Predicting active cis-regolatory regions: an epigenomic task

Ivan Lamperti and Matteo Salvi

Università degli Studi di Milano, Via Festa del Perdono 7 - 20122 Milano

Abstract. The interest in understanding and predicting the position of the cis-regulatory regions (CRRs) in the non-coding areas of DNA is increasing being an open challenge in the field of computational biology and genomics. Several studies have used Feed Forward Neural Networks (FFNN) trained on epigenomic data and one-dimentional convolutional neural network (CNN) trained on sequence data in order to locate and determine the cis-regulatory areas and in particular categorize them into active and inactive. Our aim is to demonstrate that neural networks trained on epigenomic data perform better in the task of predicting the categorization between active zone or non-active zone.

Keywords: Neural Networks · Deep Learning · Cis-Regulatory Elements

1 Introduction

The human genome is made up of about 3.2 billion pairs of nitrogen bases, only 1,5% of which are coding; the remaining 98.5% is still a subject of study.
The subset of non-coding regions of DNA that are defined at the state of art as cis-regulatory play a key role in the transcription process of coding areas. The cis-regulatory regions are typically near the gene whose transcription they regulate; in fact the prefix *cis* means "on this side". CRRs generally regulates the transcription of genes by binding to transcription factors, specific proteins that determine the activation of the region, and in the focal alteration of chromatin structure. [1]
Different tissues and cells are proof that different phenotypes can be derived from a single DNA sequence. The origin of this event is to be found in the first instance during the transcription process, when portions of DNA are transcribed into mRNA. As the Genome-wide association studies (GWAS) show, the activation of certain non-coding portions of DNA have an important impact on gene-type diseases.
Recent studies are dealing with locating cis-regulatory areas within DNA sequences using Deep Learnig techniques. Among these excellent results he had Deep-Enhancer using CNNs to locate enhancers using only sequence data. [5]
Further studies are dedicated to the prediction of active regulatory regions obtaining excellent results with both FFNN and CNN using epigenomic data and sequence data respectively. [6,7]

Our aim is to verify that in order to predict the classification between active and inactive CRRs, it is more efficient and therefore recommended to use epigenomic data rather than sequence data; the very nature of the epigenomic data gives us information on the state of a certain region with respect to the sequence data that gives us positional information. The cell lines involved in our analysis were HEPG2 and HEK293, the first consists of cancer cells of the liver carcinoma of a 15 year old Caucasian teenager, the second of human fetal kidney cells. [2,3] Our interest has focused on the classification of two particular classes of cis-regulatory regions: enhancer and promoter. The enhancers, located thousands of base pairs from the gene, binding to specific proteins called activators, cause a bending of the DNA bringing them near the promoter of the gene. The promoters, located near the extreme 5 'of the gene, have the role of activating transcription via RNA polymerase II (RNAPII). the activator can bind to some specific proteins that are called transcription factors (TFs), this protein complex that has been created is able to bind to the promoter, it can increase the ease of binding of the RNA polymerase and therefore facilitates the start of transcription (see Fig. 1). [4]

