ExplaiNN: interpretable and transparent neural networks for genomics

Gherman Novakovsky[1,†], Oriol Fornes[1,†], Manu Saraswat[1,2,3†],
Sara Mostafavi[4], Wyeth W. Wasserman[1,✉]

[1]Centre for Molecular Medicine and Therapeutics, Department of Medical Genetics, BC Children's Hospital Research Institute, University of British Columbia, Vancouver, BC V5Z 4H4, Canada
[2]Division of Computational Genomics and Systems Genetics, German Cancer Research Centre (DKFZ), Heidelberg, Germany
[3]European Molecular Biology Laboratory (EMBL), Genome Biology Unit, Heidelberg, Germany
[4]Paul G. Allen School of Computer Science and Engineering, University of Washington (UW), Seattle, USA

[†]GN, OF and MS contributed equally to this work. Co-first authors can prioritize their names when adding this reference to their résumés.

✉Correspondence to WWW (wyeth@cmmt.ubc.ca)

GN: 0000-0002-7089-5941
OF: 0000-0002-5969-3054
MS: 0000-0002-1472-8049
SM: 0000-0003-4698-1177
WWW: 0000-0001-6098-6412

Abstract

Sequence-based deep learning models, particularly convolutional neural networks (CNNs), have shown superior performance on a wide range of genomic tasks. A key limitation of these models is the lack of interpretability, slowing their broad adoption by the genomics community. Current approaches to model interpretation do not readily reveal how a model makes predictions, can be computationally intensive, and depend on the implemented architecture. Here, we introduce ExplaiNN, an adaptation of neural additive models[1] for genomic tasks wherein predictions are computed as a linear combination of multiple independent CNNs, each consisting of a single convolutional filter and fully connected layers. This approach brings together the expressivity of CNNs with the interpretability of linear models, providing global (cell state level) as well as local (individual sequence level) insights of the biological processes studied. We use ExplaiNN to predict transcription factor (TF) binding and chromatin accessibility states, demonstrating performance levels comparable to state-of-the-art methods, while providing a transparent view of the model's predictions in a straightforward manner. Applied to *de novo* motif discovery, ExplaiNN detects equivalent motifs to those obtained from specialized algorithms across a range of datasets. Finally, we present ExplaiNN as a plug and play platform in which pre-trained TF binding models and annotated position weight matrices from reference databases can be combined in a simple framework. We expect that ExplaiNN will accelerate the adoption of deep learning by biological domain experts in their daily genomic sequence analyses.

Introduction

High-throughput genomics methods such as ATAC-seq[2] or ChIP-seq[3], which respectively assess genome-wide accessibility of chromatin and binding of transcription factors (TFs), allow functional annotation of DNA elements. The sheer scale of the data generated by these methods precludes manual analyses. Machine and deep learning have become pervasive in large-scale genomic analyses due to their ability to identify meaningful features in massive datasets (reviewed in [4,5]). For instance, deep learning models have shown superior performance in predicting chromatin accessibility states[6–9], gene expression levels[10–12], and TF binding sites (TFBSs; reviewed in [13]).

While increasingly complex models are now feasible, some are opaque; they do not readily reveal the features and properties that underlie their predictions[14]. For genomics, there are multiple approaches to improve the interpretability of deep learning models, of which we focus on filter visualization and attribution methods.

For convolutional neural networks (CNNs) and related models, a powerful interpretation approach is to visualize the filters of the first convolutional layer as position weight matrices (PWMs), a common format in bioinformatics to represent TFBS patterns[15], by separately aligning the set of sequences activating each filter[16]. The resulting filter PWMs can be compared to reference TF binding profile databases like JASPAR[17] for biological insights. While filter visualization provides an overview of the genomic features that the model has learned in the first layer, the success of this approach is highly dependent on model architectural choices such as max pooling or filter sizes[18]. Moreover, it is difficult to interpret how the model combines the learned motifs within the deeper layers. Furthermore, to gain interpretability at a global level (*i.e.* how each filter influences the model's predictions), the filters must be nullified sequentially, which is both computationally intensive and dependent on arbitrary thresholds[8,9].

Attribution methods quantify the importances of individual nucleotides in the input sequences using forward- (*e.g. in silico* mutagenesis[6,19]) or back-propagation[20,21]. These importance scores can be the basis for further clustering of activating sub-sequences into PWMs[22], which, as with filter visualization, can in turn be compared to known TF motifs for biological insights. Although attribution methods provide local interpretability by identifying important nucleotides in the input sequences, quantifying the contribution of each feature to the overall model's predictions (*i.e.* global interpretability) remains challenging[23]. Noteworthy, neither approach is transparent as to how a model makes predictions.

Unlike deep learning models, linear models are interpretable and transparent: the basis on which they make predictions and the importance of the features they learn are clear. Agarwal and colleagues recently introduced neural additive models (NAMs), which combine features of deep learning and linear models[1]. NAMs compute predictions as a linear combination of outputs from independent deep learning models, each tuned to one input feature, resulting in the levels of explainability much appreciated in linear models without compromising on accuracy.

In this study, we present ExplaiNN (explainable neural networks), a fully interpretable and transparent, sequence-based deep learning model for genomic tasks inspired by NAMs. We evaluate ExplaiNN on different tasks, demonstrating that it performs as well as state-of-the-art models and allows a similar interpretability to more complex approaches, both locally and globally, but in less time and with more ease. Next, we show that the quality of the motifs learned by the convolutional filters of ExplaiNN is equivalent to those discovered by *de novo* tools on the same data. Finally, we showcase the application of ExplaiNN as a plug and play platform for JASPAR PWMs and pre-trained deep learning models with which to interpret genomic data in a transparent manner.

Results

ExplaiNN is a glass box deep learning model for genomics

ExplaiNN is a fully interpretable and transparent deep learning approach for genomic tasks trained on one-hot encoded sequences inspired by NAMs[1]. Predictions are computed as a linear combination of multiple independent CNNs (hereafter referred to as units), each of which consisting of one convolutional layer with a single filter and two fully connected layers (**Fig. 1A**). ExplaiNN provides global interpretability, as the filter of each unit can be readily mapped to a TF profile from JASPAR using Tomtom[24] (**Methods**), thus assigning a biological interpretation to that unit. Besides, the weights of each unit from the final linear layer can be visualized, akin to linear models. ExplaiNN also provides local interpretability by multiplying the output of each unit by the weight of that unit for each input sequence (hereafter referred to as unit importance scores; **Methods**).

As a proof of concept, we applied ExplaiNN to predict the binding of 50 TFs to open chromatin regions (OCRs) from a reference dataset describing the binding of 163 TFs to >1.8M 200-bp long OCRs that we repurposed for this study (**Methods**). A key hyperparameter in ExplaiNN is the number of independent units to be used. To assess the impact of this hyperparameter on model performance, we trained multiple ExplaiNN models using increasing numbers of units (from 1 to 200). As expected, the performance of ExplaiNN improved with the number of units used, plateauing at around 100 units (**Fig. 1B**). For comparison, we evaluated four additional models on the same dataset (**Methods**): a CNN with one convolutional layer (CNN_1); a CNN_1 with exponential activation function (CNN_1Exp); a deep CNN with three convolutional layers (DeepCNN); and DanQ[7], a hybrid deep learning model with convolutional and recurrent layers. Although simpler, ExplaiNN outperformed all three CNN models as measured by average area under the precision-recall curve (AUPRC) and, when using more than 100 units, nearly reached the performance of the more complex DanQ (**Fig. 1B**). Focusing on the ExplaiNN model trained with 100 units, it outperformed the DeepCNN model for most TFs, performing only slightly worse than DanQ (**Fig. 1C)**.

Next, we visualized the filters of each ExplaiNN model and assigned them TF binding modes from JASPAR, which we defined based on the hierarchically clustered groups of DNA-binding profiles included in the database (**Table S1**; **Methods**). As with performance, the number of binding modes recovered by the model increased with the number of units used (**Fig. 1D**). For comparison, we provide the number of binding modes recovered by the

DeepCNN and DanQ when applying filter visualization, as we did with ExplaiNN, or when using TF-MoDISco[22] clustering on DeepLIFT[21] attribution scores (**Methods**). Out of 33 different binding modes associated with the set of 50 TFs analyzed, ExplaiNN models trained with 100 and 300 units recovered 19 and 21, respectively, which is a similar number as DanQ (19 when applying filter visualization and 20 when using DeepLIFT and TF-MoDISco) and greater than obtained for the DeepCNN model (**Fig. 1E**).

An advantageous feature of ExplaiNN is that one can readily visualize the final linear layer weights for global interpretation purposes (**Fig. 1F**; **Methods**). For instance, units with filters annotated as FOX motifs had high positive weights for predicting the FOXA1 class. Similarly, CEBP-, CTCF-, and Ets-like units had high positive weights associated with predicting the classes of CEBP factors, CTCF, and Ets family members, respectively. However, some units had negative weights for predicting the class of their annotated TFs (**Fig. 1F**; highlighted with arrows). To delve further into the contribution of each unit to the prediction of each class, we computed unit importance scores (**Methods**). Visualization of the importance scores of a FOX-like unit in a heatmap confirmed its importance for predicting the FOXA1 and AR classes (**Fig. 1G**; top panel), in agreement with the observation that FOXA1 helps shape AR signaling in prostate cells[25]. Visualizing unit importance scores also revealed why several CTCF-like units had negative weights for predicting the CTCF class: the importances of these units for the CTCF class were negligible, suggesting that the model was not using them to make predictions for that class (**Fig. S1**). The same was true for units annotated as CEBP and Ets with negative weights for those classes (**Fig. S1**). Finally, we calculated the impact from nullifying the FOX-like filters in the DanQ model one at a time (**Fig. 1G**; bottom panel; **Methods**) and, as expected, the impact scores from the filter nullification process were consistent with the unit importance scores.

Taken together, these analyses demonstrated that ExplaiNN performs comparably to more complex models, at least for TF binding prediction. In addition, ExplaiNN provided local and global interpretation quickly and readily compared to using DeepLIFT followed by TF-MoDISco or filter visualization and nullification.

