EvoAug-TF: Extending evolution-inspired data augmentations for genomic deep learning to TensorFlow

Yiyang Yu,[1] Shivani Muthukumar[2] and Peter K Koo [1,*]

[1]Simons Center for Quantitative Biology, Cold Spring Harbor Laboratory, 1 Bungtown Rd, Cold Spring Harbor, NY, 11724, USA and [2]Commack High School, 1 Scholar Ln, Commack, NY, 11725, USA

*Corresponding author. koo@cshl.edu

Abstract

Summary: Deep neural networks (DNNs) have been widely applied to predict the molecular functions of the non-coding genome. DNNs are data hungry and thus require many training examples to fit data well. However, functional genomics experiments typically generate limited amounts of data, constrained by the activity levels of the molecular function under study inside the cell. Recently, EvoAug was introduced to train a genomic DNN with evolution-inspired augmentations. EvoAug-trained DNNs have demonstrated improved generalization and interpretability with attribution analysis. However, EvoAug only supports PyTorch-based models, which limits its applications to a broad class of genomic DNNs based in TensorFlow. Here, we extend EvoAug's functionality to TensorFlow in a new package we call EvoAug-TF. Through a systematic benchmark, we find that EvoAug-TF yields comparable performance with the original EvoAug package.

Availability: EvoAug-TF is freely available for users and is distributed under an open-source MIT license. Researchers can access the open-source code on GitHub (`https://github.com/p-koo/evoaug-tf`). The pre-compiled package is provided via PyPI (`https://pypi.org/project/evoaug-tf`) with in-depth documentation on ReadTheDocs (`https://evoaug-tf.readthedocs.io`). The scripts for reproducing the results are available at (`https://github.com/p-koo/evoaug-tf_analysis`).

Key words: Deep learning, Data augmentations, Regulatory genomics

Introduction

Deep neural networks (DNNs) have emerged as a promising tool for supervised learning of regulatory genomics data, taking DNA sequences as input and predicting the readouts of functional genomics experiments [Eraslan et al., 2019, Koo and Ploenzke, 2020]. Due to their overparameterization, DNNs are data hungry, requiring large amounts of data to learn discriminative features that ensure good generalization [Zhang et al., 2021]. However, most functional genomics experiments only observe a limited number of molecular interactions within the context of a cell. For instance, the number of binding sites available for a transcription factor can be limited to the finite number of accessible DNA within a given cell type. Hence, dataset size is a major limiting factor when analyzing functional genomics data with DNNs.

Data augmentation is a widely practiced strategy in machine learning to provide additional training samples. In practice, random transformations that maintain the same training labels are imposed on the input data, leveraging the natural symmetries in the data. For example, in natural images, the objects can undergo affine transformations, flips, color perturbations, and blurs, which do not alter the object's label. In genomics, the available transformations that can maintain the same training labels are limited to reverse complements and small random shifts [Avsec et al., 2021a, Toneyan et al., 2022].

To expand the available augmentations, EvoAug [Lee et al., 2023] was recently introduced to provide evolution-inspired data augmentations, including translocations, insertions, deletions, inversions, and mutations. In nature, genetic variation is sampled by evolution to increase phenotypic diversity. Thus, genetic mutations can alter the function of the sequence and, hence, change its label. Nevertheless, EvoAug asserts that the transformed sequences retain the same training label as the wild-type sequence, which can be considered as imposing a prior. For instance, small random translocations impose a prior that the activity of motifs is invariant to shifts. Moreover, insertions and deletions impose a prior that the distance between motifs is insignificant. Considered as a prior, evolution-inspired augmentations can introduce a bias in the DNN that may break

the underlying rules of *cis*-regulatory grammars in the data. Thus, EvoAug employs a second stage of training that finetunes the DNN on the original, unaltered data, ensuring functional integrity towards the observed biology. This two-stage training curriculum has proven beneficial for genomic DNNs using synthetic data augmentations [Lee et al., 2023] and natural data augmentations [Duncan et al., 2023], as well as for protein-based DNNs [Lu et al., 2020, 2022].

