8^{th} *International Work-Conference on Bioinformatics and Biomedical Engineering (IWBBIO' 2020), 6-8 May 2020, Granada, Spain*

Bayesian optimization improves tissue-specific prediction of active regulatory regions with deep neural networks

Luca Cappelletti[1], **Alessandro Petrini**[1], **Jessica Gliozzo**[1,4], **Elena Casiraghi**[1], **Max Schubach**[2,3], **Martin Kircher**[2,3], **Giorgio Valentini**[1]

[1] AnacletoLab, Dipartimento di Informatica, Università degli Studi di Milano, Italy

[2] Berlin Institute of Health (BIH), Berlin, Germany

[3] Charit Universittsmedizin Berlin, Berlin, Germany

[4] Fondazione IRCCS Ca Granda - Ospedale Maggiore Policlinico, Department of Dermatology, Milan, Italy

E-mail: valentini@di.unimi.it

Abstract. The annotation and characterization of tissue-specific cis-regulatory elements (CREs) in non-coding DNA represents an open challenge in computational genomics. Several prior works show that machine learning methods, using epigenetic or spectral features directly extracted from DNA sequences, can predict active promoters and enhancers in specific tissues or cell lines. In particular, very recently deep-learning techniques obtained state-of-the-art results in this challenging computational task. In this study, we provide additional evidence that Feed Forward Neural Networks (FFNN) trained on epigenetic data and one-dimensional convolutional neural networks (CNN) trained on DNA sequence data can successfully predict active regulatory regions in different cell lines. We show that model selection by means of Bayesian optimization applied to both FFNN and CNN models can significantly improve deep neural network performance, by automatically finding models that best fit the data. Further, we show that techniques applied to balance active and non-active regulatory regions in the human genome in training and test data may lead to over-optimistic or poor predictions. We recommend to use actual imbalanced data that was not used to train the models for evaluating their generalization performance.

1 Introduction

Non-coding DNA regions, which include 98% of the human genome, are that part of DNA that does not encode for structural proteins and enzymes. A subset of those regions, so-called cis-regulatory elements (CREs) determine spatiotemporal patterns of gene expression [1,2] and therefore play a key role in the control of transcription. CREs are involved in the development of different cell types/tissues, in the timing and intensity of gene expression during the cell life, in the dynamical response to changes in physiological conditions through interactions with DNA-binding transcription factors (TFs), and in the focal alteration of chromatin structure [3]. Genome-wide association studies (GWAS) discovered thousands of variants associated with diseases and traits enriched in

non-conding sequences, and several lines of research show that genetic variants in regulatory regions may be deleterious or directly involved in genetic diseases [4,5].

A great deal of research work has been devoted to the identification of CREs and to their cell-specific activation status. Such studies are essential to dissect the mechanisms underlying the modulation of gene expression and to understand the functional impact of genetic variants on human diseases. Indeed, the effect of genetic variants in non-coding regions is strongly related to the prediction of active regulatory regions (e.g. nucleosome-free regions that are accessible by TFs). Conversely, if a genetic variant, even if potentially deleterious/functionally constrained (e.g. high conservation), is located in an inactive DNA region, it is less likely to be pathogenic.

Thus, great effort has been undertaken to map TF binding sites and histone modifications across cell types and tissues [6,7,8,9,10]. In particular, the ENCODE project [8] identified promoters and enhancers in 147 cell types using a wide range of high-throughput technologies, while the FANTOM project employed CAGE (Cap Analysis of Gene Expression) technologies to broaden the spectrum of considered samples, including 1,816 human and 1,016 mouse samples [11,12]. The Roadmap Epigenomics Consortium [13] studied the epigenomic landscape of 111 representative primary human tissues and cell-lines. However, the experimental identification of CREs is still expensive and time consuming, and, despite the great deal of research effort devoted to this task, the problem is still open.

The use of computational methods, and in particular machine learning approaches, can be a crucial tool to identify the location and activation status of these regions. To this aim, initial approaches, due to the reduced set of reliable annotations, applied unsupervised learning techniques [14,15] to data from the ENCODE project [8]. However, their low accuracy (around 26%) in predicting enhancer [16], pushed the development of more sophisticated supervised learning models, such as random-forest methods [17] and AdaBoost-based models [18]. The subsequent availability of large-scale and high-resolution CREs provided by the FANTOM5 Consortium [11] enabled the development of ensembles of both support vector machines [19] and deep learning models [20,21,22,23], which model complex systems and capture high-level patterns from data, when an underlying, non-obvious structures are present. Examples of promising deep learners applied in regulatory genomics [24] for identifying CREs from sequence data, are DeepEnhancer [20] (Section 3) and BiRen [25]. On the other side, PEDLA [26], a deep hybrid method, achieve high generalization performance across different samples by learning on heterogeneous datasets (i.e. epigenomic, sequence, and conservation data). DECRES [21] is a notable FFNN model that out-performs state-of-the-art unsupervised works when predicting enhancers, promoters, and their activity in a specific human cell-line. Exploiting annotation data from FANTOM [27] and epigenomic features from ENCODE [8], it extends the FANTOM enhancer atlas of 16,988 bidirectionally transcribed loci, therefore providing the most complete annotation of CREs in the human genome so far. However, the experimental setup used to train and test DECRES distorts the actual distribution of the data. Indeed, authors not only balance the training set, which is quite common when dealing with highly-unbalanced data, but they also compute performance on a balanced test set, which may provide a biased evaluation. In this work, we concentrate on the prediction of the activity state of CREs (Section 2), and we present results (Section 4) obtained by two deep neural network models, a FFNN and a CNN (Section 3), whose optimal hyper-parameters have been selected by Bayesian optimization. Moreover, we provide a comparative evaluation to highlight the effects of different balancing schemes (Section 4).