ExplaiNN cannot capture nonlinear interactions between motifs

Given the architecture of ExplaiNN, in which each unit filter is independent of the rest, we expected that it would not be able to learn nonlinear interactions between pairs of TF motifs, including additive and multiplicative effects[26]. In contrast, DeepSTARR is a CNN trained on STARR-seq data to predict the activities of *Drosophila* developmental and housekeeping

enhancers that can capture these types of interactions[27]. For each pair of motifs analyzed, the authors performed a distance dependence analysis by sliding one along randomly generated sequences within which they embedded the second motif. Accounting for additive effects, they observed that the output of DeepSTARR increased when the two motifs were proximal, suggesting that the model had learned nonlinear interactions. To check whether this was also the case for ExplaiNN, we trained it on the same dataset as used for DeepSTARR and compared their performances by calculating the Pearson correlation coefficient (PCC) between predicted and actual enhancer activities (**Methods**). ExplaiNN performed worse than DeepSTARR on both developmental (PCC = 0.61 vs. 0.68) and housekeeping (PCC = 0.71 vs. 0.74) enhancers, which we attributed to the greater presence of nonlinear interactions in this particular dataset. Next, following the specifications from DeepSTARR, we performed a distance dependence analysis between the housekeeping TFs Dref, Ohler1, and Ohler6 (**Methods**). As a negative control, we slid the 5-mer GGGCT. As expected, DeepSTARR was able to learn distance dependencies between the three motifs (**Fig. S2**). This was not true for ExplaiNN: during the analysis, the resulting model outputs using the three motifs were similar to sliding GGGCT (**Fig. S2**). Therefore, as anticipated based on the model architecture, ExplaiNN is not suitable for nonlinear tasks such as detecting motif interactions.

ExplaiNN learns high-quality motifs comparable to *de novo* motif discovery tools

De novo motif discovery methods continue to emerge and improve[28–32]. With the dramatic escalation in the size of datasets, the execution time of these methods is increasingly a limitation. Furthermore, many *de novo* motif discovery methods are assay-specific, as exemplified by the DREAM5 challenge evaluation on protein binding microarray (PBM) data[33], requiring an extensive adjustment of parameters for their application to different assays. We sought to explore the capacity of ExplaiNN for efficient *de novo* motif discovery within a unified platform. For each of the 163 TFs from the previous dataset, we trained a model with 100 units and then visualized the filters and importance scores of each unit, resulting in 100 PWMs for each TF (**Methods**). As expected, PWMs derived from visualizing filters associated with important units performed better: for 139 TFs (85.3%), the best performing PWM was derived from the filter of one of the 10 most important units (**Fig. 2A**). Next, we applied STREME[31], a state-of-the-art method for *de novo* motif discovery, on the same sequences used for training the ExplaiNN models to discover 100 motifs for each TF (**Methods**). Pairwise comparison of the performances of the best PWMs obtained by each method revealed subtle differences (**Fig. 2B**), although PWMs discovered by STREME were superior for TFs with small training datasets (**Fig. 2C**). Notably, the execution times exhibited

by the two methods differed greatly, with ExplaiNN being >100 times faster for TFs with large training datasets of ≥50,000 sequences (**Fig. 2D**). These differences were consistent with a previous report related to the benefits of GPU-enabled *de novo* motif discovery[30].

Motivated by the success of ExplaiNN in discovering motifs in *in vivo* data and to demonstrate its capacity on data from various assays, we benchmarked ExplaiNN against other *de novo* methods, but this time using *in vitro* data (**Methods**). We downloaded publicly available HT-SELEX[34], PBM[35], and SMiLE-seq[36] data for the TF GATA3, as well as the PWMs discovered in these datasets by three assay-specific *de novo* motif discovery methods: Autoseed[37] (HT-SELEX), Seed-and-Wobble[38] (PBM), and a hidden Markov model (HMM) approach (SMiLE-seq). The performance of the best PWMs derived with ExplaiNN was consistent regardless of the type of assay (**Fig. 2E**), as were their logos (**Fig. 2F**). Focusing on specific assays, the capacity of ExplaiNN to discover *de novo* motifs in the *in vivo* and SMiLE-seq data, as measured by the performance of the best PWMs derived, was comparable to that of STREME and the HMM-based method, respectively, while outperforming Autoseed and Seed-and-Wobble in their corresponding assays (**Fig. 2E**). All GATA3 logos were very similar to each other, and were also similar to the logo from JASPAR for this TF profile (MA0037.4), derived originally by applying RSAT[32] on a mouse Gata3 ChIP-seq data from ReMap[39] (**Fig. 2F**). Taken together, this supports the potential of ExplaiNN as a fast, universal method for *de novo* motif discovery.

ExplaiNN recapitulates the *cis*-regulatory lexicon of immune cell differentiation

To further explore the capabilities of ExplaiNN on distinct data types, we compared its performance against AI-TAC in predicting chromatin accessibilities in 81 immune cell types from 6 different lineages[9]. We started with an exploratory analysis to determine the optimal number of units to train ExplaiNN. Saturation in model performance by means of average PCC between predicted and actual ATAC signals, as well as in the number of well-predicted sequences, occurred at ~250 units (**Fig. 3A**; **Methods**), however, we decided to use 300 units, which is the same number of convolutional filters used in the first layer of AI-TAC. The performance of ExplaiNN by means of average PCC was comparable to that of AI-TAC (1-2% difference; **Fig. 3A**), and the PCCs of individual sequences correlated well between the two models (**Fig. 3B**), however, AI-TAC correctly predicted more sequences (**Fig. 3A**). Next, we visualized both the filters and weights of each unit of the ExplaiNN model, identifying the same lineage-specific TF motifs reported in AI-TAC without having to undergo the computationally intensive process of filter nullification (**Fig. 3C**): NFE2, NFI, and GATA (in stem cells); POU, EBF, and PAX (in B cells); TCF3, TCF7, Ets, and AP1 (in T cells); NR1

(nuclear receptor type 1), TBX, and REL (in innate lymphoid cells); and SPI, Krüppel zinc fingers, and CEBP (in myeloid cells). Moreover, we visualized the importances of each unit to understand their influence on the model's predictions. For example, CEBP- and PAX-like units were important for predicting accessibility in most cell types of the myeloid and B lineages, respectively (**Figs. 3D** and **E**). Taken together, using ExplaiNN, we replicated the results of AI-TAC without the need to apply complex and time-consuming interpretation techniques by simply visualizing the weights and importances of each unit.

ExplaiNN is suitable for the analysis of single-cell chromatin accessibility data

Single-cell (sc) sequencing methods enable profiling of a wide range of genomic information in individual cells (reviewed in [40]), including chromatin accessibility. To explore the utility of ExplaiNN for deciphering *cis*-regulatory properties from granular sc data, we reanalyzed a recent scATAC-seq dataset cataloging 228,873 OCRs across 15,298 human pancreatic islet cells that were grouped into 12 clusters based on their accessibility profiles[41]. We trained a ExplaiNN model with 400 units (*i.e.* the number of units at which model performance plateaued) on the sc data to predict the activity of each OCR across the 12 clusters, and visualized both the filters and weights of each unit. We observed that the weights of some units exhibited cell type-specific patterns that had also been found using chromVAR[42] in the original study (**Fig. 4A**). For instance, PDX-like units had high positive weights for beta and delta cells, MAF-like units for alpha and beta cells, HNF1-like units for alpha and gamma but not for ductal cells, and FOX-like units for alpha, beta and gamma cells. However, there were also differences: some units did not exhibit high positive weights in expected cell types (*e.g.* Ets-like units had negative weights in endothelial cells), while others exhibited cell type-specific patterns not reported in the original study (**Fig. 4A**). Next, we visualized the importances of each unit for their contributions to the physiological stratification of the cells, finding that RFX-like units were important for predicting OCRs in hormone-high (*i.e.* alpha, beta, and delta type 1 cells) but not in hormone-low cells (*i.e.* alpha, beta, and delta type 2 cells) (**Fig. 4B**). It was the opposite for AP1-like units: they were important for hormone-low but not hormone-high cells (**Fig. 4C**). Taken together, ExplaiNN was able to reproduce and expand on the results from the original study while demonstrating its utility for the analysis and interpretation of scATAC-seq data.

ExplaiNN as a plug-and-play platform for TF motifs and deep learning models

In ExplaiNN, a key step during model interpretation is annotating each unit to ease biological interpretation. Given that ExplaiNN models can be conceptualized as a PWM scanning layer feeding into fully connected layers, we reasoned that initializing the weights of each unit filter

with a JASPAR profile would facilitate the interpretation process because the biological annotations of the units would be known beforehand. To confirm this, we trained ExplaiNN models with increasing numbers of units (from 300 to 1,492) on the AI-TAC dataset in which the filter weights of each unit had been initialized with JASPAR profiles (**Fig. 5A**; **Methods**). During training, the filter weights were frozen to prevent them from being refined (*i.e.* the models were only allowed to learn the weights of the fully connected and final linear layers). These JASPAR-initialized, frozen models, even the largest attempt with 1,492 units, performed much worse than both AI-TAC and an ExplaiNN model with 300 units trained from scratch (**Fig. 5B**). Still, the importances of some units were informative. For example, a unit whose filter weights had been initialized with the profile of Lef1 (MA0768.2) was important for predicting accessibility in T cells (**Fig. S3**), in agreement with the role of this TF in establishing T cell identity[43]. We attributed the overall poor performance of these frozen models to the fact that many JASPAR profiles used to initialize the filters might be from TFs irrelevant to immune cells and, therefore, a refinement process would be required to allow them to better resemble the motifs of relevant TFs. Indeed, unfreezing the filter weights during training improved the performance of the model, approaching that of AI-TAC (**Fig. 5B**). Allowing refinement resulted in >35% of the filters undergoing substantial changes; their visualization as PWMs revealed that they had become different from the original JASPAR profiles used for their initialization (Tomtom[24] *q*-value >0.05). For example, a unit whose filter weights had been initialized with the profile of TFAP2C (MA0815.1), and whose importance across the different immune lineages when freezing the filter weights during training was null, became important for predicting accessibility in B cells. Its filter was refined to such an extent that when visualized as a PWM it resembled the motif of EBF1, an important TF for maintaining B cell identity[44] (**Fig. 5C**).