However, the availability of EvoAug has been limited to a PyTorch [Paszke et al., 2019] implementation, posing a barrier for researchers working in TensorFlow [Abadi et al., 2015]. To address this demand, we present `EvoAug-TF`, a TensorFlow implementation that builds upon the same principles as EvoAug. Below, we describe EvoAug-TF, highlighting its adaptations and unique features that are specific to the TensorFlow framework. Additionally, we present experimental results showcasing the effectiveness of EvoAug-TF in improving the generalization capabilities of genomic DNNs.

Methodology and Implementation

EvoAug-TF adapts the functionality of the PyTorch-based EvoAug framework in TensorFlow (Fig. 1a), including the augmentation techniques (e.g., random transversion, insertion, translocation, deletion, mutation, and noise). EvoAug-TF employs the same two-stage training curriculum, where stochastic augmentations are applied online to each mini-batch during training, followed by a finetuning step on the original, unperturbed data. Since EvoAug-TF imposes transformations on the input data while maintaining the same labels as the wildtype sequence, in its current form, EvoAug-TF only supports DNNs that output scalars in single-task or multi-task settings. By contrast, profile-based DNNs [Kelley et al., 2018, Avsec et al., 2021b, Toneyan et al., 2022] may require transformations to the corresponding labels, which is currently not supported in EvoAug-TF.

Several adaptations were made to EvoAug-TF compared to the PyTorch implementation, incorporating key design choices specific to the TensorFlow version. To ensure compatibility with TensorFlow's graph mode and optimize training speed, `tf.Tensors` are used as substitutes for NumPy arrays, which were utilized in the PyTorch implementation. Additionally, EvoAug-TF employs `tf.while_loop` to achieve the same effects as the for loops used in the PyTorch implementation. The implementation of EvoAug-TF relies on TensorFlow 2 and its relevant libraries and dependencies; it has been tested thoroughly on versions 2.7 and 2.15.

A key difference between EvoAug-TF and EvoAug lies in the approach to applying augmentations to the mini-batch during training. In the PyTorch implementation, the number of augmentations and the augmentation type are randomly chosen for each sequence. In the TensorFlow implementation, the same number of augmentations and augmentation types are employed for each sequence. However, the selected augmentation type(s) are sampled in a stochastic manner to impose a unique perturbation to each sequence in the mini-batch, similar to EvoAug. This design choice was made to simplify the process of imposing augmentations, while still sampling a high degree of genetic variation. Moreover, for some augmentations – such as translocation, insertion, deletion, and transversion (i.e., reverse-complement) – we offer an additional option to perform the same perturbation across all of the sequences within the mini-batch (Fig. 1b), which we term batch mode. This feature provides a faster way to deploy augmentations, improving computational efficiency at the expense of sampling diversity.

Results and Discussion

To benchmark the performance of EvoAug-TF, we utilized the data and deep learning model from the DeepSTARR study [de Almeida et al., 2022]. The prediction task is set up to take as input 249 nucleotide (nt) sequences and predict enhancer activity (measured via STARR-seq [Arnold et al., 2013]) for developmental and housekeeping transcriptional promoters in D. melanogaster S2 cells as a multi-task regression. We systematically trained the DeepSTARR model with different EvoAug-TF augmentations individually and in combinations and compared their performance with the original EvoAug, using the same hyperparameters as in the original study [Lee et al., 2023]. To ensure the robustness of the results, we conducted five trials for each set of augmentations with different random initializations.

We found that models trained with EvoAug-TF augmentations achieved comparable performance to the original EvoAug-trained models with the same augmentation settings (Fig. 1c and 1d). Notably, EvoAug-TF consistently yielded improved performance compared to the original EvoAug after stage 1 (i.e., before finetuning) with the exception of random noise augmentation. However, the original EvoAug yielded slightly better performance than EvoAug-TF upon finetuning. In most cases, models trained with data augmentations led to improved performance compared to standard training.

