Report of bioinformatic project

Stefano Taverni
stefano.taverni@studenti.unimi.it

September 5, 2021

Abstract

The objective of the project is the prediction of active regulatory regions in cell line K562 through the development of different kind of deep neural networks. The main focus of the project is not the overtaking of state-of-the-art network models, but how to correctly construct such models, the setup of the experimental environment and how to obtain statistically significant and reliable results.

1 Introduction

All known organism generate their macromolecules, such as proteins, through the gene expression process. During this process, the information stored in a gene are synthesised in an end product, like protein or non-coding RNA. All the steps in the gene expression may be regulated to increase or decrease the production of specific gene products: this mechanism is called gene regulation. In particular, the regulation may take place during the transcription, the phase in which a segment of DNA is copied into RNA. During the transcription, two important Cis-regulatory elements (CREs), a subset of non-coding DNA, play fundamental roles: first of all, there is the promoter, a sequence of DNA located near the transcription start site of a gene, and it's responsable for the initiation of the transcription process; typically it can be about 100-1000 base pair long, but this length varies from different organisms and also from different genes. Then, there is the enhancer, which is also a sequence of DNA which can be tied up by another protein to increase the likelihood that transcription occurs; it's usually 50-1500 base pair long, but they are locate far away from the start site, even 1 milion base pair away.[1, 2, 3, 4, 5, 6]

The identification of active regulatory region in non-coding DNA sequences is an extremely important work because it allows us to forecast if a

genetic variant could be pathogenic or not: indeed, if a mutation happens in an incative DNA region, then it is less likely to be potentially harmful. But if a genetic variant is located in an active regulatory region, there are high chances of serial severe human diseases.

Since 98% of our human genome is composed by non-coding DNA regions, the prediction of active regulatory region is a complex task to be carried out by humans. The empoling of artifical intelligence methods is nowdays essential to identify the activation status and the location of these regions. Notable and promising machine learning methods developed so far in this field are:

- DeepEnhancer, a convolutional neural network trained to distinguish enhancers from background genomic sequences. [7]
- DECRES, a deep learning model specifically created for the identification of enhancer and promoter regions in the human genome. [8]
- BiRen, a deep-learning-based hybrid architecture which predicts enhancers using the DNA sequence alone. [9]

Given the complexity of such task, this project doens't aim to outperform state-of-the-art models, but rather it focuses on how to correctly develop different kind of neural network models from the ground up, starting from the retrival of the epigenomic and sequence data to all the preprocessing steps required to properly train the models.

2 Models

Four different kinds of predictor have been developed, in particular one ensamble method and three neural networks:

1. A random forest classifier, with the following paramters:
    - 100 trees in the forest;
    - each tree has a maximum depth of 5 nodes;
    - the minimum number of samples that should be routed to a leaf node is set to 100;
    - Gini function as split quality metric;
    - weighted classes based on their frequencies in the subsamples;

These hyperparameter aren't meant to create an outperforming classifier, but merely produce a simple and low footprint predictor to later compare more complex models.

2. A Feed-Forward neural network with the following structure:

Units	Activation function
256	ReLU
128	ReLU
64	ReLU
1	Sigmoid

The structure of this model is the one proposed in [10], in particular the one with the optimize paramters given by the Bayesian optimization. The weight adjustament function is a Stochastic Gradient Descent technique, with learning rate equals to 0.1 and batch size of 1024 samples.

3. A Convolutional Neural Network, built as follow:

Type	Activation	Units	Kernel	Notes
Conv1D	ReLU	64	5	-
Conv1d	ReLU	32	4	-
MaxPooling	-	-	-	Size=2
Conv1d	ReLU	32	4	-
MaxPooling	-	-	-	Size=2
GlobalAveragePooling	-	-	-	-
Dense	ReLU	64	-	-
Dropout	-	-	-	Rate=0.2
Dense	ReLU	32	-	-
Dropout	-	-	-	Rate=0.2
Dense	Sigmoid	1	-	-

The optimizing function is a Nadam procedure with learning rate equals to 0.002.