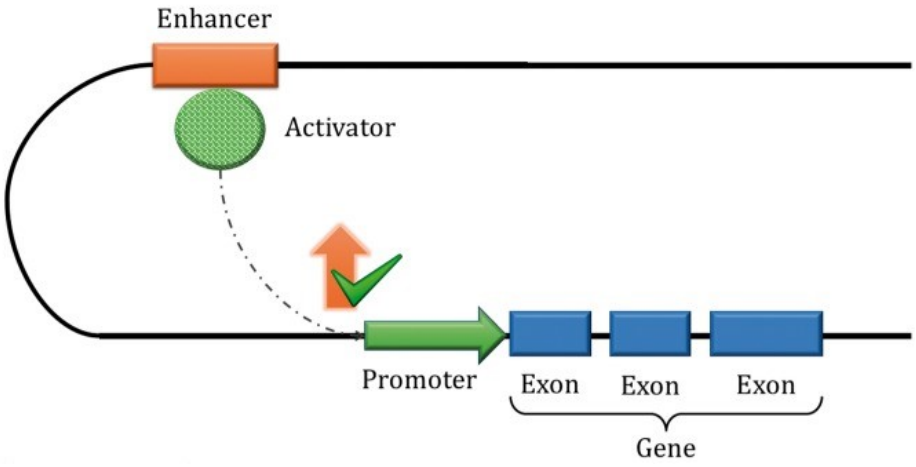


Fig. 1. In the figure the scheme of functioning of enhancers and promoters during DNA transcription [4]

2 Experimental Setup

For the purpose of our analysis we used the categorization of active and inactive regions as a task. In detail, for each cell line we are interested in identifying and

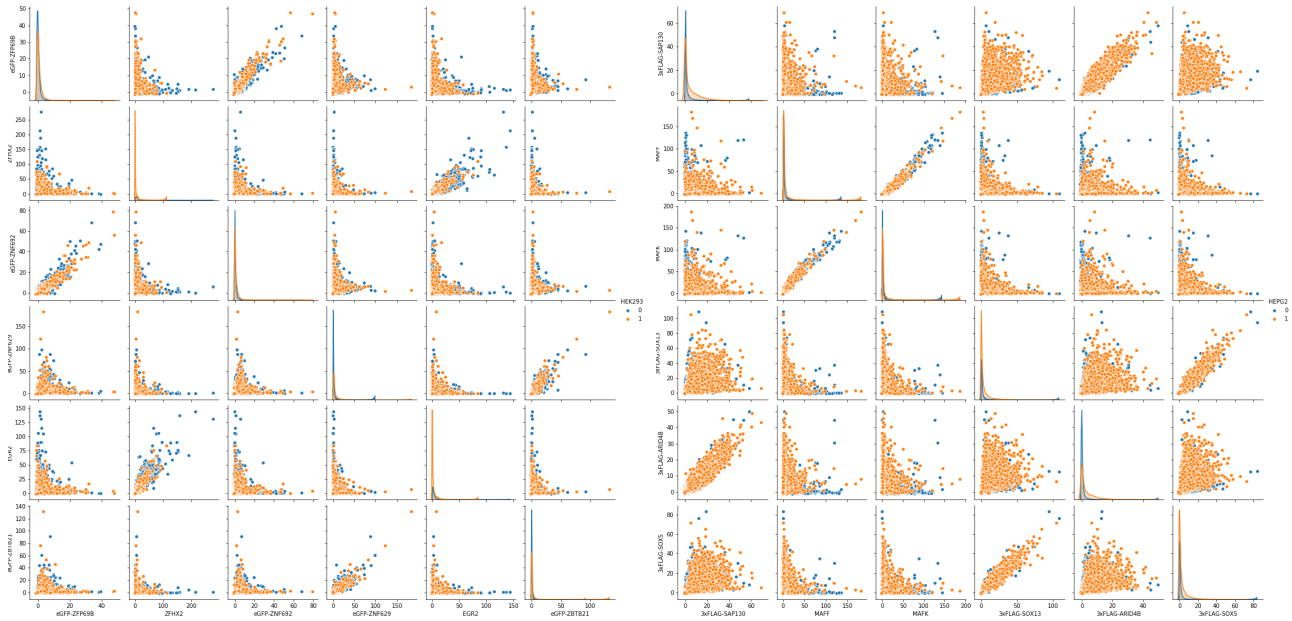
classifying active and inactive promoters (A-P and NA-P), active and inactive enanchers (A-E and NA-E).