In terms of computational costs, we found that running EvoAug-TF exhibited similar training times per epoch as EvoAug (Fig. 1e). Thus, EvoAug-TF's approach of applying the same numbers and types of augmentations to each sequence maintains a comparable computational cost as EvoAug's approach of applying different numbers and types of augmentations to each sequence. Notably, both approaches are effective in improving model performance.

To further demonstrate the breadth of the EvoAug-TF package, we explored the use of batch mode augmentations. First, we trained convolutional neural networks on ChIP-seq peak classification from the original EvoAug study using batch mode augmentations (i.e., insertion, deletion, and translocation only). As expected, models trained with EvoAug-TF augmentations in batch mode consistently led to improved performance (Fig. 1f). We also explored the effectiveness of EvoAug-TF in the low data regime with the same batch mode augmentations. Strikingly, a DeepSTARR model trained with EvoAug-TF on only 25% of the training data yields better performance than the same DeepSTARR model trained on the whole dataset with standard training (Fig. 1g). Thus, EvoAug-TF can greatly improve data efficiency when training genomic deep learning models, and batch mode is an effective way to improve performance over standard training.

In the original study, EvoAug-trained models were found to improve motif representations in attribution maps. To explore whether EvoAug-TF trained models also improve the interpretability of attribution maps, we generated Saliency Maps [Simonyan et al., 2013] for 500 sequences with the highest observed enhancer activity for the developmental promoter and a different set of 500 sequences for the housekeeping promoter in the DeepSTARR dataset. We then quantified the consistency