2 Dataset

The models were trained and tested on genomic regions of transcriptionally active enhancer and promoters downloaded from FANTOM5 [11] and matched features collected by Yifeng Li et. al [21] from ENCODE [8] for four cell lines (GM12878, HelaS3, HepG2, K562).

Yifeng Li et. al [21] defined six different classes, active enhancers (A-E), active promoters (A-P), active exons (A-X) and their inactive counterparts (I-E, I-P, I-X), using thresholds on tags per million (TPM) values of the Cap Analysis of Gene Expression (CAGE) data set downloaded from the FANTOM5 database. Classes of active and inactive exons were defined based on exon transcription levels from RNA-seq data downloaded from ENCODE[1]. Finally, an unknown class (UK) labels regions sampled from the genome, but excluding those regions overlapping FANTOM CAGE tags, exons and DNaseI peaks. The employed genomic regions and thresholds, as well as the considered features, are the same as those used in [21]. To train and test FFNN methods (see section 3), the feature set consisted of histone modification and TF binding ChIP-seq, DNase-seq, FAIRE-seq, and ChIA-PET data from ENCODE [8][2]. Further, CpG islands and phastCons scores were included in the feature set by computing the mean value of the feature signal which falls within 200-bps bins centered at each labeled region.

To train and test CNN methods (see section 3), we used sequence data obtained from the human reference genome GRCh37/hg19[3] from the UCSC repository data set[4]. In particular, each genomic region is represented by a sequence of 200 one-hot encoded nucleotides. To automate the coding process we developed an UCSC genome downloader[5]. Thus, for each cis-regulatory element, we have two different representations: the (1) a set of numeric features suitable for training FFNN models [28], and (2) nucleotide sequences to be processed with CNN models [29]. To provide some insights into the data distributions, we show projections of the training matrix for one of the cell lines (GM12878) with the two principal components computed by t-SNE [30] for the epigenomic data set (figure 1-A,B) and MCA [31][6] for sequence data (figure 1-C,D). A high level of entanglement among classes is shown in the t-SNE decomposition for task "IE vs IP" (column A in fig. 1), therefore we expect that CNNs will provide better predictions for this task. t-SNE and MCA plots for tasks "AE+AP vs ELSE" (shown, respectively, in columns B and D in fig. 1) show that the classes are relatively well separated. Hence, we expect that our models will reach good performance in such classification tasks.

[1] ENCODE Data at `ftp://hgdownload.cse.ucsc.edu/goldenPath/hg$19$/encodeDCC`.
[2] ENCODE Data at `ftp://hgdownload.cse.ucsc.edu/goldenPath/hg$19$/encodeDCC`; ENCODE fold-change values are described here `https://sites.google.com/site/anshulkundaje`
[3] `https://www.ncbi.nlm.nih.gov/assembly/GCF_000001405.13/`
[4] `https://genome.ucsc.edu/`
[5] `https://github.com/LucaCappelletti94/ucsc_genomes_downloader`
[6] For computing Multiple Correspondence Analysis we used the python package available at `https://github.com/esafak/mca`

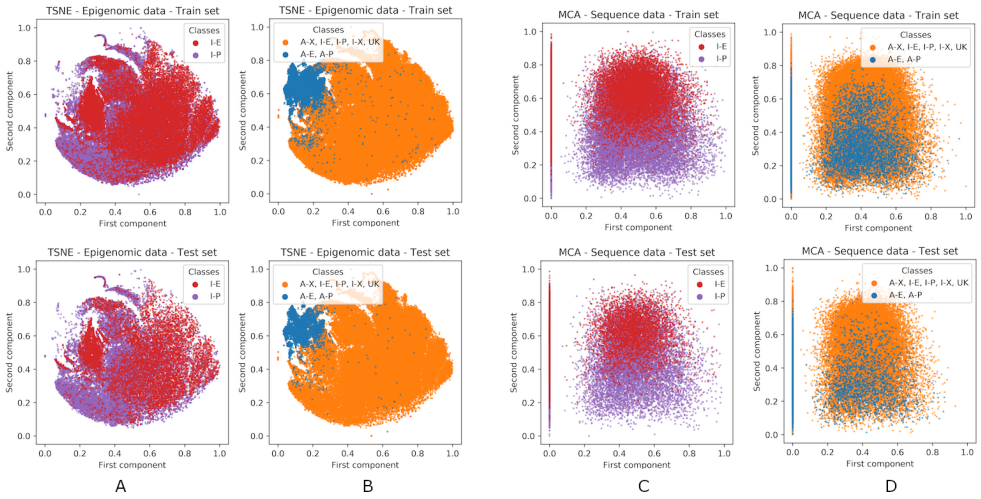


Fig. 1: Decomposition of one of the hold-outs of cell line GM12878. A, B: epigenomic data projected on the two principal components computed by TSNE; C, D: sequence data projected on the two principal components computed by MCA. Top row: training set; bottom row: test set. A, C: Inactive Enhancers vs Inactive Promoters classification task; B, D: Active Enhancers and Promoters vs anything else classification task.