Data retrieval We therefore used two types of data: epigenomic data and sequence data. The first in numerical terms are made up of numerical vectors and each element of the vector is made up of an genomic feature, the measurements of a specific characteristic of the region. Among the features considered there are: histone modifications, FTs binding, RNA transcripts, chromatin accessibility and chromatin interactions.
Great contribution in the extraction of such data was that of the Encyclopedia of DNA Elements (ENCODE) project which aims to compile a registry of candidate Regulatory Elements (cREs), which integrates all high-quality DNase-seq and H3K4me3, H3K27ac , and CTCF ChIP-seq. [8] The second type of data considered consists of sequence data, a literal representation of the genome as a string of characters in the alphabet of nitrogen bases. This data is provided by UCSC Genome Browser, an online browser that allows you to download the genome.[10] In order to have an annotation on the status of the regions, as active or inactive, the work provided by the FANTOM5 consortium, Functional ANnoTation Of the Mammalian genome, which publicly provides the DNA annotation of different living beings, has been used. [9] The data was downloaded using an automated pipeline that uses open-source packages that directly insist on previously disclosed data sources.[11,12]

Data processing Before proceeding with a data analysis, pre-processing was necessary. The sequence data, passed as a sequence of characters in the alphabet of the nitrogen bases $\{A, G, C, T\}$ where A stands for adenine, G for guanine, C for cytosine and T for thymine. They have been transformed through the One-Hot Encoding methodology into character strings in $\{0, 1\}$. The epigenomic data, passed from ENCODE as numerical vectors, were cleaned. The noise of corrupt data (NaN) has been reduced by using the 5-Nearest Neighbors algorithm and numerical have been normalized to obtain values in [0.1]. We selected only a subset of features. We used several feature selection methodologies. In the first instance, we looked for any constant beauties as they are not useful in terms of learning, but have not been found for both cell lines. Then we analyzed the linear correlation between the features and the output data. We used statistical tests based on Pearson correlation coefficient to test linear correlation and Spearman rank-order correlation coefficient to test monotone correlation. 0.01 was used as the p-value threshold. Finally, a nonlinear correlation was also tested; since computing nonlinear correlations is computationally expensive, we decided to only run the tests for the columns that are not linearly or monotonically correlated using as p-value threshold 0.01 and as correlation threshold 0.05. As previously the Spearman rank-order correlation coefficient was used for the monotone correlation and the Maximal information coefficient for the non-linear correlation. Unrelated features have been removed from the analysis. At the end, using Boruta, a wrapper built around the random forest classification algorithm, we tried to remove irrelevant features.

We verified the balance of the data: for promoters and enhancers, both in HEK293 and in HepG, the proportion between active and inactive is between 1:10 and 1: 100; no data balancing was necessary.

With the methods described above, it was possible to extract the 3 tuples of most correlated features, the scatter plots of these tuples relating to the enhancers in Fig. 2 were reported by way of example. The features are very correlated when the points are distributed along a straight line. By crossing the results with the graphs on the main diagonal, it can be seen how some features have a clear division between the two types of regions and others do not. This makes us guess that by using a large number of features it is possible to have more data to better differentiate the two statuses.

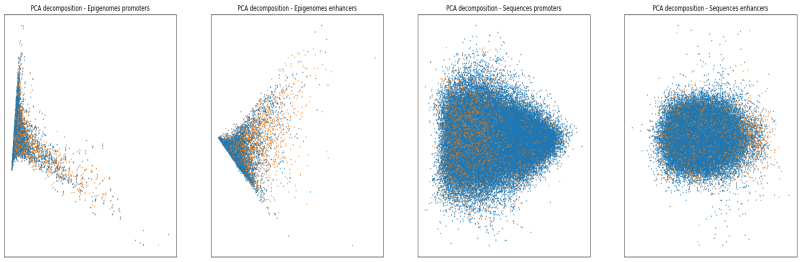


(a) HEK293 enhancers top 3 correlated tuple of features

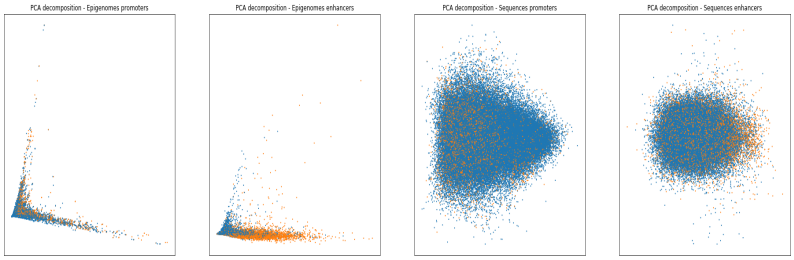
(b) HepG2 enhancers top 3 correlated tuple of features

