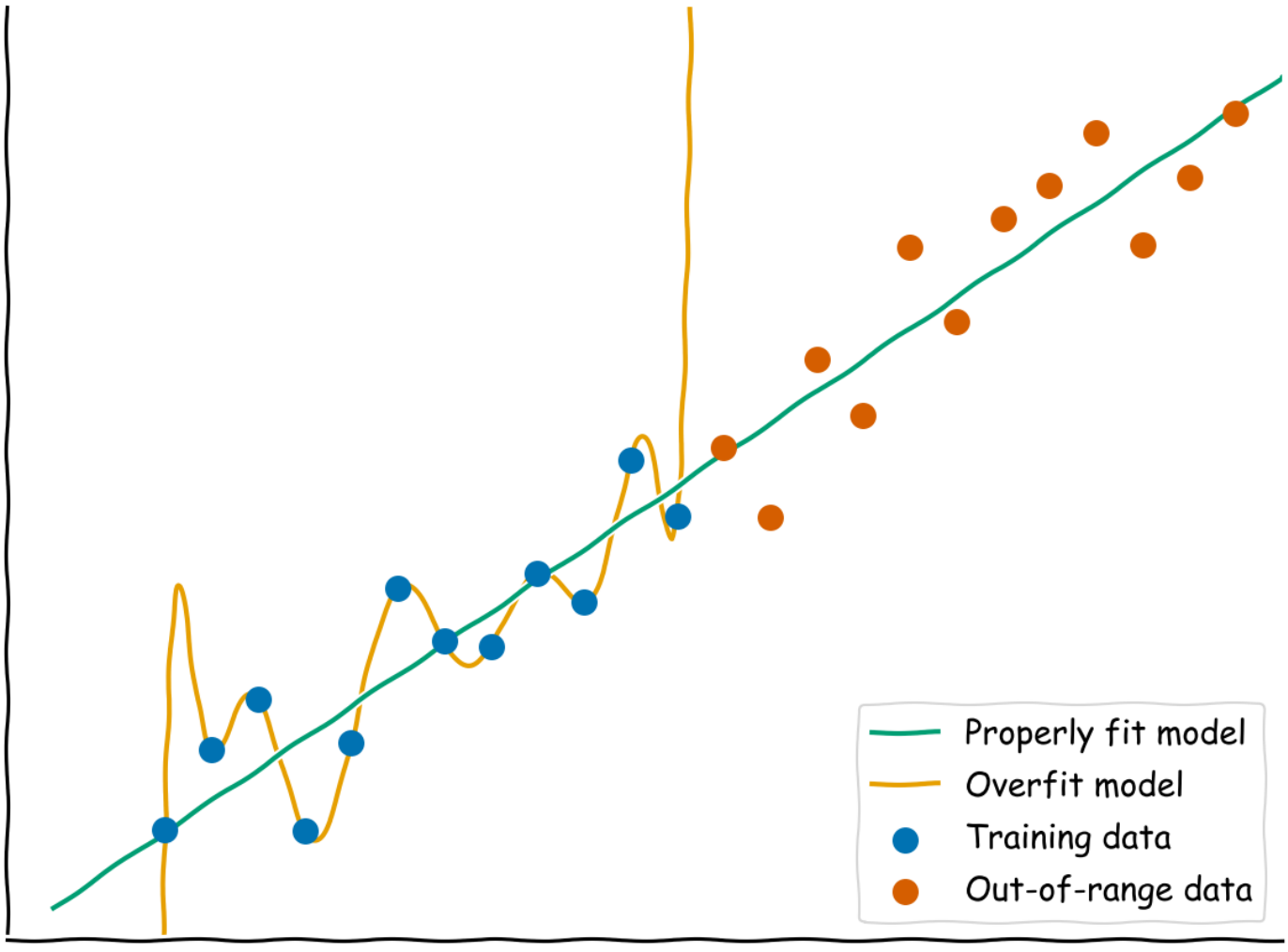

Figure 1: A visual example of overfitting and failure to generalize. While a high-degree polynomial gets high accuracy on its training data, it performs poorly on data unlike that which it has seen before. In contrast, a simple linear regression works well on both datasets. The greater representational capacity of the polynomial is analogous to using a larger or deeper neural network.