A limitation of the JASPAR-initialization-and-freezing approach was the inability to distinguish between TFs from the same family (*i.e.* TFs sharing the same class of DNA-binding domain), as they often have highly similar DNA-binding specificities[45]. In order to generate individual units that provide greater resolution, we implemented a transfer learning strategy (**Fig. 5D**; **Methods**): We pre-trained 350 single-task DanQ models, each predicting the binding of a single TF to the mouse genome, and initialized an ExplaiNN model with 350 units in which we replaced the layers of each unit with one of the pre-trained DanQ models. Akin to NAMs, we initialized a second model in which we added two fully connected layers after the DanQ models. The AUPRCs of the pre-trained DanQ models ranged from 0.51 (E4f1) to 0.96 (Snai2), with a median of 0.78 across all models (**Table S2**). Then, we fine-tuned both ExplaiNN models on the AI-TAC dataset, freezing the pre-trained DanQ models (*i.e.* their weights were not modified). The performance of the fine-tuned

models almost reached that of AI-TAC (PCC = 0.341 vs. 0.345 for AI-TAC). Next, we visualized the importances of each DanQ model unit in the ExplaiNN model without fully connected layers, allowing us to disambiguate the contribution of individual TF family members (**Methods**). For instance, the Irf4 and Irf8 units were important for predicting accessibility in different immune lineages (**Fig. 5E**), in agreement with the distinct roles of these TFs: Irf4 regulates B, T, myeloid, and dendritic cell differentiation[46], while Irf8 regulates B and myeloid cell lineages[47]. Similarly, the Pax5 unit was important for predicting accessibility in B cells, consistent with the role of this TF in establishing B lineage identity and function[48]; in contrast, the importances of the Pax3 and Pax7 units across immune lineages were negligible, in agreement with their role in regulating myogenesis[49] (**Fig. 5F**). Finally, we applied UMAP[50] to cluster the sequences based on their unit outputs in the second ExplaiNN model, resulting in three main clusters that were associated with ATAC signals in alpha/beta T, myeloid, and B cells (**Figs. 5G** and **S4**). The outputs of some units were in strong agreement with a biologically relevant cluster. For example, the Bcl11b unit outputs were confined within the boundaries of the alpha/beta T cell cluster, consistent with its role in the differentiation and survival of these lymphocytes[51], the Cebpa unit outputs within the myeloid cluster, in agreement with its role in myeloid differentiation[52], and the Ebf1 unit outputs within the B cluster (**Figs. 5G** and **S4**). Taken together, replacing the units of ExplaiNN with pre-trained high-resolution TF binding models achieved performance levels comparable to state-of-the-art deep learning models while providing the additional value of gaining insights into the roles of individual TFs. This flexibility of incorporating different components into ExplaiNN offers the potential for increased performance while retaining interpretability.

Discussion

ExplaiNN is an explainable deep learning approach designed for straightforward discovery of genomic features contributing to model performance for a broad range of predictive tasks related to DNA sequence data. Inspired by the recently introduced NAMs method, the architecture of ExplaiNN is based on multiple, simple, independent CNN units that recognize sequence patterns. The outputs of these units are subsequently combined in a manner akin to classic regression analysis techniques. We benchmarked ExplaiNN on diverse tasks, including sequence-based prediction of TF binding, chromatin accessibility, both from bulk and single-cell data, enhancer activity, and *de novo* motif discovery. Through this array of diverse applications, ExplaiNN performed comparably to leading methods specialized for the independent tasks.

ExplaiNN combines the expressiveness of CNNs and interpretability of linear models, thus providing a transparent and interpretable view of the model's decision making without sacrificing predictive capacity. While deep learning models have been successful at revealing novel biological insights from high-throughput sequencing data, outperforming traditional PWM-based approaches such as motif enrichment analysis, their adoption in the genomics field remains largely in the hands of deep learning experts. This is in part a consequence of their perceived opaque nature and lack of consensus on best practices for their interpretation, making it difficult for the non-specialist to apply these methods routinely. There is a growing need for deep learning models that are explainable and easy to use[14]. ExplaiNN is one such approach. Because of its linearity, it can be conceptualized as a simple combination of TF binding motifs, either discovered *de novo* or user-provided, however, for this very reason, it excludes higher-order interactions between them (*e.g.* TF cooperativity). It may therefore be surprising to some that in practice ExplaiNN performs so well across a wide range of predictive tasks. One reason behind this could be that excluding higher-order interactions acts as a regularizer leading to more accurate predictions than multi-layer CNN models on simple tasks such as TF binding prediction. Another reason could be that the presence of higher-order interactions in the genomics explored datasets could be marginal, in agreement with recent studies suggesting that *cis*-regulatory logic may be simpler than believed[53,54]. Alternatively, these kinds of complex interactions may be highly specific to individual enhancers, and thus not detectable by models generalized for analysis across the genome. As exemplified by the DeepSTARR analysis in this report, comparisons between ExplaiNN and non-linear models can enable researchers to assess the benefit from allowing non-linear interactions. In such cases where there is not substantial benefit, the use of the simpler, readily interpretable method may be preferred.

The emergence of new methods for regulatory sequence analysis is a recurrent process. An important early approach to identifying recurring subsequences associated with TFBSs used the expectation-maximization algorithm to learn a PWM from a set of unaligned regulatory sequences[55]. These individually learned PWMs could then be used in a linear model to make predictions regarding various properties of a given sequence[56]. ExplaiNN could be perceived as a direct extension of this approach, wherein a large number of PWMs are simultaneously learned, and their activation values linearly combined to make predictions in an end-to-end fashion.

As a simple linear model, ExplaiNN has the potential to be applied to multiple problems. This flexibility offers the opportunity for users to become proficient with the system as they work across or within topics. As presented here for *de novo* motif discovery, the flexibility to apply ExplaiNN to both ChIP-seq data and multiple special classes of *in vitro* TF binding data has the promise to reduce the number of methods that must be mastered.

There remains an ample variety of potential directions for further development of ExplaiNN, split between enhanced technical capabilities and user empowerment. It should be possible to integrate a more complex model architecture within the system to account for the “missing complexity” (*e.g.* non-linearity). As implemented, ExplaiNN cannot address analyses requiring large receptive fields (*e.g.* Basenji[57] or Enformer[12]), but it should be possible to implement such capacity within the system. Incorporating more advanced visualization techniques and data processing tools would ease the adoption of ExplaiNN by scientists working on applied problems who currently rely on heritage methods for regulatory sequence analysis.

The software for using ExplaiNN is provided open-source in a well-documented repository, and it is our hope that it will lead to or inspire widespread use of interpretable deep learning methods.

Online methods

Model architectures

Unless otherwise specified, each ExplaiNN unit consisted of:

- 1st convolutional layer with 1 filter (19x4), batch normalization, exponential activation and max pooling (7x7);
- 1st fully connected layer with 100 nodes, batch normalization, ReLU activation and 30% dropout; and
- 2nd fully connected layer with 1 node, batch normalization and ReLU activation.

The final layer of ExplaiNN is linear: one fully connected output layer with *n* outputs (*e.g.* 50 in the initial TF binding prediction task or 81 when predicting chromatin accessibility states across immune cells).

For the CNN_1 and CNN_1Exp architectures, we used the following specifications:

- 1st convolutional layer with 100 filters (19x4), batch normalization, ReLU (for CNN_1) or exponential (for CNN_1Exp) activation and max pooling (7x7);
- 1st fully connected layer with 1000 nodes, batch normalization, ReLU activation and 30% dropout;
- 2nd fully connected layer with 1000 nodes, batch normalization, ReLU activation and 30% dropout; and
- Fully connected output layer with 50 outputs.

For the DeepCNN (adapted from Basset[8]):

- 1st convolutional layer with 100 filters (19x4), batch normalization, ReLU activation and max pooling (3x3);
- 2nd convolutional layer with 200 filters (7x1), batch normalization, ReLU activation, and max pooling (3x3);
- 3rd convolutional layer with 200 filters (4x1), batch normalization, ReLU activation, and max pooling (3x3);
- 1st fully connected layer with 1000 nodes, batch normalization, ReLU activation and 30% dropout;
- 2nd fully connected layer with 1000 nodes, batch normalization, ReLU activation and 30% dropout; and
- Fully connected output layer with 50 outputs.

For DanQ[7]:

- 1st convolutional layer with 320 filters (26x4), ReLU activation, 20% dropout, and max pooling (13x13);
- 2 bi-directional LSTM layers with hidden state size 320 and 50% dropout;
- 1st fully connected layer with 925 nodes and ReLU activation; and
- Fully connected output layer with 1 or 50 outputs (depending on the task).

For DeepSTARR[27]:

- 1st convolutional layer with 256 filters (7x4), padding of size 3, batch normalization, ReLU activation function and max pooling (2x2);
- 2nd convolutional layer with 60 filters (3x1), padding of size 1, batch normalization, ReLU activation function and max pooling (2x2);
- 3rd convolutional layer with 60 filters (5x1), padding of size 2, batch normalization, ReLU activation function and max pooling (2x2);
- 4th convolutional layer with 120 filters (3x1), padding of size 1, batch normalization, ReLU activation function and max pooling (2x2);
- 1st fully connected layer with 256 nodes, batch normalization, ReLU activation function and 40% dropout;
- 2nd fully connected layer with 256 nodes, batch normalization, ReLU activation function and 40% dropout; and
- Fully connected output layer with 2 outputs.

The AI-TAC architecture was provided by Maslova and colleagues[9]. All architectures were implemented using the PyTorch framework[58].

Interpretability with ExplaiNN

First, we converted the filter of each unit into a PWM by following the specifications from the AI-TAC manuscript: For each filter, we built a position frequency matrix (PFM) by aligning all 19-mers (*i.e.* 19 bp-long DNA sequences) activating that filter's unit by ≥50% of its maximum activation value in correctly predicted sequences. The resulting PFM was then transformed into a PWM by setting the background uniform nucleotide frequency to 0.25. Next, the PWM derived from each filter was mapped to one or more profiles from the vertebrate collection of the JASPAR 2020 database[17] using Tomtom[24] (version 5.3.0; q-value ≤0.05), which, in turn, were used to annotate that filter's unit. For example, a unit whose filter's PWM were similar to the profiles of members from the SOX TF family would be annotated as "SOX-like".

For global interpretability, we visualized the weights of each unit in the final linear layer of the model (*e.g.* using a heatmap). Usually, these weights were associated with the importance of the unit for each task. In some cases, however, they were highly positive or negative for a task in which the unit was not activated by the input sequences. To overcome this limitation and obtain more fine-grained unit importances, for each unit and for each task, we computed the product of the unit's activation for each sequence with the final layer weight for that task, and visualized them (*e.g.* using a boxplot). For visualization, for each unit, we only included the products from correctly predicted sequences activating that unit's filter by ≥50% of its maximum activation value. In addition, for each unit and for each task, the median of these products was used to assess the importance of that unit for that task.

Interpretability with DeepLIFT and TF-MoDISco

For each correctly predicted sequence in the test set, we generated DeepLIFT[21] importance scores with 10 reference sequences using the Captum library[59]. We used TF-MoDISco[22] with default settings to obtain motifs from DeepLIFT importance scores.