Fig. 2. in this figure we can see the enhancers scatter plots of HEK293 and HepG2's top 3 correlated tuple of features, in blue inactive regions, in orange active ones. On the main diagonal you can see the distribution of active and inactive regions

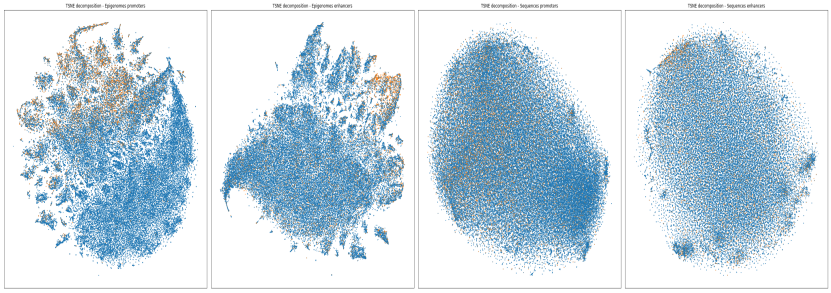
Two graphic tools were used to highlight the spatial distribution: Principal component analysis (PCA) and t-distributed Stochastic Neighbor Embedding (t-SNE). [13,14] For both cell lines it can be noted that in the representation of epigenomic data a separation between the active, in orange, and non-active, in blue, zones is more visible than the sequence data in which the data are more mixed (see Fig. 3).



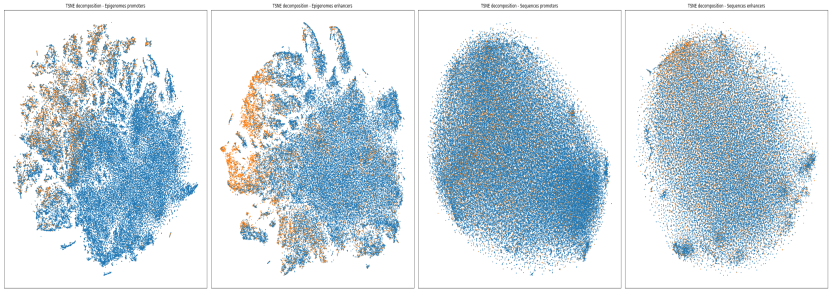
(a) HEK293 PCA decompositions



(b) HepG2 PCA decompositions



(c) HEK293 t-SNE decompositions



(d) HepG2 t-SNE decompositions

Fig. 3. Each row shows the graphs of each cell line created: 3a and 3b with PCA, 3c and 3d with t-SNE. 1st and 3rd lines: HEK293, 2nd and 4th lines: HepG2. Each column represents the different tasks. First two columns: epigenomes promoters and enhancers, last two columns: sequences promoters and enhancers

3 Models

The main tasks, i.e. the identification of active CRRs, can be seen as a binary classification task. For this purpose, we used a MultiLayer Perceptron (MLP) and a Feed Forward Neural Network (FFNN) for processing epigenomic data ([15,16]) while for sequence data a Convolutional Neural Network ([17]). We mainly use 3 base model from which we obtained the models used for this specific tasks.
In any model, it is crucial to define the set of parameters necessary to reach the goal. Such information include but are not limited to the number of layers (with theirs parameters), learning rate and number of epochs.

MLP A MultiLayer Perceptron is a simple neural network that consist of an input layer, an output layer and as many as necessary hidden layers. Regarding MLP, it is very important to define the numbers of hidden layers and the number of neurons for each layers. Were tested MLP with 2 and 3 layers with different number of units each(see Table 1). The activation function was Rectified Linear Unit (ReLU) for all layers except the last one. Being in fact a classification task, a layer with only one neuron and a sigmoid function as activation function was used. The Nadam algorithm was used to adjust the weight values during training and batch size was set to 1024 examples.