3 Methods

The identification of active regulatory regions can be modeled as a binary classification task. To perform our experiments, we used FFNN [28] for processing epigenomic data, while sequence data were analyzed using CNN [29]. For each method (FFNN and CNN) we developed a "fixed" baseline model, and an "optimized" model, with hyper-parameters selected by Bayesian optimization[7]. Therefore, we obtained four different models, which we call *fixed-FFNN*, *fixed-CNN*, *Bayesian-FFNN* and *Bayesian-CNN* (the presented models were developed using Keras [32] with the TensorFlow backend [33].).

Bayesian Optimization. When designing a neural network (NN) for a given classification task, the choice of architecture (number of layers, neurons and activation functions) and setting of learning hyper-parameters (e.g. optimizer algorithm, batch size, learning rate) are critical for achieving reliable and high performances. At the state-of-the-art, no well-accepted nor unified method for finding the appropriate hyper-parameters for a given task has been defined, and model selection is generally performed either manually or automatically by methods such as "Grid search", which exhaustively searches in the hyper-parameter space, or "random search" [34], which evaluates only a random sequence of hyper-parameter combinations. In this work, we exploit "Bayesian Optimization" (BO) [35], since it has proven to be an effective and cost efficient solution to hyper-parameter optimization. Briefly, the idea is that the "objective function", characterized by high cost for the evaluation of each point of a bounded domain, can be approximated by building a probabilistic model (the "surrogate function") which is relatively cheaper to query. Optimization can then be performed by substituting the objective with the surrogate function. As the surrogate function

[7] `https://scikit-optimize.github.io/`

fixed-FFNN		*Bayesian-FFNN*
Units	**Activation**	**Units hyper-parameters**
16	ReLU	{**256**, 128, 64, 32, 16, 8, 4, 2}
4	ReLU	{**128**, 64, 32, 16, 8, 4, 2}
2	ReLU	{**64**, 32, 16, 8, 4, 2}
1	Sigmoid	1

Parameters	*fixed-FFNN*	*Bayesian-FFNN*
LR	0.5	[**0.1**, 0.5]
LR decay	0.1	[**0.01**, 0.2]
L2 reg.	0.0	[0, .., **0.001**, .., 0.1]
Batch size	32	[32, .., **100**, .., 256]
Optimizer	SGD	SGD
Max no. epochs	64	[32, **1000**]

Table 1: The architecture (left) and learning parameters (right) of the *fixed-FFNN* and the *Bayesian-FFNN* models. The hyper-parameter space explored by BO is shown in square brackets (continuous interval) or curly brackets (finite set). Selected values are in bold. LR stands for learning rate.

represents an "a priori" distribution of the objective function, given some observations obtained by evaluation of the objective, it is possible to exploit Bayes's rule to generate an "a posteriori" estimation of the (objective) function and then update the probabilistic model (surrogate function). Candidates for the observations are suggested through an appropriate "Acquisition function"[36] which uses the information gained by the probabilistic model (estimated by the n already observed points) for suggesting the next $n + 1$ candidate. Depending on the task, it is possible to select the most appropriate acquisition function from a wide array of choices. A common trait of these functions is that they all act upon the criteria of "exploration versus exploitation", so that the sequence of suggested points will provide a better overlook of the objective function (exploration) or a better identification of its maximum/minimum (exploitation).

Fixed-FFNN and Bayesian-FFNN. We designed a baseline *fixed-FFNN* whose architecture is composed by three cascading fully-connected layers (with 16, 4 and 2 neurons, respectively) with the Rectified Linear Unit (ReLU) [37] activation function. A final layer structured as a single neuron with sigmoid activation function acts as output layer, computing the final binary predictions. During network training, weight values were adjusted using a Stochastic Gradient Descent technique with a relatively small value for the batch size hyper-parameter (32), learning rate 0.5, learning rate decay of 0.1, l_2 regularizer of 0.0, no momentum, and a maximum of 64 epochs.

Starting from *fixed-FFNN*, we developed *Bayesian-FFNN* by automatic model selection. Instead of using the computationally expensive grid search as in [21], where it is used to set the number of hidden layers - from 0 to 3, the number of their units - from 0 to a maximum of 256, 128, 64 neurons in the first, second and third layer, respectively, the initial learning rate, and the l_2-regularization amount in continuous intervals not reported by the authors, we use Bayesian optimization [38] to maximize the mean AUPRC computed over the validation sets of 10 internal hold-outs (see 4). More precisely, Bayesian optimization chooses the network architecture from the same search space as that used in [21], while the other hyper-parameters are equal to those of *fixed-FFNN*. The *fixed-FFNN* and *Bayesian-FFNN* models are summarized in table 1. In particular, we note that the chosen number of layers in *Bayesian-FFNN* was often equal to that of *fixed-FFNN*, the number of chosen units was always bigger than that of *fixed-FFNN*, and the learning parameters were often selected in the lower spectrum of the continuous search interval, except for the maximum number of epochs.