Training, validation and test datasets

To obtain human *in vivo* TF binding data, we repurposed a previously described data matrix aggregating the binding of 163 TFs to 1,817,918 200-bp-long DNase I hypersensitive sites (DHSs) in 52 cell and tissue types[60]. For each TF-DHS pair, a "1" was used to indicate that the DHS was accessible and bound by the TF (*i.e.* the DHS and at least one ChIP-seq peak summit of that TF from ReMap 2018[39] overlapped), a "0" that the DHS was accessible but not bound by the TF, and a null sign ("∅") that it was not accessible to the TF for binding (*i.e.* unresolved). For the CNN_1, CNN_1Exp, DeepCNN, DanQ, and ExplaiNN models predicting the binding of 50 TFs, we extracted a slice of the matrix including the row vectors of the 50 TFs, removing any column vectors with unresolved elements. The resulting resolved regions for all 50 TFs were randomly split into training (80%), validation (10%) and test (10%) sets using the "train_test_split" function from scikit-learn[61] (datasets were always randomly split in this way). For *de novo* motif discovery, since the number of bound versus unbound regions of all TFs were imbalanced, we subsampled the set of unbound regions to a 50:50 ratio for each TF while accounting for their %GC content distributions. The resulting datasets were randomly split into training (80%), validation (10%) and test (10%) sets while maintaining an equal proportion of bound and unbound regions.

Mouse *in vivo* TF binding data was obtained as follows: First, we downloaded non-redundant mouse ChIP-seq peaks for 350 DNA-binding TFs[62] from ReMap. We resized each ChIP-seq

peak to 201 bp by extending its summit 100 bp in both directions. Furthermore, we retrieved an atlas of 1,802,604 DHS regions in the mouse genome[63], which we also resized to 201 bp around the center of each DHS. In both cases, we applied BEDTools slop[64]. We then created a set of negative sequences for each TF by subsampling non-overlapping regions from the DHS atlas while matching the %GC content distribution of its ChIP-seq peaks. The resulting sequences for each TF were randomly split into training (80%), validation (10%) and test (10%) sets while keeping the same ratio between positive and negative sequences.

PBM data was downloaded from UniPROBE[35] (Gata3; UP00032; clone ID pTH1024). Probe signal intensities were quantile normalized (QN) using the “quantile_transform” function from scikit-learn. Probes were 60 bp long, including both the de Bruijn and linker sequences. The arrays AMADID #015681 and #016060 were used for training and validation, respectively.

HT-SELEX[34] and SMiLE-seq[36] data were retrieved from the Sequence Repository Archive (run ids: ERR1003435, ERR1003437, ERR1003439, ERR1003441, and SRR3405148). For HT-SELEX data, we treated each cycle as an independent class as in Asif and Orenstein[65], thereby removing the need for negative sequences. Reads were randomly split into training (80%) and validation (20%) sets while preserving the proportions between reads from each cycle. For SMiLE-seq data, reads were left- and right-clipped 7 and 64 bp, respectively, for a final length of 30 bp corresponding to the randomized DNA. A set of negative sequences were obtained by dinucleotide shuffling using BiasAway[66] (version 3.3.0). Sequences were randomly split into training (80%) and validation (20%) while maintaining an equal proportion between positives and negatives.

Binarised human pancreatic islet scATAC-seq data was obtained from [41]. Data was denoised by training PeakVI[67] with 21 latent dimensions using scvitools[68]. The denoised profiles were used to compute the mean accessibility of each peak in the 12 clusters identified in the original study. Peaks were resized to 600 bp around the center of each peak using BEDTools slop. Peaks from chromosome 1 were used for validation, peaks from chromosomes 10, 11, and 12 for testing, and peaks from the remaining chromosomes for training.

The AI-TAC and DeepSTARR datasets were obtained from the original publications, and we used the same data splits.

Model training

Unless otherwise specified, we trained all models using the Adam optimizer[69]. We applied one-hot encoding to convert nucleotides into 4-element vectors (*i.e.* A, C, G, and T), set the learning rate to 0.003 and batch size to 100, and used an early stopping criteria to prevent overfitting when the model performance on the validation set did not improve. The training and validation sets included the reverse-complement of each sequence. Sequences with Ns were discarded.

De novo PWMs

For each of the 163 TFs from the data matrix, we trained one ExplaiNN model with 100 units using its training and validation sets, and then visualized the filters and importance scores of each unit, resulting in 100 PWMs per TF. As baseline, we applied STREME[31] (version 5.3.0) on the set of training sequences of each TF to also generate 100 PWMs: We set the fraction of sequences held-out for *p*-value computation to 0 (option --hofract), the maximum length of the PWMs to 19 bp (*i.e.* the filter size in ExplaiNN; option --maxw), and the number of output PWMs to 100 (option --nmotifs). Next, for each TF and for each of its PWMs, we evaluated the PWM performance by keeping the score of the best hit from scanning that PWM along both strands of each sequence in the test set and computing the AUPRC.

Like above, for each GATA3 *in vitro* assay (*i.e.* HT-SELEX, PBM and SMiLE-seq), we trained one ExplaiNN model with 100 units using the training and validation sets, and visualized the filters of each unit, resulting in 100 PWMs for each assay. However, the task of the ExplaiNN model trained on HT-SELEX data was multiclass classification: the model tried to predict the cycle of origin of each input read, with the expectation that the last cycles would be enriched for bound sequences of the TF. In addition, for PBM data, the task of the ExplaiNN model was to infer QN intensity signals of each probe.

For each TF and for each assay, the best PWM was selected based on its performance on the corresponding validation set.

Plug-and-play

First, we resized all 746 profiles from the JASPAR 2020 vertebrate collection to 19 bp. Then, we removed redundant profiles based on their Tomtom similarity, obtaining four different sets of non-redundant profiles of sizes 150, 250, 375 and 500. Next, we computed the reverse complement of each profile, doubling the number of profiles in each non-redundant set to 300, 500, 750 and 1000, and raising the number of profiles in the JASPAR 2020 vertebrate

collection to 1492. We trained ExplaiNN models with increasing numbers of units (300, 500, 750, 1000, and 1492) on the AI-TAC dataset in which the filter weights of each unit had been initialized with the JASPAR profiles. To initialize filter weights with profiles from JASPAR, we followed the specifications from DanQ: We reformatted JASPAR profiles as PWMs using Biopython[70] and then converted them to filter weights by subtracting 0.25 from the probability of each nucleotide at each PWM position. During training, nullification of gradients could be applied to freeze the filter weights.

For each of the 350 TFs from the mouse *in vivo* TF binding data, we trained a DanQ model using the training and validation sets of that TF, and assessed its performance by computing the AUPRC on the test set. We then scored the sequences from the AI-TAC dataset keeping the outputs from each model (pre- or post-sigmoid). Next, we combined the outputs of each sequence in one fully connected layer with 81 output nodes. Alternatively, each DanQ output could be embedded in its own fully-connected neural network with one layer and 200 nodes, resulting in 350 processed outputs ultimately combined in one fully connected layer with 81 outputs.

UMAP clusters were obtained and plotted using the UMAP Python library[50] (version 0.5.2).

Code availability

ExplaiNN is open-source software distributed under the MIT license. It is available on GitHub (https://github.com/wassermanlab/ExplaiNN) accompanied by the IPython notebooks that we used for the different experiments.

Data availability

The TF binding matrices, and the code for generating them, are available on GitHub as 2D numpy arrays[71] (https://github.com/wassermanlab/TF-Binding-Matrix). The filter weights from JASPAR profiles, and the code for generating them, are also available on GitHub as pickles (https://github.com/wassermanlab/PWM-to-filter-weights).

Author contributions

MS and GN conceived the idea; OF, GN and MS designed and performed the analyses; GN, MS and OF wrote the code; GN developed the Python library; OF and GN created the figures with input from MS; WW and SM provided advice on all aspects of the work; all authors participated in the writing and editing of the manuscript.

Acknowledgements

We thank the members of the Mostafavi and Wasserman labs for providing useful feedback. We thank Dora Pak and Jonathan Chang for administrative and IT support, respectively. We thank UBC's Advanced Research Computing (ARC) (https://arc.ubc.ca/) for enabling HPC.

Funding

GN was supported by an International Doctoral Fellowship from the University of British Columbia. OF, MS and WWW were supported by grants from the Canadian Institutes of Health Research (PJT-162120), Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant (RGPIN-2017-06824), and the BC Children's Hospital Foundation and Research Institute. SM acknowledges support from the Canadian Institute for Advanced Research (CIFAR). Equipment enabled by NSERC Research Tools and Instruments Grant (RTI-2020-00778) to SM and WWW.

Figures

Fig. 1: The ExplaiNN model and its application to TFBS prediction in OCRs. (**A**) ExplaiNN takes as input one-hot encoded DNA sequences. Architecturally, it is composed of multiple independent CNNs (*i.e.* units), each of which comprising one convolutional layer with a single filter, batch normalization, exponential activation, and max pooling, and two fully connected layers with batch normalization, ReLU activation, and dropout. The final linear layer of ExplaiNN (*i.e.* the output) combines the outputs from each unit (denoted here using Xs). (**B**) Performances (AUPRC; y-axis) of ExplaiNN models trained using increasing numbers of units (x-axis) on predicting the binding of 50 TFs in OCRs (green line). The performances of DanQ[7], a deep CNN with 3 convolutional layers (*i.e.* DeepCNN), and two shallow CNNs with 1 convolutional layer (*i.e.* CNN_1 and CNN_1Exp featuring an exponential activation function instead of ReLU) are provided as baselines (gray lines). (**C**) Pairwise comparison of the individual performances (AUPRC) of the 50 TFs from the previous dataset between the ExplaiNN model trained using 100 units and either DanQ (green dots) or the DeepCNN (gray dots). (**D**) Number of binding modes (y-axis) detected with ExplaiNN using increasing numbers of units (x-axis). (**E**) Number of binding modes detected with DanQ, the DeepCNN, and ExplaiNN trained using 100 and 300 units on the previous dataset using either filter visualization or TF-MoDISco[22] clustering on DeepLIFT[21] attribution scores (pinstriped). The 50 TFs in the dataset are represented by 33 unique binding modes (dashed line), some of which are detected using different combinations of models and interpretive approaches (green); other detected binding modes (*i.e.* different from those 33) are shown in grey. (**F**) Heatmap of the final linear layer weights of the ExplaiNN model trained using 100 units, with rows representing units with assigned biological annotations based on their Tomtom[24] similarity to known TF profiles from the JASPAR database[17] and columns representing the 50 TFs predicted by the model. More than one filter can learn the same TF motif representation, but some may not contribute to the model's predictions (black arrows).

(**G**) (top) Visualization of importance scores for a unit annotated as FOXA1 from the ExplaiNN model trained using 100 units. This unit contributes the most to the prediction of FOXA1 binding. (bottom) Filter nullification analysis for the seven DanQ filters annotated as FOX TFs. The results are consistent with the unit importance scores. AUPRC, area under the precision-recall curve; CNN, convolutional neural network; OCR, open chromatin region; ReLU, rectified linear unit; TF, transcription factor; TFBS, TF binding site.