Layers	**Units**	**Activation**
Fully connected	{**128**, 256}	ReLU
Fully connected	{32, **64**, 128}	ReLU
Fully connected	{16, **32**, 64}	ReLU
Output	1	Sigmoid

Table 1. Architecture of MLP

FFNN A Feed Forward Neural Network is similar to MLP but it allows other kinds of layer except for fully connected. A base FFNN was defined (see 2) in which the first two layers were fully connected layers with 256 and 128 neurons respectively. Then we used a batch normalization followed by a ReLU activaction function. Three more fully connected layers with 64, 32, 16 neurons were added with a 0.3 dropout and a final output layer with 1 neuron and sigmoid function. Any fully connected layer used ReLU as activation function except for the last one.

A few modifications were tested with this architecture. In particular, we tried to use a ThresholdedReLU instead of standard ReLU. The best performance was achieved with a ThresholdedReLU with a threshold of 0.05. We also noted that a AlphaDropout, i.e. a standard Dropout which keep mean and variance

Layers	**Units**	**Activation**	**Notes**
Fully connected	256	ReLU	
Fully connected	128	ReLU	
BatchNormalization			
Activation		ReLU	
Fully connected	64	ReLU	
Dropout			{**0.3**, 0.5}
Fully connected	32	ReLU	
Fully connected	16	ReLU	
Output	1	Sigmoid	

Table 2. Architecture of FFNN

of input, had better results overall. To process sequence data, the first layer was replaced by Flatten and BatchNormalization-Activation were replaced by another Dropout 0.3. Again, the Nadam Algorithm was used to adjust the weight values.

CNN A Convolutional Neural Network is a deep neural network inspired by the human visual cortex. We used a 1D-CNN only to analyze raw data. The first step was to reshape the input data to match the desired shape. It was followed by two blocks of 2-dimensional convolutional layers with a dropout 0.3. The first two convolutional layers has 64 units and a 10x2 kernel, the third 32 units and a 10x2 kernel and the last two 32 units with a 10x1 kernel. After another Dropout 0.03, we added a small FFNN with a Flatten and two fully connected layers with 32 and 16 neurons respectively and ReLU as activation function.

Layers	**Type**	**Units**	**Kernel**	**Activation**	**Notes**
1	Reshape				200x4
2	Convolutional	{**64**, 128}	10x2		
1	Dropout				{**0.3**, 0.5}
1	Convolutional	{**16**, 32, 64}	10x2		
2	Convolutional	{**16**, 32}	10x1		
1	Dropout				{**0.3**, 0.5}
1	Flatten				
1	Fully connected	32		ReLU	
1	Fully connected	16		ReLU	
1	Output	1		Sigmoid	

Table 3. Architecture of CNN

Hold-outs The hold-outs were generated randomly by splitting the datasets into training set (80%) and test set (20%). The classification tasks of both epigenomic and sequence data used 50 hold-outs.

Metrics The performance of each model is measured by using mainly Area Under the Receiver-Operating Curve (AUROC [19]) and Area Under the Precision-Recall Curve (AUPRC [20]) but also Balanced Accuracy and Loss were taken into account. AUROC is a performance measurement for classification problems and it is used to represent degree or measure of separability. In other words, how well the model is predicting true positive and true negative. AUPRC is another performance measurement and it is used to consider both recall (find all true positive) and precision (no false positive). Loss represents the sum of errors made for each example in training or validation and balanced accuracy is the percentage of correct predictions w.r.t. the total predictions for each class. To measuring performance we only considered AUROC and AUPRC because the others were not very informative.