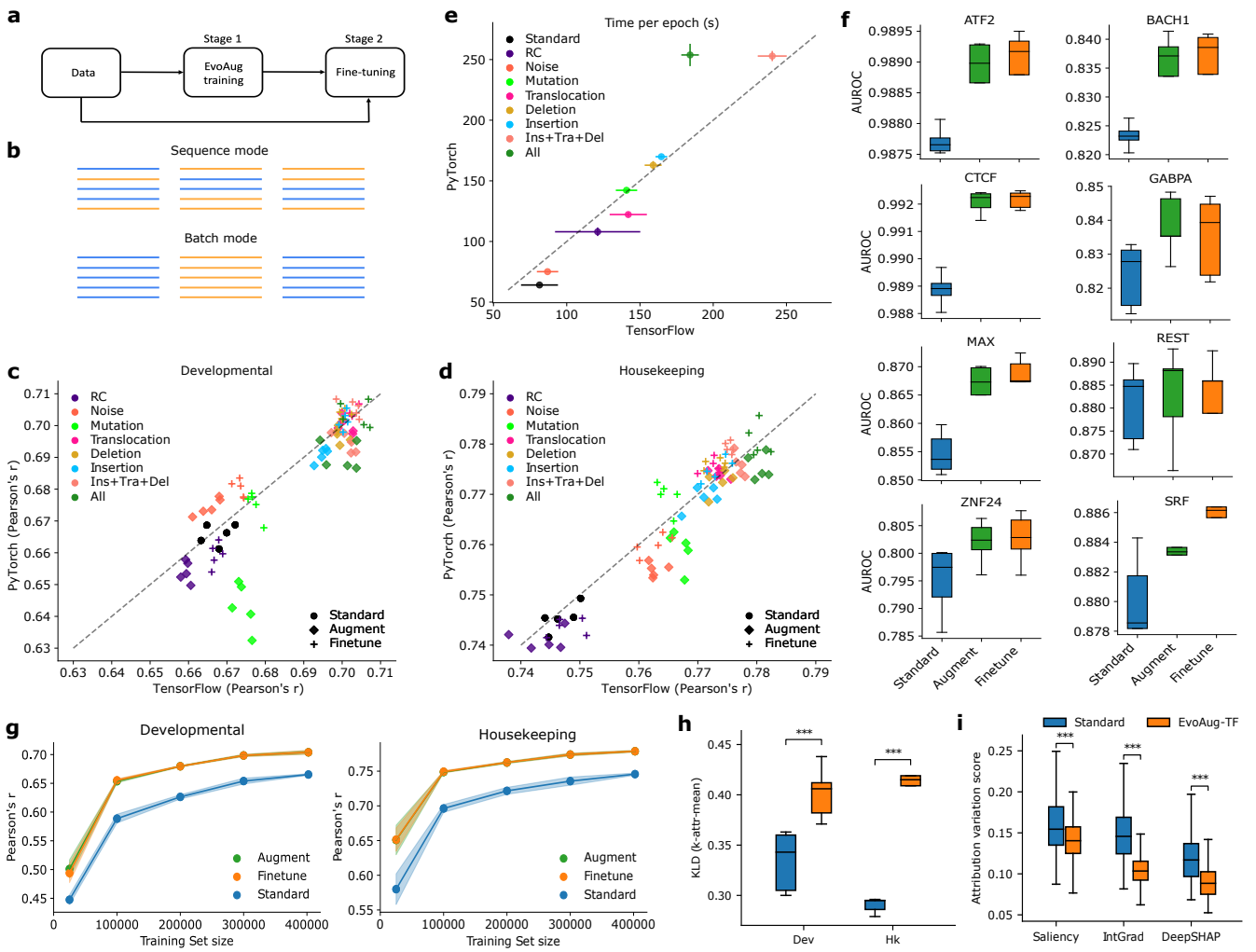

Fig. 1: Performance comparison between EvoAug and Standard training. **a** Flowchart illustrating EvoAug-TF's two-stage training curriculum. **b** Schematic demonstrating the difference between sequence mode and batch mode, where each column represents a mini-batch of sequences and different colors represent different data augmentations. **c, d** Performance comparison between EvoAug (PyTorch-based) and EvoAug-TF (TensorFlow-based) with DeepSTARR models trained with individual or combinations of augmentations (i.e., insertion + translocation + deletion; all augmentations) and finetuned on original STARR-seq data for two promoters: developmental (**c**) and housekeeping (**d**). **e** Comparison of the training time per epoch between EvoAug (PyTorch) and EvoAug-TF (TensorFlow). **c-e** Each scatter plot includes the data from 5 trials with random initialization. **f** Box-plots comparing the performance (area under the receiver operating characteristic curve, AUROC) of a convolutional neural network trained with standard training, EvoAug-TF augmentations, and with finetuning on various Chip-seq peak classification tasks. **g** Performance comparison of DeepSTARR models with various training methods fit to training sets with varying levels of down-sampling. EvoAug-TF augmentations employed insertion + translocation + deletion in batch mode. The shaded region represents the standard deviation of the mean across five models with different random initializations. **h** Box-plot comparison of the consistency of patterns within Saliency Maps, as measured by the *k*-attr-mean, for DeepSTARR models with standard training or EvoAug-TF augmentations (All). **i** Box-plot comparison of the attribution variation score, calculated according to the root-mean-squared attribution scores, for attribution maps generated by different methods (i.e., Saliency Maps, Integrated Gradients, and DeepSHAP) using a DeepSTARR model trained with standard training or EvoAug-TF augmentations (All). **h,i** Mann-Whitney U test with p-values less than 0.001 (***) and 0.01 (**). Boxes represent five identical models with different random initializations.

of salient patterns across a population of attribution maps using the Kullback-Liebler Divergence (KLD) between the distribution of locally embedded attribution scores versus an uninformative prior, which is termed *k*-attr-mean metric [Majdandzic et al., 2022]. A higher KLD suggests that the attribution scores reflect similar recurring patterns across the 500 attribution maps. Indeed, EvoAug-TF-trained DeepSTARR yielded significantly higher KLD scores than standard training (Fig. 1h).

