# Ten Quick Tips for Deep Learning in Biology

*This manuscript (permalink) was automatically generated from Benjamin-Lee/deep-rules@fa1a03c on February 28, 2019.*

## 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
  Department of Genetics, University of Wisconsin-Madison; Department of Bacteriology, 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


# Introduction

Deep learning (DL), a subfield of machine learning (ML) implementing artificial neural networks with many layers, is 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, a lack of concise recommendations for biological applications of DL hamper 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 few resources articulate general DL best practices to the scientific community. Most instructional literature focuses on ML broadly, rather than DL 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 techniques to biology, we aim to improve the overall quality and reporting of DL in the literature, enabling more researchers to apply these powerful methods to generate biological insights.

## 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 data that are relevant to the problem at hand; non-salient data can hamper performance or lead to spurious conclusions. 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 throughly 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, data that is confounded, biased, skewed, or of low quality will produce models of dubious performance and limited generalizability.

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.

Extreme cases warrant testing the robustness of the model and metrics on simulated data for which the ground truth is known. Said simulations can be used to verify the correctness of the model's implementation as well.

## Tip 2: Use traditional methods to establish performance baselines

Before diving into a fancy thousand-layer neural network, always implement at least a simple model to establish an adequate performance baseline. For example, researchers can build multinomial logistic regression or random forest models using the same software framework that is being used for DL and evaluate its classification performance. This approach will help researchers with assessing the complexity of the task at hand and debugging more complex DL architectures. The utility of these methods is evidenced by the recent development of hybrid models which combine DL and simpler models to improve robustness, interpretability, and confidence estimation [11,12]. Depending on the amount of available data and the type of tasks, DL models may not necessarily be the best performing one. As an illustration, the simple baseline models by Rajkomar et al. [13] 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.

It is worth noting that conventional machine learning methods (e.g., support vector machines, random forests) are also likely to benefit from parameter tuning. It can be tempting to train baseline models with these conventional methods using default parameters, which may provide acceptable but not stellar performance, but then tune the parameters for DL models to optimize performance. Hu and Greene [14] discuss a Continental Breakfast Included effect by which unequal hyperparameter tuning skews the evaluation of methods, especially those with performance that varies substantially with modest changes to hyperparameters. Those wishing to compare methods should tune the parameters of traditional and DL 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 tuned.

## 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 have to expect to train many networks with a range of different parameter and hyperparameter settings. Despite improved framework ease-of-use and on-demand cloud computing resources this means DL can be very demanding, often requiring significant infrastructure and patience to achieve state-of-the-art performance [15]. 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 good coding practices (for example, version control, continuous integration etc.) in order to be effective and interpretable. This organization is also key to being able to efficiently share your work and to update your model as new data becomes available.

One specific reproducibility pitfall that is often missed is the default use of non-deterministic algorithms by CUDA/CuDNN backends when training on 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 [16].

Similar to Tip 4, try to start with a relatively smaller 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 [17]). These need to be carefully considered and their impacts evaluated (see Tip 6).

## 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; 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 data that seems to be a good fit to test your hypothesis, but the act of obtaining the data can reveal numerous problems such as the data is unstructured when it is supposed to be structured, crucial metadata such as sample stratification is 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 data is obtained, it is easy to begin analyzing data 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 [18]), 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 [19]. 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.

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). Finally, it is important to note that with the exception of very specific cases of unsupervised data analysis, it is generally the case that a molecular or imaging dataset does not have much value without appropriate clinical or demographic data; this must always be balanced with the need to protect patient privacy (see Tip 10). Looking at these data 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.

## 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 [20], 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 data 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 data 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 [arxiv:1803.01271].

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 [21]. Transfer learning is most useful when pre-training and target datasets are of similar nature [21]. For this reason, it is important to search for similar data that is 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 [22] for the model that is applied to grayscale medical image classification [23]. 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 [24] for genomics models. Moreover, learned features can be helpful even when pre-training task was different from the target one [25]. 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 in combination with transfer learning [26].

## Tip 6: Expect to tune hyperparameters extensively and systematically

Deep neural networks have the ability to approximate arbitrary continuous functions, as long as the neural network contains enough hidden nodes [27]. However, this flexibility makes the training process somewhat challenging. Users should expect to systematically evaluate the impact of numerous hyperparameters when they aim to apply deep neural networks to new data or challenges.

Neural network architectures also have their own odd nuances that affect hyperparameter portability. For example, in variational autoencoders (VAEs) there are two elements that are being optimized, reconstruction and distribution loss [28]. In common implementations, the relative

weights of each are a function of the number of input features (more increase the importance of reconstruction loss) and the number of features in the latent space (more increase the importance of the distribution loss). Users who apply a VAE architecture to a new dataset with more input features, even without changing any hyperparameters, alter the relative weights of the components of the loss function.

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 an input of human and compute time for hyperparameter optimization.

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

Overfitting is one of the most significant dangers faced by a deep learning practitioner. 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.