Fig. 2: *De novo* motif discovery with ExplaiNN. (**A**) Average performances (AUPRC; y-axis) by rank (x-axis) of PWMs derived from training ExplaiNN models with 100 units on *in vivo* datasets of 100 TFs and then visualizing the filter of each unit (*i.e.* 100 PWMs per TF). The rank of each PWM is given by the importance of its unit. The gray dots indicate the rank of the best performing PWM for each TF. (**B**) Pairwise comparison of the individual performances (AUPRC) of the best PWMs derived for each TF from the previous dataset using ExplaiNN (y-axis) or STREME (x-axis). (**C**) Performance difference (*i.e.* ΔAUPRC) of the previous PWMs (x-axis) is plotted with respect to the dataset size of the corresponding TF (x-axis). (**D**) Execution time (in seconds; y-axis) of the *de novo* motif discovery application of ExplaiNN (green dots) and STREME (gray dots) is plotted with respect to the dataset size of the corresponding TF (x-axis). (**E**) Performances (AUPRC; y-axis) of PWMs derived from different experimental assay datasets related to the TF GATA3 by different methods (x-axis), including ExplaiNN (green bars) and four assay-specific methods (grey bars). (**F**) GATA3 logos derived from the dataset of each experimental assay using ExplaiNN or the assay-specific method. The JASPAR[17] logo for this TF profile (MA0037.4), derived by applying RSAT[32] on the mouse Gata3 ChIP-seq data from ReMap[39], is shown at the top. AUPRC, area under the precision-recall curve; HMM, hidden Markov model; PBM, protein binding microarray; PWM, position weight matrix; S&W, Seed-And-Wobble; TF, transcription factor.

Fig. 3: Application of ExplaiNN in predicting chromatin accessibility in the mouse immune system. (**A**) Performances (average PCC; y-axis; green) and number of well-predicted sequences (secondary y-axis; gray) for ExplaiNN models (solid lines) with increasing numbers of units (x-axis) and AI-TAC (dashed lines) trained on OCRs in 81 immune cell types from 6 different lineages[9]. (**B**) Pairwise comparison of the individual performances (PCC) of the OCRs from the previous dataset between the ExplaiNN model trained using 300 units (y-axis) and AI-TAC (x-axis). The Pearson correlation coefficient (R) of the individual OCR performances between the two methods is shown at the lower right corner. (**C**) Heatmap of the final linear layer weights of the ExplaiNN model trained using 300 units, with columns representing units with assigned biological annotations based on their

Tomtom[24] similarity to known TF profiles from the JASPAR database[17] and rows representing the 81 immune cell types coloured by lineage: stem cells (navy blue), B cells (turquoise), alpha/beta (forest green) and gamma/delta T cells (olive green), innate lymphoid cells (pink), and myeloid cells (purple). The logos derived from visualizing the filters of selected lineage-specific units are shown at the right. (**D** and **E**) Visualizations of importance scores coloured by lineage for two units from the previous model annotated as CEBP and PAX, revealing their importance to the prediction of chromatin accessibilities in myeloid[52] and B cell lineages, respectively. The logos of the filters of these units are shown at the top. OCR, open chromatin region; PCC, Pearson correlation coefficient; TF, transcription factor.

Fig. 4: ExplaiNN analysis and interpretation of scATAC-seq data of human pancreatic islets. (**A**) Heatmap of the final linear layer weights of an ExplaiNN model with 400 units trained on OCRs from human pancreatic islet sc data[41], with rows representing units with assigned biological annotations based on their Tomtom[24] similarity to known TF profiles from the JASPAR database[17] and columns representing the 12 clusters of cells based on their accessibility profiles and coloured by cell type: alpha type 1 cells (purple), beta type 1 cells (navy blue), delta type 1 cells (turquoise), alpha type 2 cells (forest green), beta type 2 cells (light green), delta type 2 cells (yellow), acinar cells (olive green), ductal cell (orange), endothelial cells (red), gamma cells (brown), immune cells (dark brown), stellate cells (gray). The logos derived from visualizing the filters of selected units are shown at the right. (**B** and **C**) Visualizations of importance scores coloured by cell cluster for two units from the previous model that were annotated as RFX and AP1 (TRE site), revealing their respective importance to the prediction of chromatin accessibilities in hormone-high and hormone-low cells. OCR, open chromatin region; scATAC-seq, single-cell ATAC-seq; TF, transcription factor.

Fig. 5: Initializing ExplaiNN with JASPAR profiles and DanQ models. (**A**) Schematic representation of the proposed ExplaiNN model in which the convolutional filters of each unit are initialized with JASPAR[17] profiles. (**B**) Performances (average PCC; y-axis) and number of well-predicted sequences (secondary y-axis; black dots) for different models trained on OCRs from the AI-TAC dataset[9]: an ExplaiNN model with 300 units (green) and different ExplaiNN models trained with increasing numbers of units (300, 500, 750, 1000, 1492) initialized with JASPAR profiles (light green). The filters from the JASPAR-initialized ExplaiNN models could be freezed during training (pinstriped). The performance of AI-TAC is provided as baseline (gray). (**C**) Refinement example for a filter whose weights had been initialized with the JASPAR profile of TFAP2C (MA0815.1). Freezing the filter weights during training was detrimental (top): the importance scores of that filter's unit across all immune

lineages was null. However, if filter weight refinement was allowed during training (*i.e.* no freezing; bottom), that filter was modified until it resembled the motif of EBF1, an important TF for maintaining B cell identity[44], and that same unit became important for predicting accessibility in B cells. (**D**) Schematic representation of the proposed transfer learning strategy in which ExplaiNN units are replaced with pre-trained DanQ[7] models that can be followed (right) or not (left) by fully connected layers. (**E** and **F**) We used the transfer learning strategy on the left to train one ExplaiNN model with 350 units on the AI-TAC dataset in which the units had been replaced with 350 different pre-trained DanQ models, each predicting the binding of a single TF to the mouse genome. During the training process of the ExplaiNN model, the DanQ models were frozen (*i.e.* their weights were not modified). Unit importance scores of DanQ models belonging to different members of the IRF and PAX TF families are shown. (**G**) We repeated the same process but using the transfer learning strategy on the right: each unit was replaced with a pre-trained DanQ model but adding two fully connected layers after each model. Then, we applied UMAP[50] to cluster the OCRs based on their unit outputs. (top) UMAP clusters display cell-type specificity (*e.g.* alpha/beta T and myeloid cells). (bottom) The outputs of the Bcl11b and Cebpa DanQ model units strongly agree with their biologically relevant clusters. OCR, open chromatin region; PCC, Pearson correlation coefficient; TF, transcription factor; UMAP, uniform manifold approximation and projection.

Fig. S1: From top to bottom, visualization of importance scores for three units annotated as CTCF, one unit annotated as CEBP, and one unit annotated as Ets from an ExplaiNN model trained using 100 units on predicting the binding of 50 TFs in OCRs. OCR, open chromatin region.

Fig. S2: Cooperativity (residual fold change; y-axis) plotted as a function of distance (x-axis) between the motifs of the housekeeping TFs Dref (top row), Ohler1 (middle row), and Ohler6 (bottom row) for DeepSTARR[72] (left column) and ExplaiNN (right column). The 5-mer GGGCT is provided as a negative control (blue). TF, transcription factor.

Fig. S3: Visualization of importance scores coloured by lineage of a unit from an ExplaiNN model initialized with 300 JASPAR profiles and trained on the AI-TAC dataset[9] with freezing. The unit, which corresponded to the JASPAR profile of Lef1 (MA0768.2), was important for predicting accessibility in T cells, in agreement with the role of this TF in establishing T cell identity[43]. TF, transcription factor.

Fig. S4: We trained one ExplaiNN model with 350 units on the AI-TAC dataset[9] in which the units had been replaced with 350 different pre-trained DanQ[7] models, each predicting the binding of a single TF to the mouse genome. During the training process of the ExplaiNN model, the DanQ models were frozen (*i.e.* their weights were not modified). TF, transcription factor.

Tables

Table S1: List of TF binding modes used in this study.

Table S2: Individual performances of 350 DanQ models trained on mouse *in vivo* binding data in predicting DNA binding of their respective TFs.

References

1. Agarwal, R. *et al.* Neural Additive Models: Interpretable Machine Learning with Neural Nets. *ArXiv200413912 Cs Stat* (2021).
2. Buenrostro, J. D., Giresi, P. G., Zaba, L. C., Chang, H. Y. & Greenleaf, W. J. Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, DNA-binding proteins and nucleosome position. *Nat. Methods* **10**, 1213–1218 (2013).
3. Johnson, D. S., Mortazavi, A., Myers, R. M. & Wold, B. Genome-Wide Mapping of in Vivo Protein-DNA Interactions. *Science* **316**, 1497–1502 (2007).
4. Libbrecht, M. W. & Noble, W. S. Machine learning applications in genetics and genomics. *Nat. Rev. Genet.* **16**, 321–332 (2015).
5. Eraslan, G., Avsec, Ž., Gagneur, J. & Theis, F. J. Deep learning: new computational modelling techniques for genomics. *Nat. Rev. Genet.* **20**, 389–403 (2019).
6. Zhou, J. & Troyanskaya, O. G. Predicting effects of noncoding variants with deep learning–based sequence model. *Nat. Methods* **12**, 931–934 (2015).
7. Quang, D. & Xie, X. DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences. *Nucleic Acids Res.* **44**, e107–e107 (2016).
8. Kelley, D. R., Snoek, J. & Rinn, J. L. Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks. *Genome Res.* **26**, 990–999 (2016).
9. Maslova, A. *et al.* Deep learning of immune cell differentiation. *Proc. Natl. Acad. Sci.* **117**, 25655–25666 (2020).
10. Zhou, J. *et al.* Deep learning sequence-based ab initio prediction of variant effects on expression and disease risk. *Nat. Genet.* **50**, 1171–1179 (2018).
11. Agarwal, V. & Shendure, J. Predicting mRNA Abundance Directly from Genomic Sequence Using Deep Convolutional Neural Networks. *Cell Rep.* **31**, (2020).
12. Avsec, Ž. *et al.* Effective gene expression prediction from sequence by integrating

long-range interactions. *Nat. Methods* **18**, 1196–1203 (2021).