Fixed-CNN and Bayesian-CNN. To analyze raw DNA sequence data, we used 1D Convolutional Neural Networks. Like for FFNN models, we firstly developed a fixed model to assess whether this approach effectively recognizes active regulatory regions. After obtaining promising

fixed-CNN						*Bayesian-CNN*				
Layers	**Type**	**Units**	**Kernel**	**Activation**	**Notes**	**Layers**	**Type**	**Units**	**Kernel**	**Notes**
3	Conv	64	5	ReLU	-	3	CBM	64	5	-
1	MP	-	-	-	size 2	1	MP	-	-	Size 2
3	Conv	128	3	ReLU	-	1	CBM	{32, **64**, 128}	{5, **10**}	-
1	MP	-	-	-	size 2	1	MP	-	-	Size 2
3	Conv	128	3	ReLU	-	1	Flatten	-	-	-
1	AP	-	-	-	-	1	Dense	{10, 32, **64**}	-	-
1	Dropout	-	-	-	P. 0.5	1	Dropout	-	-	P. 0.1
2	Dense	10	-	ReLU	-	1	Dense	{10, 32, **64**}	-	-
1	Dropout	-	-	-	P. 0.5	1	Dropout	-	-	P. 0.1
1	Dense	1	-	Sigmoid	-	1	Dense	1	-	-

Table 2: Architecture of the *fixed-CNN* (left)and *Bayesian-CNN* (right) models. Square brackets show the explored hyper-parameter space. Selected values are in bold. MP refers to max-pooling 1D layer, AP refers to average-pooling 1D layer and CBM refers to a convolutional layer with batch normalization.

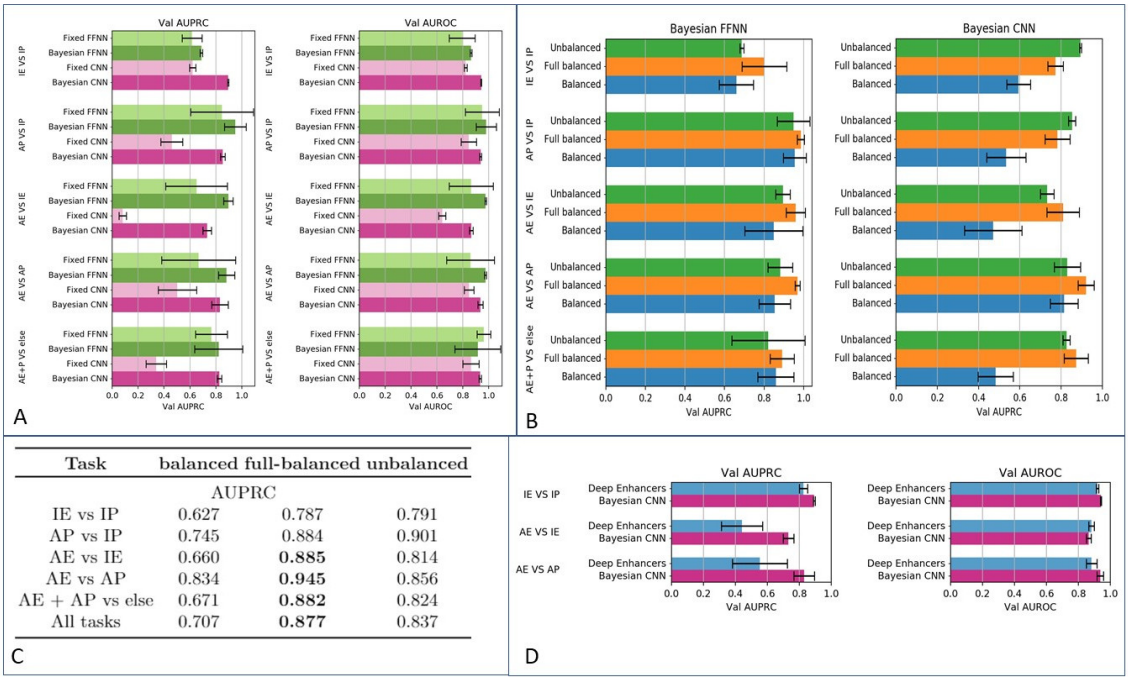
results, we aimed at improving performance by automatically selecting the CNN model parameters through Bayesian optimization. The *fixed-CNN* model is outlined in the left of table 2. The core of the network is composed of three consecutive blocks, each consisting of three (consecutive) convolutional layers followed by one 1-dimensional max/average pooling layer. The number of units in the three convolutional layers of each block, as well as the filter sizes, are fixed. A filter size of 5 for the first three convolutional layers was chosen as this represents a reasonable motif size. As for the FFNN models, all neurons in each layer have ReLU activation function with the exception of the output layer, where the output neuron has sigmoid activation. The Nadam algorithm [39] was used to adjust weight values, learning rate was set to 0.002, and batch size set to 100 examples.