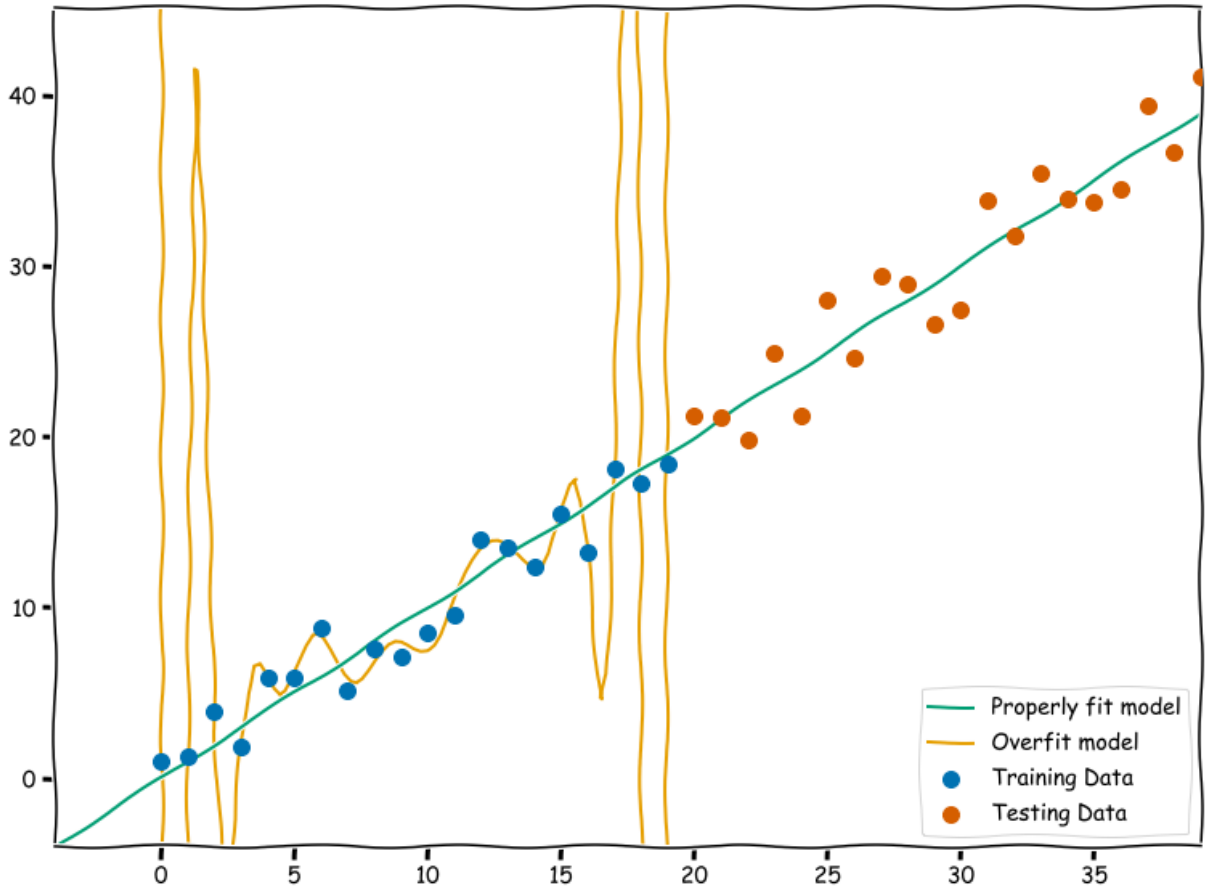


Figure 1: A visual example of overfitting. While a high-degree polynomial gets high accuracy on its training data, it performs poorly on data that is has not seen before, whereas a simple linear regression works well. The greater representational capacity of the polynomial is analogous to using a larger 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). Additionally, there are a variety of techniques to reduce overfitting during training including data augmentation and regularization techniques such as dropout [29] and weight decay [30]. 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 [31]. 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, one must be sure that their data are not skewed or biased, such as by having confounding and scientifically irrelevant variables that the model can pick up on [32]. In this case,

simply holding out test data is insufficient. The best remedy for confounding variables is to know your data and to test your model on truly independent data.

## Tip 8: Do not necessarily consider a DL model as a black box

## 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 [33,34]. 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 pneumonia 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). By probing data domains where the model succeeds and contrasting with domains where the model fails, we can identify the internal logic and deduce scientific conclusions. In this way, we can move beyond fitting predictive models toward building understanding.

## Tip 10: Don’t share models trained on sensitive data

One of the greatest opportunities for deep learning in biology is the ability for deep learning techniques to incorporate representation learning to extract information that can not readily be captured by traditional methods [35]. The abundance of features for each training example means that the representation learning of the deep learning models can capture information-rich

abstractions of data during the training process. Therefore with both deep learning and traditional machine learning models (*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. Applying deep learning to images of cats from the internet does not pose significant ethical, legal, or privacy problems; this is not the case when dealing with classified, confidential, trade secret, or other types of biological data that cannot be shared.