13. Koo, P. K. & Ploenzke, M. Deep learning for inferring transcription factor binding sites. *Curr. Opin. Syst. Biol.* (2020) doi:10.1016/j.coisb.2020.04.001.
14. Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.* **1**, 206–215 (2019).
15. Stormo, G. D. DNA binding sites: representation and discovery. *Bioinformatics* **16**, 16–23 (2000).
16. Alipanahi, B., Delong, A., Weirauch, M. T. & Frey, B. J. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat. Biotechnol.* **33**, 831–838 (2015).
17. Fornes, O. *et al.* JASPAR 2020: update of the open-access database of transcription factor binding profiles. *Nucleic Acids Res.* **48**, D87–D92 (2020).
18. Koo, P. K. & Eddy, S. R. Representation learning of genomic sequence motifs with convolutional neural networks. *PLOS Comput. Biol.* **15**, e1007560 (2019).
19. Nair, S., Shrikumar, A., Schreiber, J. & Kundaje, A. fastISM: performant in silico saturation mutagenesis for convolutional neural networks. *Bioinformatics* btac135 (2022) doi:10.1093/bioinformatics/btac135.
20. Sundararajan, M., Taly, A. & Yan, Q. Axiomatic Attribution for Deep Networks. *ArXiv170301365 Cs* (2017).
21. Shrikumar, A., Greenside, P. & Kundaje, A. Learning Important Features Through Propagating Activation Differences. *ArXiv170402685 Cs* (2019).
22. Shrikumar, A. *et al.* Technical Note on Transcription Factor Motif Discovery from Importance Scores (TF-MoDISco) version 0.5.6.5. *ArXiv181100416 Cs Q-Bio Stat* (2020).
23. Koo, P. K. & Ploenzke, M. Interpreting Deep Neural Networks Beyond Attribution Methods: Quantifying Global Importance of Features. 6.
24. Gupta, S., Stamatoyannopoulos, J. A., Bailey, T. L. & Noble, W. S. Quantifying similarity between motifs. *Genome Biol.* **8**, R24 (2007).

25. Teng, M., Zhou, S., Cai, C., Lupien, M. & He, H. H. Pioneer of prostate cancer: past, present and the future of FOXA1. *Protein Cell* **12**, 29–38 (2021).
26. Grossman, S. R. *et al.* Systematic dissection of genomic features determining transcription factor binding and enhancer function. *Proc. Natl. Acad. Sci.* **114**, E1291–E1300 (2017).
27. de Almeida, B. P., Reiter, F., Pagani, M. & Stark, A. DeepSTARR predicts enhancer activity from DNA sequence and enables the de novo design of synthetic enhancers. *Nat. Genet.* 1–12 (2022) doi:10.1038/s41588-022-01048-5.
28. Kulakovskiy, I. V., Boeva, V. A., Favorov, A. V. & Makeev, V. J. Deep and wide digging for binding motifs in ChIP-Seq data. *Bioinformatics* **26**, 2622–2623 (2010).
29. Grau, J., Posch, S., Grosse, I. & Keilwagen, J. A general approach for discriminative de novo motif discovery from high-throughput data. *Nucleic Acids Res.* **41**, e197 (2013).
30. Quang, D., Guan, Y. & Parker, S. C. J. YAMDA: thousandfold speedup of EM-based motif discovery using deep learning libraries and GPU. *Bioinformatics* **34**, 3578–3580 (2018).
31. Bailey, T. L. STREME: accurate and versatile sequence motif discovery. *Bioinformatics* (2021) doi:10.1093/bioinformatics/btab203.
32. Santana-Garcia, W. *et al.* RSAT 2022: regulatory sequence analysis tools. *Nucleic Acids Res.* gkac312 (2022) doi:10.1093/nar/gkac312.
33. Weirauch, M. T. *et al.* Evaluation of methods for modeling transcription factor sequence specificity. *Nat. Biotechnol.* **31**, 126–134 (2013).
34. Yin, Y. *et al.* Impact of cytosine methylation on DNA binding specificities of human transcription factors. *Science* **356**, (2017).
35. Hume, M. A., Barrera, L. A., Gisselbrecht, S. S. & Bulyk, M. L. UniPROBE, update 2015: new tools and content for the online database of protein-binding microarray data on protein–DNA interactions. *Nucleic Acids Res.* **43**, D117–D122 (2015).
36. Isakova, A. *et al.* SMiLE-seq identifies binding motifs of single and dimeric transcription factors. *Nat. Methods* **14**, 316–322 (2017).

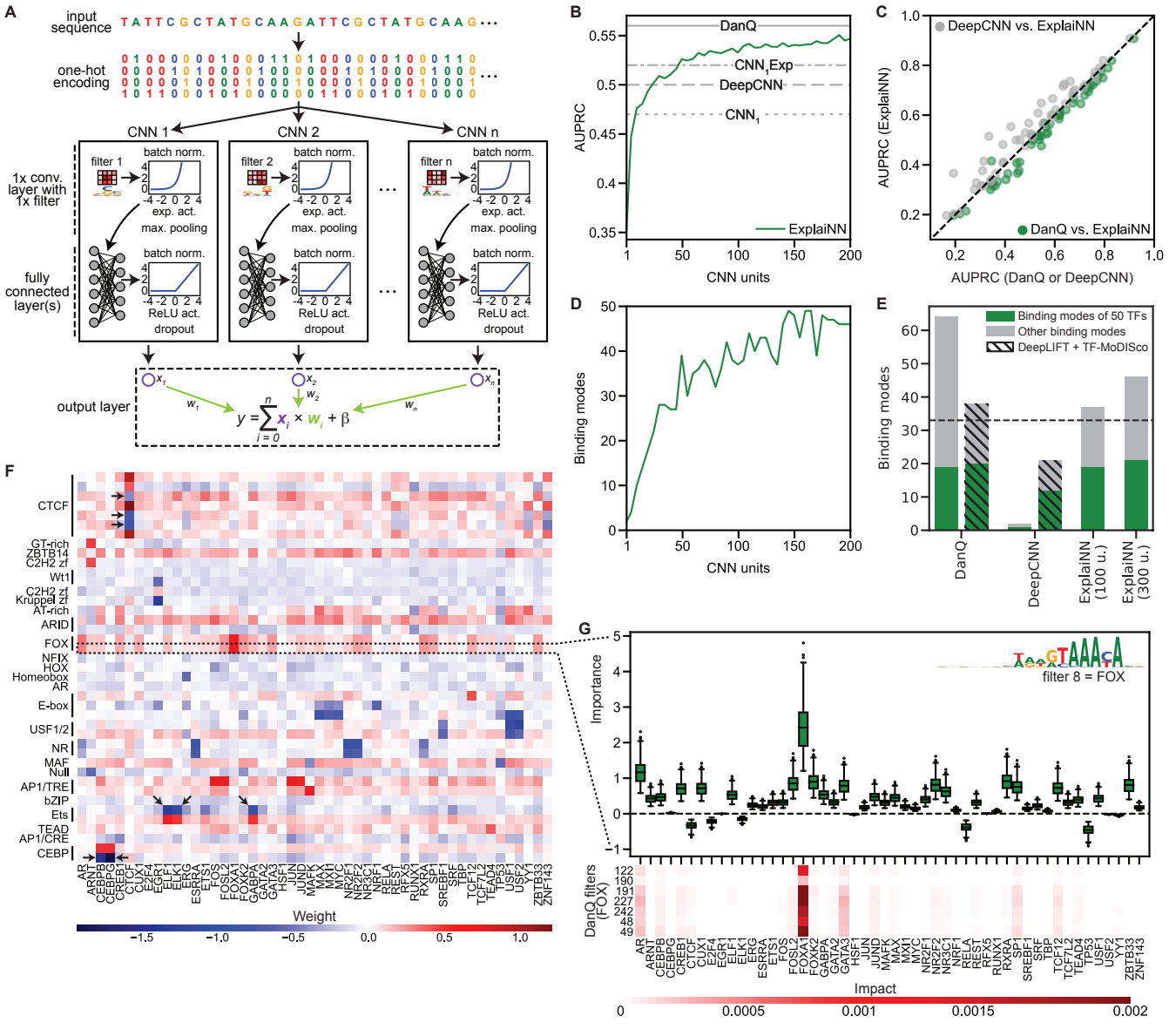
37. Nitta, K. R. *et al.* Conservation of transcription factor binding specificities across 600 million years of bilateria evolution. *eLife* **4**, e04837 (2015).
38. Berger, M. F. *et al.* Compact, universal DNA microarrays to comprehensively determine transcription-factor binding site specificities. *Nat. Biotechnol.* **24**, 1429–1435 (2006).
39. Chèneby, J., Gheorghe, M., Artufel, M., Mathelier, A. & Ballester, B. ReMap 2018: an updated atlas of regulatory regions from an integrative analysis of DNA-binding ChIP-seq experiments. *Nucleic Acids Res.* **46**, D267–D275 (2018).
40. Stuart, T. & Satija, R. Integrative single-cell analysis. *Nat. Rev. Genet.* **20**, 257–272 (2019).
41. Chiou, J. *et al.* Single-cell chromatin accessibility identifies pancreatic islet cell type– and state-specific regulatory programs of diabetes risk. *Nat. Genet.* **53**, 455–466 (2021).
42. Schep, A. N., Wu, B., Buenrostro, J. D. & Greenleaf, W. J. chromVAR: inferring transcription-factor-associated accessibility from single-cell epigenomic data. *Nat. Methods* **14**, 975–978 (2017).
43. Shan, Q. *et al.* Tcf1 and Lef1 provide constant supervision to mature CD8+ T cell identity and function by organizing genomic architecture. *Nat. Commun.* **12**, 5863 (2021).
44. Nechanitzky, R. *et al.* Transcription factor EBF1 is essential for the maintenance of B cell identity and prevention of alternative fates in committed cells. *Nat. Immunol.* **14**, 867–875 (2013).
45. Ambrosini, G. *et al.* Insights gained from a comprehensive all-against-all transcription factor binding motif benchmarking study. *Genome Biol.* **21**, 114 (2020).
46. Yanai, H., Negishi, H. & Taniguchi, T. The IRF family of transcription factors. *OncoImmunology* **1**, 1376–1386 (2012).
47. Wang, H. & Morse, H. C. IRF8 regulates myeloid and B lymphoid lineage diversification. *Immunol. Res.* **43**, 109 (2008).
48. Cobaleda, C., Schebesta, A., Delogu, A. & Busslinger, M. Pax5: the guardian of B cell identity and function. *Nat. Immunol.* **8**, 463–470 (2007).
49. Buckingham, M. & Relaix, F. PAX3 and PAX7 as upstream regulators of myogenesis.