Considering that the *fixed-CNN* architecture has many weights that need to be set, we applied Bayesian optimization to simplify its architecture. We maximized the mean AUPRC computed over the validation sets of 10 internal hold-outs to choose the number of blocks from 1 to 3, and to choose, for each layer, a number of units lower than that of the *fixed-CNN* model. In table 2, the architecture of the *fixed-CNN* model is shown, together with the meta-structure of the *Bayesian-CNN* model. Again, the Nadam algorithm was used to estimate the weight values, the learning rate is set to 0.002, and the batch size to 100.

DeepEnhancer. We compared our *Bayesian-CNN* model with the five DeepEnhancer networks [20], which are 1D-CNNs with one-hot encoded sequence data for distinguishing enhancers from background sequences. With DeepEnhancer, the best performing model is 4conv2pool4norm [20] which consists of four convolutional layers, each one followed by a batch normalization operation. In the second and fourth layer, batch normalization is followed by a max-pooling layer. The first two convolutional layers contain 128 kernels of shape 1×8, while the other convolutional layers contain 64 kernels with shape 1×3. They are followed by two dense layers, with 256 and 128 neurons respectively, interleaved with a dropout layer (ratio 0.5), while a final 2-way softmax layer computes the classification probability results. The ReLU activation function is employed in the dense layers. To distinguish the activity state of enhancers versus promoters we used the network structure and hyper-parameters of the 4conv2pool4norm model [20], but modified the output layer by substituting the 2 output neurons (with softmax activation) with one single output neuron with

a sigmoid activation function to generate the final binary predictions, in line with our FFNN and CNN models.

4 Experimental Results



Task	balanced	full-balanced	unbalanced
	AUPRC		
IE vs IP	0.627	0.787	0.791
AP vs IP	0.745	0.884	0.901
AE vs IE	0.660	**0.885**	0.814
AE vs AP	0.834	**0.945**	0.856
AE + AP vs else	0.671	**0.882**	0.824
All tasks	0.707	**0.877**	0.837

Fig. 2: A: Comparison of deep neural network models with fixed parameters and Bayesian optimized parameters. The plotted AUPRC (left) and AUROC (right) are averaged over cell lines. Black bars represent standard deviations. B: Comparison of the mean AUPRC obtained by *Bayesian-FFNN* (left) and *Bayesian-CNN* (right) among the three different balancing setups for each the five classification tasks. C: Average AUPRC compared across the different experimental setups. For each experimental setup (columns) and task (rows), we report the AUPRC averaged with respect to the Bayesian optimized FFNN and CCN models and the four cell lines. Bold text highlights significantly best results according to Wilcoxon rank sum tests at 0.01 significance level between full-balanced and unbalanced experimental settings; both unbalanced and full-balanced significantly outperform the balanced setting. D: Comparison between *Bayesian-CNN* and DeepEnhancer. The plotted AUPRC (Left) and AUROC (Right) are averaged across multiple hold-out.

Experimental setup. For each of the cell lines described in Section 2, we tested our methods on the five dichotomic tasks introduced in [21]: Inactive Enhancers vs Inactive Promoters (IE vs IP), with average imbalance ratio (AIR) equal to 2.57 (the imbalance ratio is measured as the ratio of the cardinalities of the higher represented and the lower represented class); Active Promoters vs Inactive Promoters (AP vs IP), with AIR = 7.70; Active Enhancers vs Inactive Enhancers (AE vs

IE) with AIR = 22.32; Active Enhancers vs Active Promoters (AE vs AP) with AIR = 7.17; Active Enhancers and Promoters vs anything else ((AE + AP) vs ELSE) with AIR = 18. Note that the five classification tasks have different class imbalance ratios, with ratios ranging from 2.5 to 38.5.

All classification tasks are executed using 10 randomly generated (external) hold-outs, each composed by splitting the data set into training (containing 70% of samples) and test set (containing 30% of samples). Classification tasks involving model selection were performed using additional 10 internal hold-outs (with the same proportion, 70%-30%, of train and test samples). The internal hold-outs are generated by randomly splitting each training set 10 times, thus forming 10 (internal) training and validation sets, used in Bayesian Optimization to select the best model by maximizing performance (AUPRC) on the validation sets. Training features are normalized using MinMax scaling between 0 and 1. The same normalization is applied on validation and test sets.

Performance is measured by using both Area Under the Receiver-Operating Curve (AUROC) [40] and Area Under the Precision-Recall Curve (AUPRC) [41] metrics computed over all test sets in the 10 hold-outs. While AUROC is a de-facto standard for evaluating classifier performance, AUPRC is more suitable when dealing with unbalanced data sets [42,43,44].

To investigate the effects of class balancing in training and test set data, all experiments were repeated three times with the following balancing setups: 1) *"full-balanced"* setup (proposed in [21]): training and test set are randomly sampled with respectively 70% and 30% of samples. The training set is then randomly downsampled to at most 3000 samples per class. This provides a balanced training set. The 30% of samples in the test set are also randomly downsampled to generate a corresponding test set with proportion of A-E:A-P:A-X:I-E:I-P:I-X:UK=1:1:1:2:2:1:10, according to the setup proposed in [21]. Note that the described test set subsampling greatly reduces class imbalance for all available cell lines. 2) *"balanced"* setup: only the training set is balanced as described in 1); 3) *"unbalanced"* setup: any balancing is avoided, maintaining the imbalance that characterizes the original class distribution.