Techniques to train deep neural networks without sharing unencrypted access to data are being advanced through implementations of homomorphic encryption [36,37]. However, 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 [38]. These risks are more significant in deep learning than traditional machine learning because models have more representational capacity. The large number of model weights, even in a relatively small network, allow deep learning to model high-dimensional non-linear relationships among features. Such models can learn more nuanced features of specific data, but these features may be so specific that they may reveal the underlying sensitive data. When training deep learning models on sensitive data, using privacy preserving techniques [39], such as differential privacy [40,41,42], can help to mitigate risks as long as the assumptions underlying these techniques are met.

# Conclusion

# 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) 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. **Ten Quick Tips for Deep Learning in Biology. Contribute to Benjamin-Lee/deep-rules development by creating an account on GitHub**
Benjamin Lee
(2019-02-28) https://github.com/Benjamin-Lee/deep-rules

5. **Open collaborative writing with Manubot**
Daniel S. Himmelstein, David R. Slochower, Venkat S. Malladi, Casey S. Greene, Anthony Gitter
(2019-02-21) https://greenelab.github.io/meta-review/

6. **Ten quick tips for machine learning in computational biology**
Davide Chicco
*BioData Mining* (2017-12) 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* (2018-11-13) https://arxiv.org/abs/1811.12808v2

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

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* (2018-03-13) https://arxiv.org/abs/1803.04765v1

12. **To Trust Or Not To Trust A Classifier**
Heinrich Jiang, Been Kim, Melody Y. Guan, Maya Gupta
*arXiv* (2018-05-30) https://arxiv.org/abs/1805.11783v2

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

14. **Parameter tuning is a key part of dimensionality reduction via deep variational autoencoders for single cell RNA transcriptomics**
Qiwen Hu, Casey S Greene
*Cold Spring Harbor Laboratory* (2018-08-05) https://doi.org/gdxxjf
DOI: 10.1101/385534

15. **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

16. **Deep Learning SDK Documentation**
NVIDIA
(2018-11-01) https://docs.nvidia.com/deeplearning/sdk/cudnn-developer-guide/index.html#reproducibility

17. **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

18. **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

19. **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

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

21. **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

22. **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

23. **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

24. **Kipoi: accelerating the community exchange and reuse of predictive models for genomics**
Ziga Avsec, Roman Kreuzhuber, Johnny Israeli, Nancy Xu, Jun Cheng, Avanti Shrikumar,

Abhimanyu Banerjee, Daniel S Kim, Lara Urban, Anshul Kundaje, … Julien Gagneur
*Cold Spring Harbor Laboratory* (2018-07-24) https://doi.org/gd24sx
DOI: 10.1101/375345

25. **CNN Features Off-the-Shelf: An Astounding Baseline for Recognition**
Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, Stefan Carlsson
*2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2014-06) https://doi.org/f3np4s
DOI: 10.1109/cvprw.2014.131

26. **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* (2016) https://doi.org/gfvs28
DOI: 10.1109/tbdata.2016.2573280

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

28. **Auto-Encoding Variational Bayes**
Diederik P Kingma, Max Welling
*arXiv* (2013-12-20) https://arxiv.org/abs/1312.6114v10

29. **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

30. **A Simple Weight Decay Can Improve Generalization**
Anders Krogh, John A. Hertz
*Proceedings of the 4th International Conference on Neural Information Processing Systems* (1991) http://dl.acm.org/citation.cfm?id=2986916.2987033
ISBN: 9781558602229

31. **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

32. **Confounding variables can degrade generalization performance of radiological deep learning models**
John R. Zech, Marcus A. Badgeley, Manway Liu, Anthony B. Costa, Joseph J. Titano, Eric K.

Oermann
*arXiv* (2018-07-02) https://arxiv.org/abs/1807.00431v2

33. **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

34. **Intelligible Models for HealthCare**
Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, Noemie Elhadad
*Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '15* (2015) https://doi.org/gftgxk
DOI: 10.1145/2783258.2788613

35. **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* (2015-09-30) https://arxiv.org/abs/1509.09292v2

36. **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

37. **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, Sim Jun Jie, Benjamin Hong Meng Tan, Xiao Nan, Khin Mi Mi Aung, Vijay Ramaseshan Chandrasekhar
*arXiv* (2018-11-02) https://arxiv.org/abs/1811.00778v1

38. **Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures**
Matt Fredrikson, Somesh Jha, Thomas Ristenpart
*Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS '15* (2015) https://doi.org/cwdm
DOI: 10.1145/2810103.2813677

39. **A generic framework for privacy preserving deep learning**
Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert,

Jonathan Passerat-Palmbach
*arXiv* (2018-11-09) https://arxiv.org/abs/1811.04017v2

40. **Deep Learning with Differential Privacy**
Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, Li Zhang
*Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS'16* (2016) https://doi.org/gcrnp3
DOI: 10.1145/2976749.2978318

41. **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
*Cold Spring Harbor Laboratory* (2017-07-05) https://doi.org/gcnzrn
DOI: 10.1101/159756

42. **Privacy-Preserving Distributed Deep Learning for Clinical Data**
Brett K. Beaulieu-Jones, William Yuan, Samuel G. Finlayson, Zhiwei Steven Wu
*arXiv* (2018-12-04) https://arxiv.org/abs/1812.01484v1