Attribution methods are sensitive to local function properties and thus can yield spurious attribution scores for reasons that are not biological [Majdandzic et al., 2022]. Similar to the robustness test for model predictions introduced in [Toneyan et al., 2022], we developed a translational robustness test to quantitatively assess the robustness of attribution scores to small shifts in the input sequence. The assumption is that when no significant attribution scores are present near the ends of the sequence, small shifts to the sequence should not affect the importance of binding sites. Hence, the attribution scores for the binding sites should also shift with the translations while maintaining the same importance levels. In the translational robustness test, the input sequence was randomly translated by up to 30nt in either direction with `np.roll`, the (corrected [Majdandzic et al., 2023]) attribution scores were calculated for the translated sequence, and then the inverse translation was imposed on the attribution maps to align with the wild type attribution map. This process was repeated 20 times for each sequence, which resulted in 20 translated attribution maps realigned to the input sequence. Next, a variation score was calculated using the root-mean-squared error to summarize the total variation across the aligned attribution maps with a scalar value. As expected, DeepSTARR trained with EvoAug-TF yields a significantly lower variability score compared to standard training for Saliency Maps [Simonyan et al., 2013], Integrated Gradients [Sundararajan et al., 2017], and DeepSHAP [Lundberg and Lee, 2017] (Fig. 1i). Thus, EvoAug-TF training has the benefit of resulting in more robust attribution maps for various attribution methods.

Each augmentation has corresponding hyperparameters that can be tuned to optimize performance gains. Previously, EvoAug identified hyperparameters through a simple grid search, focusing on one augmentation at a time. The EvoAug-TF package provides examples that show how to integrate EvoAug-TF with comprehensive hyperparameter searches, such as population-based training [Jaderberg et al., 2017] and the asynchronous hyperband algorithm [Li et al., 2018] provided by Ray Tune [Liaw et al., 2018]. These should offer an alternative strategy to help navigate the combinatoric search space of discovering optimal hyperparameters when deploying multiple augmentations.

Conclusion

EvoAug-TF is a TensorFlow implementation of EvoAug (a PyTorch package) that provides the ability to train genomic DNNs with evolution-inspired data augmentations. Our results demonstrate the effectiveness of EvoAug-TF to improve generalization and model interpretability with attribution methods. We found that models incorporating EvoAug-TF augmentations achieved comparable or improved performance with similar computational efficiency as the original EvoAug models in PyTorch. Similar to EvoAug, EvoAug-TF is extensible – it can easily accommodate new types of custom augmentations within its data augmentation framework. While the current implementation only supports model outputs as scalars in single- or multi-task settings, we plan to extend these capabilities for multivariate predictions in the future to provide data augmentations for quantitative models that output profiles of read coverages [Kelley et al., 2018, Avsec et al., 2021b, Toneyan et al., 2022]. Overall, EvoAug-TF offers a new tool for TensorFlow users in the genomics research community to improve data efficiency in training genomic deep learning models, leading to improved generalization and interpretability with attribution analysis.

Conflict of interest

None declared.

Author contributions statement

Y.Y. and P.K. conceived the experiments, Y.Y. developed EvoAug-TF, conducted the experiments, and analyzed the results. S.M. developed the attribution translational robustness test and performed the experiments that used the test. Y.Y., S.M., and P.K. wrote and reviewed the manuscript.

Acknowledgments

The authors would like to thank Ziqi (Amber) Tang and Chandana Rajesh for support in setting up Ray Tune, Jakub Kaczmarzyk for support in setting up ReadTheDocs documentation, and Anirban Sarkar and Alessandro Crnjar for support on the translational robustness test for attribution maps.

Funding

Research reported in this publication was supported in part by the National Institute Of General Medical Sciences of the National Institutes of Health under Award Number R01GM149921 and the National Human Genome Research Institute of the National Institutes of Health under Award Number R01HG012131. This work was performed with assistance from the US National Institutes of Health Grant S10OD028632-01.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015.