Bayesian optimization improves prediction performance of FFNN and CNN models.

To investigate whether automatic model selection can improve performance we assessed the two FFNN learning models (using epigenomic features) and the two CNN models (based only on sequence) for the five classification tasks, by using the unbalanced setup. For each combination of model/task/data set, mean AUPRC and AUROC computed over the 10 repetitions and over the cell lines are shown in Figure 2 A. Performance of "fixed" and "Bayesian" models on each task were compared by applying Wilcoxon signed rank tests (at 0.01 significance level) [45,46,47] to detect statistically significant differences in the mean values of the classifiers' AUPRC and AUROC distributions. For AUPRC, Wilcoxon test showed a statistically significant difference in means between *Bayesian-FFNN* and *fixed-FFNN* in all tasks. We remark that *Bayesian-FFNN* achieved a better AUPRC value than *fixed-FFNN* for all AUPRCs computed over all hold-outs, tasks, and cell lines, i.e. *Bayesian-FFNN* outperformed *fixed-FFNN* in 200 out of 200 comparisons.

Comparing performance of the two CNN models, *Bayesian-CNN* significantly outperformed *fixed-CNN*. Indeed, when looking again at the full list of AUPRC values across hold-outs, tasks, and cell lines, *Bayesian-CNN* always outperformed *fixed-CNN*. Our results show that model selection by Bayesian optimization improves AUPRC results.

Regarding AUROC results, *Bayesian-FFNN* scores better average ratings than *fixed-FFNN* in all tasks with the exception of "AE+AP vs else". Also, Wilcoxon tests detected a statistically significant difference in mean between *Bayesian-FFNN* and *fixed-FFNN* in all tasks but one ("AE+AP vs else"). Considering all AUROC values computed over the 10 hold-outs, five tasks, and four cell lines, *Bayesian-FFNN* outperformed the *fixed-FFNN* 93% of the times (186 out of 200). Comparing CNN

models, we note again that *Bayesian-CNN* always outperforms *fixed-CNN*. These results suggest that Bayesian model selection is a valid aid for improving both AUPRC and AUROC values of CNN models. This is also true for *Bayesian-FFNN*, as model selection improved the results in both AUPRC and AUROC for almost all tests.

The Bayesian FFNNs outperform Bayesian CNNs in tasks where we need to predict active versus inactive regulatory regions, i.e A-E vs I-E or A-P vs I-E, (Figure 2 B). This is not surpris- ing as epigenetic features are more informative than pure sequence in predicting whether a regulatory region is active or not in a specific cell-type. On the contrary, when we need to discriminate between different types of regulatory regions (i.e. I-E vs I-P or A-E vs A-P) Bayesian CNNs outperform Bayesian FFNNs, as CNNs seem to extract DNA sequence motifs or characteristics (like GC or CpG content) that distinguish enhancers from promoters.

Bayesian CNN models show comparable results with state-of-the-art methods. Due to the very good performance achieved by *Bayesian-CNN*, we decided to further assess its capability for detecting active regulatory regions from raw DNA sequences by comparing it with 4conv2pool4norm net, the best performing DeepEnhancer neural network model (section 3, [20]). Precisely, we used the unbalanced setup and the four cell lines to perform the three classification tasks involving enhancers ("IE vs IP", "AE vs IE" and "AE vs AP"). Using 4conv2pool4norm for these classification tasks is appropriate, as DeepEnhancer networks have been developed for recognizing enhancers against the background genome, and the original authors [20] state that it can be used for similar tasks.
Both models were assessed using multiple hold-outs. In figure 2-D we show, for each of the three tasks, the mean AUPRC (left) and the mean AUROC (right). Wilcoxon tests confirmed that the difference in means of the computed AUPRC and AUROC distributions are statistically significant. Looking at individual AUPRC results, *Bayesian-CNN* outperforms the DeepEnhancer 4conv2pool4norm model 199 times out of 200. For AUROC values, 4conv2pool4norm outperforms *Bayesian-CNN* in only one task. These results suggest that Bayesian optimization is able to select models competitive with state-of-the-art CREs classifiers. Moreover, we confirm that a CNN model trained on sequence data may achieve accurate results in the prediction of cis-regulatory element activity.

Test set balancing may lead to over-optimistic results. In [21], to deal with the data imbalance that characterizes the prediction of active regulatory regions, the authors balanced both training and test data. We reproduced this experimental setup that we name "full-balanced". In addition, we test a "balanced" setup where we only balance the training set, and we compare both with the "unbalanced" experimental setup, for which results have been presented so far (see Section 4).

Table 2-C reports the average AUPRC across Bayesian FFNN and CCN models for the three different balancing techniques. In the last row, average AUPRC values over all tasks are reported for each balancing setup. Comparing the AUPRCs in table 2-C and in figure 2-B, we note that the balanced experimental setup is the one with the worst performance in all tasks. Wilcoxon tests confirm that, on average, the full-balanced setup produces the highest AUPRC scores.