4 Results

Epigenomic Data Epigenomic data proved to be very suitable for a classification task. As far as we can tell, the models showed that they were more reliable in prediction of active CRRs for HepG2 than HEK293 as we can see from the metrics (see fig. 4,6,5,7). In fact, the four models with HepG2 data out performed their counterpart with HEK293 data. Comparing promoters and enhancers predictions, it can be seen that the latter has better results for both AUROC and AUPRC. They have an higher value on average and a smaller confidence interval.

Sequence Data Instead the sequence data showed to be not very well suitable to predict active CRRs. All models showed very bad performance on training and almost perfect prediction on test, except for MLP models (see fig. 8,9,10,11). A AUROC value of 0.5 suggests that the model is not training at all and a very high test value that it has probably an overfitting problem. Even with sequence data, HepG2 predictions proved to have better results than HEK293 predictions.

Comparing models Wilcoxon's statistical test [21] was used for both epigenomic and sequence data to determine which model was statistically best. The results are summarized in the tables 4, 5, 6 and suggest that the best models for epigenomics tests are FFNN, however for sequence data there is no statistical evidence that FFNN work better than CNN. As mentioned above, the results obtained with the models trained with the sequence data do not meet the standards for considering them models that correctly predict the active CRRs.

Sequence type	**Train set**	**Test set**
Enhancers	$FFNN2 > FFNN1 > FFNN3 > MLP$	$FFNN2 > FFNN3 > FFNN1 > MLP$
Promoters	$FFNN2 > FFNN3 > (FFNN1 \equiv MLP)$	$FFNN2 > FFNN3 > FFNN1 > MLP$

Table 4. Wilcoxon's test (epigenomic data) for HEK293

Sequence type	**Train set**	**Test set**
Enhancers	$(FFNN1 \equiv FFNN2) > FFNN3 > MLP$	$(FFNN1 \equiv FFNN2) > FFNN3 > MLP$
Promoters	$FFNN3 > FFNN2 > (FFNN1 \equiv MLP)$	$FFNN3 > FFNN2 > FFNN1 > MLP$

Table 5. Wilcoxon's test (epigenomic data) for HEPG2

Sequence type	**Train set**	**Test set**
Enhancers	$FFNN > MLP > CNN$	$MLP \equiv FFNN \equiv CNN$
Promoters	$FFNN > MLP > CNN$	$MLP \equiv FFNN \equiv CNN$

Table 6. Wilcoxon's test (sequence data) for both HEK293 and HEPG2

5 Conclusions

The goal of this experiment may be to find which data are more informative: epigenomic or sequence data. The answer can be trivial but is not. The aim of this work was to show that only in predicting whether a regulatory regions is active or not, i.e. A-E vs I-E and A-P vs I-P, epigenomic data are more suitable than sequence data.
The results clearly showed that these task are without any doubt, an epigenomic task. Any models with epigenomic data, even the simplest, outperformed any other models with sequence data.
Our study also demonstrated that it is not necessary to develop very complex models to obtain satisfying results. As we can see from the metrics, MLP behaved very well and it was at least twice as fast than the other models.
In further studies we could compare other task, such as active regions vs inactive regions. We expect that in some cases, like the ones where we need to discriminate different regions, sequence data could produce better results compared to epigenomic data.
Finally, a further effort that could be made would be the creation of neural network models capable of learning information both from epigenomic data and sequence data to combine the different types of information.