4. A Multi-Modal Neural Network, which allows to combine two different models and with different input spaces into one single model. The two previous neural networks are combined in the input layer, they compute in parallel throughout the hidden layers until they reach the last hidden layer, which is concatenated to create a single hidden layer for the Multi-Modal model. Then, the joint hidden layer is connected to a Dense layer with 64 units and ReLU activation function, ending in

a Dense layer with 1 unit and Sigmoid activation function. This model can be constructed in two different way: suppling the pre-trained neural networks, perfoming only the fine tuning of the last hidden layer; or giving as input the shape of the input data, creating the internal neural networks from scratch and then concatenate them. In both cases, a Nadam optimizing procedure is used to tune the weights.

3 Considered tasks

This work studies two different tasks: active enhancers vs inactive enhancers (AEvsIE) and active promoters vs incative promoters (APvsIP) in the cell line K562 from HG38 dataset. This means that the predictors should be trained in order to identify which region of the non-coding DNA has an actual function of promoter or enhancer, avoiding misclassifications.

The HG38 is the reference human genome released on December 2013, and its function is to represent an example of how the set of genes are distributed in the human beings [11]. While the K562 cell line was derived from leukemic cells obtained in December 1970 from a pleural effusion of a 53-year-old female suffering from chronic myelogenous leukemiafor about 4 years [12].

4 Dataset source

Two different kinds of dataset are used to build the predictors.

The epigenomic datasets, for instance the enanchers and promoters, are retrived thanks to the automated pipeline[1] which downloads all the data from ENCODE, a consortium providing a comprehensive parts list of functional elements in the human genome, including elements that act at the protein and RNA levels, and regulatory elements that control cells and circumstances in which a gene is active [13]. After that, the regulatory elements are linked to their activation status, retriving the labels from FANTOM, a consortium which provides functional annotation for the mammalian genome [14].

The other dataset is directly the HG38 human genome assembly, retrived thanks to another automated pipeline[2] which fetches and elaborates the information directly from [11].

[1] `https://github.com/AnacletoLAB/epigenomic_dataset`

[2] `https://github.com/LucaCappelletti94/ucsc_genomes_downloader`

5 Preprocessing

The applied preprocessing steps are:

- binarization of the labels, because they are downloaded as boolean values, and they are transformed into integer values to perform some data visualization;
- imputation of missing values via the median strategy, a robust statistical indictor;
- scaling of the features through sklearn RobustScaler[3], which removes the median and scales the data according to the quantile range, making this scaling robust to outliers.

These preprocessing steps are wrapped in a sklearn Pipeline[4], allowing an efficient and conveninet way to compute the statistic values only on the training set, and then separately scales the test set, avoding any kind of data leakage.

6 Data correlation and distribution

The dataset of AEvsIE presents a strong class imbalance: indeed, only 0.873% of the samples leads to an active region. This situation doens't appear in the APvsIP task, where 21.657% of the sample are active region. This situation is better visualized in figure 1.

Note that a region is considered active when it has a TPM value at least of 1, while it is labeled as inactive with values lower than 1 for APvsIP task, while these thresholds are set to 0 for AEvsIE task.

Concerning the feature correlation, the Pearson correlation coefficient has been computed between each feature: this is useful to identify any kind of linear correlation between the features, in order to remove the ones with higher correlation (0.95 threshold score, with p value equals to 0.01), favoring the features with more entropy. Figures 2 and 3 show the top 3 correlated features for both datasets.