We hypothesize that a reduced performance of the balanced setup may be due to sub-sampling of the training set for balancing, which requires to discard a relatively large amount of training samples. This ultimately affects data coverage during training and the neural network may not be able to effectively learn the intra-class variability, which results in a reduced generalization capability.

In contrast, the better performance obtained by the full-balanced experimental setup (figure 2-B) could be the result of a distortion in the distribution of the test data (i.e. artificial increment of the ratio minority/majority class), thus leading to an over-optimistic estimation of the generalization capabilities of the predictor. Furthermore, test set balancing is not always feasible or possible, for instance when predicting the activity of not previously annotated CREs, since in this application the true labeling is not known.

5 Conclusion

This work showed that Bayesian optimization has the potential of increasing the performance of deep neural networks trained for predicting active regulatory regions in specific cell lines. Further, results show that balancing the test set may lead to an over-optimistic estimation of the generalization performance of the model, while naive balancing of the training data may lead to poor generalization results. To improve the achieved performance, in future works we aim at enlarging the spectrum of the optimized learning parameters, as well as their exploration space.

References

1. Latchman, D.S.: Transcription factors: an overview. International journal of experimental pathology **74** (1993) 417–422
2. Mora, A., Sandve, G.K., Gabrielsen, O.S., Eskeland, R.: In the loop: promoter-enhancer interactions and bioinformatics. Briefings in bioinformatics **17** (2016) 980–995
3. Lambert, S.A., Jolma, A., Campitelli, L.F., Das, P.K., Yin, Y., Albu, M., Chen, X., Taipale, J., Hughes, T.R., Weirauch, M.T.: The human transcription factors. Cell **172** (2018) 650–665
4. Schubach, M., Re, M., Robinson, P.N., Valentini, G.: Imbalance-aware machine learning for predicting rare and common disease-associated non-coding variants. Scientific Reports **7** (2017)
5. Rentzsch, P., Witten, D., Cooper, G., Shendure, J., Kircher, M.: Cadd: predicting the deleteriousness of variants throughout the human genome. Nucleic Acids Res. **47** (2019) D886–D894
6. Javierre, B., et al.: Lineage-specific genome architecture links enhancers and non-coding disease variants to target gene promoters. Cell **167** (2016) 1369–1384
7. Bernstein, B., et al.: The nih roadmap epigenomics mapping consortium. Nature biotechnology **28** (2010) 1045
8. Dunham, I., Kundaje, A., F. Aldred, S., J. Collins, P., Davis, C., Doyle, F., Epstein, C., Frietze, S., Harrow, J., Kaul, R., Khatun, J., Lajoie, B., Landt, S., Lee, b.k., Pauli Behn, F., Rosenbloom, K., Sabo, P., Safi, A., Sanyal, A., Birney, E.: The ENCODE Project Consortium: An integrated encyclopedia of DNA elements in the human genome. Nature **489** (2012) 57–74
9. Shen, Y., Yue, F., McCleary, D.F., Ye, Z., Edsall, L., Kuan, S., Wagner, U., Dixon, J., Lee, L., Lobanenkov, V.V., et al.: A map of the cis-regulatory sequences in the mouse genome. Nature **488** (2012) 116
10. Zhu, J., Adli, M., Zou, J.Y., Verstappen, G., Coyne, M., Zhang, X., Durham, T., Miri, M., Deshpande, V., De Jager, P.L., et al.: Genome-wide chromatin state transitions associated with developmental and environmental cues. Cell **152** (2013) 642–654
11. Noguchi, S., Arakawa, T., Fukuda, S., Furuno, M., Hasegawa, A., Hori, F., Ishikawa-Kato, S., Kaida, K., Kaiho, A., Kanamori-Katayama, M., et al.: FANTOM5 CAGE profiles of human and mouse samples. Scientific data **4** (2017) 170112
12. Lizio, M., Harshbarger, J., Shimoji, H., Severin, J., Kasukawa, T., Sahin, S., Abugessaisa, I., Fukuda, S., Hori, F., Ishikawa-Kato, S., et al.: Gateways to the FANTOM5 promoter level mammalian expression atlas. Genome Biology **16** (2015) 22