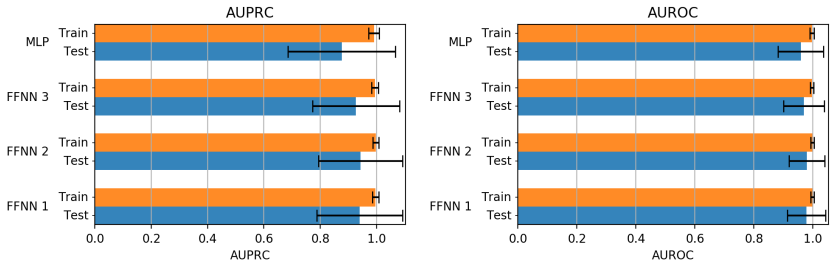


Fig. 4. HEK293 - AE vs IE using epigenomic data

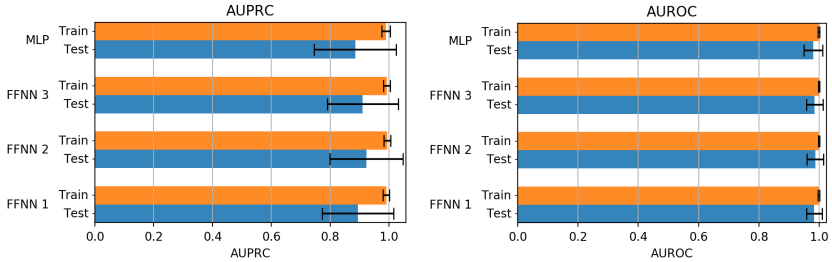


Fig. 5. HEK293 - AP vs IP using epigenomic data

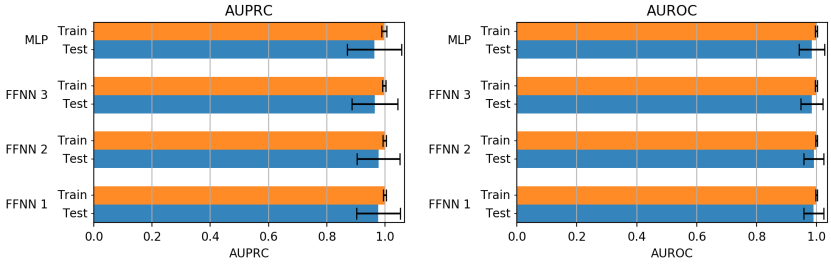


Fig. 6. HepG2 - AE vs IE using epigenomic data

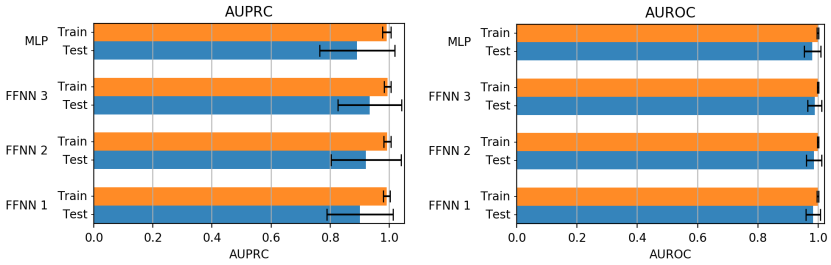


Fig. 7. HepG2 - AP vs IP using epigenomic data

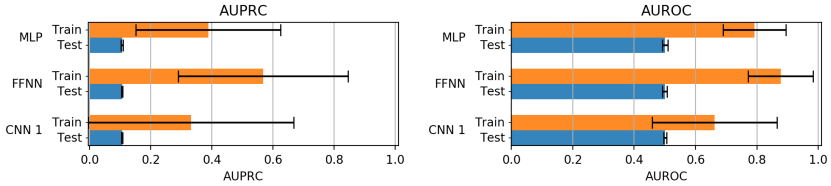


Fig. 8. HEK293 - AE vs IE using epigenomic data

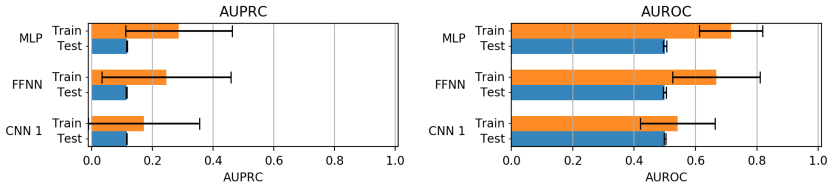


Fig. 9. HEK293 - AP vs IP using sequence data

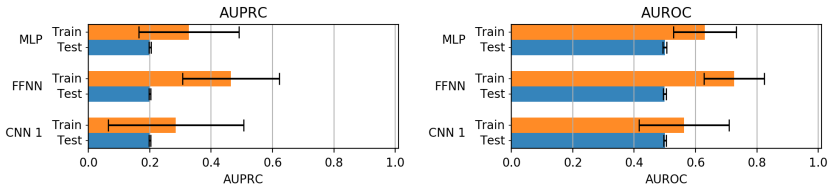


Fig. 10. HepG2 - AE vs IE using sequence data

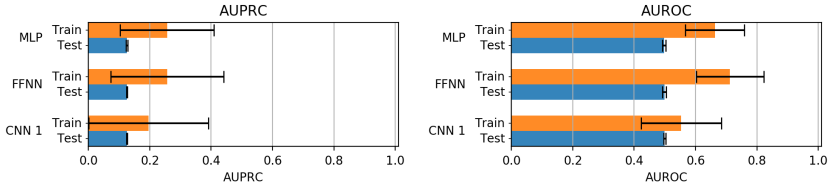


Fig. 11. HepG2 - AP vs IP using sequence data

References

1. Samuel A Lambert et al. “The human transcription factors”. In: Cell 172.4 (2018), pp. 650–665.
2. Hep G2, American Type Culture Collection
3. HEK 293, American Type Culture Collection
4. Rojano E, Seoane P, Ranea JAG, Perkins JR. Regulatory variants: from detection to predicting impact. Brief Bioinform. 2019;20(5):1639-1654. doi:10.1093/bib/bby039
5. Min X, Zeng W, Chen S, Chen N, Chen T, Jiang R. Predicting enhancers with deep convolutional neural networks. BMC Bioinformatics. 2017;18(Suppl 13):478. Published 2017 Dec 1. doi:10.1186/s12859-017-1878-3
6. Cappelletti L. et al. (2020) Bayesian Optimization Improves Tissue-Specific Prediction of Active Regulatory Regions with Deep Neural Networks. In: Rojas I., Valenzuela O., Rojas F., Herrera L., Ortuño F. (eds) Bioinformatics and Biomedical Engineering. IWBBIO 2020. Lecture Notes in Computer Science, vol 12108. Springer, Cham