It is also necessary to investigate which feature has no correlation with the output: this can help to reduce the burden of useless feature in the classification tasks. The study of this kind of correlation has been done thanks to three

[3] `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html`

[4] `https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html`

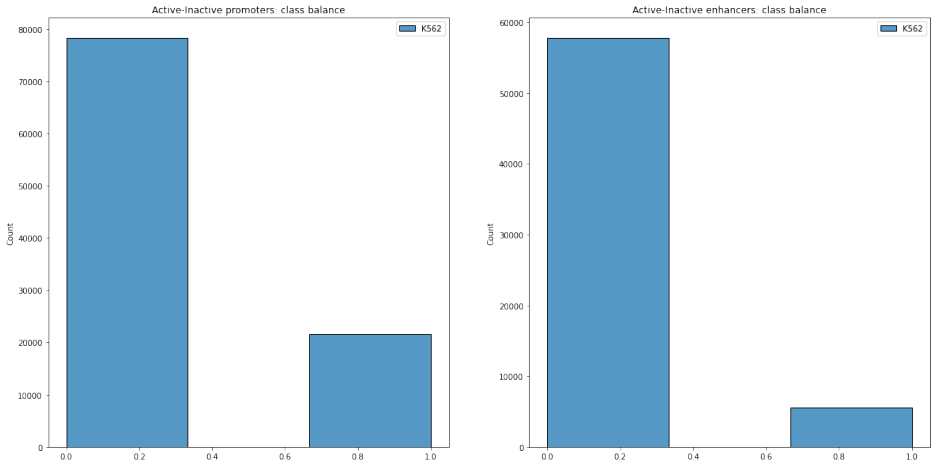

Figure 1: class balance for APvsIP (right) and AEvsIE (left) tasks

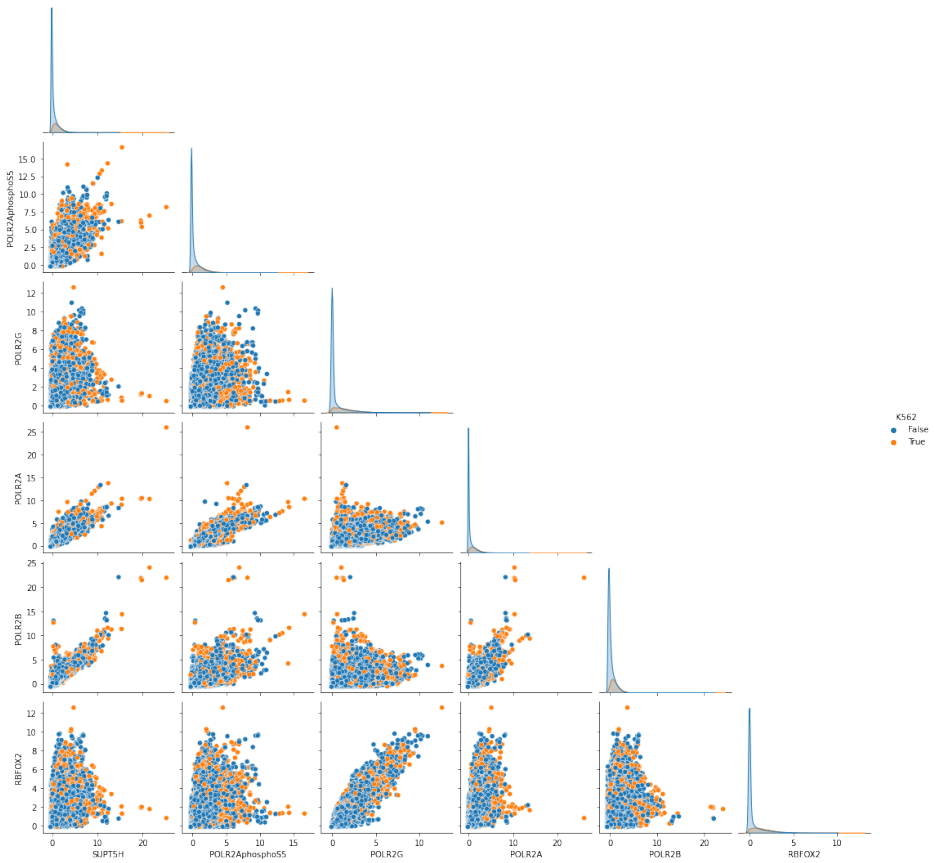

Figure 2: three most correlated features in APvsIP dataset

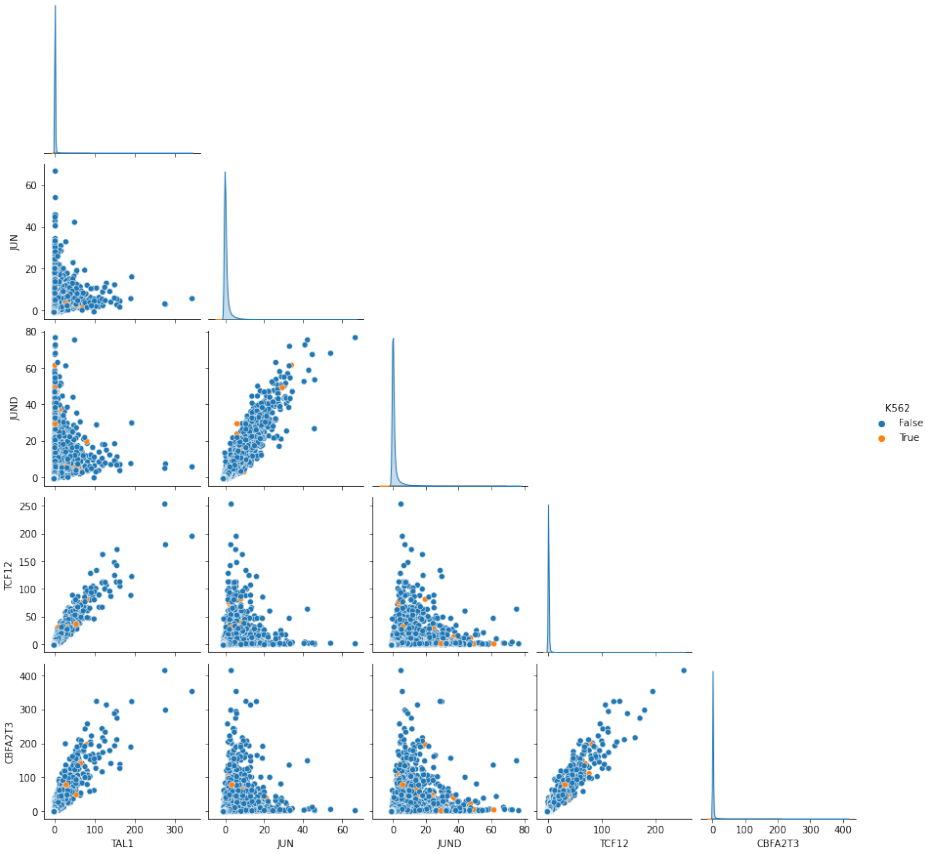

Figure 3: three most correlated features in AEvsIE dataset

different coefficients. First of all, the Pearson coefficient helps to identify any linear relation between the single feature and the output values; then, the Spearman correlation coefficient identifies monotonic relations, both linear or not. Finally, the maximal information coefficient (MIC) is computed only on the subset of variables discovered by the previuos two metrics, in order to filter out any misleading classification, and also to save some time. This phase leads to the identification of the following features as non correlated with the output:

- promoters: THRAP3, NCOA4
- enhancers: H3K36me3, XRCC3, SRSF7, ZNF318, SNRNP70, ZNF830, THRA, SAFB2, U2AF2, STAG1, RBM17, ZKSCAN3, NFE2L1, HNRNPH1, RBM34, ZNF408, FOXJ3, ILK, ZNF280A, whole-genome shotgun bisulfite sequencing, ZNF778, DLX4, FOXA1, MCM7, RBM15, PCBP2, ZBTB8A