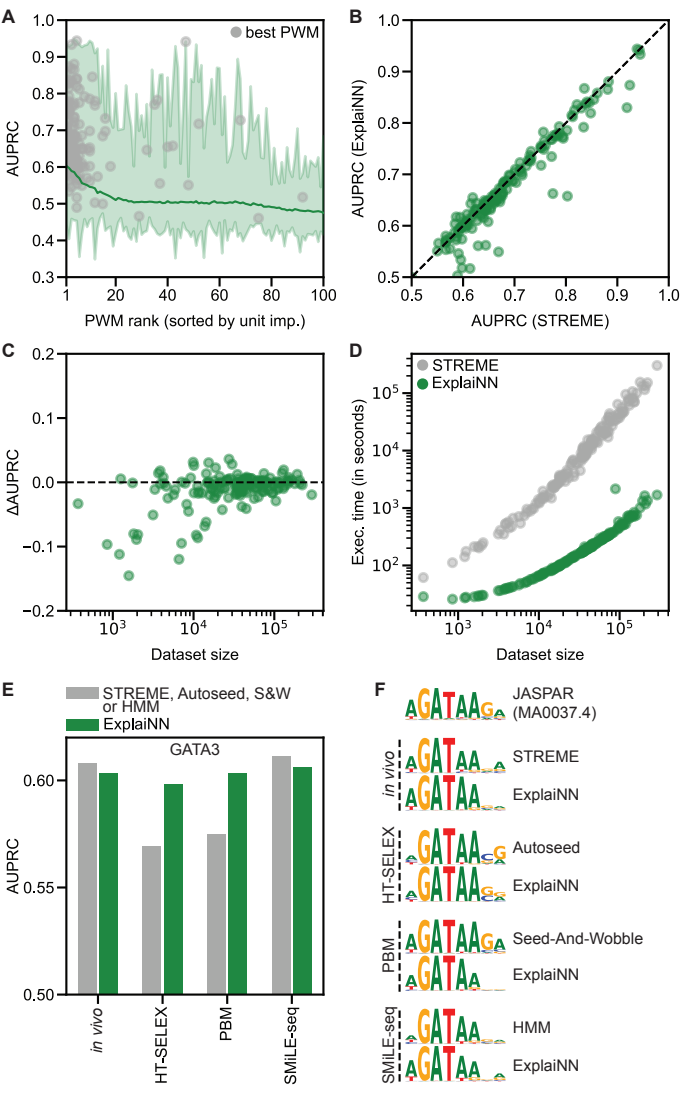
Semin. Cell Dev. Biol. **44**, 115–125 (2015).

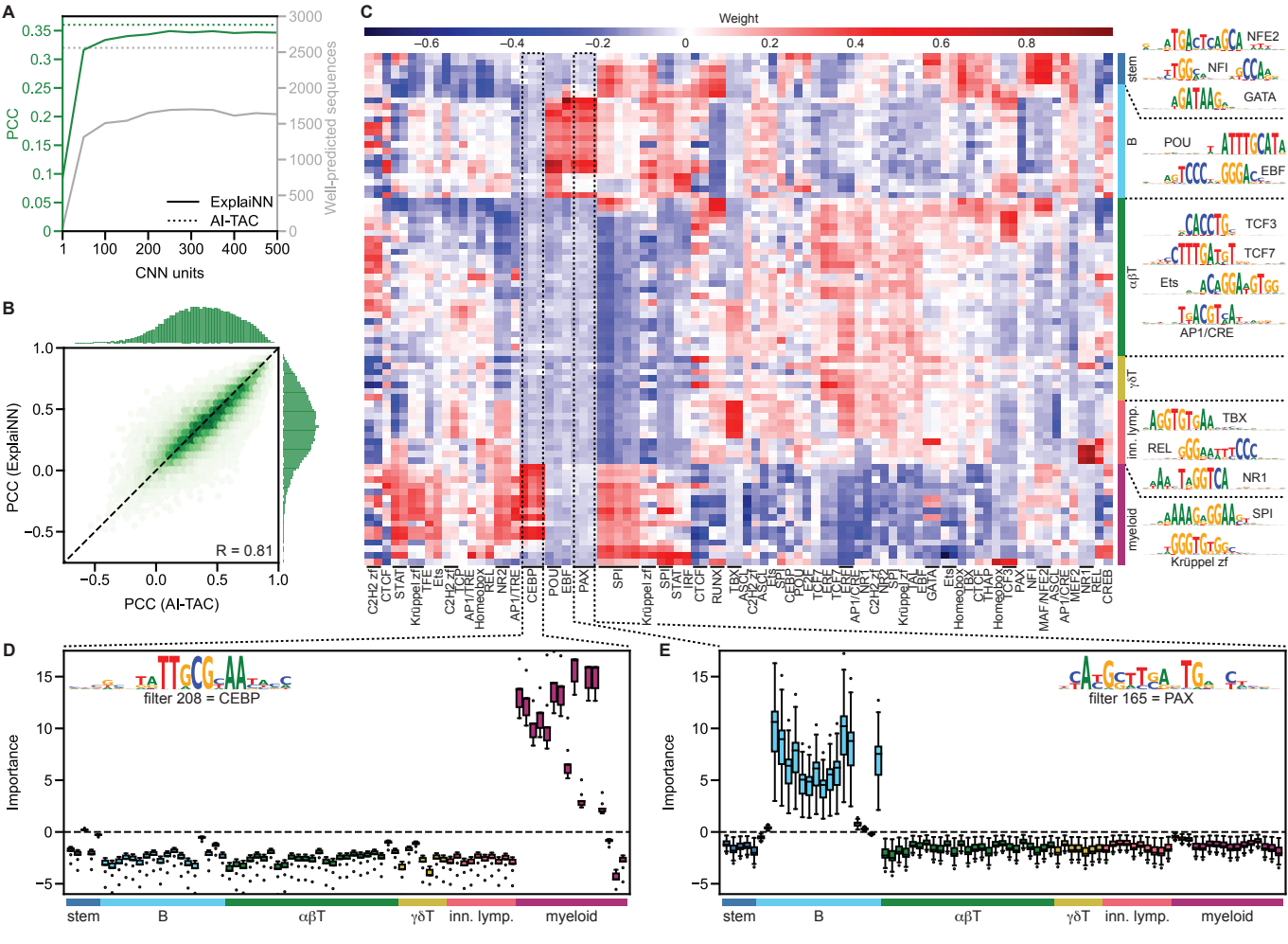
50. McInnes, L., Healy, J. & Melville, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv180203426 Cs Stat* (2018).
51. Wakabayashi, Y. *et al.* Bcl11b is required for differentiation and survival of αβ T lymphocytes. *Nat. Immunol.* **4**, 533–539 (2003).
52. Pabst, T. *et al.* Dominant-negative mutations of CEBPA, encoding CCAAT/enhancer binding protein-α (C/EBPα), in acute myeloid leukemia. *Nat. Genet.* **27**, 263–270 (2001).
53. Wei, B. *et al.* A protein activity assay to measure global transcription factor activity reveals determinants of chromatin accessibility. *Nat. Biotechnol.* **36**, 521–529 (2018).
54. Patel, Z. M. & Hughes, T. R. Global properties of regulatory sequences are predicted by transcription factor recognition mechanisms. *Genome Biol.* **22**, 285 (2021).
55. Bailey, T. L. & Elkan, C. Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Mach. Learn.* **21**, 51–80 (1995).
56. Wasserman, W. W. & Fickett, J. W. Identification of regulatory regions which confer muscle-specific gene expression11Edited by G. Von Heijne. *J. Mol. Biol.* **278**, 167–181 (1998).
57. Kelley, D. R. *et al.* Sequential regulatory activity prediction across chromosomes with convolutional neural networks. *Genome Res.* **28**, 739–750 (2018).
58. Paszke, A. *et al.* PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Adv. Neural Inf. Process. Syst.* **32**, 8026–8037 (2019).
59. Kokhlikyan, N. *et al.* Captum: A unified and generic model interpretability library for PyTorch. *ArXiv200907896 Cs Stat* (2020).
60. Novakovsky, G., Saraswat, M., Fornes, O., Mostafavi, S. & Wasserman, W. W. Biologically relevant transfer learning improves transcription factor binding prediction. *Genome Biol.* **22**, 280 (2021).
61. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).
62. Lovering, R. C. *et al.* A GO catalogue of human DNA-binding transcription factors.

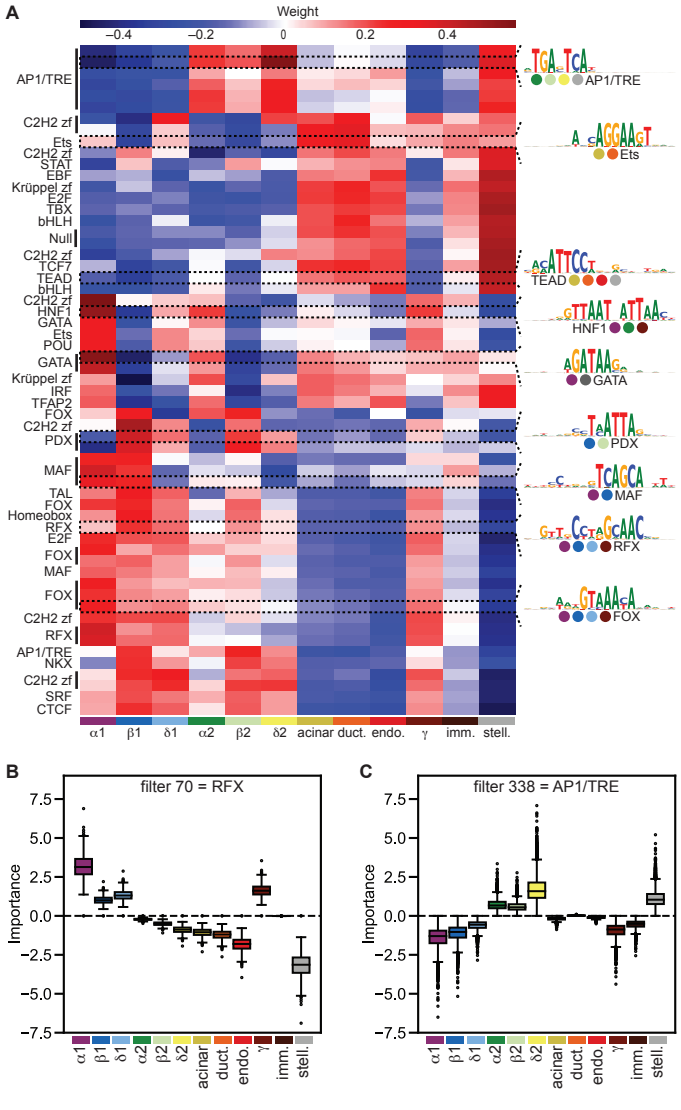
Biochim. Biophys. Acta BBA - Gene Regul. Mech. **1864**, 194765 (2021).