7. Genome-Wide Prediction of cis-Regulatory Regions Using Supervised Deep Learning Methods Yifeng Li, Wenqiang Shi, Wyeth W. Wasserman bioRxiv 041616; doi: https://doi.org/10.1101/041616
8. ENCODE Project: https://www.encodeproject.org
9. FANTOM: https://fantom.gsc.riken.jp/
10. UCSC Genome Browser: https://genome.ucsc.edu/
11. Cappelletti L., epigenomic dataset
12. Cappelletti L., ucsc genomes downloader
13. Principal component analysis
14. Laurens van der Maaten and Geoffrey Hinton. “Visualizing data using t-SNE”. In: Journal of machine learning research 9.Nov (2008), pp. 2579–2605. http://www.cs.toronto.edu/~hinton/absps/tsne.pdf
15. MLP and SVM Networks - a Comparative Study Classification: MLP vs SVM
16. “Deep learning in neural networks: An overview”. In: Neural networks 61 (2015), pp. 85–117.
17. Kunihiko Fukushima. “Neocognitron: A Self- Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position”. In: Biological Cybernetics 36 (1980), pp. 193–202.
18. Timothy Dozat. “Incorporating Nesterov Momentum into Adam”. In: 2015. URL : .
19. Viv Bewick, Liz Cheek, and Jonathan R Ball. “Statistics review 13: Receiver operating characteristic curves”. In: Critical Care 8 (2004), pp. 508–512.
20. Kendrick Boyd, Kevin H. Eng, and C. David Page. “Area under the Precision-Recall Curve: Point Estimates and Confidence Intervals”. In: Machine Learning and Knowledge Discovery in Databases. Ed. by Hendrik Blockeel et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 451–466. ISBN : 978-3-642-40994-3.
21. Wilcoxon’s Test, Encyclopaedia Britannica