7 Feature selection

Taking inspiration from [15], even though the application field is slightly different from the one we are facing in this paper, the feature selection method applied is a Recursive Feature Elimination (RFE) with Support Vector Machine (SVM) as internal classifier. RFE is an iterative process of backward feature elimination, and works by taking an external classifier and training it on the dataset; then the features are ranked according to a ranking criterion, and finally the feature with the smallest rank are removed. RFE can work with a subset of feature greater than 1 at each step, removing more than one feature at each iteration. The process ends as soon as the required number of features is reached. As interal classifier, SVM is a linear classifier that creates an hyperplane or a set of hyperplanes in a higher dimensional space. It chooses the hyperplane that maximize the distance between the two classes, reducing the generalization error. SVM can be adapted to cope with nonlinear separable dataset, by choosing a kernel function to transform the feature space.

In this work, the used implementations of these methods are taken from Sklearn[5, 6], where RFE is tuned by choosing 250 features as number of features to select, and 10 features to remove at each iteration. While SVM is

[5] `https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html`

[6] `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html`

created with a radial basis function as kernel, log loss as loss function and L1 regularization.

8 Data visualization

In order to visualize the dataset, two different techniques of dimensionality reduction have been used, since the dataset belongs to an high dimensional space: the Principal Component Analysis (PCA) and t-Distributed Stochastic Neighbor Embedding (t-SNE).

First, PCA computes the principal components, a set of uncorrelated variables derived from the original variables of the data through orthogonal transformation. The first component retains the highest variance of the original dataset, while the subsequent compontes must be orthogonal to the previous one. Since PCA is sensitive to the scale of the variables, it is necessary to scale the features with respect to their standard deviation.

Then, t-SNE is a statistical nonlinear dimensionality reduction technique well-suited for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions. Specifically, it models each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability. [16]. Since the perplexity of t-SNE procedure highly influnces the output of decomposition, it might be useful to test different values of perplexity and inspect how the visualization changes, obtaining more reliable results.

Figures 4 and 5 are obtained by using the Sklearn implentation of PCA[7] with 2 components to select, while t-SNE implentation comes from Multicore t-SNE library[8], with perplexity value of 30, 40 and 50. Before computing the t-SNE, a preliminary decomposition of the dataset to 50 components has been performed via PCA.

9 Holdouts

The test size is about 20% of the original samples, and 10 different holdouts are computed. Since the class imbalance must be taken into account, each training and test set is created thanks to a stratified split of the dataset,

[7] `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html`

[8] `https://github.com/DmitryUlyanov/Multicore-TSNE`

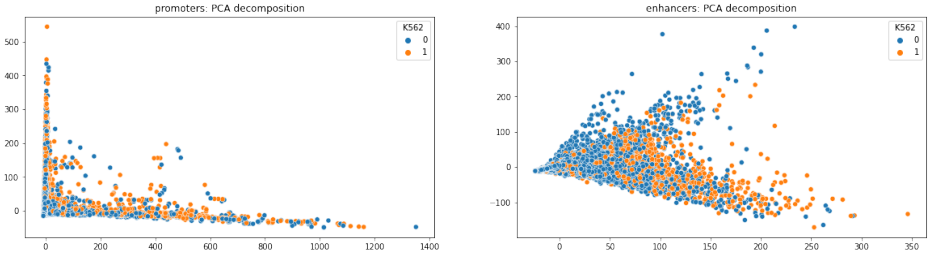

Figure 4: PCA decomposition for promoters and enhancers.