13. Kundaje, A., Meuleman, W., Ernst, J., Bilenky, M., Yen, A., Heravi-Moussavi, A., Kheradpour, P., Zhang, Z., Wang, J., Ziller, M.J., et al.: Integrative analysis of 111 reference human epigenomes. Nature **518** (2015) 317
14. Ernst, J., Kellis, M.: ChromHMM: Automating chromatin-state discovery and characterization. Nature Methods (2012)
15. Hoffman, M.M., Buske, O.J., Wang, J., Weng, Z., Bilmes, J.A., Noble, W.S.: Unsupervised pattern discovery in human chromatin structure through genomic segmentation. Nature Methods **9** (2012) 473
16. Kwasnieski, J.C., Fiore, C., Chaudhari, H.G., Cohen, B.A.: High-throughput functional testing of encode segmentation predictions. Genome Research **24** (2014) 1595–1602
17. Yip, K.Y., Cheng, C., Bhardwaj, N., Brown, J.B., Leng, J., Kundaje, A., Rozowsky, J., Birney, E., Bickel, P., Snyder, M., et al.: Classification of human genomic regions based on experimentally determined binding sites of more than 100 transcription-related factors. Genome Biology **13** (2012) R48
18. Lu, Y., Qu, W., Shan, G., Zhang, C.: Delta: a distal enhancer locating tool based on adaboost algorithm and shape features of chromatin modifications. PloS One **10** (2015) e0130622
19. Kleftogiannis, D., Kalnis, P., Bajic, V.: Deep: a general computational framework for predicting enhancers. Nucleic Acids Research (2014)
20. Min, X., Zeng, W., Chen, S., Chen, N., Chen, T., Jiang, R.: Predicting enhancers with deep convolutional neural networks. BMC bioinformatics **18** (2017) 478
21. Li, Y., Shi, W., Wasserman, W.W.: Genome-wide prediction of cis-regulatory regions using supervised deep learning methods. BMC Bioinformatics **19** (2018) 202
22. Hinton, G., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science (2006)
23. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence (2013)
24. Park, Y., Kellis, M.: Deep learning for regulatory genomics. Nature Biotechnology **33** (2015) 825
25. Yang, B., Liu, F., Ren, C., Ouyang, Z., Xie, Z., Bo, X.C., Shu, W.: BiRen: Predicting enhancers with a deep-learning-based model using the DNA sequence alone. Bioinformatics **33** (2017)
26. Liu, F., Li, H., Ren, C., Bo, X.C., Shu, W.: Pedla: Predicting enhancers with a deep learning-based algorithmic framework. Scientific Reports **6** (2016) 28517
27. Andersson, R., Gebhard, C., Miguel-Escalada, I., Hoof, I., Bornholdt, J., Boyd, M., Chen, Y., Zhao, X., Schmidl, C., Suzuki, T., Ntini, E., Arner, E., Valen, E., Li, K., Schwarzfischer, L., Glatz, D., Raithel, J., Lilje, B., Rapin, N., Bagger, F.O., Jorgensen, M., Andersen, P.R., Bertin, N., Rackham, O.J.L., Burroughs, A.M., Baillie, J.K., Ishizu, Y., Shimizu, Y., Furuhata, E., Maeda, S., Negishi, Y., Mungall, C.J., Meehan, T.F., Lassmann, T., Itoh, M., Kawaji, H., Kondo, N., Kawai, J., Lennartsson, A., Daub, C.O., Heutink, P., Hume, D.A., Jensen, T.H., Suzuki, H., Hayashizaki, Y., Müller, F., Forrest, A.R.R., Carninci, P., Rehli, M., Sandelin, A.: An atlas of active enhancers across human cell types and tissues. Nature **507** (2014) 455–461
28. Schmidhuber, J.: Deep learning in neural networks: An overview. Neural Networks **61** (2015) 85–117
29. Fukushima, K.: Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics **36** (1980) 193–202
30. van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of Machine Learning Research **9** (2008) 2579–2605
31. Hierlemann, A., Schweizer-Berberich, M., Weimar, U., Kraus, G., Pfau, A., Göpel, W.: Pattern recognition and multicomponent analysis. Sensors update **2** (1996) 119–180
32. Chollet, F., et al.: Keras. `https://github.com/fchollet/keras` (2018)
33. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016)
34. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. Journal of Machine Learning Research **13** (2012) 281–305

35. Swersky, K., Snoek, J., Adams, R.P.: Multi-task bayesian optimization. In Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q., eds.: Advances in Neural Information Processing Systems 26. Curran Associates, Inc. (2013) 2004–2012
36. Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., de Freitas, N.: Taking the human out of the loop: A review of bayesian optimization. Proceedings of the IEEE **104** (2016) 148–175
37. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML-10). (2010) 807–814
38. Snoek, J., Larochelle, H., Adams, R.P.: Practical bayesian optimization of machine learning algorithms. In: Proc. of the 25th International Conference on Neural Information Processing Systems - Volume 2. NIPS'12, USA, Curran Associates Inc. (2012) 2951–2959
39. Dozat, T.: Incorporating nesterov momentum into adam. (2015)
40. Bewick, V., Cheek, L., Ball, J.R.: Statistics review 13: Receiver operating characteristic curves. Critical Care **8** (2004) 508 – 512
41. Boyd, K., Eng, K.H., Page, C.D.: Area under the precision-recall curve: Point estimates and confidence intervals. In Blockeel, H., Kersting, K., Nijssen, S., Železný, F., eds.: Machine Learning and Knowledge Discovery in Databases, Berlin, Heidelberg, Springer Berlin Heidelberg (2013) 451–466
42. Fawcett, T.: An Introduction to ROC Analysis. Pattern Recogn. Lett. **27** (2006) 861–874
43. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering **21** (2009) 1263–1284
44. Saito, T., Rehmsmeier, M.: The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. PLOS ONE **10** (2015) 1–21
45. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics Bulletin **1** (1945) 8083
46. Pratt, J.W.: Remarks on zeros and ties in the wilcoxon signed rank procedures. Journal of the American Statistical Association **54** (1959) 655–667
47. Derrick, B., White, P.: Comparing two samples from an individual likert question. International Journal of Mathematics and Statistics **18** (2017)