The simplest way to combat overfitting is to detect it. This can be done by splitting the dataset into three parts: a training set, a tuning set (also commonly called a validation set in the machine learning literature), and a test set. By exposing the model solely to the training data during fitting, a researcher can use the model's performance on the unseen test data to measure the amount of overfitting. 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). In addition, there are a variety of techniques to reduce overfitting during training including data augmentation and regularization techniques such as dropout [36] and weight decay [37]. Another way, as described by Chuang and Keiser, is to identify the baseline level of memorization of the network by training on the data with the labels randomly shuffled and to see if the model performs better on the actual data [38]. If the model performs no better on real data than randomly scrambled data, then the performance of the model can be attributed to overfitting.

Additionally, in biology and medicine it is critical to consider independence when defining training and test sets. For example, a DL model for pneumonia detection in chest X-rays performed well but failed to generalize to outside hospitals because they were able to detect which hospital the image was from and exploited this information when making predictions [39]. Similarly, when dealing with sequence data, holding out data that are evolutionarily related or share structural homology to the training data can result in overfitting. In these cases, simply holding out test data selected from a random partition of the training data is insufficient. The best remedy for confounding variables is to know your data and to test your model 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. Futhermore, be cognizant of the danger of skewed or biased data artificially inflating accuracy.

Tip 8: Your DL models can be more transparent

Model interpretability is a broad concept. 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. In much of the ML literature, including in our guidelines, the concept of model interpretability refers to the ability to identify the discriminative features that influence or sway the predictions. ML 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.

Model interpretability is particularly important in biomedicine, where subsequent decision making often requires human input. For example, while prediction rules can be derived from high-throughput molecular datasets, most affordable clinical tests rely on lower dimensional measurements of a limited number of biomarkers. Selecting those biomarkers to support decision making is an important modeling and interpretation challenge. Many authors attribute a lower uptake of DL tools in healthcare to interpretability challenges [40,41]. Strategies to interpret both ML and DL models are rapidly emerging, and the literature on the topic is growing at an exponential rate [42]. Therefore, instead of recommending specific methods for either DL-specific or general-purpose model interpretation, we suggest consulting [43] which is freely available and continually updated.

Model interpretation is an open, active area of research. It is becoming more feasible to interpret models with many parameters and non-linear relationships, but in many cases simpler models remain substantially easier to interpret than more complex ones. 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, interpretable model that gives up only a little performance when compared to a very complex one more useful in many settings. 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 to defining biomarker panels that enable clinical applications.

Tip 9: Don't over-interpret predictions

Once we have trained an accurate deep model, we often want to use it to deduce scientific findings. In doing so, we need to take care to correctly interpret the model's predictions. We know that the basic tenets of machine learning also apply to deep learning (Tip 1), but because deep models can be difficult to interpret intuitively, there is a temptation to anthropomorphize the models. We must resist this temptation.

A common saying in statistics classes is "correlation doesn't imply causality". While we know that accurately predicting an outcome doesn't imply learning the causal mechanism, it can be easy to forget this lesson when the predictions are extremely accurate. A poignant example of this lesson is [44,45]. In this study, the authors evaluated the capacities of several models to predict the probability of death for patients admitted to an intensive care unit with pneumonia. Unsurprisingly, the neural network model achieved the best predictive accuracy. However, after fitting a rule-based model, the authors discovered that the hospital data implied the rule `HasAsthma(x) => LowerRisk(x)`. This rule contradicts medical understanding - having asthma doesn't make pneumonia better! This rule was supported by the data (pneumonia patients with a history of asthma tended to receive more aggressive care), so the neural network also learned to make predictions according to this rule. 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 toward building understanding and deducing scientific conclusions, probe data domains where your model succeeds and contrast them with domains where your model fails in order to identify your model's internal logic, taking care to avoid overinterpreting or anthropomorphizing the model.

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 [46]. One of the greatest opportunities for deep learning in biology is the ability for these techniques to extract information that cannot readily be captured by traditional methods [47]. The representation learning of the deep learning models can capture information-rich abstractions of multiple features of the data during the training process. However, 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. Thus, with both deep learning and certain traditional machine learning methods (*e.g. k*-nearest neighbors models, which learn by memorizing the full training data), it is imperative not to share models trained on sensitive data.

Techniques to train deep neural networks without sharing unencrypted access to data are being advanced through implementations of homomorphic encryption [48,49], but adversarial training techniques such as model inversion attacks can be used to exploit model predictions to recover recognizable images of people's faces used for training [50]. Privacy preserving techniques [51], such as differential privacy [52,53,54], can help to mitigate risks as long as the assumptions underlying these techniques are met. These techniques provide a path towards a future where models can be shared, but more software development and theoretical advances will be required to make these techniques easy to apply in many settings. Until then, don't share models trained on sensitive data.