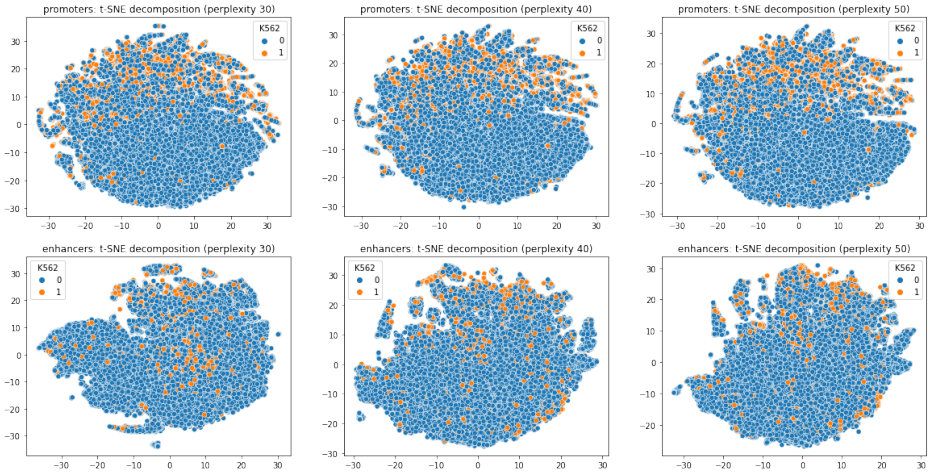

Figure 5: t-SNE decomposition for promoters and enhancers with perplexity level of 30, 40 and 50 respectively.

where the percentage of samples for each class is preserved in each fold[9].

10 Result analysis

The inspected metrics during the result analysis are accuracy, AUROC, AUPRC. The accuracy is the number of correctly predicted observations divided by the total number of samples. AUROC (area under the receiver operating characteristic) misures the discrimination capability of the model, and is calculated as the area under the ROC curve, a plot of the true positive rate against the false positive rate. The closer the metric is to 1, the better will be the discriminating power of the model. An AUROC value of 0.5 corresponds to a random classifier, and so the model is completely useless. Since auroc may lead to overly optimistic results in the case where the class of negative examples is much larger than the class of positive examples, it is necessary to exploit another metric. AUPRC (area under the precision-recall curve) is well-suited performance metric when dealing with imbalanced data. Unlike AUROC, which has a baseline value of 0.5, AUPRC baseline value is the fraction of positive examples over the total number of examples. This means that AUPRC is problem dependent.

In order to compare the different models according to the previous metrics, a Wilcoxon signed-rank test is performed. It is a non parametric statistical test used to compare two different data samples, assuming that they derive from the same population, they are randomly and indipendently generated and they are ordinal. As null hypothesis it assumes that the difference between pairs has a symmetric shape around zero. When the null hypothesis is rejected, one of the two models is statistically better than the other; otherwise, they are statistically indistinguishable.

The tests performed are the comparison between the same model trained with and without feature selection, and the comparison between the effectiveness of the models on the dataset without feature selection. In both cases, the significance level is set to 0.01.

The implementation of the Wilcoxon test is the one provided by Scipy[10].

[9] `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html`

[10] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wilcoxon.html`

11 Results

The barplots in figures 6 and 7 show the training and test performance of the models when trained with and without feature selection for both promoters and enhancers tasks.

Figure 6: Performance of the models without feature selection

At first glance, we can draw several conclusions:

- for both tasks, the perfomance of the models with and without feature selection are practically the same. Indeed, running the Wilcoxon test for the same model when the feature selection is performed or not gives the same results, except for the random forest that leads to better results without feature selection in the enhancer task without feature selection.

- the Multi-Modal Neural Network trained from scratch and the one created with pre-trained models show similar performance. This happens in both tasks, and the results show off a clear overfitting of the models: the values of the metrics are far too good during the training phase than during the test phase, especially for the enhancer task and for the

Figure 7: Performance of the models with feature selection

AUPRC metric. The Wilcoxon test proves that the two versions of the Multi Modal Neural Network are statistically indistinguishable.