Cosmas D Arnold, Daniel Gerlach, Christoph Stelzer, Łukasz M Boryń, Martina Rath, and Alexander Stark. Genome-wide quantitative enhancer activity maps identified by starr-seq. *Science*, 339(6123):1074–1077, 2013.

Žiga Avsec, Vikram Agarwal, Daniel Visentin, Joseph R Ledsam, Agnieszka Grabska-Barwinska, Kyle R Taylor, Yannis Assael, John Jumper, Pushmeet Kohli, and David R Kelley. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature Methods*, 18(10):1196–1203, 2021a.

Žiga Avsec, Melanie Weilert, Avanti Shrikumar, Sabrina Krueger, Amr Alexandari, Khyati Dalal, Robin Fropf, Charles McAnany, Julien Gagneur, Anshul Kundaje, et al. Base-resolution models of transcription-factor binding reveal soft motif syntax. *Nature Genetics*, 53(3):354–366, 2021b.

Bernardo P de Almeida, Franziska Reiter, Michaela Pagani, and Alexander Stark. Deepstarr predicts enhancer activity from dna sequence and enables the de novo design of synthetic enhancers. *Nature Genetics*, 54(5):613–624, 2022.

Andrew G Duncan, Jennifer A Mitchell, and Alan M Moses. Improving the performance of supervised deep learning for regulatory genomics using phylogenetic augmentation. *bioRxiv*, 2023.

Gökcen Eraslan, Žiga Avsec, Julien Gagneur, and Fabian J Theis. Deep learning: new computational modelling techniques for genomics. *Nature Reviews Genetics*, 20(7):389–403, 2019.

Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. Population based training of neural networks. *arXiv 1711.09846*, 2017.

David R Kelley, Yakir A Reshef, Maxwell Bileschi, David Belanger, Cory Y McLean, and Jasper Snoek. Sequential regulatory activity prediction across chromosomes with convolutional neural networks. *Genome Research*, 28(5):739–750, 2018.

Peter K Koo and Matt Ploenzke. Deep learning for inferring transcription factor binding sites. *Current Opinion in Systems Biology*, 19:16–23, 2020.

Nicholas Keone Lee, Ziqi Tang, Shushan Toneyan, and Peter K Koo. Evoaug: improving generalization and interpretability of genomic deep neural networks with evolution-inspired data augmentations. *Genome Biology*, 24(1):105, 2023.

Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. Massively parallel hyperparameter tuning. *arXiv 1810.05934*, 2018.

Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. Tune: A research platform for distributed model selection and training. *arXiv 1807.05118*, 2018.

Alex X Lu, Amy X Lu, Iva Pritišanac, Taraneh Zarin, Julie D Forman-Kay, and Alan M Moses. Discovering molecular features of intrinsically disordered regions by using evolution for contrastive learning. *PLOS Computational Biology*, 18(6): e1010238, 2022.

Amy X Lu, Alex X Lu, and Alan Moses. Evolution is all you need: phylogenetic augmentation for contrastive learning. *arXiv 2012.13475*, 2020.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30, 2017.

Antonio Majdandzic, Chandana Rajesh, Ziqi Tang, Shushan Toneyan, Ethan L Labelson, Rohit K Tripathy, and Peter K Koo. Selecting deep neural networks that yield consistent attribution-based interpretations for genomics. In *Machine Learning in Computational Biology*, pages 131–149. PMLR, 2022.

Antonio Majdandzic, Chandana Rajesh, and Peter K Koo. Correcting gradient-based interpretations of deep neural networks for genomics. *Genome Biology*, 24(1):1–13, 2023.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. 2019.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv 1312.6034*, 2013.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

Shushan Toneyan, Ziqi Tang, and Peter K Koo. Evaluating deep learning for predicting epigenomic profiles. *Nature Machine Intelligence*, 4(12):1088–1100, 2022.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3): 107–115, 2021.