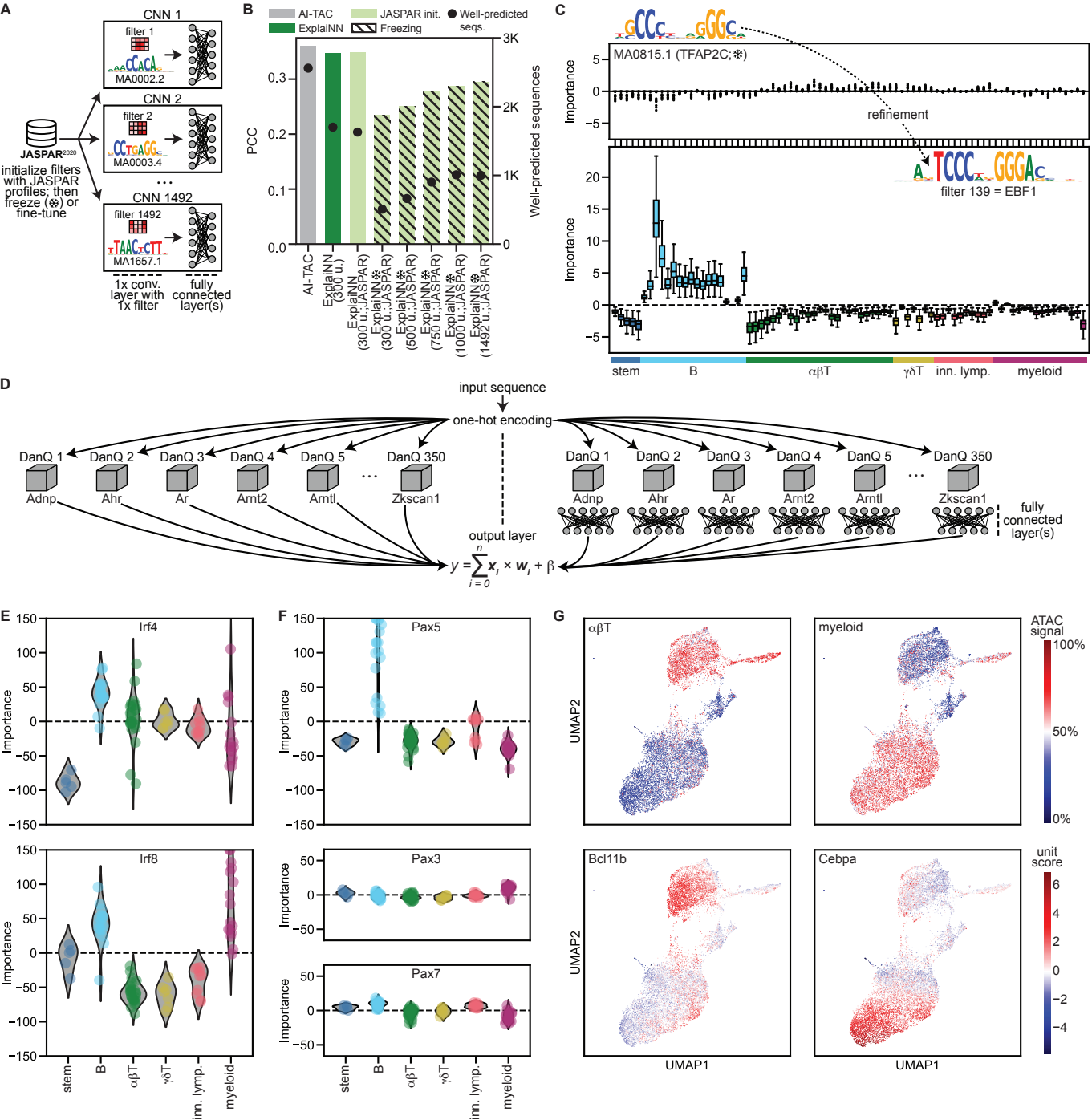
63. Breeze, C. E. *et al.* Atlas and developmental dynamics of mouse DNase I hypersensitive sites. 2020.06.26.172718 (2020) doi:10.1101/2020.06.26.172718.
64. Quinlan, A. R. & Hall, I. M. BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics* **26**, 841–842 (2010).
65. Asif, M. & Orenstein, Y. DeepSELEX: inferring DNA-binding preferences from HT-SELEX data using multi-class CNNs. *Bioinformatics* **36**, i634–i642 (2020).
66. Khan, A., Riudavets Puig, R., Boddie, P. & Mathelier, A. BiasAway: command-line and web server to generate nucleotide composition-matched DNA background sequences. *Bioinformatics* **37**, 1607–1609 (2021).
67. Ashuach, T., Reidenbach, D. A., Gayoso, A. & Yosef, N. PeakVI: A deep generative model for single-cell chromatin accessibility analysis. *Cell Rep. Methods* **2**, 100182 (2022).
68. Gayoso, A. *et al.* A Python library for probabilistic analysis of single-cell omics data. *Nat. Biotechnol.* **40**, 163–166 (2022).
69. Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. *ArXiv14126980 Cs* (2017).
70. Cock, P. J. A. *et al.* Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* **25**, 1422–1423 (2009).
71. Harris, C. R. *et al.* Array programming with NumPy. *Nature* **585**, 357–362 (2020).
72. Almeida, B. P. de, Reiter, F., Pagani, M. & Stark, A. DeepSTARR predicts enhancer activity from DNA sequence and enables the de novo design of enhancers. 2021.10.05.463203 (2021) doi:10.1101/2021.10.05.463203.


A
input sequence
TATTCGCTATGCAAGATTCGCTATGCAAG...
one-hot encoding
CNN 1
CNN 2
CNN n
1x conv. layer with 1x filter
filter 1
filter 2
filter n
batch norm.
exp. act.
max. pooling
fully connected layer(s)
ReLU act.
dropout
output layer
$y = \sum_{i=0}^{n} x_i \times w_i + \beta$
B
DanQ
CNN1Exp
DeepCNN
CNN1
ExplaiNN
AUPRC
CNN units
C
DeepCNN vs. ExplaiNN
DanQ vs. ExplaiNN
AUPRC (ExplaiNN)
AUPRC (DanQ or DeepCNN)
D
Binding modes
CNN units
E
Binding modes of 50 TFs
Other binding modes
DeepLIFT + TF-MoDISco
Binding modes
DanQ
DeepCNN
ExplaiNN (100 u.)
ExplaiNN (300 u.)
F
CTCF
GT-rich
ZBTB14
C2H2 zf
Wt1
C2H2 zf
Kruppel zf
AT-rich
ARID
FOX
NFIX
HOX
Homeobox
AR
E-box
USF1/2
NR
MAF
Null
AP1/TRE
bZIP
Ets
TEAD
AP1/CRE
CEBP
Weight
G
filter 8 = FOX
Importance
DanQ filters (FOX)
Impact


A
best PWM
AUPRC
PWM rank (sorted by unit imp.)
B
AUPRC (ExplaiNN)
AUPRC (STREME)
C
ΔAUPRC
Dataset size
D
STREME
ExplaiNN
Exec. time (in seconds)
Dataset size
E
STREME, Autoseed, S&W or HMM
ExplaiNN
GATA3
AUPRC
in vivo
HT-SELEX
PBM
SMiLE-seq
F
JASPAR (MA0037.4)
in vivo
STREME
ExplaiNN
HT-SELEX
Autoseed
ExplaiNN
PBM
Seed-And-Wobble
ExplaiNN
SMiLE-seq
HMM
ExplaiNN


A
PCC
Well-predicted sequences
ExplaiNN
AI-TAC
CNN units
B
PCC (ExplaiNN)
PCC (AI-TAC)
R = 0.81
C
Weight
stem
B
αβT
γδT
inn. lymp.
myeloid
NFE2
NFI
GATA
POU
EBF
TCF3
TCF7
Ets
AP1/CRE
TBX
REL
NR1
SPI
Krüppel zf
D
filter 208 = CEBP
Importance
stem
B
αβT
γδT
inn. lymp.
myeloid
E
filter 165 = PAX
Importance
stem
B
αβT
γδT
inn. lymp.
myeloid


A
Weight
−0.4
−0.2
0
0.2
0.4
AP1/TRE
C2H2 zf
Ets
C2H2 zf
STAT
EBF
Krüppel zf
E2F
TBX
bHLH
Null
C2H2 zf
TCF7
TEAD
bHLH
C2H2 zf
HNF1
GATA
Ets
POU
GATA
Krüppel zf
IRF
TFAP2
FOX
C2H2 zf
PDX
MAF
TAL
FOX
Homeobox
RFX
E2F
FOX
MAF
FOX
C2H2 zf
RFX
AP1/TRE
NKX
C2H2 zf
SRF
CTCF
α1
β1
δ1
α2
β2
δ2
acinar
duct.
endo.
γ
imm.
stell.
AP1/TRE
Ets
TEAD
HNF1
GATA
PDX
MAF
RFX
FOX
B
filter 70 = RFX
Importance
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
α1
β1
δ1
α2
β2
δ2
acinar
duct.
endo.
γ
imm.
stell.
C
filter 338 = AP1/TRE
Importance
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
α1
β1
δ1
α2
β2
δ2
acinar
duct.
endo.
γ
imm.
stell.


A
CNN 1
filter 1
MA0002.2
CNN 2
filter 2
MA0003.4
CNN 1492
filter 1492
MA1657.1
JASPAR2020
initialize filters with JASPAR profiles; then freeze (❄) or fine-tune
1x conv. layer with 1x filter
fully connected layer(s)
B
AI-TAC
ExplaiNN
JASPAR init.
Freezing
Well-predicted seqs.
PCC
Well-predicted sequences
AI-TAC
ExplaiNN (300 u.)
ExplaiNN (300 u.,JASPAR)
ExplaiNN❄ (300 u.,JASPAR)
ExplaiNN❄ (500 u.,JASPAR)
ExplaiNN❄ (750 u.,JASPAR)
ExplaiNN❄ (1000 u.,JASPAR)
ExplaiNN❄ (1492 u.,JASPAR)
C
MA0815.1 (TFAP2C;❄)
Importance
refinement
filter 139 = EBF1
stem
B
αβT
γδT
inn. lymp.
myeloid
D
input sequence
one-hot encoding
DanQ 1
DanQ 2
DanQ 3
DanQ 4
DanQ 5
DanQ 350
Adnp
Ahr
Ar
Arnt2
Arntl
Zkscan1
fully connected layer(s)
output layer
$y = \sum_{i=0}^{n} x_i \times w_i + \beta$
E
Irf4
Irf8
Importance
stem
B
αβT
γδT
inn. lymp.
myeloid
F
Pax5
Pax3
Pax7
G
αβT
myeloid
Bcl11b
Cebpa
UMAP1
UMAP2
ATAC signal
100%
50%
0%
unit score