Conclusion

Deep learning techniques have the potential for wide use in biology, meeting or exceeding the performance of both humans and the current state-of-the art algorithms in a variety of tasks. Beyond simply achieving good predictive performance, deep learning has the potential to generate novel biological insights that could assist the progress of fundamental 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 will serve as a starting point for the discussion of best practices for deep learning as they apply to biology, not as an ending point.

Acknowledgements

The authors would like the 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 · PMID: 29618526 · PMCID: 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 · PMID: 29295994 · PMCID: 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 · PMID: 30087331 · PMCID: PMC6081423

4. **Benjamin-Lee/deep-rules GitHub repository**
   Benjamin Lee
   GitHub (2018) https://github.com/Benjamin-Lee/deep-rules

5. **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/

6. **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 · PMID: 29234465 · PMCID: PMC5721660

7. **Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning**
   Sebastian Raschka
   arXiv (2018-12-04) https://arxiv.org/abs/1811.12808

8. **Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms**
   Thomas G. Dietterich
   Neural Computation (1998-10) https://doi.org/fqc9w5
   DOI: 10.1162/089976698300017197 · PMID: 9744903

9. **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 · PMID: 25738806 · PMCID: PMC4349800

10. **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 · PMID: 29096442 · PMCID: PMC5741413

11. **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

12. **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

13. **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 · PMID: 31304302 · PMCID: PMC6550175

14. **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 · PMID: 29086090 · PMCID: PMC5489441

15. **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

16. **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

17. **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. Eglen, Daniel S. Katz, … Juan Antonio Vizcaíno
PLOS Computational Biology (2016-07-14) https://doi.org/gbrb39
DOI: 10.1371/journal.pcbi.1004947 · PMID: 27415786 · PMCID: PMC4945047

18. **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/pjb
DOI: 10.1371/journal.pcbi.1003285 · PMID: 24204232 · PMCID: PMC3812051

19. **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

20. **Deep Learning SDK Documentation**
NVIDIA

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

21. **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

22. **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 · PMID: 11726920

23. **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 · PMID: 20838408 · PMCID: PMC3880143

24. **Correcting for experiment-specific variability in expression compendia can remove underlying signals**
Alexandra J. Lee, YoSon Park, Georgia Doing, Deborah A. Hogan, Casey S. Greene
Cold Spring Harbor Laboratory (2020-05-03) https://doi.org/gg7m5s
DOI: 10.1101/2020.05.03.066597

25. **Neural Networks: Tricks of the Trade**
Lecture Notes in Computer Science
(2012) https://doi.org/gfvtvt
DOI: 10.1007/978-3-642-35289-8

26. **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

27. **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

28. **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

29. **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 · PMID: 27730417 · PMCID: PMC5267603

30. **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

31. **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

32. **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

33. **Approximation capabilities of multilayer feedforward networks**
Kurt Hornik
Neural Networks (1991) https://doi.org/dzwxkd
DOI: 10.1016/0893-6080(91)90009-t

34. **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)

35. **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)

36. **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

37. **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

38. **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/acschembio.8b00881 · PMID: 30336670

39. **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 · PMID: 30399157 · PMCID: PMC6219764

40. **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/gfgtzx
DOI: 10.1109/jbhi.2016.2636665 · PMID: 28055930

41. **Towards trustable machine learning**
Nature Biomedical Engineering
(2018-10-10) https://doi.org/gfw9cn
DOI: 10.1038/s41551-018-0315-x · PMID: 31015650

42. **On Interpretability of Artificial Neural Networks**
Fenglei Fan, Jinjun Xiong, Ge Wang
arXiv (2020-01-09) https://arxiv.org/abs/2001.02522

43. **Interpretable Machine Learning**
Christoph Molnar
https://christophm.github.io/interpretable-ml-book/

44. **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

45. **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/gftgxk
DOI: 10.1145/2783258.2788613

46. **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/gdqfcn
DOI: 10.1371/journal.pcbi.1005399 · PMID: 28358831 · PMCID: PMC5373508

47. **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

48. **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 · PMID: 30180163 · PMCID: PMC6138421

49. **Towards the AlexNet Moment for Homomorphic Encryption: HCNN, theFirst 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

50. **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

51. **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

52. **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

53. **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

54. **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