- the CNN is the worst model amongst the others. It shows a clear underfitting, meaning that the depth of the network is not the correct one, and it is not capable of extracting relevant information from the dataset.
- The Feed-Forward Neural Network is the most powerful model and it seems quite robust. Wilcoxon test confims that it outperforms all the other models in both tasks.
- All the models perform better in the promoters task instead of the enhancers, meaning that predicting the activation status of enanchers is harder than predicting the activation status of promoters.

12 Further improvements

There are several improvements that can been done to further investigate the performance of the models:

- changing the number of features to select during the recursive feature elimination procedure;
- changing the number of features to remove at each step during the recursive feature elimination procedure;
- increasing the depth of the CNN and/or changing the holdout rate;
- trying differnt values of TPM thresholds;
- trying to transform the classification task into a regression task, since the created models take into account this switch.

References

[1] Wikipedia, the Free Encyclopedia, Regulatory Sequence, https://en.wikipedia.org/wiki/Regulatory_sequence,

[2] Wikipedia, the Free Encyclopedia, Regulation of gene expression, https://en.wikipedia.org/wiki/Regulation_of_gene_expression,

[3] Wikipedia, the Free Encyclopedia, Promoters, https://en.wikipedia.org/wiki/Promoter_(genetics),

[4] Wikipedia, the Free Encyclopedia, Cis-regulatory element, https://en.wikipedia.org/wiki/Enhancer_(genetics)

[5] Wikipedia, the Free Encyclopedia, Transcription, https://en.wikipedia.org/wiki/Transcription_(biology),

[6] Wikipedia, the Free Encyclopedia, Cis-regulatory element, https://en.wikipedia.org/wiki/Cis-regulatory_element

[7] Min, X., Zeng, W., Chen, S. et al. Predicting enhancers with deep convolutional neural networks. BMC Bioinformatics 18, 478 (2017). https://doi.org/10.1186/s12859-017-1878-3

[8] Li, Y., Shi, W. & Wasserman, W.W. Genome-wide prediction of cis-regulatory regions using supervised deep learning methods. BMC Bioinformatics 19, 202 (2018). https://doi.org/10.1186/s12859-018-2187-1

[9] Bite Yang, Feng Liu, Chao Ren, Zhangyi Ouyang, Ziwei Xie, Xiaochen Bo, Wenjie Shu,
BiRen: predicting enhancers with a deep-learning-based model using the DNA sequence alone, Bioinformatics, Volume 33, Issue 13, 1 July 2017, Pages 1930–1936, https://doi.org/10.1093/bioinformatics/btx105

[10] Luca Cappelletti, Alessandro Petrini, Jessica Gliozzo, Elena Casiraghi, Max Schubach, Martin Kircher, and Giorgio Valentini. Bayesian optimization improves tissue-specific prediction of active regulatory regions with deep neural networks. In Springer, editor, Bioinformatics and Biomedical Engineering, IWBBIO 2020, Lecture Notes in Computer Science, 2020.

[11] Kent WJ, Sugnet CW, Furey TS, Roskin KM, Pringle TH, Zahler AM, Haussler D. The human genome browser at UCSC. Genome Res. 2002 Jun;12(6):996-1006.

[12] Carmen B. Lozzio, Bismarck B. Lozzio, Human Chronic Myelogenous Leukemia Cell-Line With Positive Philadelphia Chromosome, Blood, Volume 45, Issue 3, 1975, Pages 321-334, ISSN 0006-4971, https://doi.org/10.1182/blood.V45.3.321.321.

[13] The ENCODE Consortium, 2021, Stanford University, https://www.encodeproject.org/

[14] FANTOM, 2014, RIKEN https://fantom.gsc.riken.jp/

[15] Guyon, I., Weston, J., Barnhill, S. et al. Gene Selection for Cancer Classification using Support Vector Machine. Machine Learning 46, 389–422 (2002). https://doi.org/10.1023/A:1012487302797

[16] Wikipedia, the Free Encyclopedia, t-distributed stochastic neighbor embedding, https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding