SCHOOL OF SCIENCES

DEPARTMENT OF COMPUTER SCIENCE AND BIOMEDICAL INFORMATICS

Machine learning algorithms for large-scale transcriptomic and genetic data

PhD DISSERTATION

Dimitris Grigoriadis

Supervisor: Artemis Hatzigeorgiou, Professor

Lamia, 2022

Dedicated to my family and friends,

Dissertation committee

Abstract

This study is part of the research areas of bioinformatics and large-scale data analysis by utilizing Machine Learning algorithms. Current research focuses on regulatory regions with two major biological objectives. Creating robust machine learning algorithms for single nucleotide resolutions transcription start site discovery and functional characterization of variants in regulatory regions.

First biological objective for the current study is the characterization of regulatory promoter region of genes. CAGE is a protocol that offers a clear advantage when studying the dynamics related to transcription initiation, alternative promoter usage and the identification of enhancer RNAs. Despite the increased popularity of this protocol, CAGE is not absent from the list of experimental methods that suffer from biological and technical noise which can significantly diminish the robustness of downstream analyses. Thus, the need for computational methods emerges, that can accurately increase the signal-to-noise ratio in CAGE data, resulting in error-free transcription start site annotation and quantification of regulatory region usage.

DiS-TSS, an annotation agnostic computational framework, that for the first time utilizes digital signal processing inspired features customized on the peculiarities of CAGE data. Features from the spatial and frequency domains are combined with a robustly trained Support Vector Machines model to accurately distinguish between peaks related to real transcription initiation events and biological or protocol-induced noise. When benchmarked on experimentally derived data on active transcription marks as well as annotated TSSs, DiS-TSS was found to outperform existing implementations, by providing on average ~11k positive predictions and an increase in performance by ~5% based on in the experimental and annotation-based evaluations.

The DeepTSS algorithm, which is a novel computational method for processing CAGE samples, that combines genomic signal processing (GSP), structural DNA features,

evolutionary conservation evidence and raw DNA sequence with Deep Learning (DL) to provide single-nucleotide TSS predictions with unprecedented levels of performance.

Second biological objective applies in the field of genetics by assessing the survival risk of cancer patients. Cancer prognosis is a highly sensitive process where patients' risk state and survival outcome are accessed. For the prediction of survival prognosis in cancerous patients, many approaches have been introduced, including coding and/or non-coding expression profiles, metadata (age, cancer stage, sex etc), methylation profiles and medical images (MRI, CT scan etc). Thus, genomic alterations in regulatory gene regions lack consideration and extensive exploratory analysis.

A novel deep clustering technique is utilized, using the advantages of autoencoders in DNA sequencing data to accurately separate high and low risk patients in eight different primary cancer tissues. More than 1000 primary tumor whole genomes have been analyzed, retrieved from the ICGC repository. Overall findings suggest that promoter regulatory regions play a key role in survival risk prognosis in eight tissues with two levels of mortality for the Kaplan-Meier curves. Thus, promoter loci with genomic variations have underling patterns able to distinguish between high and low risk cancer patients.

Development of reliable machine learning, as well as deep learning algorithms in current dissertation may pave the way for deciphering biological problems and contribute to advancing the field of gene regulation, as well as advance genetics by discovering the functional impact of variants in regulatory regions.

Acknowledgements

That’s all folks!

The current study was carried out form early 2019 to late 2022, for the University of Thessaly at the Hellenic Pasteur Institute. Since many friends and colleagues have helped me carry out this study, I wish to write a few lines to thank them.

I would like to express my gratitude to my professor and chair of the committee Artemis Hatzigeorgiou for the opportunity and her support thought-out this study. Additionally, I want to thank all six members of the scientific committee for their time and interest.

Thanks, should also go to all my colleagues in the lab. PhD students and post docs still there and those who left creating carries elsewhere. Special thanks to Nikos Perdikopanis who was there in my first steps of my master thesis and in my PhD. Also, I want to thank Giorgos Georgakilas who gave me immense help within the scope of Deep neural networks.

To my family and friends, I want to express my gratitude for supporting me in all my years of study.

Table of Contents

Abbreviations

ANN: Artificial neural network
AUC: Area Under the Curve
CAGE: Cap Analysis of Gene Expression
CNN: Convolutional Neural Network
CNN: Convolutional neural network
ChIP-Seq: Chromatin ImmunoPrecipitation followed by Sequencing
DL: Deep Learning
DSP: Digital Signal Processing
FFT: Fast Fourier Transform
FNN: Feedforward neural network
GSP: Genomic Signal Processing
LDA: Linear Discriminant Analysis
ML: Machine Learning
NGS: Next Generation Sequencing
NN: Neural Network
PCA: Principal Component Analysis
RBF: Radial Basis Function
RL: Reinforcement Learning
RNN: Recurrent Neural Network
ReLU: Rectified Linear Unit
SL: Supervised Learning
SVMs: Support Vector Machines
TFBS: Transcription Factor Binding Site
TSS: Transcription Start Sites
UCSC: University of California Santa Cruz
UMAP: Uniform Manifold Approximation and Projection
t-SNE: t-distributed Stochastic Neighbor Embedding

Table of Figures

Table of Tables

1. Introduction

1.1. Machine learning

Machine learning (ML) is a field focusing on the use of data and algorithms in a way that imitates the human way of learning. Mathematical models and algorithms used in machine learning, are build based on training data aiming to make predictions on new unknown data. There are many fields of knowledge such as medicine[1], bioinformatics, computer vision and more that make use of machine learning algorithms to solve complex problems.

1.1.1. Designing machine learning algorithms

To design robust machine learning models, one needs to well address the nature of the problem trying to answer from the data. In other words, asking the right questions from the data is a crucial step. Thus, a comprehensive knowledge of the domain is very important as well as creating robust training sets. Data must undergo quality control and a certain transformation to meet the appropriate structure before feeding them in the ML model. This stage remains highly important for the overall accuracy of the final trained model and there are several stages such as feature extraction and feature selection where robustness of the training set is built.

1.1.2. Feature extraction

Feature extraction is the process which transforms the raw data into numerical features that can be processed while preserving the information in the original dataset. Thus, feature extraction yields better results compared to applying ML directly on raw datasets.

1.1.3. Feature selection

Feature selection in machine learning is the process of selecting a subset of relevant variables, as more capable predictors to use for the training process of the ML model. Feature selection techniques are mainly used for simplifying the models, shorten training time or bypass the curse of dimensionality by applying dimensionality reduction techniques (see chapter 1.2.5 Dimensionality reduction). Feature selection can be described in three main categories: filters, wrappers and embedded.

With filter type feature selection methods, variables are selected regardless of the model. They observe the correlation of the variables with models output for evaluating the separation ability [2]. This type of feature selection suppresses variables with low separation ability. The rest of the variables will be involved in the training process of a classification, or a regression model and will be used to classify data. These methods are particularly effective in computation time and robust to overfitting.

Wrapper methods evaluate subsets of variables in a repetitive process observing the model output and trying to maximize the separation ability [3]. The best feature (forward selection) is selected for each loop, or the worst is excluded (backward selection). There are two main drawbacks with Wrapper methods. Firstly, there is a high risk of overfitting when the number of observations is insufficient and secondly, computation time may grow exponentially when the number of variables is large.

Embedded methods are a combination of filter and wrapper methods. Implementation is carried out by algorithms that have their own feature selection methods. Amongst these methods LASSO [4] and RIDGE regression [5] remain very popular techniques. Thus, they both have inbuilt penalization functions to reduce overfitting.

1.2. Categorization of Machine Learning algorithms

ML algorithms, depending on the type of analysis and the degree of supervision needed, are separated into four learning categories. These include supervised learning, unsupervised learning, semi-supervised and reinforcement learning.

1.2.1. Supervised learning

Supervised learning (SL) is the machine learning task of learning a labeled training set of variables[6]. In SL, each observation is a pair that consists of an input vector and a desired output value. Supervised learning algorithms learn the labeled training set and generate an inferred function which is utilized to perform predictions on unknown new data points.

SL is used in both classification algorithms, where data points correspond to a certain class and regression predictive models where the relationship between independent and dependent variables is investigated. SL predictive accuracy depends on the quality of the training datasets.

1.2.2. Unsupervised learning

Unsupervised learning (UL) is a machine learning type that learns patterns from unlabeled data points and there is no prior knowledge for the data labels. Algorithms within the UL scope aim to produce patterns, clusters or association rules that can robustly describe the input data points. Popular clustering algorithms (e.g., k-means) fall into this category.

1.2.3. Semi-supervised learning

Semi-supervised learning is a hybrid approach for machine learning that combines a small portion of labeled data with a large quantity of unlabeled data points for the training set. It lies between unsupervised and supervised learning types, and thus it is considered a special instance of weak supervision.

1.2.4. Reinforcement learning

Reinforcement learning (RL) is another type of machine learning and differs from supervised learning by not needing labeled input/output data point pairs to be presented. Instead, it focuses on finding balance between exploring and exploiting the dataset [7]. Partially supervised RL algorithms can combine the advantages of supervised and RL algorithms [8].

1.2.5. Dimensionality reduction

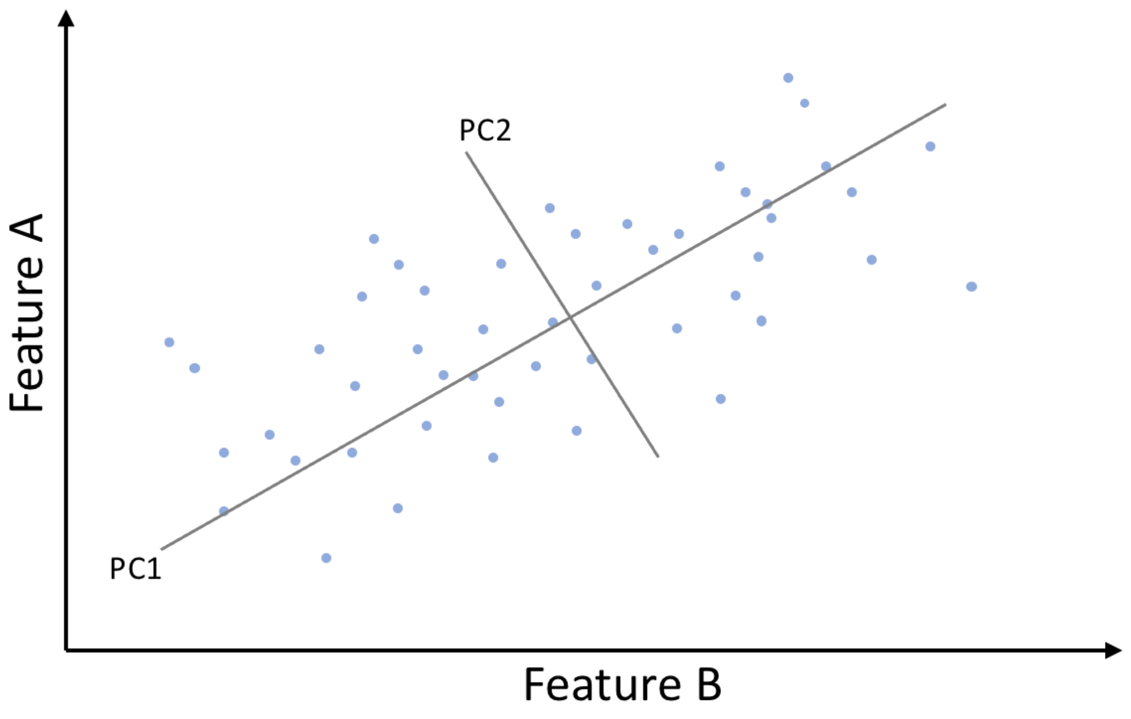

Figure 1: PCA transformation. Features A and B are transformed into a new coordinate system PC1 and PC2. (Figure has been created for the needs of the current thesis)

Principal component analysis (PCA) is a method suitable for the analysis of big datasets with a high dimensional space. PCA reduces the dimensions of a dataset while trying to minimize any information loss. Data are linearly transformed into their major components and projected into fewer dimensions, where most of the variation is explained. Thus, PCA mathematical approach lies in the maximization of the covariance while the new coordinate system demands the eigenvalues and eigenvectors of the covariance matrix.

Linear discriminant analysis (LDA), is also a technique widely used for dimensionality reduction. In this case, except of the principal components that maximize the variance within data, LDA aims to maximize the differences between groups. Thus, the data tend to maintain the information of the classes of origin.

t-distributed stochastic neighbor embedding (t-SNE) is a statistical method for the visualization of high-dimensional datasets. T-SNE is giving each datapoint a location in a lower-dimensional map and is based on a Stochastic Neighbor Embedding [9]. t-SNE is a nonlinear dimensionality reduction technique for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions. To visualize the data in low dimensions t-SNE creates a probability distribution over pairs of high-dimensional objects in such a way that similar objects are assigned a higher probability while dissimilar points are assigned a lower probability. Furthermore, t-SNE defines another probability distribution for the points in the low-dimensional space, while it minimizes the Kullback–Leibler divergence (KL divergence) between the two distributions with respect to the locations of the points.

Uniform Manifold Approximation and Projection (UMAP) is another dimensionality reduction technique that can be used to visualize patterns of clustering in high-dimensional data. UMAP remains very similar to t-SNE technique and attempts to create local clusters without preserving global structure of the data. It is a widely used approach for single cell RNA sequencing data, able to denote different cell types.

1.2.6. Assessment of machine learning models

While building ML models, part of the data is excluded from the training process to later serve as an evaluation set. Comparing the model predictions on this “unseen” set with the true labels of the data will produce several accuracy metrics which indicate the performance of the model. Among the most popular and robust metrics are precision, sensitivity (or recall) and specificity. Regarding classification problems (see chapter 1.3.1 Classification) and for visualizing the performance of the model, Area Under the Curve (AUC) or Receiver Operating Characteristics (ROC) are widely used as performance indicators. Wider area under the curve, created by sensitivity and 1 – specificity axis, indicates better degree or measure of separability for the predictive model.

$$\text{sensitivity} = \frac{\text{number of true positives}}{\text{number of true positives} + \text{number of false negatives}}$$

$$\text{specificity} = \frac{\text{number of true negatives}}{\text{number of true negatives} + \text{number of false positives}}$$

Lastly, F1 score combines the precision and sensitivity of a classifier into a single metric by taking their harmonic mean. Also allows for better balance between the two metrics, especially in case of unbalanced datasets.

$$F = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

1.3. Categorization of Machine Learning applications

1.3.1. Classification

Classification algorithms refer to the process of data categorization in different classes or categories. With the training process followed for creating predictive classifiers, output models can identify the designated label for each unknown data point with high accuracy. Several classifiers are thoroughly explained in chapter "1.4 Machine learning algorithms".

1.3.2. Regression

Regression analysis is a process for estimating the relationship between a dependent variable and a set of independent variables. This type of analysis aims to estimate the value of the independent variable.

Linear regression is a very popular and robust form of regression analysis, where a line is calculated that most closely fits the data in respect to ordinary least squares (OLS). OLS calculates the unique line, which minimizes the sum of squared differences between the true data and the line.

1.3.3. Clustering

Clustering [10] is the process where data points are separated into subsets in such a way that objects in the same group are more related to each other than those in the other groups [11].

There are many different approaches for clustering datasets that led to many algorithms, comprising two major categories, Hierarchical and Partitional. The former construct new clusters with every step, while utilizing the algorithm and the latter construct all clusters simultaneously (see chapter 1.4.8 for k-means clustering algorithm).

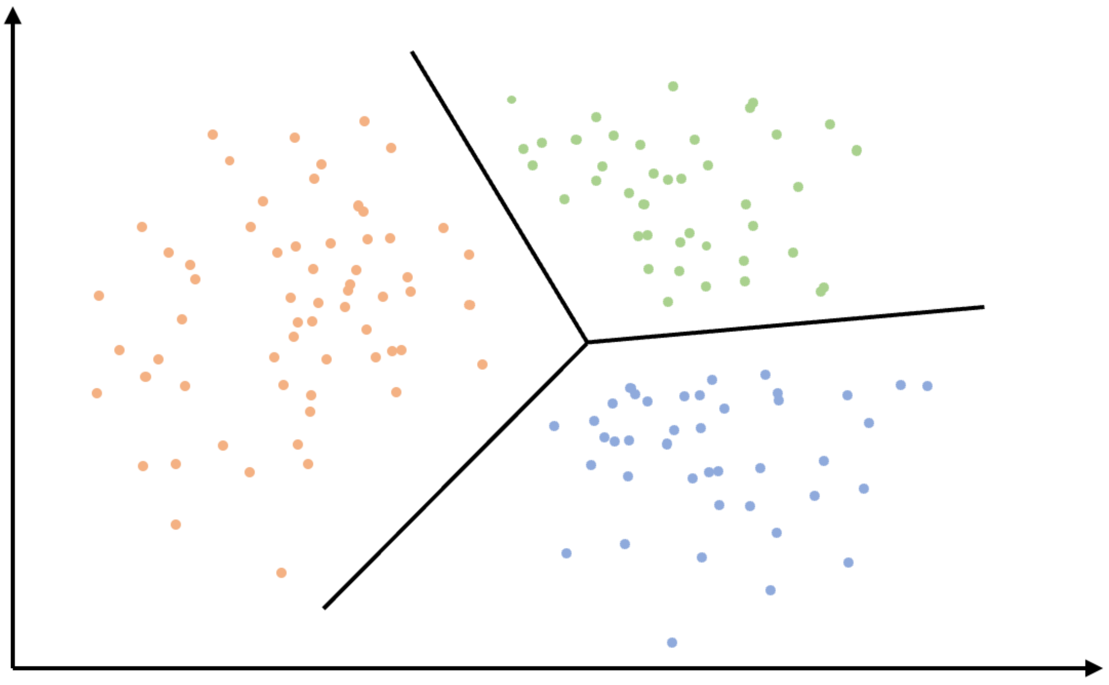

Figure 2: Clustering analysis. Data points are separated according to capable features. (Figure has been created for the needs of the current thesis)

1.4. Machine learning algorithms

The following chapters focus on ML algorithms, utilized in current study, for solving large scale biological problems.

1.4.1. Naive Bayes classifier

Naive Bayes classifier is a supervised classification ML algorithm which assumes that features of a measurement are independent of each other. Thus, these features depend on each other, and all properties independently contribute to the classification probability of an observation. For each class, the algorithm calculates the probability for the observation and classifies it in the class with the highest probability. The conditional probability is calculated from the following formula:

$$P(X \mid C_i) = \prod_{\chi=1}^{n} P(x_k \mid C_i)$$

1.4.2. Support Vector Machines

Support Vector Machines (SVMs) are categorized as a supervised learning technique and are utilized for both classification and regression analysis. SVMs first appeared by Vladimir Vapnik in 1963 [12] and remains one of the most popular method for predictive analysis. SVMs map training data points in space in a way that they maximize the width of the gap between the two categories. A separating line/surface is created between surfaces supported by the samples on margins which are called support vectors (Figure 3). New predictions are mapped in the space of the hyperplane and a probability is predicted for which class/category they belong to. In conclusion SVMs try to maximize the distance between the classes while minimizing the classification error. SVMs have been used for many bioinformatics tasks [13].

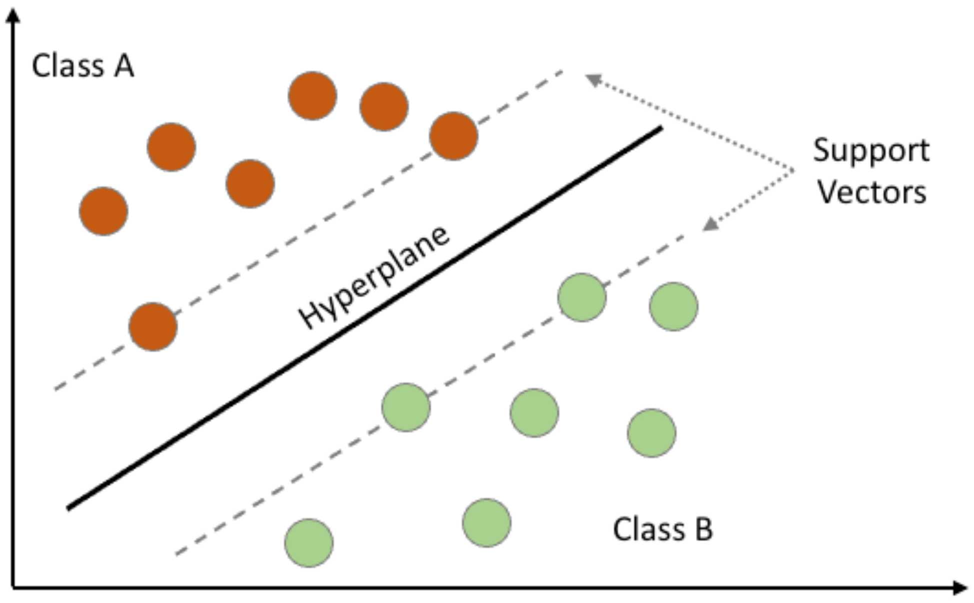

Figure 3: SVMs. Maximum margin Hyperplane. Support vectors are created by the samples on the margins. (Figure has been created for the needs of the current thesis)

1.4.3. Random forest

Random forest is a supervised ensemble learning technique utilized for classification and regression problems and operates by contracting a multitude of decision trees (see chapter 1.4.7) during the training process. In the case of classification analysis, random forest output is the class which was selected by the majority of the trees. For regression problems, the output of the algorithm is the mean or average prediction of the individual trees. Random forest overcomes the overfitting problem during training of decision trees but gradient boosted trees demonstrate higher accuracy [14]. However, input data characteristics may affect performance.

The first random forest algorithm was introduced by Tin Kam Ho [15] in 1995 and utilized the random subspace method [16], which was a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg [17].

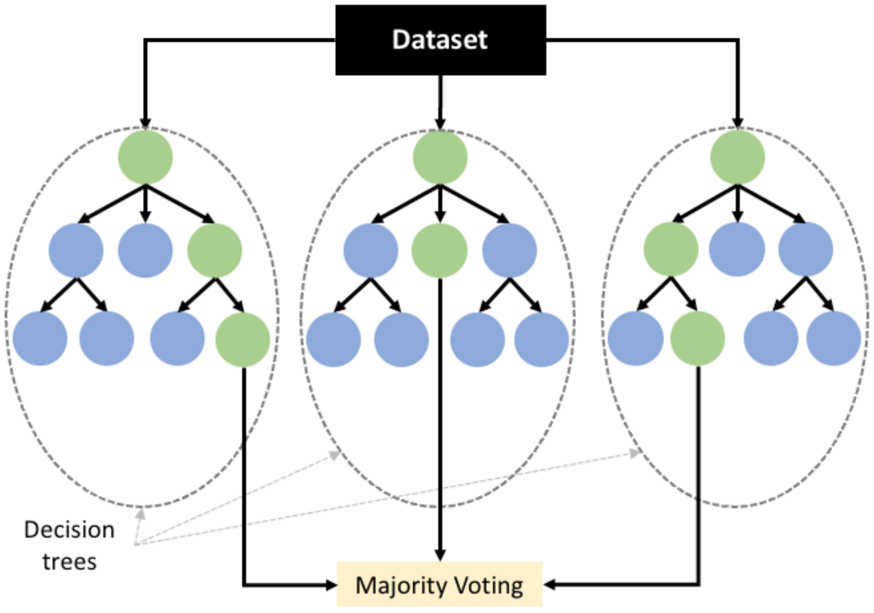

Figure 4: Overview of Random Forest algorithm. The final prediction comes from the majority vote of the individual decision trees. (Figure has been created for the needs of the current thesis)

1.4.4. K-nearest neighbors

K-nearest neighbors algorithm (K-NN) is a non-parametric supervised learning method which was first developed by Evelyn Fix and Joseph Hodges [18] and is used for solving both classification and regression problems. The KNN algorithm assumes that similar data points will be in closer proximity. K-NN uses the k nearest data points of the training set, to obtain the mean (in case of regression analysis) or for defining the class (for classification problems). The "K" value is a hyperparameter of the algorithms and directly affects the performance of the method. Several distance metrics can be used for assigning the data points to the corresponding class such as Euclidian distance, cosine similarity and chi-square distance. Metrics are used depending on the dataset and type of the analysis.

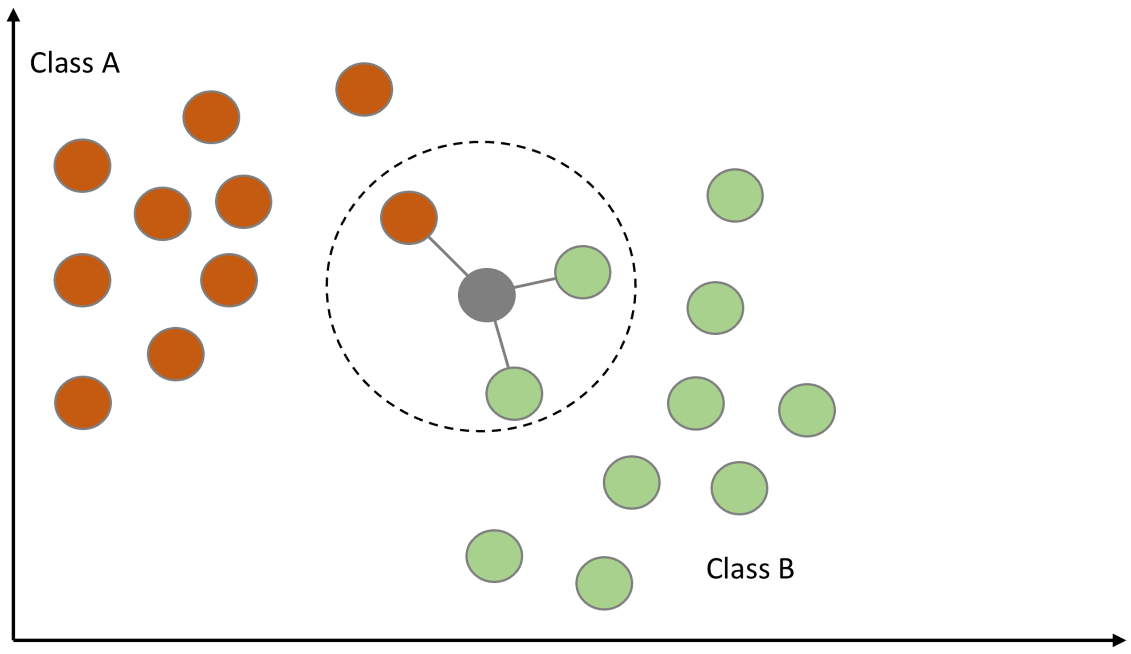

Figure 5: K-NN. Current K value is 3 (K=3). The grey data point, which is under classification, the majority of the 3 nearest neighbors define the class. The datapoint will be classified in "class B". (Figure has been created for the needs of the current thesis)

1.4.5. Linear regression

Linear regression can be referred as a linear approach for modeling the relationship between a dependent and one or more independent variables. In the case of one independent variable the method is called simple linear regression and for more than one, the process is called multiple linear regression [19]. Thus, a mathematical equation can describe the above relationship. The following form describes the relationship of y dependent variable,

$$\mathrm{y} = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$$

where β are the coefficients, with the independent variables χ. Furthermore, there are several methods for calculating β coefficients such as least squares method or maximum likelihood. Linear regression models are usually fitted using the least squares approach by achieving the best-fit regression line. Regression models fall under the category of supervised learning methods.

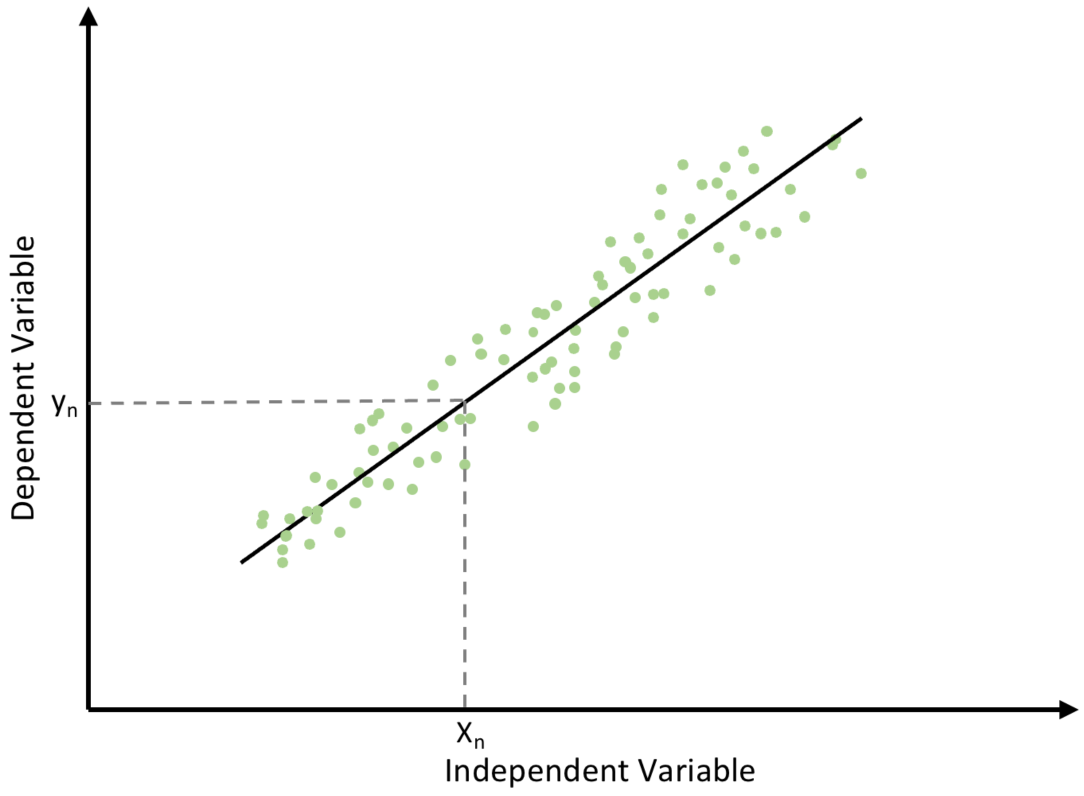

Figure 6: Example of linear regression. Observations are assumed as the result of random deviations from a relationship between the dependent variable (y) and an independent (x) variable. (Figure has been created for the needs of the current thesis)

1.4.6. Logistic regression

Logistic regression is a classification method widely used when the dependent variable is categorical [20]. Usually, the method is used with a binary dependent variable while the independent variables have continuous or discrete values. Furthermore, multinomial logistic regression is an extension of binary logistic regression that allows for more than two categories of the dependent variable.

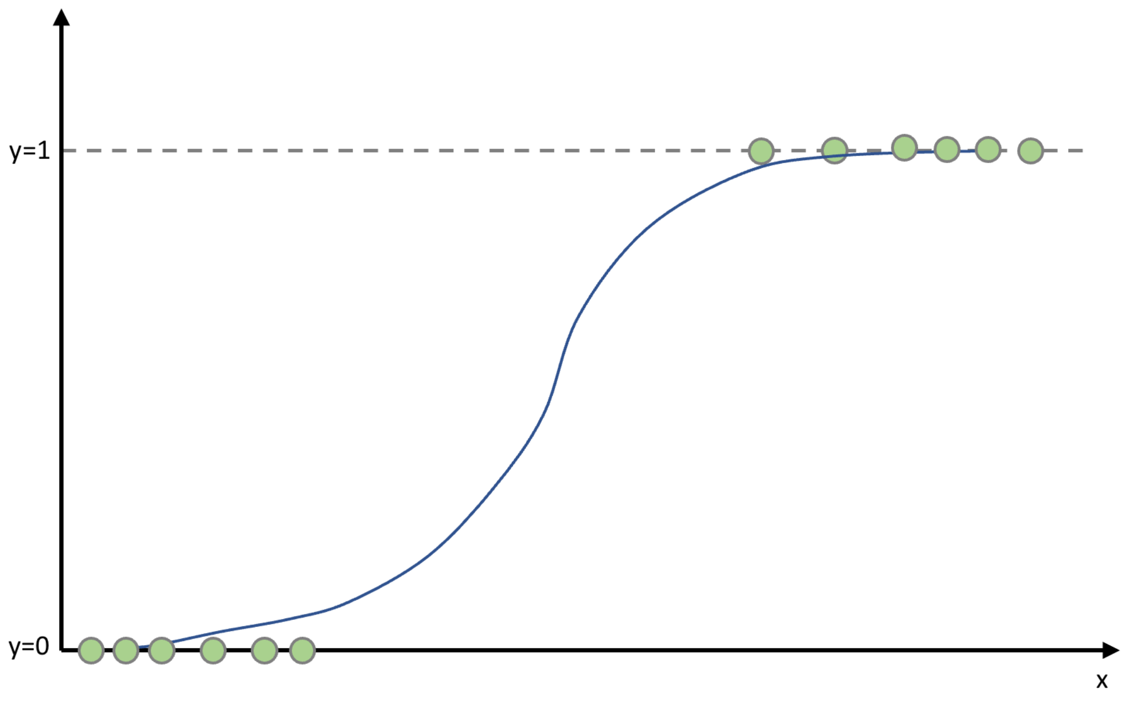

Figure 7: Logistic regression. Independent variable (x) has discrete values compared to dependent variable (y) which is binary [0, 1]. (Figure has been created for the needs of the current thesis)

In contrast with linear regression that aims to describe the data points with a line, in logistic regression a sigmoid function is fitted on the data by tuning several parameters for the best fit. Thus, sigmoid function imposes the output values between [0,1]. New observations are given a probability to belong to either class.

1.4.7. Decision trees

Decision Trees (DTs) is a non-parametric supervised learning method that can be utilized for classification and regression analysis. DTs aim to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. In decision analysis, a decision tree can be used to represent decisions and support decision making visually. Also, DTs are represented with nodes and leafs in a tree-like model of decisions. Amongst the algorithms proposed for DTs, ID3 [21] (which works only for discrete data) and CART [22] (for discrete and continuous data) are the most popular.

ID3 utilizes a top to down approach and for each node the best feature regarding the data separation is selected. Furthermore, ID3 uses entropy as a measure for the data homogeneity. The process is repeated until all sub-trees consist of homogeneous data. On the other hand, CART creates binary trees and utilizes brute force search for optimal separation of the data.

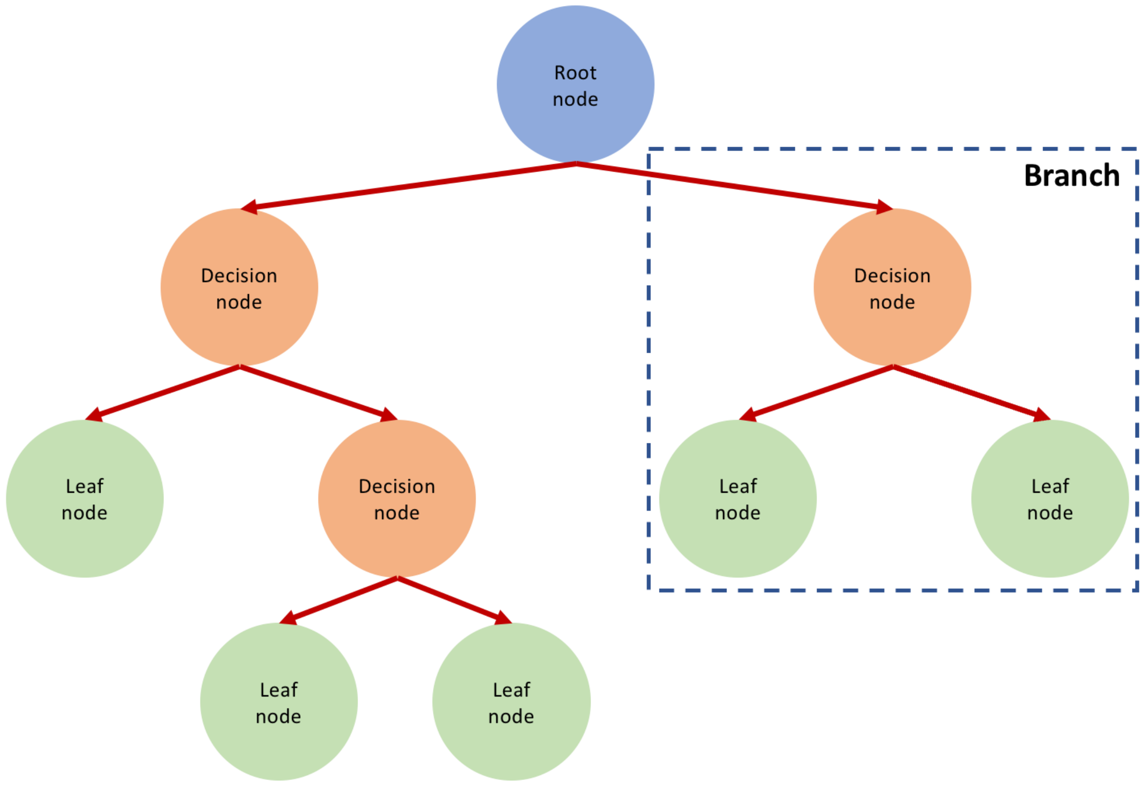

Figure 8: Decision tree overview. (Figure has been created for the needs of the current thesis)

1.4.8. K-means

K-means is an unsupervised clustering algorithm for separating observations in groups. Centroids for each group are randomly placed. In each step of the algorithm the mean of each cluster is calculated and represents the new centroid for each cluster (Figure 10). The squared Euclidean distance metric (or other metrics e.g., mahalanobis distance) is calculated for estimating the current mean of each cluster.

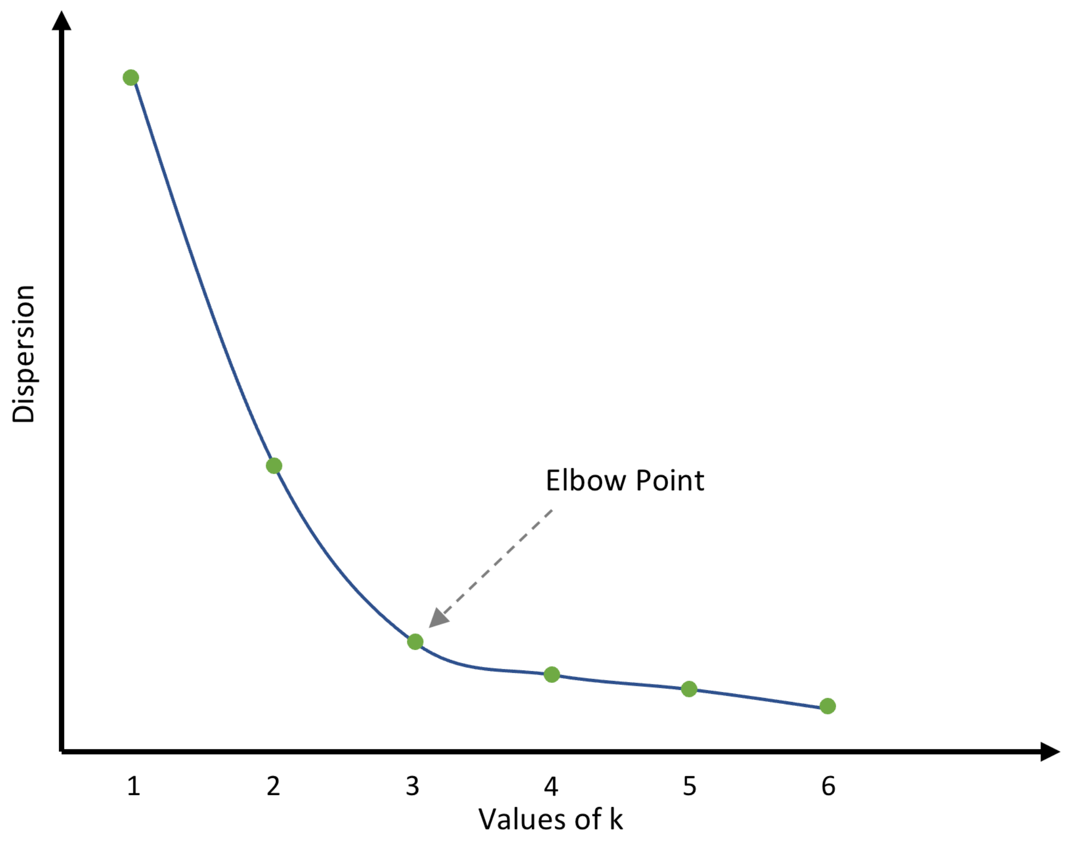

Figure 9: Elbow methods. Optimal number of clusters is three (3) as indicated by the elbow point. (Figure has been created for the needs of the current thesis)

“K” value indicates the number of centroids/clusters to be created and is an input parameter. There are several methods for finding the optimal k value with the elbow method finding great popularity (Figure 9). K-means clustering tries to minimize within-cluster variances. Also, the algorithm has a loose relationship to the k-nearest neighbor classifier (see chapter 1.4.4 K-nearest neighbors).

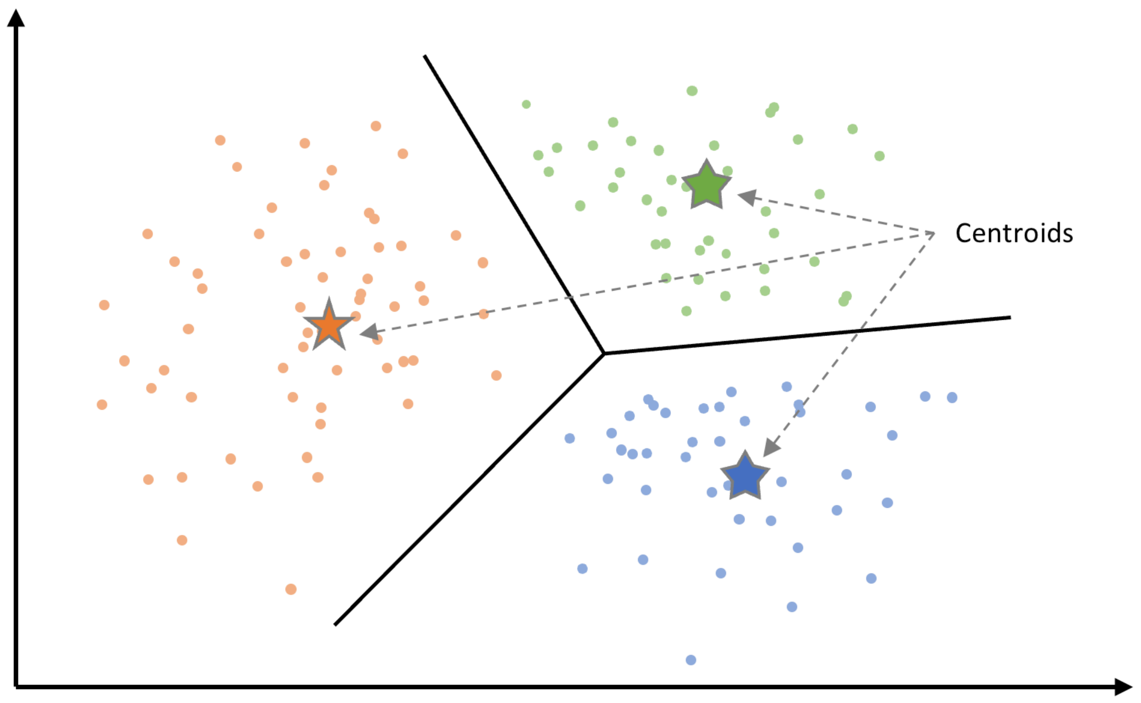

Figure 10: K-means clustering. Stars represent the 3 centroids (Green, blue, orange). Black lines indicate the borders between the three (3) classes. (Figure has been created for the needs of the current thesis).

1.5. Neural networks

1.5.1. Artificial neural networks

An artificial neural network (ANN) is a method of connected nodes (or else called artificial neurons) inspired from biological neural systems (like the human brain) and can perform calculations. These nodes are connected, similarly to neuron synapses in a biological organism, and transmit signals to nearby neurons. Artificial neurons receive signals, processes them, and transmits signals to neighbor connected neurons. These connections are called edges and both neurons and edges have weights assigned to them which adjusts the learning process (Figure 11).

This approach was initially adopted in 1943 with the first neural network presented by McCulloch & Pitts [23]. During the 1980s neural networks were improved with Hopfield model [24] and the introduction of the back propagation algorithms [25].

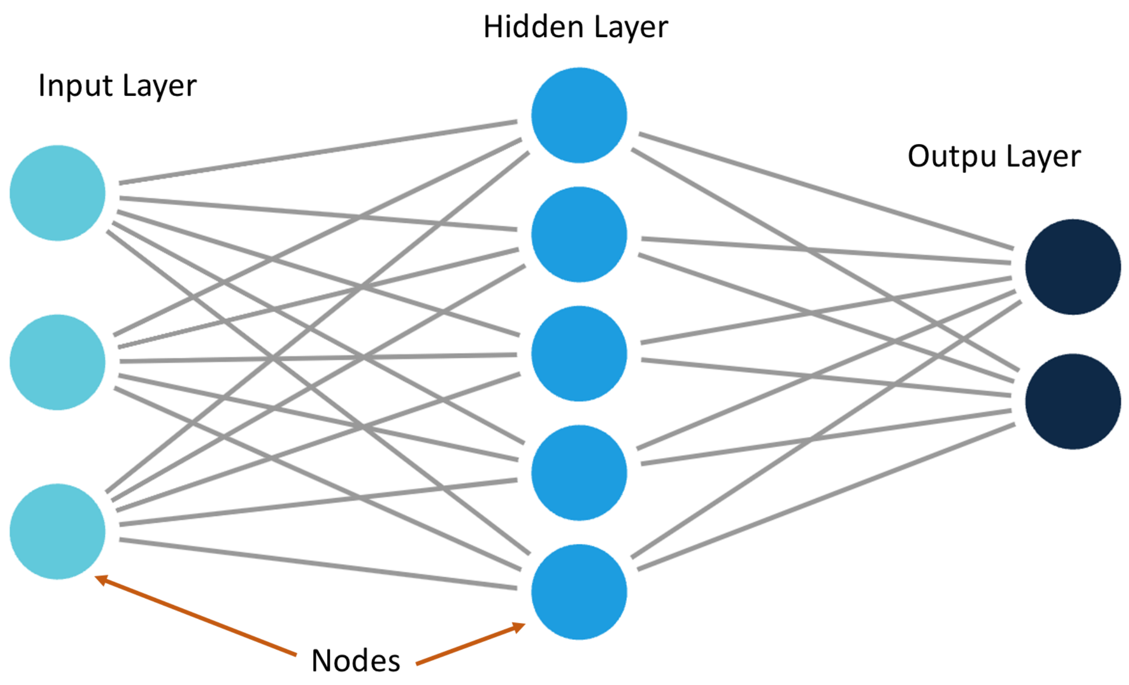

Figure 11: Overview of an ANN (Figure has been created for the needs of the current thesis).

1.5.2. Recurrent neural network

A recurrent neural network (RNN) is a type of ANN where connections between neighboring nodes can create a cycle that allows output from some nodes to affect future inputs to the same nodes which allows the network to demonstrate a dynamic behavior (Figure 12).

RNNs derive from feedforward neural networks and use feedback loops to process a sequence of data which informs the final output. These feedback loops (also often described as memory) allow information to persist. RNNs are widely used for tasks such as speech recognition [26] and natural language processing [27].

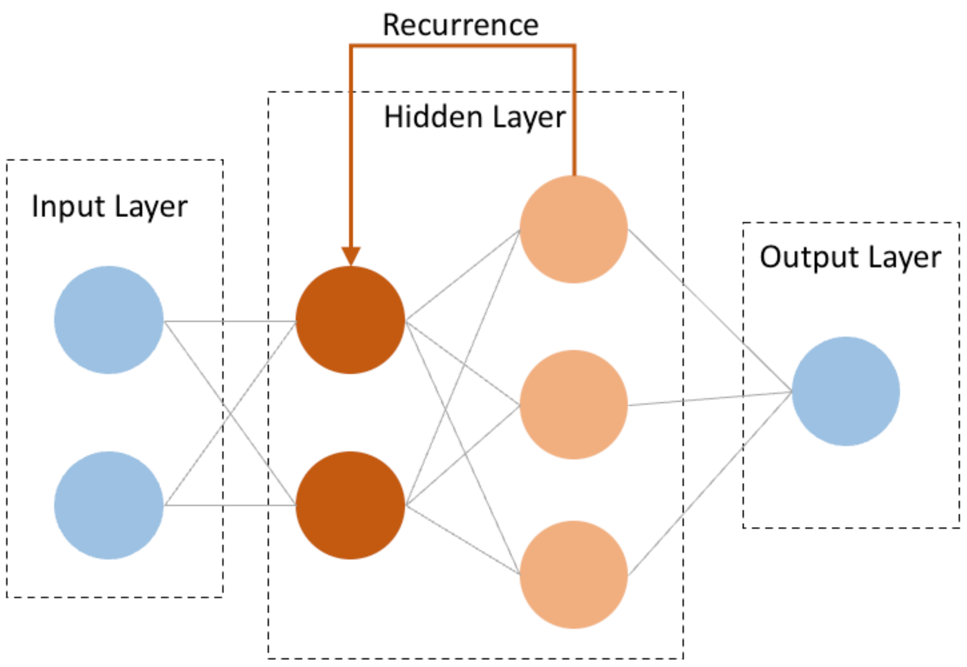

Figure 12: Overview of RNN (Figure has been created for the needs of the current thesis).

1.5.3. Feedforward neural network

A feedforward neural network (FNN) is an ANN where connections between the nodes do not create a cycle [28] like in RNNs. FNN was the first type of an ANN [2] where the information is passed forward to the next nodes (in only one direction) through the input nodes (Figure 13).

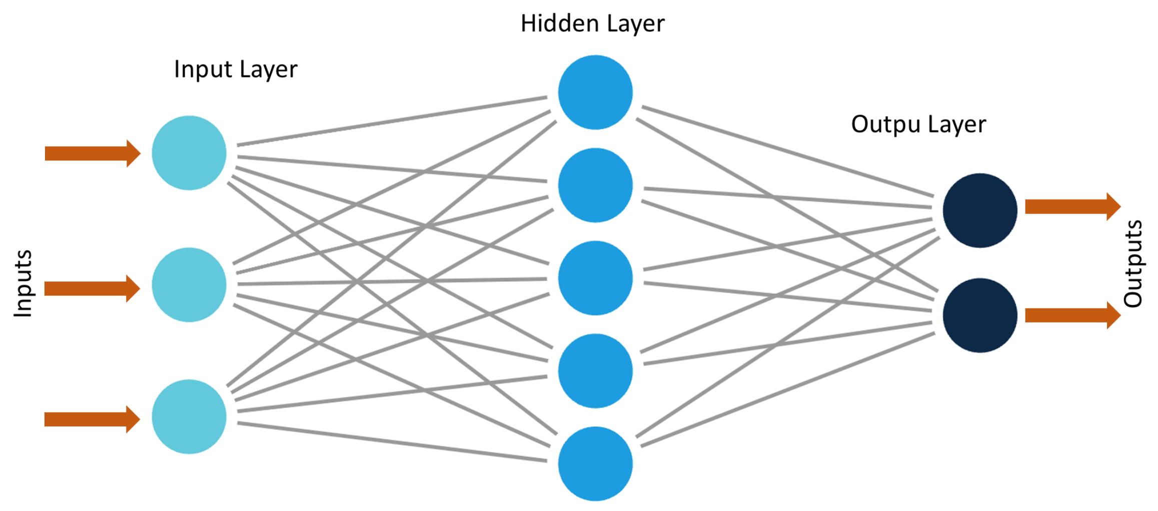

Figure 13: Overview of FNN (Figure has been created for the needs of the current thesis).

The simplest form of the FNN is the single layer perceptron [23]. In a single layer perceptron, inputs are fed into the layer and multiplied by the corresponding weights. All values are then summed up to produce a sum of the weighted input values. Finally, the sum is compared to a threshold previously set, and the value produced takes values (usually zero or one) [0,1] if below or above the threshold (Figure 14).

Single layer perceptron is an important model of FNN and as a supervised learning method it is used for classification tasks. Using the delta rule [29], which is a gradient descent learning rule for updating the weights of the inputs, the network can compare the outputs of the nodes with the intended values, allowing the network to readjust the weights through the training process to improve its accuracy.

In the multi-layered perceptron algorithm, the process of updating the weights is nearly analogous. However, it is defined more specifically as a back-propagation method. In such cases, each hidden layer within the network is adjusted according to the output values produced by the final layer.

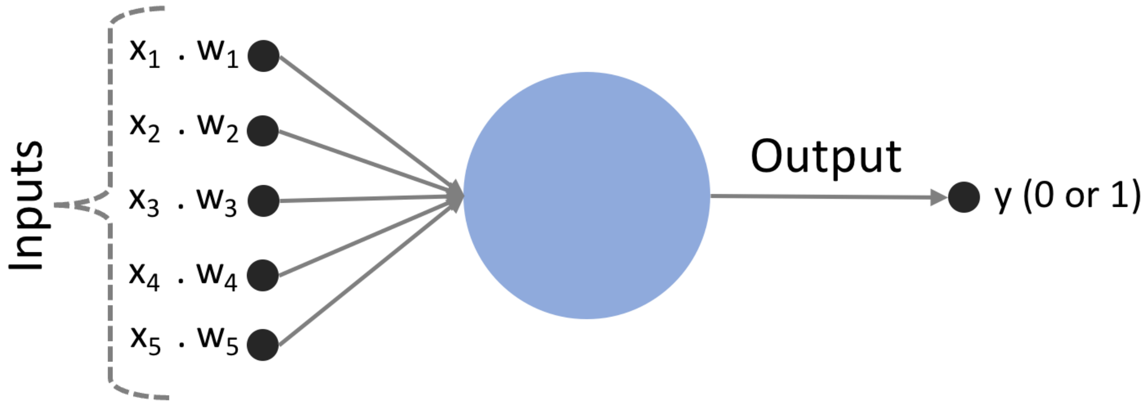

Figure 14: Single layer perceptron overview (Figure has been created for the needs of the current thesis)

1.5.4. Autoencoder

Autoencoders are an unsupervised learning technique and a subcategory of feedforward neural networks. They can compress the input into a lower dimensional space with minimum loss and reconstruct the output to the original data (Figure 15).

Autoencoders use three main components, the encoder, the code, and the decoder. The encoder is responsible for compressing the input producing the code, and the decoder reconstructs the code to the original input data. Autoencoders are mainly used as a dimensionality reduction technique or for compressing data. Autoencoders are trained as an ANN, with the backpropagation method.

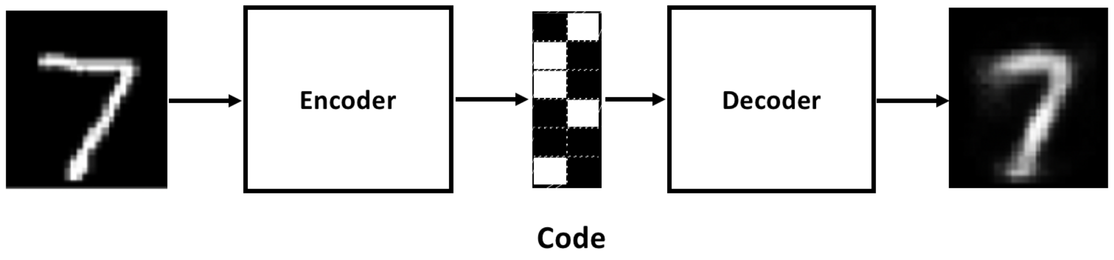

Figure 15: Autoencoder architecture. Input is a number image (seven). Output can maintain the core information of the image with partial information loss. (Figure has been created for the needs of the current thesis)

1.5.5. Long short-term memory

Long short-term memory (LSTM) [30] is an ANN which, unlike standard FNN networks, has feedback connections. These networks combine the advantages of the RNNs and take it a step further by expanding lifetime of the memory, thus their name combines the two phrases "long-term memory" and "short-term memory". LSTM is used for tasks as handwriting recognition [31], speech recognition [26] and more and these networks have become very popular in the last decade. The LSTM architecture aims to achieve a short-term memory, like RNNs that can last thousands of timesteps, and therefore become "long short-term memory" [30]. LSTMs networks thrive in prediction for time series data where lags of unknown duration between important events may happen.

1.5.6. Convolutional neural network

A convolutional neural network (CNN or ConvNet) is another class of ANNs, mainly used in computer vision to classify image data [32]. Thus, CNNs are deep learning algorithms that can find patterns within object inputs and are able to differentiate the inputs in classes. The preprocessing of the inputs required in a CNN is minimal in contrast to other classification algorithms.

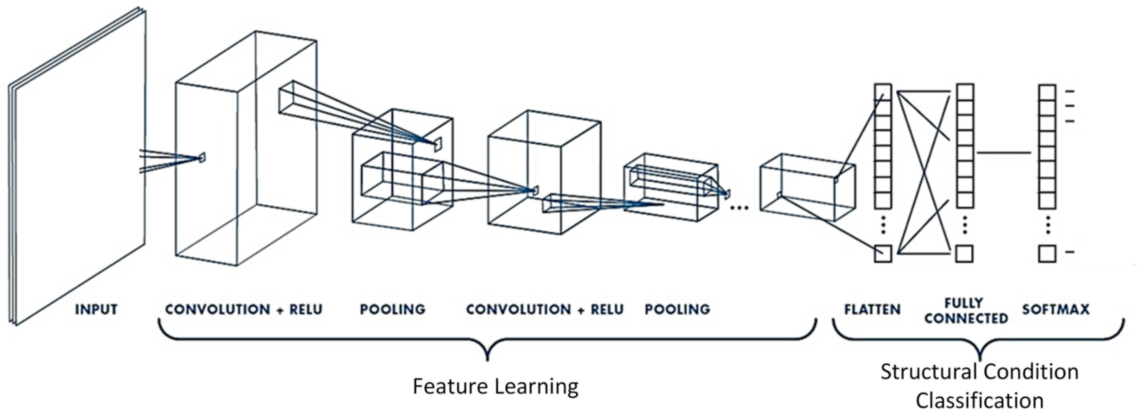

Figure 16: CNN architecture [33].

A CNN can robustly capture the Spatial and Temporal dependencies within an input through the application of relevant filters. Their architecture enables a better fitting to the input dataset due to the reduction in the number of parameters involved and reusability of weights. CNNs demonstrate superior performance especially when classifying images compared to conventional ML algorithms. There are three main types of layers in CNN architecture, convolutional layer, Pooling layer, and fully connected layer.

Convolutional layers are the core building block of a CNN architecture. They are the first layer of a CNN and can be followed by additional convolutional layers or pooling layers with the fully connected layer in the final stage of the architecture. Complexity is increased as more layers are added to the CNN and greater aspects of the inputs can be identified. Primary layers focus on simple features (like edge detection) and downstream layers start to recognize elements or shapes. Hence, convolutional layers are where the majority of computation occurs.

1.6. Deep learning

Deep Learning [34] is a subfield of machine learning and learns by discovering intricate structures in the data. Such models use multiple layers to progressively extract higher level features from the raw inputs. Hence, each level in a deep learning model learns to transform the inputs into a more abstract representation.

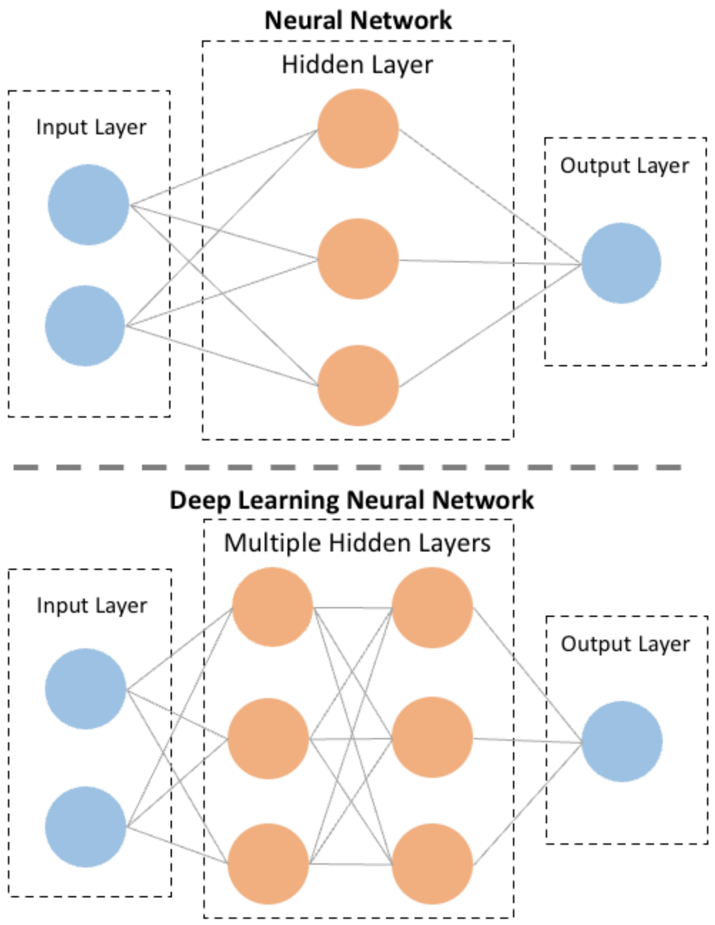

Figure 17: Neural network vs deep learning neural network. (Figure has been created for the needs of the current thesis)

Moreover, deep learning model architectures such as deep neural networks, deep reinforcement learning, autoencoders, convolutional neural networks and lately Transformers [35] (which is a model mainly used in NLP) have been successfully applied to fields including computer vision [36], speech recognition, NLP, bioinformatics [37], drug design, medical image analysis, and games [38], where they have produced highly accurate results and in some cases even surpassing human performance.

1.7. Big biological data

With next generation sequencing techniques (see chapter 1.7.1) flourishing in the past years, the scientific community has entered an era of big biological data and the need for analyzing and extracting meaning out of them arose. There are five main types of data that are massively growing and utilized to answer biological questions including, gene expression data, DNA, RNA, and protein sequence data, protein to protein interaction (PPI), pathway data, and gene ontology (GO) [39].

For this study, ML techniques are applied on a large collection of big biological data, aiming to make predictions and correlations for gaining knowledge about major, state-of-the-art biological problems.

1.7.1. Next Generation Sequencing

DNA double helix structure was discovered in 1953 by James Watson and Francis Crick [40]. Rosalind Franklin, who was an X-ray crystallographer, also contributed to this huge scientific discovery [41]. Watson and Crick were awarded the Nobel Prize in Physiology or Medicine in 1962, shortly followed by Robert Holley in 1968 for sequencing the first RNA molecule [42]. The combination of these immense discoveries paved the way for sequencing the DNA double helix. In 1977 Fredrick Sanger [43] developed the chain-termination method, which uses a sequence of interest as a template for a PCR and adds modified nucleotides, called dideoxyribonucleotides, on the DNA strand during the extension step [44]. By 1986, the first automated DNA sequencing method had been developed. Further ahead, in 2005, the second generation (2G) of sequencing (or NGS) emerged, and enabled the amplification of millions of copies of DNA fragments in a parallel structure [45] surpassing the limitations of sanger sequencing. The main goal of NGS compared to first generation sequencing technologies was to lower the cost and improve the performance of the method.

NGS technologies offer many advantages over alternative sequencing techniques, including the ability to generate sequencing reads in a cost-effective, fast, and sensitive way. Nevertheless, there are also some drawbacks, including poor interpretation of homopolymers and incorporation of incorrect dNTPs by polymerases which result in some sequencing errors. The need for deeper sequencing coverage arose, because of short read lengths, and progress in this domain will enable more accurate final genome assembly

[46]. The major drawback of NGS techniques is the need for PCR amplification prior to sequencing that is associated with PCR bias in the library preparation stage.

The 3G sequencing bypasses the need for PCR amplification and is capable of sequencing single molecules. Stephen Quake introduced the first single molecule sequencing technology [47], by obtaining sequence information with the use of DNA polymerase. The method was monitoring the incorporation of fluorescently labeled nucleotides to DNA strands with single base resolution.

1.7.2. CAGE-Sequencing

CAP Analysis of Gene Expression (CAGE) is a protocol introduced in 2003 [48] for capturing and quantification of capped RNA 5’ ends.

CAGE produces a set of short nucleotide sequences (or tags) approximately 20 to 21 nucleotides long that correspond to the TSSs for the vast majority of human transcripts with the exception of RNAs without the 5’-cap such as rRNAs. Mapping the CAGE tags to the genome allows the quantification of RNA transcripts within a biological sample and simultaneously the identification of the tissue/cell-type specific TSSs and thereby their promoters.

Assigning identified TSSs and promoters to either known or novel genes, a process called annotation, remained a challenging bioinformatics task for several years. In 2010 the CAGEscan [49] methodology was introduced and provided a link of TSSs detected by CAGE to downstream sequence regions contributing to novel promoter discovery and offering new perspectives into the relations of transcribed RNAs to their neighboring genomic and/or transcriptomic elements.

A short overview of the CAGE-sequencing protocol (Figure 18) and downstream analysis is described below. First, reverse transcription and cDNA synthesis is performed with random priming to ensure the capture of both polyadenylated and non-polyadenylated RNA. Then, the RNA 5’cap and 3' ends are biotinylated followed by RNAse I treatment where non-hybridized, single-stranded RNAs are digested. Streptavidin beads are used to capture the biotinylated cap-site of complete cDNAs, a process called Cap-trapping, and separate them from RNAs without the 5'-cap such as rRNA, truncated RNA, and incompletely reversely transcribed RNAs that remain in the solution.

After selection, cDNA is further processed for sequencing. First, it is released from the beads, then a linker is ligated to the single stranded cDNA and second strand synthesis is performed. The linker sequence contains a recognition site for type III restriction endonuclease that cleaves a fragment of approximately 27 nucleotides into the cDNA. The short fragments are then amplified and sequenced.

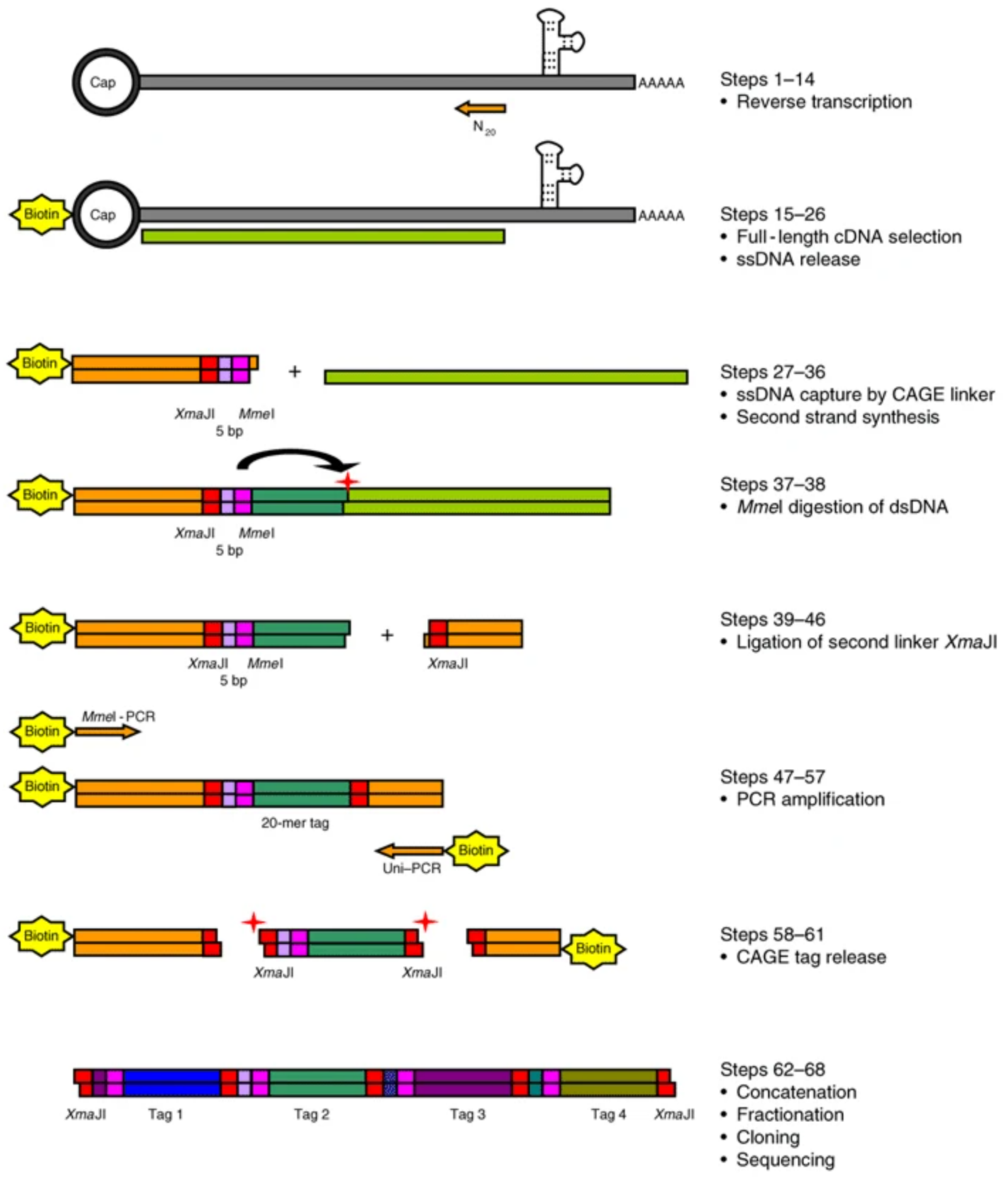

Figure 18: Overview CAGE protocol [50].

The output of the CAGE protocol is a set of short nucleotide sequences along with their observed counts. CAGE tags quantify the abundance of RNA transcripts within a biological sample.

1.7.3. WGS-Sequencing

Whole-genome sequencing (WGS) is a concise method for analyzing the entire DNA of a genome. Rapidly growing WGS genomic data revolutionized biosciences and supported scientists in characterizing the mutations involved in cancer progression, and tracking disease outbreaks.

Maxam–Gilbert sequencing [51] and Sanger sequencing [52] were the most popular DNA sequencing methods in the 1970s. Several whole bacteriophages as well as animal viral genomes were sequenced with these methods, but in the 1990s the need arose for more rapid and automated sequencing methods.

The entire DNA sequence of human chromosome 22 (the shortest human chromosome) was published in 1999 [53]. In 2000, the second genome of Drosophila melanogaster, a popular animal in experimental research, was sequenced [54] and the first plant genome (Arabidopsis thaliana) wassequenced by 2000 [55]. In 2001, a draft of the entire human genome sequence was published [56] and the genome of the laboratory mouse Mus musculus was completed in 2002 [57]. Lastly, in 2014 WGS was introduced to clinics [58].

With NGS, sequencing costs and time to produce large volumes of data with state-of-the-art sequencers made WGS a very powerful tool for advanced genomic research. All the exons of an individual's DNA, which provide instructions for making proteins, have been sequenced and are available in many public repositories. Mutations occurring within exons have attracted great attention within the scientific community and many of them have been related with several diseases. However, DNA variations that occur outside the exons can also affect regulatory elements and therefore gene regulation. WGS can determine variations in any part of the genome, including promoter loci which is the main subject of this study.

1.7.4. ChIP-Sequencing

ChIP-sequencing is an NGS method utilized to identify binding sites of DNA-associated proteins with the DNA by combining chromatin immunoprecipitation (ChIP) with massively parallel DNA sequencing. It can be used to precisely map global binding sites for any protein of interest (e.g., Transcription factors).

The ChIP protocol is divided in five steps (Figure 19). The first step is cross-linking [59] using formaldehyde and large batches of the DNA for obtaining a useful amount. The cross-links are created between DNA and the protein, but also between RNA and other proteins. Second step includes the process of chromatin fragmentation which breaks chromatin in chunks to get high quality DNA pieces for the ChIP analysis. Fragments are then divided in 500bp (or under) pieces [60] for better genome mapping quality. Next step utilizes chromatin immunoprecipitation, the process that enhances specific crosslinked DNA-protein complexes by making use of antibodies against proteins of interest which also removes non-specific binding sites. DNA recovery and purification takes place to separate the DNA from the protein and the final step includes qPCR and parallel sequencing.

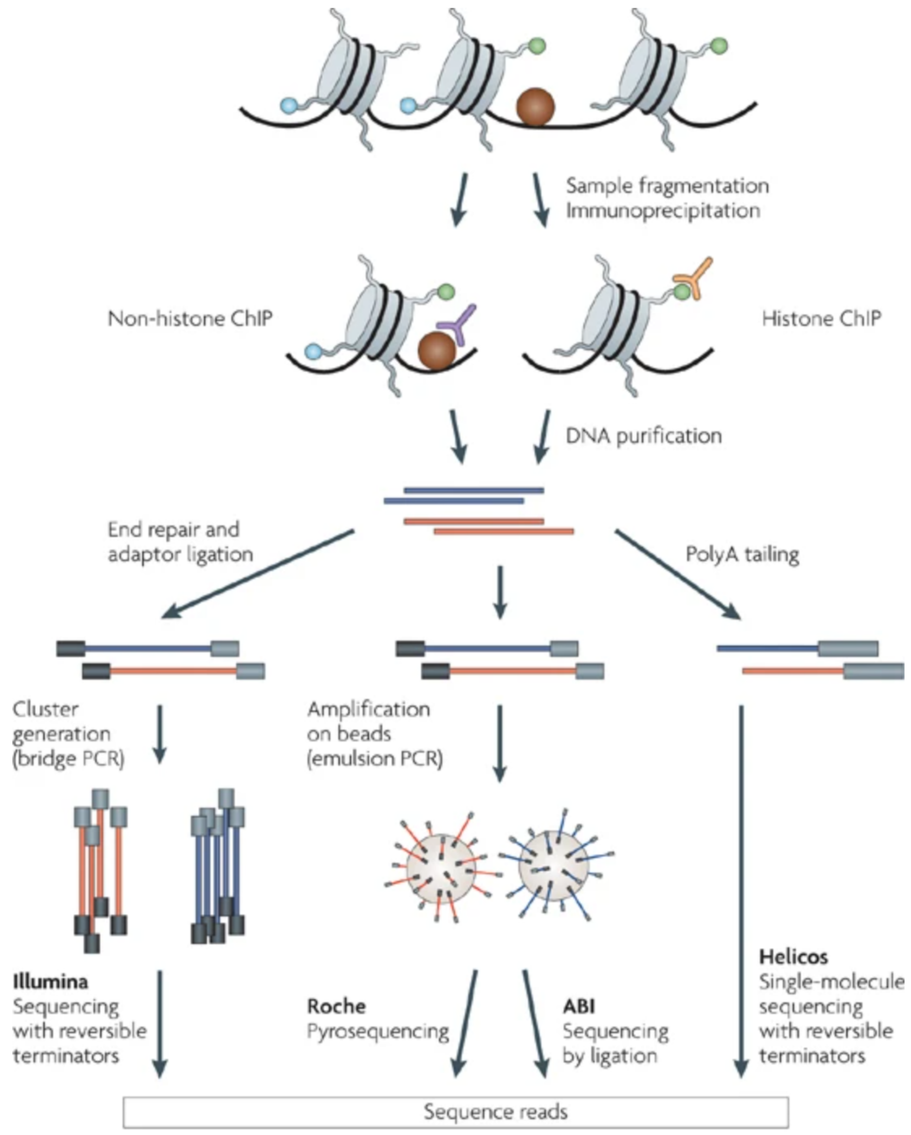
Sample fragmentation
Immunoprecipitation
Non-histone ChIP
Histone ChIP
DNA purification
End repair and adaptor ligation
PolyA tailing
Cluster generation (bridge PCR)
Amplification on beads (emulsion PCR)
Illumina
Sequencing with reversible terminators
Roche
Pyrosequencing
ABI
Sequencing by ligation
Helicos
Single-molecule sequencing with reversible terminators
Sequence reads

Figure 19: Overview of a ChIP–seq experiment. [61]

1.7.5. RNA-Sequencing

RNA sequencing (RNA-seq) is a sequencing method that utilizes NGS to discover and quantify RNA within a biological sample at a given moment [62]. It is widely used to identify and analyze alternatively spliced transcripts, post-transcriptional modifications, gene fusion and changes in gene expression or differences in gene expression between different groups [63].

The protocol’s main steps are described below and might vary between different sequencing platforms [64] (Figure 20).

RNA is isolated from the tissue and mixed with Deoxyribonuclease which reduces the amount of genomic DNA. The amount of RNA degradation is checked with gel and capillary electrophoresis. RNA quality and total abundance are taken into consideration for the subsequent library preparation, sequencing, and analysis steps.

For analyzing the signals of interest, isolated RNA can be kept untouched or filtered for RNA with 3' polyadenylated (poly(A)) tails to include only mRNA. Otherwise, RNA can be depleted of ribosomal RNA (rRNA) and filtered for RNA that binds specific sequences. RNA with 3' poly(A) tails is composed of mature, processed, coding sequences [65].

Poly(A) selection has several limitations regarding RNA biotype detection. Many RNA biotypes are not polyadenylated, including many noncoding RNA and histone-core protein transcripts, or are regulated via their poly(A) tail length and thus might not be detected after poly(A) selection [66]. Thus, poly(A) selection can display increased 3' bias, especially if RNA quality is low [67]. To avoid such limitations ribosomal depletion is used where rRNA is removed which typically represents more than 90% of the total RNA in a biological sample [68]. Small RNA targets, such as miRNAs, can be further isolated through size selection process with exclusion gels, magnetic beads, or commercial kits.

RNA is reversed transcribed into cDNA, a step called cDNA synthesis. DNA is more stable and allows for amplification using sequencing technology. Fragmentation is performed to purify sequences that are the appropriate length for the sequencing machine and size selection will remove small sequences [69].

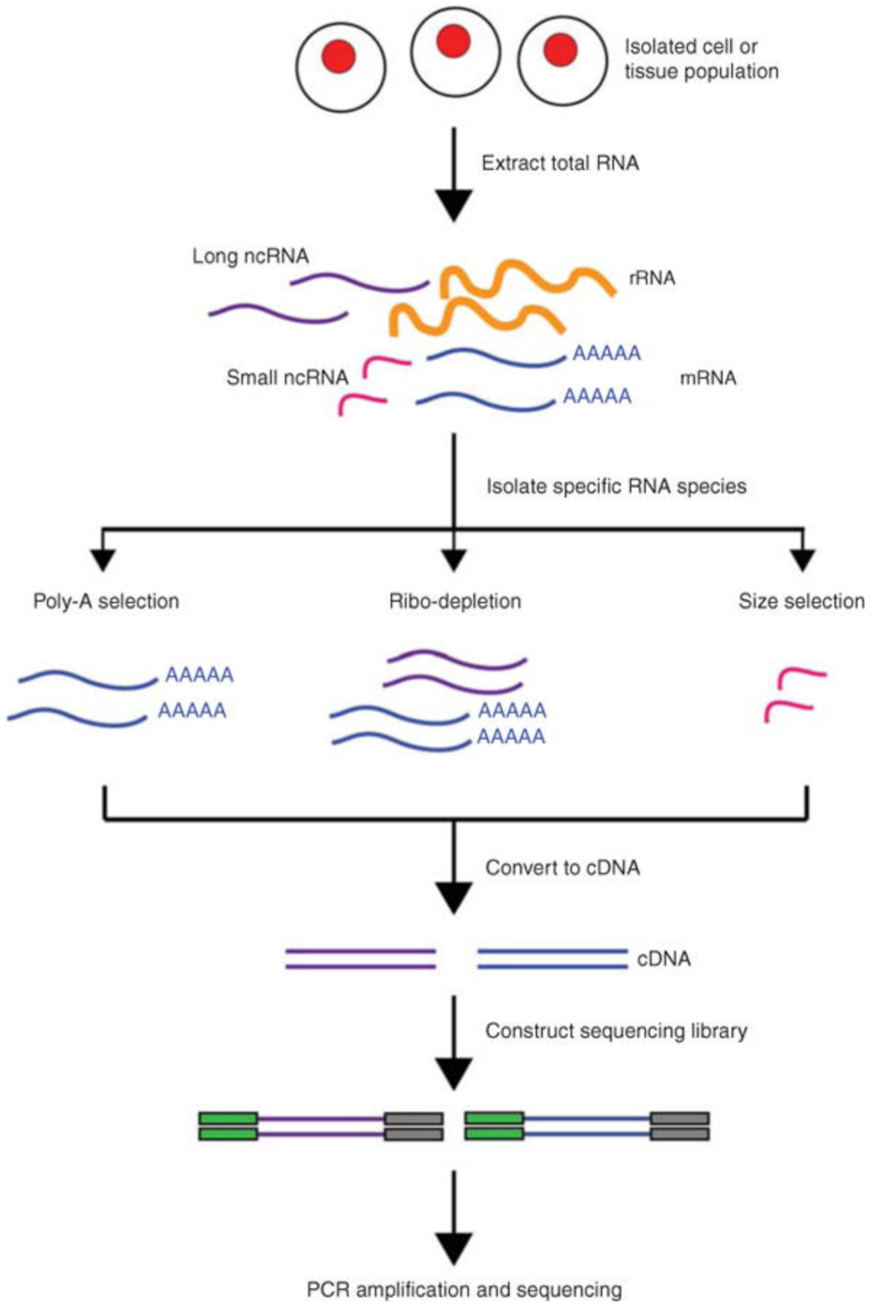

Figure 20: Overview of RNA-Seq. [70]

1.7.6. DNase-Sequencing

DNaseq-seq (or DNase I Hypersensitive sites) is a method in molecular biology used to identify the areas that are sensitive to deoxyribonuclease (DNase I) allowing the detection of active regulatory footprints (enhancers, promoters, etc.) on the genome. The method is based on genome-wide sequencing of regions sensitive to cleavage by DNase I [71].

DNase-seq (DNase I hypersensitive sites sequencing) is a method in molecular biology used to identify the location of regulatory regions, based on the genome-wide sequencing of regions sensitive to cleavage by DNase I.

The protocol treats DNA-protein complexes with DNase l, followed by DNA extraction and DNA sequencing. Sequences bound by regulatory proteins are protected from DNase l digestion. By utilizing deep sequencing, representation of regulatory proteins location comes with high accuracy genome wide [72] (Figure 21).

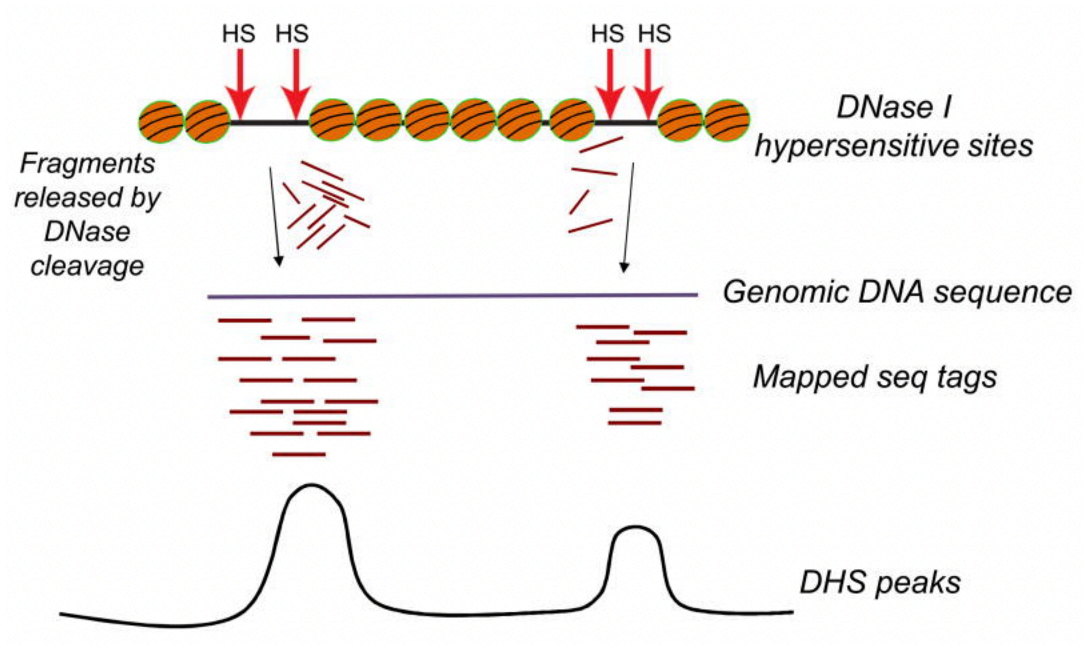
HS
HS
HS
HS
DNase I hypersensitive sites
Fragments released by DNase cleavage
Genomic DNA sequence
Mapped seq tags
DHS peaks

Figure 21: DNase-sequencing protocol overview [73].

1.8. Promoters of coding and non-coding genes

1.8.1. Promoters

Promoters are DNA sequences approximately 1000bp upstream and 500bp downstream of the genes (strand specific - towards the 5' region of the sense strand) (Figure 22) that contain regulatory elements of gene transcription. Several proteins, such as transcription factors, bind on them to initiate transcription of an RNA transcript. Several proteins, such as transcription factors, bind on them to initiate transcription of an RNA transcript.

Produced RNA transcripts may encode for a protein (mRNA), or they may have a different functional role in the cell. RNA polymerase is recruited on site to initiate transcription of the gene. Thereby, promoters have similar DNA sequence patterns with some minor variations. Promoter identification is a crucial problem in gene discovery and in understanding the regulation of genes in a cell [74].

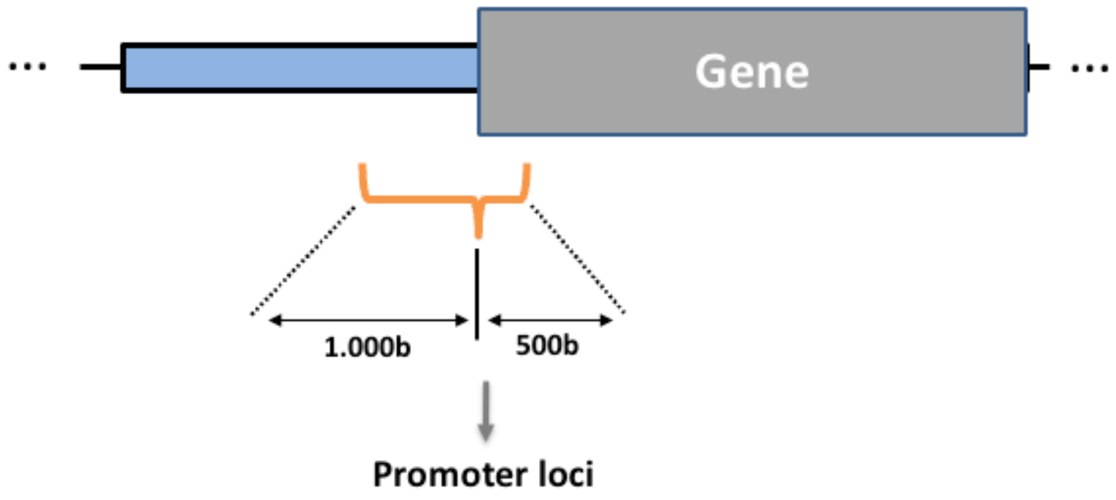

Figure 22: Promoter loci. (Figure has been created for the needs of the current thesis).

This study focuses on promoter DNA regions, with a big portion dedicated to TSS identification from CAGE datasets (and thereby promoter loci identification).

1.8.2. Algorithms for transcription start site identification from CAGE data

During the past decade, several *in silico* methods have been developed for the analysis of CAGE sequencing data sets. These methods (RECLU [75], PARACLU [76], CAGEr [77]) aim to extract peaks from the CAGE signal and quantify the abundance of RNA transcripts.

All algorithms aim to remove any bioproducts and false positive peaks from CAGE signal while preserving true gene TSS. Some of these implementations include ML techniques for the accurate identification of TSS with ADAPT-CAGE [78] exhibiting the highest accuracy amongst them.

1.9. Genetic variations in DNA

Variants are alterations in the nucleic acid sequence of the genome of a living organism. Genetic variants occur frequently in all living organisms. Some variants are beneficial, some neutral, and some are harmful. Variants can be classified in two categories based on cell type.

Germline variants [79] are DNA alterations in gamete cells and can be passed from parent to offspring. On the other hand, somatic variants [80] are present only in specific cells that are not part of the germline and thus are not inherited. Somatic variants are acquired during life progression, usually due to environmental factors or due to errors in the cell division process. Some somatic variants may lead to the development of diseases such as cancer (Figure 23). Both somatic and germline variants have been reported to play a key role in many cancers.

There are many different variant types that may occur in the DNA and can be classified in two categories, small-scale and large-scale variants. Small-scale variants involve a single gene. These mutations alter one or a few nucleotide bases of the DNA. Large-scale variants involve changes in the structure of one or more chromosomes and many genes may be affected at once.

1.9.1. Point variants

A point variant can occur in a genome when a single base pair is added, deleted, or changed. While most point mutations are not harmful, they might have functional impact, including changes in gene expression or alterations in final encoded protein. Point variants may involve either substitution, insertion, deletion, and inversion changes to the DNA.

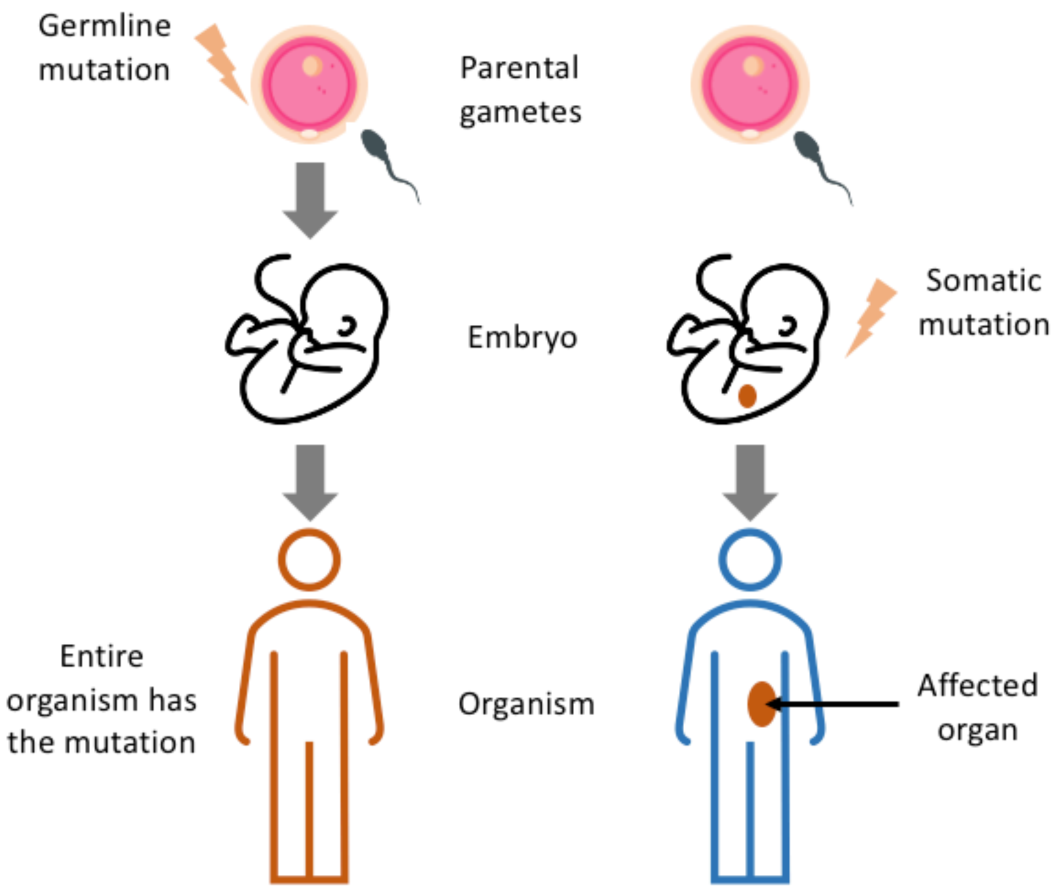

Figure 23: Germline vs somatic mutations. (Figure has been created for the needs of the current thesis).

1.9.2. Types of point variants

Substitution variant occurs when one or more bases in the sequence is replaced by the same number of bases. Insertion when a base is added to the sequence and deletion when a base is deleted from the sequence. Insertion and deletions are usually called indels. Also, inversion variants occur when a segment of the DNA is reversed (Figure 24).

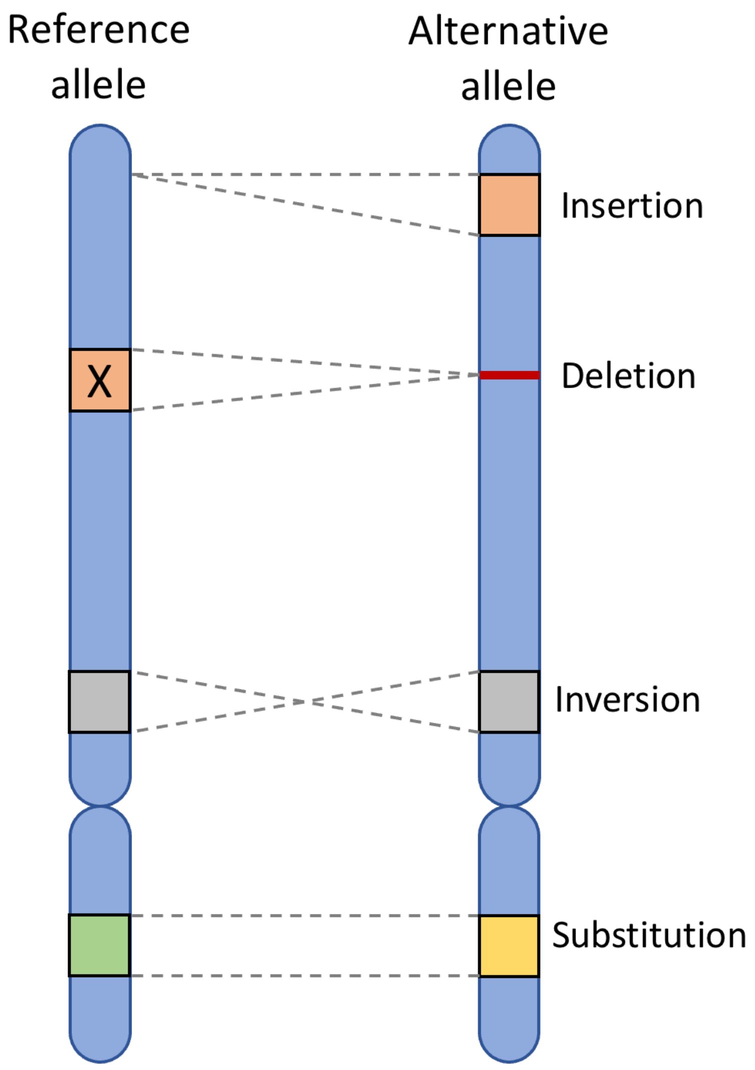

Figure 24: Types of point mutations. (Figure has been created for the needs of the current thesis).

1.9.3. Effects of point variants

Point variants are classified in silent, missense, nonsense, and frameshift categories based on their effect. Silent variants occur when the DNA change does not alter the amino acid sequence of the polypeptide. Missense variants occur when the DNA change alters a single amino acid in the produced polypeptide chain and nonsense when the DNA change creates a premature end codon that intervenes in the polypeptide which remains incomplete. Finally, frameshift mutations occur when the addition or removal of a base alters the reading frame of the gene.

1.9.4. Effects of large-scale variants

Large-scale variants include Copy Number Variation (CNV) which is a type of variant where large portions of the DNA are inserted, repeated, or completely lost. These DNA regions vary between 10Kb and 5000Kb long. Other large-scale variants include duplication of genes when there is an increase in the number of copies for a gene, deletion of large chromosome regions, loss of one copy of a gene where previously two copies were present or loss of both copies of the same gene and movement of DNA sections to another location.

1.9.5. SNPs

Single-nucleotide polymorphisms (SNPs) are substitutions of a single nucleotide at a specific position within the genome. SNPs are the most common type of genetic variation, and a variant is called an SNP if it occurs with a frequency above one percent (>1%) in a population (Figure 25). There are more than 600 million known SNPs across populations around the world.

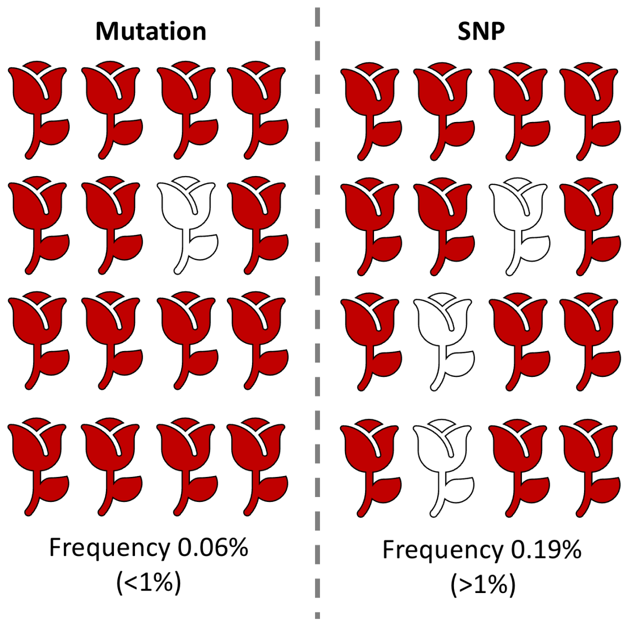

Figure 25: Demonstration of SNPs vs mutations in a population of roses. Assuming there is a variant that turns red roses to white, depending on the frequency the variant is called SNP or Mutation. (Figure has been created for the needs of the current thesis)

Most SNPs have no effect on an organism’s health. However, some of these genetic variations have proven to be very important in the study of health. SNPs may help predict an individual’s response to a certain drug, susceptibility to an environmental factor, and risk of developing diseases. SNPs can also be used to track the inheritance of disease-associated genetic variations within ancestors. Hence, a particular SNP may not cause a disorder, although some of them have been associated with certain diseases. Such type of analysis for discovering SNPs correlated with disorders or diseases, is called genome wide association study (GWAS).

1.9.6. Genome wide association studies

GWAS are utilized to identify associations between DNA regions and diseases (or traits). This method analyzes the whole genome of a population seeking for SNPs, to identify differences in phenotypes between groups. These studies are searching for hundreds or thousands of SNPs at the same time. These SNPs are present at a higher frequency in cases compared to controls in a particular locus (Figure 26). SNPs found to be associated with a disease can help pinpoint genes or regulatory elements which are more likely to correlate with the disease development.

Therefore, GWAS represent a promising way for studying SNPs across the genome and deciphering complex and common diseases in which high abundance of SNPs contribute to a person’s risk for developing them. GWAS studies have identified SNPs associated with several traits and diseases including heart, Parkinson, Crohn’s and more diseases. Furthermore, SNPs have been correlated with response to certain drugs as well as susceptibility to specific environments.

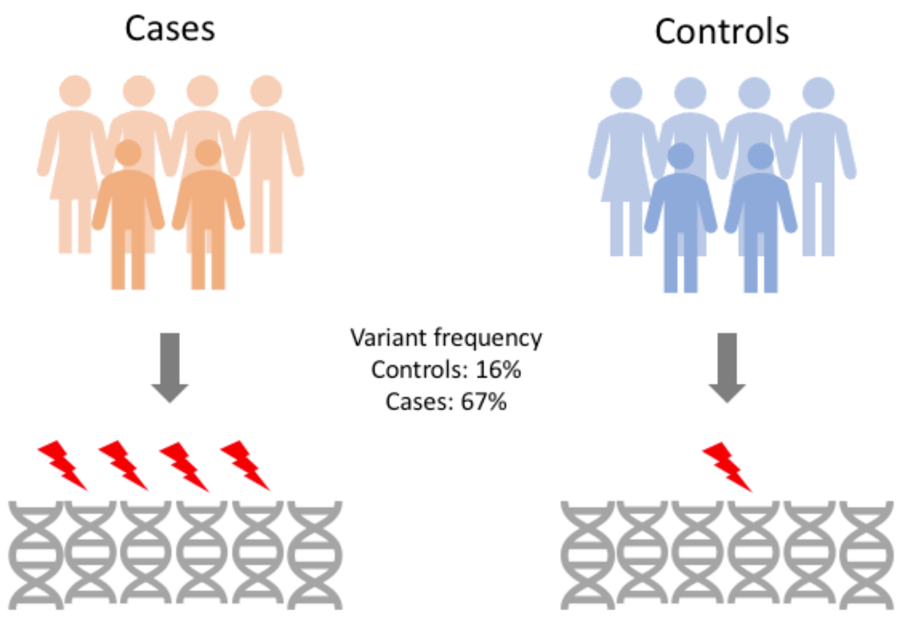

Figure 26: Overview of allele distribution across population. (Figure has been created for the needs of the current thesis)

1.9.7. Mutations

Mutations are genomic variations that occur in less than one percent (<1%) of a population. These types of variants have been associated with several diseases and disorders, such as cancer development.

There are two categories in the context of mutations. Passenger mutations which are defined as those that do not confer cancer phenotypes and driver mutations which describe changes in the DNA sequence that cause cells to become cancerous. Driver mutations are targeted in treatment of cancer patients by drugs that can target specific mutations.

Discovery of driver mutations was mainly focused on protein coding genes [81] but recently analyses were conducted for the non-coding part of the genome [82]. However, passenger mutations have also been proposed and used for analyzing and identifying primary and metastatic cancer patterns [83].

1.9.8. Mutational Signatures

Mutational signatures are characteristic combinations of mutations arising from specific mutagenic processes such as DNA replication infidelity, exogenous and endogenous genotoxin exposures, defective DNA repair pathways and DNA enzymatic editing [84].

A crucial step for analyzing diseases through mutational signature profiles, is to understand the underlying mutational processes [85] (Figure 27). This goal is enhanced by utilizing non-negative matrix factorization (NMF) [86] algorithms where the components of an object (i.e. a mutational signature) are extracted.

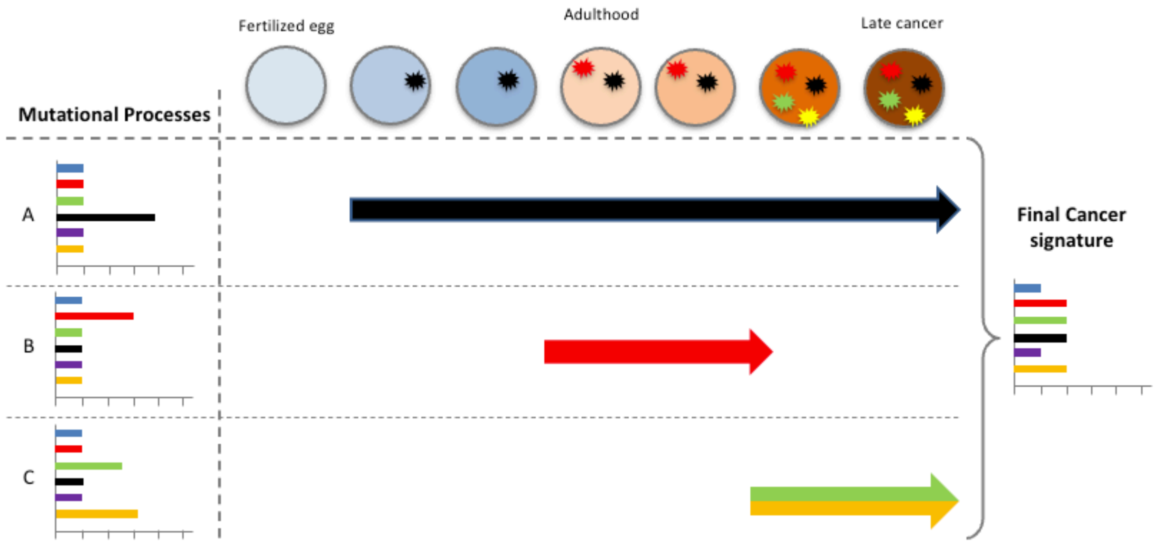

Figure 27: Mutational processes thought the lineage of a cell [87]*. (Figure has been created for the needs of the current thesis)*

Thus, deciphering mutational signatures in various cancers, provides insight for the biological context of the disease, the mechanisms involved in carcinogenesis [88], as well as exposure to environmental factors [89]. Furthermore, mutational signatures profiling has been proven successful for the management and use of targeted therapies in cancer patients contributing to personalized medicine. Hence, these profiles have been proven as a robust way for analyzing and studying carcinogenesis through evolution of life.

Large-scale analysis has revealed many different mutational signature profiles in the context of human cancers and has been greatly enhanced with the public repositories of ICGC and TCGA [90]. Cosmic [91] is one of the largest databases storing mutational signatures that were identified from the analyses of the Pan-Cancer Analysis of Whole Genomes (PCAWG) dataset (Figure 28). Signature profiles can be analyzed based on Single Base Substitution (SBS), Doublet Base Substitution (DBS) and small insertions and deletions (ID). For each of these three categories, signatures have been defined and correlated with a particular mutational process. By calculating the cosine similarity between signatures of interest and cosmic curated signatures, it is possible to study and understand the underlying mechanisms for most cancers.

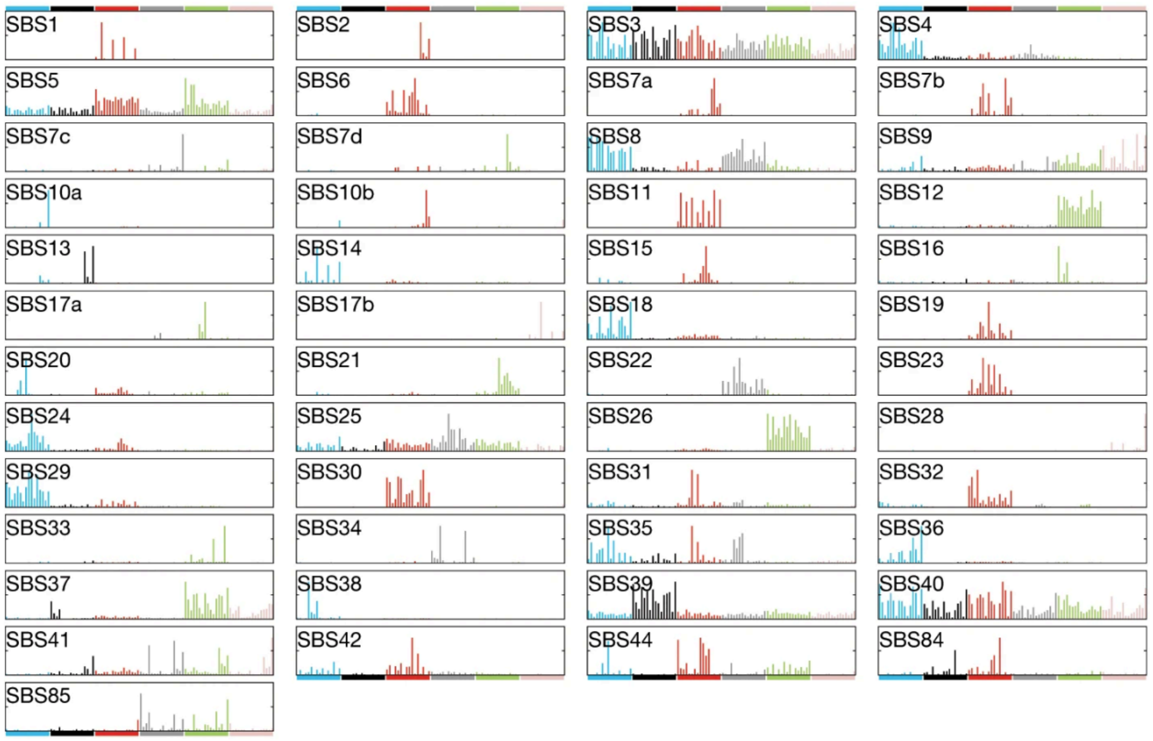

Figure 28: Profiles of SBS signatures [85].

1.9.9. Beneficial mutations

It is well known that mutations that cause changes in DNA sequence are highly possible to be harmful for an organism. On the other hand, there might be occasions that these

DNA variations can be beneficial in certain environments. Certain mutations may enable an organism to withstand harsh environmental conditions better than wild-type organisms. In such cases they will tend to become more common within the population through natural selection.

For example, a mutation allowed humans to express the enzyme lactase immediately after they naturally weaned away from breast milk. Thus, adults are enabled to digest lactose, which is likely one of the most beneficial mutations in human evolution [92].

Another example of mutated organisms that may be benefited are bacteria. They can develop antibiotic resistance when exposed to antibiotic drugs and amongst their population only those with great resistance tend to survive [93]. This phenomenon poses a challenge to modern medicine as infection by these bacteria is harder to be treated.

2. Transcription start site identification of coding and non-coding genes

2.1. Previous implementations

Current chapter analyzes all current implementations for the analysis of CAGE datasets for TSS identification.

2.1.1. ADAPT-CAGE

ADAPT-CAGE [78] is the most recent algorithm published in 2020 and utilizes Ensemble classifiers to distinguish between true TSS sites and noise or other bioproducts. The algorithm can be utilized directly on CAGE reads however, in contrast to existing implementations, it is also based on structural and promoter-associated motif features extracted from the underlying DNA sequence. Also, CAGE reads are filtered based on their mapping quality and 5 prime ends are grouped into peaks based on user provided distance. ADAPT-CAGE features derive from the polymerase II motifs, structural DNA features and expression profiles.

2.1.2. PARACLU

PARACLU [76] an algorithm which was published in 2007, discovers clusters of different density and length for CAGE tags. The algorithm utilizes a density parameter d, and thereafter reports the segments of each chromosome that maximize the following formula:

$$Number\ of\ events\ in\ the\ segment - d \cdot (size\ of\ the\ segment\ in\ nucleotides)$$

Therefore, distance "d" favors loci with high tag abundance while simultaneously penalizes longer regions.

2.1.3. RECLU

RECLU [75] was published in 2004 and comprises an expansion of the PARACLU algorithm. RECLU can utilize biological replicates as well as different and more vigorous filtering of the results.

2.1.4. CAGEr

CAGEr [77] was the current implementation before ADAPT-CAGE and was published in 2015. CAGEr filters quality of the input data and removes the "G" nucleotide from the 5' prime ends when necessary. All tags are then grouped into clusters. CAGEr can also supply results about differentiated TSSs between different samples and identify TSS shifting events by analyzing several CAGE experiments.

2.1.5. TSRchitect

TSRchitect [94] which is a more versatile implementation, is able to utilize data from a spectrum of transcriptome profiling techniques including CAGE, PEAT, RAMPAGE, and STRIPE-Seq, while the algorithms is able to also handle experiments with replicates. Furthermore, TSRchitect can perform directly on bam file types.

2.2. DiS-TSS

Despite the widespread usage of CAGE-seq data, there is increasing evidence in the literature that unveil alarming levels of noise embedded in CAGE datasets, raising issues related to the specificity of CAGE as a TSS identification protocol [75], [77], [95]. According to these studies CAGE can identify capping events involving diverse locations of transcribed loci such as different splicing products that may be classified as transcriptional noise.

To this end DIANA Signal-TSS (DiS-TSS) [13] was implemented, an *in silico* TSS identification approach that combines CAGE data with digital signal processing (DSP) derived features and ML to provide highly accurate and single-nucleotide resolution predictions (Figure 29). CAGE reads are represented in the time series domain as signal vectors, after basic pre-processing takes place regarding mapping quality and tag-cluster aggregation. DiS-TSS directly extracts signal features inspired by the DSP field [96] that are related to the signal's structural properties which can accurately distinguish real transcription initiation events from biological and/or protocol-induced noise (Figure 30). Subsequently these features are forwarded into an SVM model that has been trained on CAGE signal from annotated TSSs and non-promoter regions.

2.2.1. Overview of DiS-TSS

DiS-TSS algorithm can operate with two different modes (Figure 29). Regarding the first mode of operation users can provide input of CAGE alignments in bam format and the algorithm will score all candidate TSSs. In the second mode, users can provide BED formatted genomic loci along with the bam alignments. The algorithm will analyze the bed file and analysis will focus on the selected loci instead of the whole genome. Initially, CAGE tags with low mapping quality (less than 10) are removed and the remaining tags are aggregated into clusters using a distance parameter cutoff (default value = 25). Peaks are further filtered based on the normalized (tags per million - tpm) expression level threshold (default = 1). For each peak, DiS-TSS identifies the position with the highest 5' end tag coverage and flags it as the peak's representative for future analysis. The following step involves DSP-derived feature extraction which captures the characteristics of each peak's signal. Subsequently, the features are vectorized and combined to an SVM model which has been trained to distinguish between CAGE signal from real transcription initiation events and technical or biological noise.

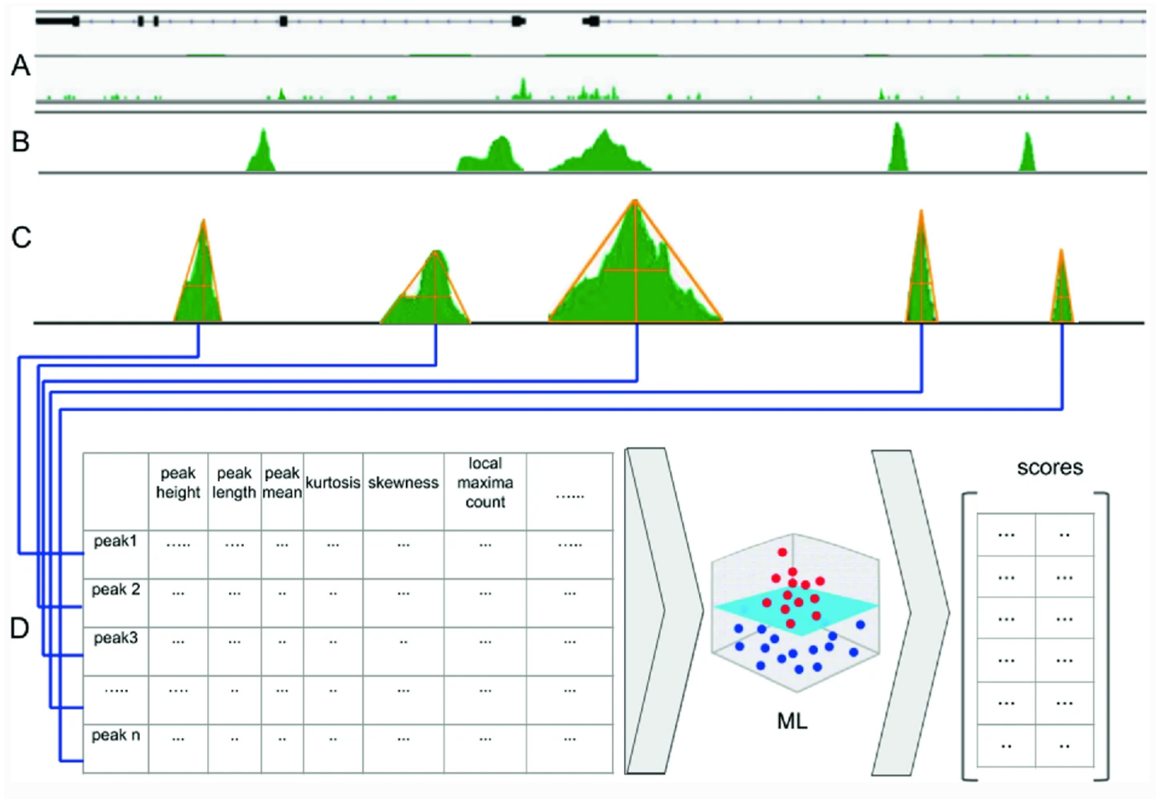

Figure 29: DiS-TSS pipeline overview. (A) Genome browser view of raw CAGE signal. (B) Initial step that summarizes the aggregation of CAGE tags into clusters/peaks. (C) Extraction of features related to peak properties. (D) Assembly of feature vectors prior to using the DiS-TSS ML model that produces the final score for each peak.

2.2.2. Feature Analysis and Selection

The thirteen (13) most popular features which were inspired from the DSP field for analyzing and unveiling embedded patterns in digital signals were extracted. These features are tightly related to the signal's structural properties [97] and by conducting exploratory analysis revealed that they can distinguish structural differences between TSS-associated CAGE signal and transcriptional noise (Figure 30). The list of features includes kurtosis, height, width, prominence, peak area, skewness, peak length, local maxima mean, variance of local maxima and count of local maximum peaks. Additionally, after transferring the signal from space to frequency domain by utilizing fast Fourier transform (FFT), mean and variance of the frequency amplitudes are calculated as well as the frequency with higher energy inside the power spectrum (which is known as dominant frequency) [98].

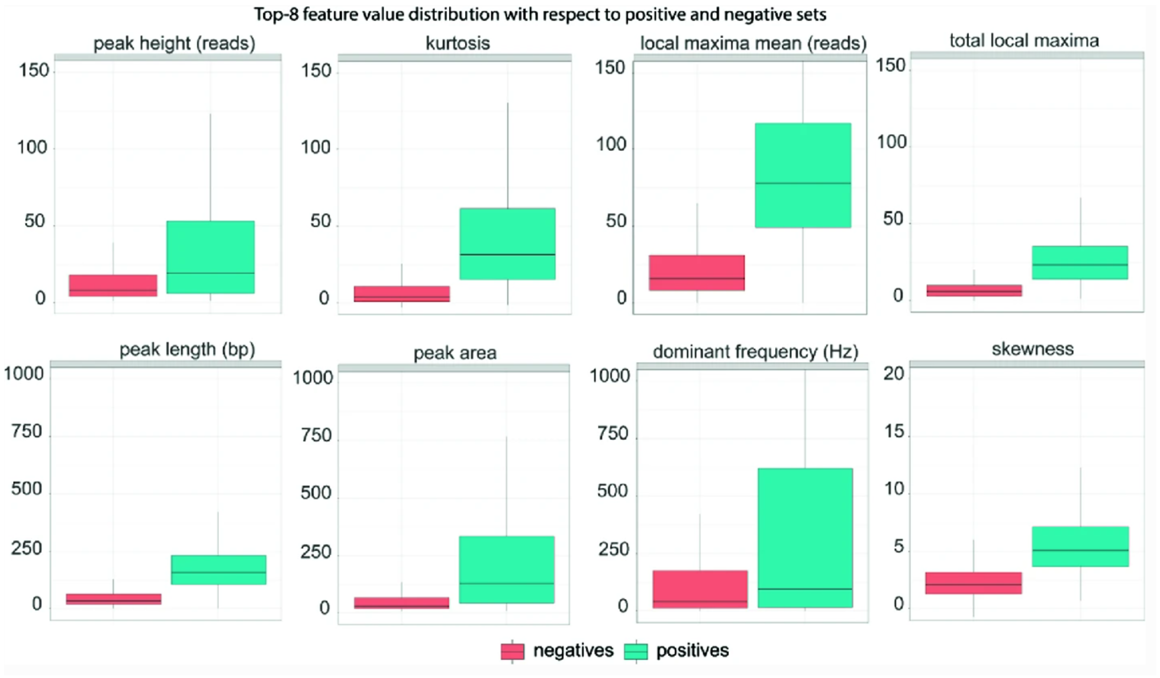

Figure 30: Distribution of the top-8 feature values in the positive and negative sets.

To identify each feature importance and keep only those having high predictive power, the Recursive Feature Elimination (RFE) [99] method was utilized. RFE fits a model using different combinations of features and eliminates the weak ones until a certain performance limit is reached. In this study, RFE was conducted by utilizing a radial basis function (RBF) SVM model for scoring different subsets of feature. Therefore, eight (8) total features were selected to contribute as the best performing set which includes skewness, peak length, peak height, peak area, dominant frequency, mean and count of local maxima as well as kurtosis (Figure 30). All remaining features contribution to the algorithm's performance was having a negative efect and were therefore excluded from any further analysis. Features that do not provide better model performance are usually removed prior to the training procedure to achieve greater generalization of the model and reduce the overall computational complexity.

2.2.3. Training of DiS-TSS

CAGE peaks (N = 38,439) were extracted from H9 cell line samples, as described in chapter 2.2.1 (Overview of DiS-TSS) section and formed the basis for training the algorithm (Figure

31). CAGE peaks that were localized closer than 1 kb from annotated protein-coding TSSs and overlapped H3K4me3 and Polymerase II ChIP-Seq enriched regions, formed the positive set (N = 11,304). To avoid the inclusion of promoter-proximal genomic loci that are rich in functional information (i.e. promoters), CAGE peaks that were found in regions flanking promoters (up to 9 kb in each direction from the 1 kb window mentioned above) were removed entirely from the dataset. Intergenic peaks that did not overlap any H3K4me3 and Polymerase II ChIP-Seq enriched regions were selected as the negative set (N = 11,579). The same pipeline was used for generating a benchmarking set of 75,127 CAGE peaks (32,310 positives and 42,817 negatives) in K562 cells which was used to compare the performance of DiS-TSS with existing algorithms outside the biological context of DiS-TSS training. The importance of utilizing an evaluation dataset from a cell line (K562) with an entirely different expression profile than the samples used for training is paramount for exploring the possibility of overfitting the model on putative cell-specific structural properties of CAGE signal or sequencing batch effects.

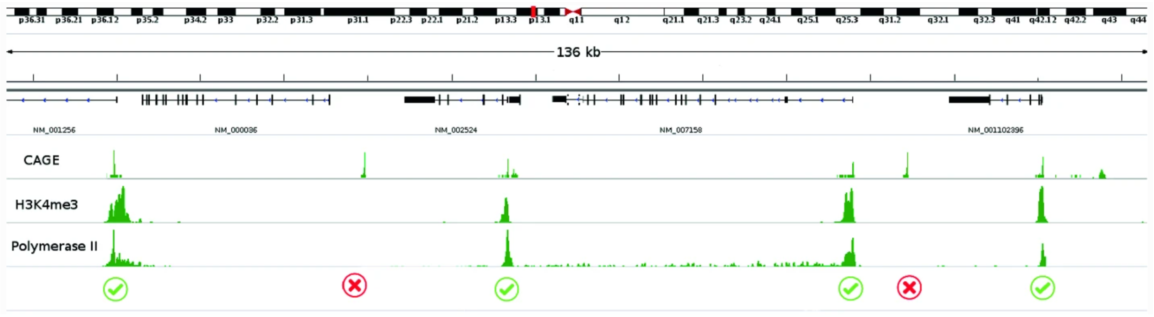

Figure 31: Genome browser view of an example genomic locus depicting the process of assigning CAGE peaks to the positive or negative sets. The distinction between the two sets depends on the co-occupancy of H3K4me3 and Polymerase II ChIP-Seq peaks. CAGE peaks occupied by both active transcription signals are assigned to the positive set. Peaks that do not exhibit coverage by any of the two signals were assigned to the negative set, while peaks that were occupied either by H3K4me3 or Polymerase II were not considered for training or for the labeled data evaluation.

2.2.4. Evaluation of DiS-TSS

Two distinct types of evaluation strategies were utilized for benchmarking the performance of CAGE TSS predictors, including DiS-TSS, CAGER, RECLU and TSRchitect. The first strategy is mainly experimentally driven and utilizes Polymerase II, H3K4me3 and transcription factor ChIP-Seq occupancy. The second evaluation approach is based only on gene annotations that were used to segment the genome into positive and negative zones in terms of promoter functionality. Both evaluation processes were applied on K562 cell line datasets to further explore the generalization ability of our algorithm and provide a benchmarking environment outside the cell type context of DiS-TSS training set. In Table 1 all evaluation results are summarized.

Table 1: Summarized evaluation results in K562 cells, based on experimental data and annotated protein coding TSSs. From left to right, each column shows the number of predictions overlapping with at least one TFBS, H3K4me3 and Polymerase II peaks, the true positive and false positive zones (as defined by annotated protein coding TSSs). DiS-TSS results are based on a 0.5 score cutoff.

Algorithm	**Total positive predictions**	**Number of predictions overlapping with:**			
		TFBS (%)	**H3K4me3 (%)**	**Polymerase II (%)**	**TP zone (%)**
DiS-TSS	10,937	10,125 (92.57)	9,619 (87.94)	9,574 (87.53)	8,915 (81.51)
CAGER	14,465	12,681 (87.66)	11,732 (81.1)	12,076 (83.48)	10,819 (74.79)
RECLU	12,282	10,646 (86.67)	9,958 (81.07)	9,763 (79.49)	9,420 (76.69)
TSRchitect	7,281	4,705 (64.62)	4,161 (57.14)	4,222 (57.98)	4,128 (56.69)

Experimentally Driven Evaluation

For the purposes of the evaluation process, ChIP-Seq datasets were utilized based on known marks of active transcription such as H3K4me3 [100]–[102] and Polymerase II in H9 and K562 cell lines. After applying a score cutoff of 0.5 for DiS-TSS, the overlap of positive predictions with H3K4me3 and Polymerase II ChIP-Seq peaks was calculated. The same overlap was also calculated for CAGER, RECLU, and TSRchitect predictions. 87.95% (9,616 out of 10,937) of positive DiS-TSS predictions were found enriched in H3K4me3 (Figure 32A) and 87.53% (9,574) in Polymerase II ChIP-Seq signal (Figure 32B), 81.1% and 83.48% of positive CAGER predictions, 81.07% and 79.49% for RECLU and 57.14% and 57.98% for TSRchitect respectively.

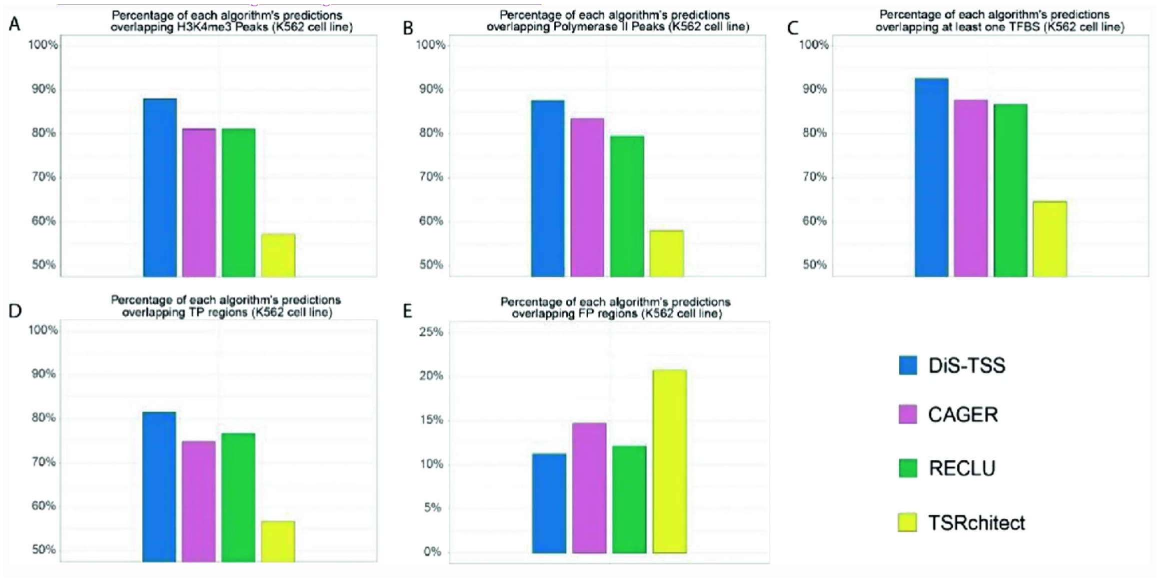

Figure 32: Experimental evaluation of algorithms based on annotated protein-coding TSSs and H3K4me3, Polymerase II and TF ChIP-Seq peaks in K562 cells. Percentage of each algorithm's predictions overlapping (A) H3K4me3 and (B) Polymerase II enriched regions as well as (C) at least one TFBS. (D) Percentage of predictions that were found in promoters (±1 kb around annotated TSSs, true positive regions) and (E) outside of promoters (flanking true positive regions and up to ±50 kb from the TSSs, false positive regions).

An additional approach for the evaluation was also utilized, based on the cross-tissue compendium of transcription factor ChIP-Seq binding sites provided by the ENCODE consortium [103] (Txn Factor track). The abundance of TFBSs overlapping with the positive predictions set provided by each algorithm in K562 cells was calculated (Figure 32C). A total of 92.57% (10,125 out of 10,937) of DiS-TSS predictions were found to overlap with at least one TFBS, while 87.66%, 86.67% and 64.16% were the results for CAGER, RECLU and TSRchitect respectively.

Evaluation Based on Protein-Coding Transcript Annotation

To further explore the performance limits of CAGE-oriented TSS predictors, an evaluation approach exclusively based on annotated protein coding transcripts was utilized. The region surrounding each transcript start (±1000b) was considered as the positive zone and the flanking region, up to 50000b in both directions, as the negative zone. For every algorithm, positive predictions within the positive zone were considered true positives (TP) while those found in the negative zone were flagged as false positives (FP). DiS-TSS achieved the highest performance (Figure 32D, E) with 81.51% TP (8,915 out of 10,937), followed by RECLU with 76.69% (9,420 out of 12,282), CAGER with 74.79% (7,479 out of 14,465) and TSRchitect with 56.69% (4,128 out of 7,281).

2.2.5. Discussion

The above presented algorithm attempts to overcome the limitations and provide a breakthrough in CAGE- mediated TSS identification. DiS-TSS is an annotation-agnostic and the first algorithm to combine digital signal processing, on CAGE datasets for extracting peak shape related features, and Machine Learning for effectively distinguishing real TSS-related CAGE enriched regions from biological and technical induced noise. Hence, through this study, the field of digital signal processing will be introduced to the transcriptomics community which can effectively be combined with Machine Learning to provide answers to complex and diverse biological questions.

The evaluation process of algorithms with different types of functionalities is far from straightforward and requires careful planning. Thus, the current 2-fold strategy involved two major points. At first an evaluation based on the enrichment of the algorithms' predictions with H3K4me3, Polymerase II and TFBS ChIP-Seq signal separately, and second, a benchmarking approach solely based on annotated protein-coding transcripts and their promoters. The advantage of the previous evaluation method is that for each comparison there are several advantages and disadvantages as they can complement one another. They also illustrate a spherical view of the algorithms' performance and provide knowledge for critical components of their applicability and possible functional limitations. The algorithm was trained, and the performance was evaluated in two different cell types with basically different gene expression profiles, to test the robustness against overfitting on cell-specific signal structural properties or sequencing batch effects. H9 cells are one of the most widely distributed and well researched embryonic stem cell lines in contrast with K562 cells that are extensively utilized as a myelogenous leukemia model. The gene expression diversity was the main reason for selecting these two cell types, in addition to their widespread and popularity usage in Consortia such as the ENCODE [103] and FANTOM [104]. The availability of CAGE, H3K4me3, Polymerase II and TFBS ChIP-Seq data for both cell types, also played a decisive role for actively selecting them for this study.

The contribution of this study gravitates around the novel application of digital signal processing techniques applied on transcriptomic data and the extraction of feature types that when combined with a robust ML model can achieve high performance in the problem of CAGE-oriented TSS identification.

2.3. DeepTSS

Another robust highly sensitive approach for approaching the TSS identification problem is introduced in this chapter. By combining the knowledge gained from the previous implementation (DiS-TSS), a novel computational method for processing CAGE-oriented samples, that combines genomic signal processing (GSP), structural DNA features, evolutionary conservation evidence and raw DNA sequence with Deep Learning to provide single-nucleotide TSS predictions with unprecedented levels of performance is presented.

In this chapter DeepTSS [105], which is a fully updated extension of the previous method (DiS-TSS) [13] for distinguishing between TSS-associated CAGE signal and biological or technical noise, will be thoroughly analyzed. DeepTSS is a computational framework for single-nucleotide resolution and accurate TSS identification that combines GSP, sequence and evolutionary conservation inputs, CAGE data and DL (Figure 33C). The algorithm handles basic pre-processing of the inputs and works directly with aligned tags overlapping the CAGE peaks. Hence, tags are transformed to signal vectors in the time domain and GSP-inspired features are calculated. Additionally, the DNA sequence corresponding to CAGE signal peaks is one-hot encoded and structural features are extracted along with the evolutionary conservation score which is calculated by phyloP [106]. Furthermore, each feature domain is forwarded as input to a separate convolutional layer, which is a branching scheme, already proved to work successfully and utilized in the context of precursor micro RNA prediction [107] . All outputs of the layers are concatenated and directly forwarded to the densely connected part of the ML framework. A multifaceted benchmarking strategy was utilized based on annotated experimental data and genomic loci, DeepTSS was found to outperform previously published implementations for distinguishing real transcription initiation events from biological or protocol-induced noise.

2.3.1. Annotation and experimental data

Pre-aligned CAGE datasets in bam format (GRCh38 assembly), from H9 and K562 cells with sample codes CNhs11917, CNhs12334 respectively and the corresponding collapsed CAGE tags (5' end) contained in ctss files were downloaded from the FANTOM repository [104]. DeepTSS and ADAPT-CAGE can be applied directly on bed or bam files with pre-calculated representatives of the CAGE peak and iTiSS can only perform on bam file types. All other implementations, PARACLU, RECLU and CAGER can only be utilized on the ctss files.

Regarding ChIP-Seq datasets for Polymerase II and H3K4me3 with sample codes ENCFF281VBW, ENCFF773FKD, ENCFF757WPX and ENCFF261REY were retrieved from the ENCODE repository [103] in bed narrow peak file format and the UCSC liftover software was used to migrate them from GRCh37 to GRCh38 assembly coordinates.

Also, genomic locations of TFBS from 161 TFs regarding 91 cell types were downloaded from the ENCODE 'Txn Factor' track in UCSC. The protein-coding gene annotation was downloaded from Ensembl v98 [108]. Furthermore, 100-way per-nucleotide phyloP evolutionary conservation score was retrieved from UCSC.

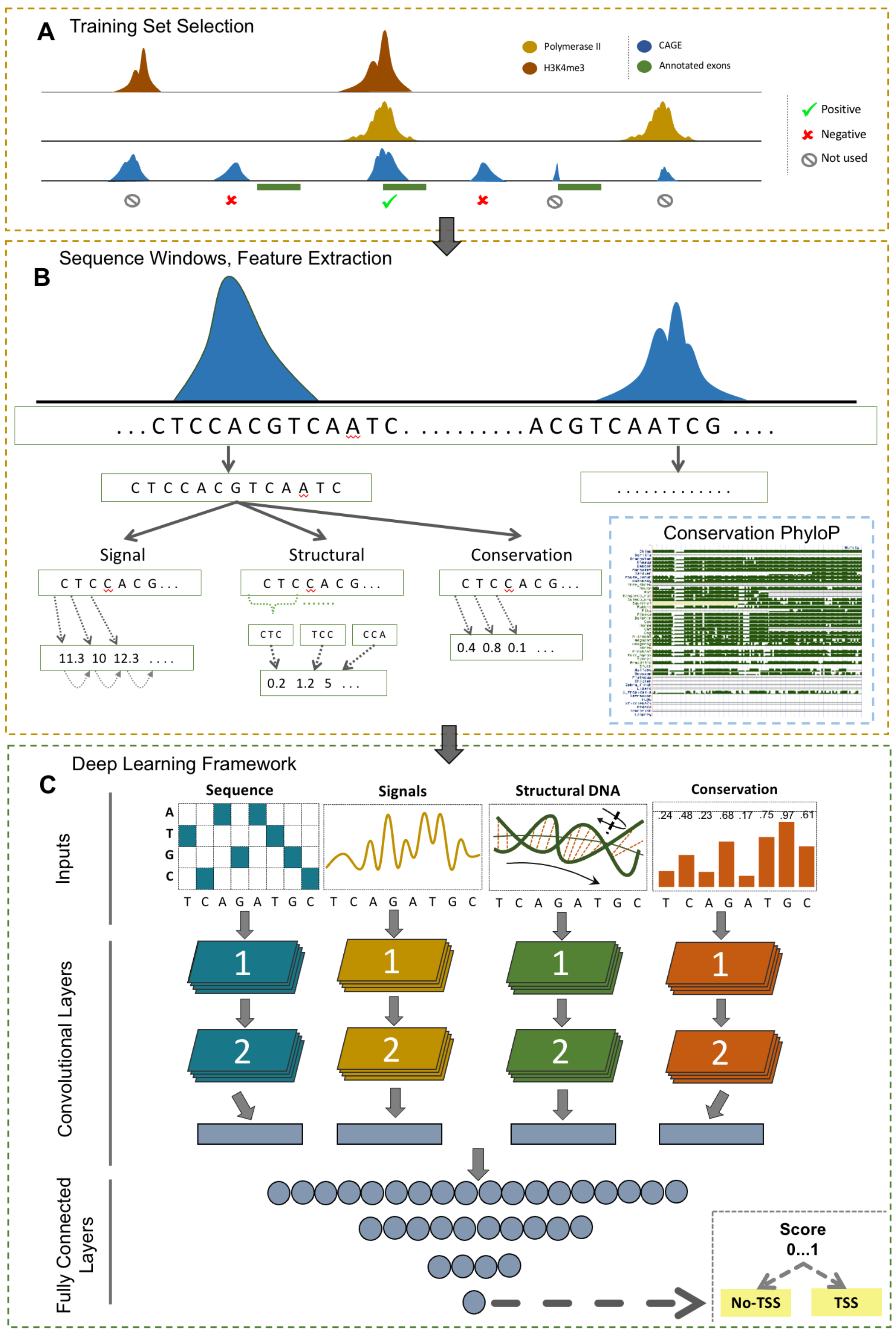

Figure 33: Overview of training set selection, feature extraction and DeepTSS DL architecture. A) Synopsis of the process for labeling H9 CAGE peaks as positive or negative samples. Peaks exhibiting an overlap with annotated protein-coding

gene TSSs as well as both H3K4me3 and Polymerase II enriched loci were labeled as positives while those that overlapped with either of the two marks but not with annotated TSSs were removed from any subsequent analysis. CAGE peaks that did not overlap with any of the two marks and annotated TSSs were marked as negatives. B) For each peak representative (position with highest amount of overlapping 5' end of tags) we extracted the centered underlying sequence (600bp) and proceeded to extract four distinct feature categories. The one-hot encoded version of the sequence, the GSP-inspired and structural DNA sequence-based features as well as the per nucleotide evolutionary conservation evidence. C) DL architecture of DeepTSS, specifically designed for exploiting each individual feature type. The architecture consists of 4 distinct convolutional branches for processing the different feature types. The first branch operates on the one-hot encoded version of the input DNA sequence, the second and third on GSP and structural DNA features respectively, and the fourth on the evolutionary conservation evidence. All branches are designed with 2 consecutive convolutional layers and their output is concatenated prior to the application of the fully connected part of the network. The final output is based on a sigmoid activation function.

2.3.2. Overview of DeepTSS

DeepTSS framework can be utilized directly with pre-aligned CAGE reads in bam file format. All CAGE tags that don't meet a quality threshold mapping provided by the user (default = 10) are excluded from further analysis. All remaining reads are grouped into peaks based on a user-defined distance parameter (default = 50bp). Subsequently, the expression level (normalized in tags per million - tpm) of each CAGE peak is calculated and those below a user-defined cutoff (default = 1) are excluded. For each peak, the position with the greatest number of overlapping 5' tag ends is spotted and selected as representative of the peak. On the other hand, users can provide their own CAGE peak representatives in bed format, and they are forwarded directly as the input of DeepTSS framework. DeepTSS performs all required feature extraction and directly proceeds on utilizing the DL model for scoring each observation.

2.3.3. Network architecture

One-hot encoded DNA sequence surrounding the CAGE peak representatives is utilized within the DeepTSS framework, GSP and structural DNA features are extracted directly from the underlying sequence, and evolutionary conservation is extracted as calculated by phyloP. The two branches of GSP and structural DNA features, standardization of features took place separately. Regarding the conservation score no processing was utilized (for regions not having a score, DeepTSS assigns zeros).

For each input type category, a separate convolutional branch has been created, which is composed of two convolutional layers (Figure 33C). The calculations from all four branches are concatenated and forwarded to the final fully connected layer of the framework. All branches operate on a 600 nucleotide bases window size. A multitude of models were trained and tested with many different kernel sizes, filter numbers and nodes for the dense layers and the final combination which reached the best performance on the test set was selected. The one-hot encoded DNA sequence, structural and GSP feature branches were constructed with 16/12 kernel sizes and 20/10 filters in the convolutional layers. The evolutionary conservation branch is consisted of 32 filters for the first convolution layer and 16 regarding the second layer with kernel sizes 16 and 8 respectively. Range of values for the number of filters in both layers was 20, 30 and 40. The range for the kernel length was 20/15, 16/12, and 12/8. The fully connected part is comprised of three layers with 120/60/25 nodes, while the range of tested values was 140/80/40, 120/60/25, and 100/60/20. Leaky ReLU [109] which was adopted as the activation function regarding all layers was followed by batch normalization. The sigmoid function has been utilized for the final layer of the architecture which calculates the algorithm’s output.

All architectures were trained with binary cross entropy loss function, for 60 epochs having enabled an early stopper parameter which was adjusted on ten epochs patience. Most of the hyper-parameters were tuned by training many different models of the framework for more than 60 epochs exhausting out all possible combinations, which concluding in batch size fixed at 256, 0.001 learning rate with ‘Adam’ optimizer and dropout rate of 0.2 (Supplementary Table 8). Furthermore, models were tested with 64, 128, 256 and 512 batch sizes, 0.01, 0.001, 0.0005 and 0.0001 learning rates and 0.2, 0.25, 0.3, 0.35 dropout rates. A grid search has been utilized for the hyper-parameters tuning approach which learns each configuration of the hyper-parameters and indicates the best performing combination.

2.3.4. Feature extraction

Various GSP-inspired features were specifically selected as approximations of the DNA physicochemical properties in the form of distinct time series. Z-Curve [110] which comprises three different signal vectors, each providing a unique representation of the DNA sequence. The three components Xn, Yn and Zn correspond to an irrespective nucleotide distribution where Xn describes the distribution of purines/pyrimidines, Yn the functional group of the bases (amino or keto) and Zn the strength of the hydrogen bonds between base pairs (strong H-bond or weak H-bond). DNA-walk representation [111] describes a graph where a step upwards is taken if the current nucleotide is a pyrimidine and vice versa for a purine. In contrast to Z-curve, DNA walk does not consider the previous nucleotide. Paired numeric [112] incorporates the complementarity DNA sequences. Tetrahedron representation [113] is a fixed mapping method where all four nucleotides are considered as the four vertices of a regular tetrahedron. DeepTSS also utilizes structural DNA features associated with promoter regions such as bendability and propeller twist [114].

All spatial signals are calculated based on a 600bp window around the representative CAGE peak. The window size has been decided based on the performance of multiple models trained with different values, and its application was achieved with a stride of 1bp. Z-Curve and tetrahedron representations are multidimensional signals composed of three vectors while DNA walk and paired numeric are one-dimensional vectors. These signal features can describe a genomic sequence, identify hidden periodicities and nucleotide distributions that cannot be revealed with conventional methods.

Structural features were calculated by using a sliding window and converting each 600bp sequence to overlapping 3-mers or 2-mers. For bendability, the input sequences were fragmented into overlapping 3-mers (1bp stride), and for each 3-mer we assigned a score that was derived from earlier biochemical studies [115]. The dimensionality of the resulting vector was 598. The same strategy was applied for propeller twist [116], [117], with the only difference being that this feature is based on 2-mers. Therefore, the resulting vector had 599 values.

These features effectively transform the input DNA sequences into time series. In addition, each input sequence is also transformed into its one-hot encoded version and the phyloP-derived numerical representation of its evolutionary conservation. Regardless of the feature type, the application of the neural network convolutional process ensures that local patterns and more abstract combinations of them across different feature types that maximize the DL model's capacity for distinguishing between positive and negative CAGE peaks will be identified.

2.3.5. Training of DeepTSS

CAGE peaks and their representatives (N=38,439) were extracted from the H9 sample, as described in chapter 2.3.2 (Overview of DeepTSS), and used for training our model (Figure 33B). Peaks that overlapped H3K4me3 and Polymerase II ChIP-Seq enriched loci and were positioned within 1kb from annotated protein-coding TSSs, comprised the positive set (N=11,304). Intergenic CAGE peaks not overlapping with either H3K4me3 or Polymerase II bound regions formed the negative set (N=11,579). Promoter-proximal CAGE peaks that were localized in regions flanking annotated promoters (9kb in each direction and outside of the previously mentioned 1kb window) were removed entirely from all analyses, to avoid the putative inclusion of functionally rich information in the negative set. A benchmarking set of 75,127 CAGE peaks (32,310 positives and 42,817 negatives) from the K562 sample was generated and used to query the generalization capacity of DeepTSS on data from a biological context that was not included in the training process, and to compare its performance with previously existing implementations.

Chromosomes 15 and 14 were completely left out of the training process. The former was used for testing the models during the optimal hyperparameters search and the latter as an evaluation set during training. For all loci in the training and validation sets, a sliding window of 600bp (the window based on which the input features are calculated) was used. Initially, the window was placed 100bp upstream of the CAGE peak representative position and then moved to its final position 100bp downstream, with a 25bp stride. Therefore, for each sample in the initial set of CAGE peak representatives, we generated 7 additional samples where the input DNA sequence, based on which all input features are calculated, is not centered on the representative. With this approach, we augmented the input set by generating more samples and forced our model to remain unaffected by any potential biases regarding the position of the CAGE peak representatives within the input sequences [107].

2.3.6. Evaluation of DeepTSS

Regardless of their common objective, the evaluation process of algorithms with diverse feature extraction processes and mathematical modeling is far from trivial. For comparing DeepTSS, ADAPT-CAGE, TOMETOOLS, CAGER, RECLU, PARACLU and iTiSS processes were specifically designed to calculate an unbiased estimate of each algorithm's performance. The first benchmark is based only on protein-coding gene annotations that were used to divide the genome into promoter (positive) and non-promoter (negative) regions. The second benchmark consists of purely experimental data including H3K4me3 and transcription factor ChIP-Seq enriched regions as well as a segmentation of the human genome into chromatin states as calculated by ChromHMM [118], downloaded from the Roadmap Epigenomics Project.

DeepTSS was trained on CAGE samples from H9 cells, and all benchmarks were based on K562 datasets to query the algorithm's ability to generalize on unseen data.

Evaluation based on protein-coding gene annotation

The evaluation of TSS predictors was based on a benchmark created by annotated protein-coding genes. For each TSS the proximal region (+/- 500b) was labeled as the positive zone, and the flanking region (+/- 50.000b) as the negative zone (Supplementary

Figure 50). Positive predictions overlapping the positive zones were assigned as true positives (TP) and the rest were considered false positives (FP). Any prediction falling within a negative zone and exhibiting an overlap with H3K4me3 ChIP-Seq peaks was flagged as a TP instead of FP.

Positive zone is defined as a +/−500b window centered on the annotated TSS as it has been reported to be occupied by TSSs across different tissues (e.g alternative TSSs) [119] or under different conditions [120].

Hence, to observe the performance based on a different point of view, instead of evaluating each CAGE peak individually, we considered that a positive prediction is a gene exhibiting at least one overlapping CAGE peak (algorithms' output) with its TSS vicinity (+/− 500bp).

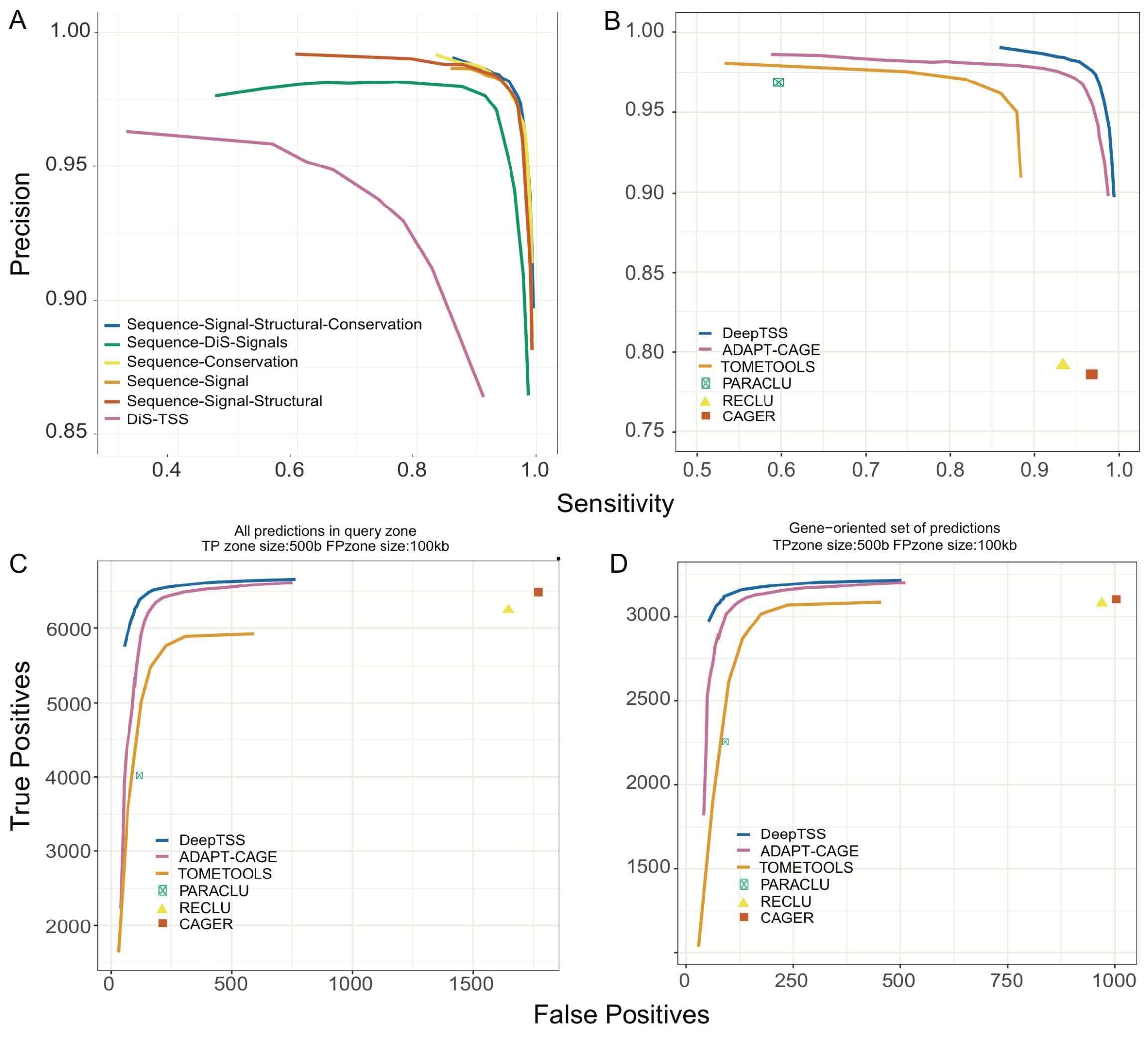

Figure 34: Annotation-based evaluation in K562 cells, based on known protein-coding TSSs. For all algorithms, we applied multiple score cutoffs to calculate performance metrics in a wide range of prediction confidence and generate Precision-Sensitivity as well as TP-FP curves. A) Comparison of distinct DeepTSS models trained with different combinations of input feature types and our initial GSP-inspired algorithm, DiS-TSS [26], that used SVM to model the data. The green curve corresponds to a DL model trained on the GSP features used in DiS-TSS and the one-hot encoded version of the raw DNA sequence. B) DeepTSS, ADAPT-CAGE and TOMETOOLS performance as measured with precision and sensitivity. Trade-off between TPs and FPs in the CAGE peak- (C) and gene-oriented (D) evaluation.

For DeepTSS and ADAPT-CAGE, several score cutoffs were applied to explore their performance in the full score range (Figure 34). PARACLU, RECLU, CAGER and iTiSS do not provide a score for every CAGE peak. Instead, their output is the equivalent of DeepTSS and ADAPT-CAGE predictions after applying a score cutoff. For this reason, precision-recall curves were not calculated for PARACLU, RECLU, CAGER and iTiSS, which are denoted as points in the graphs. A comparison of all algorithms on this benchmark is also presented in Table 2. PARACLU, RECLU, CAGER and ADAPT-CAGE were used with default settings, and DeepTSS with a score cutoff of 0.9.

Table 2: Evaluation results in K562 cells, based on annotated protein-coding gene TSSs. From left to right, the number of total positive predictions of each algorithm is shown, the number of TPs and FPs in both CAGE- and gene-oriented benchmarks as well as the performance in terms of precision and sensitivity on default parameters.

Algorithm	**Total positive predictions**	**All predictions in query zone**		**Gene–oriented set of predictions**		**Protein-coding TSS annotation**	
		TP	FP	TP	FP	Precision	Sensitivity
DeepTSS	31,443	6,398	123	3,122	91	0.98	0.96
ADAPT-CAGE	31,177	6,294	172	3,091	125	0.97	0.94
CAGER	14,465	6,489	1,771	3,102	1,003	0.97	0.78
PARACLU	9,453	4,016	129	2,258	95	0.97	0.60
RECLU	11,558	6,257	1649	3,082	970	0.93	0.79
TOMETOOLS	30.689	5,765	228	3,016	174	0.96	0.86
iTiSS	1,734	98	37	118	40	0.72	0.01

Evaluation based on ChromHMM-derived genome annotation and experimental data

Current section presents the comparison results based on two evaluation benchmarks on K562 cells (Supplementary Table 6). The first one is based on ChromHMM, a well-established ML algorithm for genome segmentation on different chromatin states from the analysis of six histone modifications (Figure 35A-B, Supplementary Figure 49A-B). The second is based on purely experimental data related to H3K4me3 and TFBS occupancy (Figure 35C-D, Supplementary Figure 49C-D). Score cutoff of 0.9 was applied to DeepTSS and default settings (Supplementary Table 7) on ADAPT-CAGE, TOMETOOLS, PARACLU, RECLU, CAGE and iTiSS for isolating their positive predictions. A summary of the comparison results of this evaluation process is shown in Table 3.

Table 3: Summarized evaluation results in K562 cells, based on chromatin states, experimental data and annotated protein coding TSSs. From left to right, each column shows the number of total positive predictions of each algorithm, the percentage overlapping with chromatin states associated with active and weak/repressed transcription, and at least one TF and H3K4me3 peak.

Algorithm	**Total positive predictions**	**ChromHMM active transcription**	**ChromHMM weak transcription**	**TF ChIP-Seq peaks**	**H3K4me3 ChIP-Seq peaks**
DeepTSS	31,443	96.66% (30,376)	3.33% (1,047)	98.27% (30,898)	92.04% (28,939)
ADAPT-CAGE	31,177	95.90% (29,885)	4.09% (1,276)	97.72% (30,466)	91.12% (28,409)
CAGER	14,465	84.62% (12,195)	15.37% (2,215)	87.67% (12,681)	81.11% (11,732)
PARACLU	9,453	94.01% (8,868)	5.98% (565)	95.26% (9,005)	92.06% (8,702)
RECLU	11,558	93.35% (10,773)	6.64% (767)	94.32% (10,902)	89.96% (10,397)
TOMETOOLS	30,689	92.59% (28,395)	7.40% (2,272)	95.46% (29,296)	88.12% (27,044)

iTiSS	1,734	41,58% (721)	58,41% (1,013)	48,12% (848)	41,94% (739)

For the first benchmark the percentage of each algorithm's positive predictions with active transcription (group 1) and repressed or weak transcription chromatin states (group 2) as annotated by ChromHMM (Figure 35A) was calculated. 96.66% (30,376) of DeepTSS's positive predictions were found to overlap group 1 and 3.33% (1,047) group 2. ADAPT-CAGE performance was 95.90% (29,885) and 4.09% (1,276) for group 1 and 2 respectively, TOMETOOLS 92.59% (28395) and 7.40% (2272), PARACLU 94.01% (8,868) and 5.98% (565), RECLU 93.35% (10,773) and 6.64% (767), CAGER 84.62% (12,195) and 15.37% (2,215) and iTiSS 41.58% (721) and 58.41% (1,013).

Regarding the negative predictions (Figure 35B), DeepTSS exhibited an overlap of 28.74% (4,576) with group 1 and 71.25% (11,343) with group 2, 31.31% (5,067) and 68.68% (11,114) for ADAPT-CAGE, 39.42% (6609) and 60.57% (10155) for TOMETOOLS, 61.25% (18,522) and 38.75% (11,718) for PARACLU, 69.72% (27,147) and 30.27% (11,790) for RECLU, 57.63% (13,643) and 42.36% (10,029) for CAGER and 73.94% (34,405) and 26.05% (12,121) for iTiSS.

Furthermore, the second evaluation approach was utilized with the scope of exploring the occupancy of TF and H3K4me3 ChIP-Seq peaks in the vicinity of positive predictions (Figure 35C & D for TF and H3K4me3 respectively). For DeepTSS, 98.27% (30,898) of positive predictions overlapped with at least one TFBS and 92.04% (28,939) with H3K4me3, ADAPT-CAGE 97.72% (30,466) and 91.12% (28,409), TOMETOOLS 95.46% (29296) and 88.12% (27044), PARACLU 95.26% (9,005) and 92.06% (8,702), RECLU 94.32% (10,902) and 89.96% (10,397), CAGER 87.67% (12,681) and 81.11% (11,732) and iTiSS 48.12% (848) and 41.94% (739).

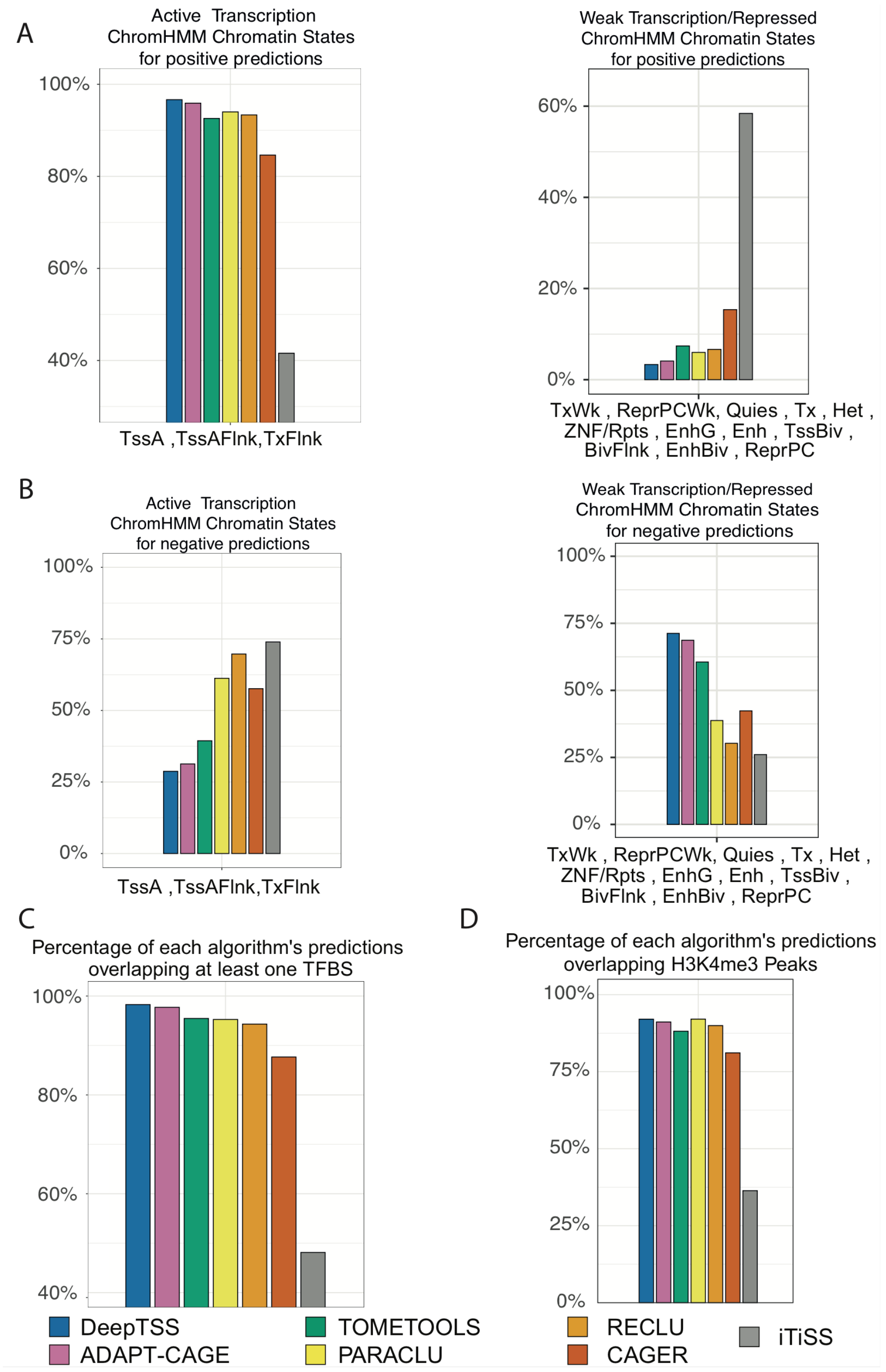

Figure 35: Evaluation of algorithms based on H3K4me3 and TF ChIP-Seq peaks as well as ChromHMM-derived chromatin states from the analysis of six histone modifications in K562 cells. Percentage of each algorithm's positive (A) and negative (B) predictions overlapping chromatin states associated with genomic regions exhibiting active (left panel) and weak/repressed (right panel) transcription. Percentage of the algorithms' positive predictions with at least one TFBS (C) and H3K4me3 peak (D) derived from ChIP-Seq.

2.3.7. Application of existing algorithms

CAGER, RECLU, PARACLU and iTiSS were applied on the ctss files provided by FANTOM with default parameters and the results we considered as their positive predictions (Supplementary Table 7). Regarding previous implementation, ADAPT-CAGE and TOMETOOLS are the only methods that utilize ML to filter out noise from the CAGE signal. ADAPT-CAGE and DiS-TSS were also used with default parameters (Supplementary Table 7). For TOMETOOLS, 1,048,124 scored CAGE enriched loci were downloaded from all cell types profiled by FANTOM for human species. Liftover from UCSC was utilized to lift the coordinates from hg37 to hg38. To discover TOMETOOLS predictions for the K562 cell line, we overlapped the FANTOM scored TSS file with the K562 CAGE peaks (N = 47,377). This way, we generated TOMETOOLS predictions in H9 and K562 cells (Supplementary Table 4). The score threshold (0.228) was chosen based on the algorithm's documentation and was used in the evaluation process shown in Figure 34. In Figure 35 benchmark, multiple score thresholds were applied.

The computational times for ADAPT-CAGE, CAGER, RECLU, PARACLU and iTiSS are ~13 hours, ~46 mins, ~13 mins, ~4 sec and ~30 mins respectively, for ~47,000 CAGE peaks. TOMTOOLS peaks were downloaded directly from the Fantom5 repository.

2.3.8. Software requirements and benchmarking

DeepTSS was developed with Python 3.7, and TensorFlow (version 2.2) with Keras API (version 2.4.3) for implementing the DL part of the framework. All dependencies and a thorough documentation can be found at the GitHub repository.

To apply DeepTSS, users must provide a CAGE bam file or a bed file with precalculated CAGE peak representatives, the corresponding human genome assembly in fasta format, and the evolutionary conservation score as calculated by phyloP in bigWig format. If the phyloP score is not provided, the evolutionary conservation branch will not be utilized automatically, and predictions will be based on the remaining feature types.

DeepTSS was benchmarked in terms of computational cost on a computer running on an Intel Xeon E5-2630 v3 @ 2.40GHz and a total of 8 threads was utilized for performance benchmarking, to simulate the average CPU capacity of personal computers. The time cost for predicting approximately 40,000 CAGE peak representatives was ~4 minutes on average, with all convolutional branches enabled.

DeepTSS is freely available to the public and can be downloaded at the following GitHub repository: https://github.com/DianaLaboratory/DeepTSS.

2.3.9. Discussion

DeepTSS is a novel DL-based computational framework for removing noise from CAGE data and maximizing the probability that the remaining CAGE signal corresponds to transcription initiation events. DeepTSS operates on a seemingly unrelated spectrum of features that are used as input to a DL architecture that was specifically designed to exploit each individual feature type. In contrast to existing implementations, DeepTSS does not require any kind of prior feature engineering process since it relies on convolutional layers directly embedded in the DL architecture that can readily identify patterns and only utilize the important ones for the classification task. DeepTSS was found to outperform existing state-of-the-art implementations when evaluated on a meticulously designed strategy that included experimental data and high-quality genome annotations.

This study highlights the importance of ML, and specifically DL, in providing solutions to removing inherent flaws in experimental methods that are the bread-and-butter of contemporary Molecular Biology research. Reliable algorithms, like DeepTSS, can unleash the full potential of already popular protocols such as CAGE, and play a fundamental role towards unveiling key gene expression regulators as well as pushing the boundaries of non-coding RNAs implication in regulatory networks even further. Finally, DeepTSS is utilized without using gene expression quantities of CAGE peaks and works directly on the DNA paving the road for novel gene identification.

3. Analysis of variants in regulatory regions

3.1. Overview of variants in diseases

Variants affecting regulation of gene expression have been proposed as one of the main causes of human diversity. Furthermore, genomic variants implicated in regulatory mechanisms may affect an organism’s health. The importance of SNPs is underscored from GWAS, which have correlated a vast number of polymorphisms to various diseases. Alongside the SNPs, analysis of rare variants in non-coding loci has provided immense insights and has had a great impact in cancer understanding and treatment. This type of analysis paved the road for better drug discovery and precision medicine [121].

In this chapter, rare variants, or better referenced mutations, have been studied with respect to the functional impact on an organism’s genome. Furthermore, a novel methodology based on ML techniques is introduced and aims to overcome technical difficulties encountered in the context of characterizing survival outcomes of cancer patients. To this end, mutations (rare variants) are thoroughly analyzed, and patients’ survival outcome has been correlated with mutations in promoters’ loci. This study highlights that mutations in these regulatory regions can be robust predictors of cancer severity and overall patient survival potential.

3.1.1. Promoter loci

A great challenge for analyzing regulatory region profiles was to define the promoter length. It is crucial that promoter length is used with caution and that the variants analyzed are in the context of non-coding regulatory regions. To this end, Enriched DNase-seq profiles from the ENCODE consortium were analyzed. Narrow peaks already analyzed with ENCODE’s pipeline were obtained from the repository. This task was performed in order to identify regions with high open chromatin density relative to gene start. The consensus of all peaks was considered as the region of interest (Figure 36).

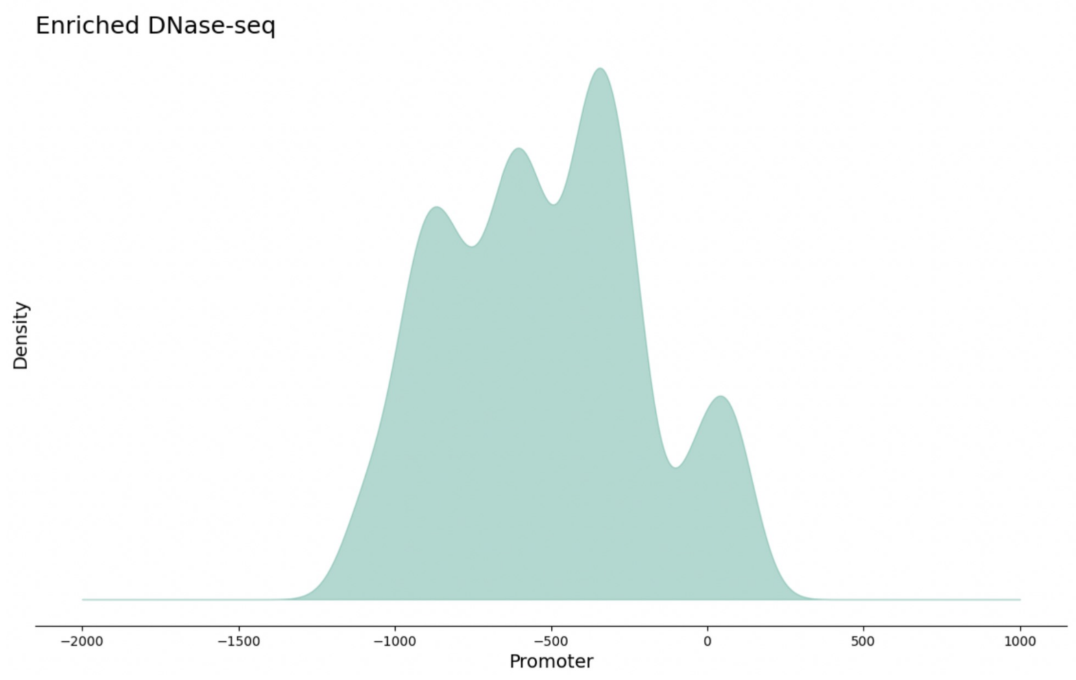

Figure 36: Enriched DNase-seq regions around the promoter. Coordinates are relative to gene start.

Thereby, promoters are considered as 2,000bp length regions, 1,500bp upstream and 500bp downstream of gene start with respect to strand (Figure 37).

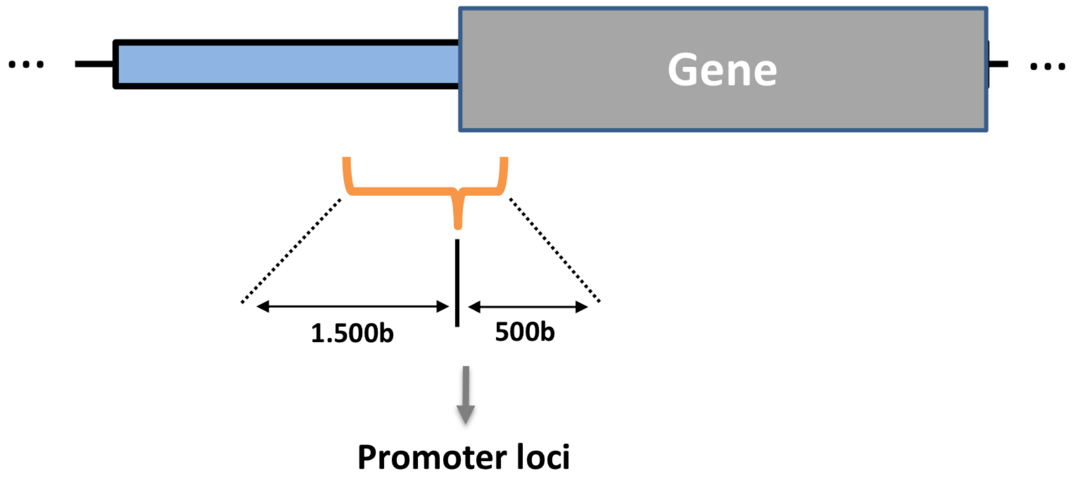

Figure 37: Promoter loci. Promoters are considered as 2,000bp length regions (1,500bp upstream and 500bp downstream with respect to strand).

3.1.2. Exploratory data analysis

In the following chapters pan-cancer analysis was conducted regarding mutations residing within promoter regions. These mutations were studied based on how they may affect the overall survival of cancer patients. To meet these requirements WGS and survival metadata were obtained from the ICGC consortium. More than 1000 whole genomes were analyzed across seven (7) different primary cancers (Brain, Esophagus, Kidney, Liver, Ovary, Pancreas and Prostate).

Indels were separated from single nucleotide variants (SNV). SNVs overlapping promoter loci were discovered in higher abundance compared to indels which were found in much lower numbers (Figure 38).

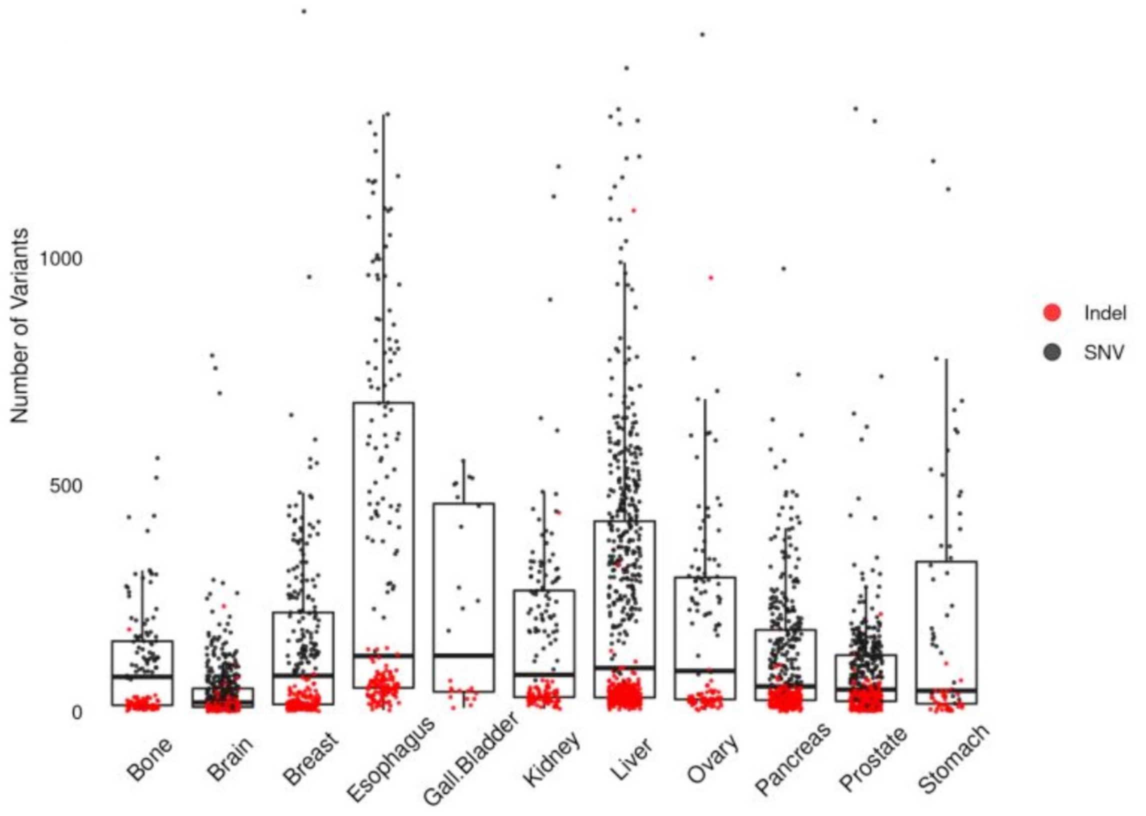

Figure 38: Abundance of variants in promoter loci genome wide pan-cancer. Each dot represents a sample. Red dots correspond to indels and dark dots to SNVs.

3.1.3. Somatic mutation-type features

Downstream analysis focuses on SNVs (the most abundant type of variant in the datasets) that reside on promoter regions of protein coding genes. Each SNV and its flanking sequences serve as features for the model. More specifically, for each donor, all SNVs are categorized across the six possible single-nucleotide changes, the 96 possible nucleotide changes plus both flanking nucleotides, the 78 possible doublet-nucleotide changes (Doublet Base Substitution, DBS) and the 28 and 96 possible insertions/deletions. This concludes in a set with 304 total features. All features were extracted with SigProfilerMatrixGenerator [122].

3.1.4. Mutational signature extraction

De novo promoter-signature discovery was utilized with SigProfilerExtractor method [123], based on the 96 trinucleotide sequence contexts for single base substitutions (Figure 39). Two signatures were obtained for each tissue's high and low risk groups, to assess the underlying mutational processes. Mutational processes from different etiologies are retrieved from the cosmic database version 3 [85]. Thus, cosine similarity and correlation metrics were calculated between original and reconstructed signatures with values greater than 0.8 and 0.7 respectively for all tissues.

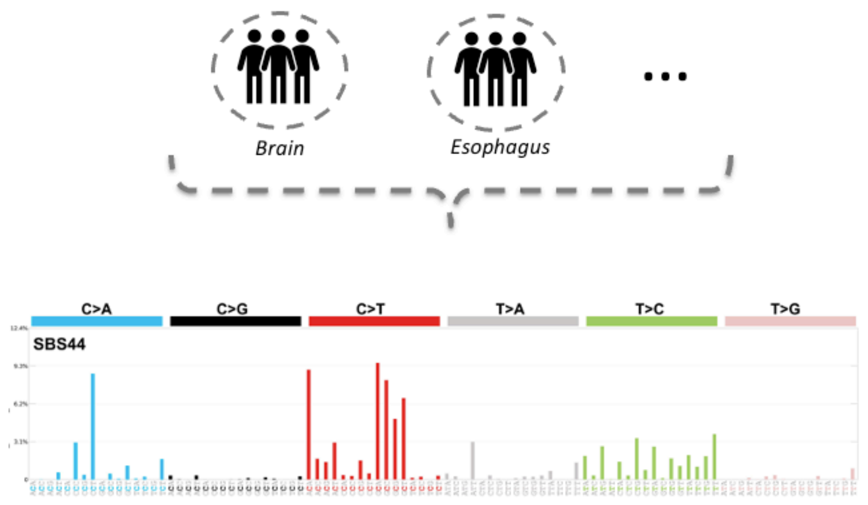

Figure 39: Mutational signatures extraction from patients in 8 different tissues. (Figure has been created for the needs of the current thesis)

3.2. Training of autoencoders

Mutation-type features were used as inputs to train multiple autoencoder models. For each model, a single convolutional branch is created, composed of 2 layers. For each tissue, a different model was trained, concluding in a total of seven (7) different models. Inputs were standardized by utilizing standard scalers. Hyperparameters were optimized, separately for each of the 8 models, by training different models for over 20 epochs, exhausting all possible combinations and evaluating the validation loss. Also, 10% of the data, for each tissue, were used as a validation for the autoencoder. Autoencoder models final layers size, vary between 10 and 20 nodes. For further reduction UMAP was utilized, reducing the dimensions to 2. All models number of neighbors parameter is set to 5 (Figure 40).

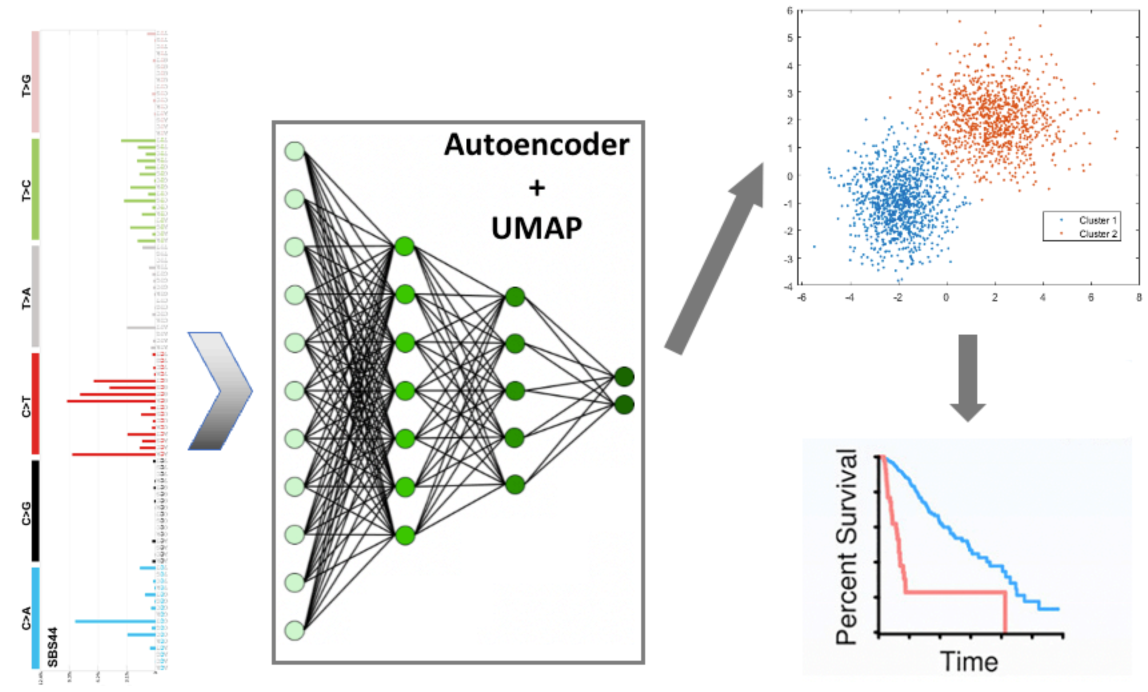

Figure 40: Pipeline overview. Mutational signatures are fed to the autoencoder and therefore to UMAP. clustering algorithm was utilized (k-means) to cluster patients and Kaplan-Meier curve estimation indicated separation between patient groups. Also, strong statistical significance was discovered for the log rank test. (Figure has been created for the needs of the current thesis)

3.2.1. Clustering

K-means, a widely used and efficient clustering algorithm, was utilized to group between samples (Figure 41, Figure 51). The elbow method was used to determine the optimal number of total clusters (see chapter 1.4.8 K-means). Thus, two (2) clusters are created as the optimal cluster number for minimizing within-cluster variance for all tissues, except for ovary that indicated an optimal number of 3 total clusters. K-means was utilized with 300 max iterations.

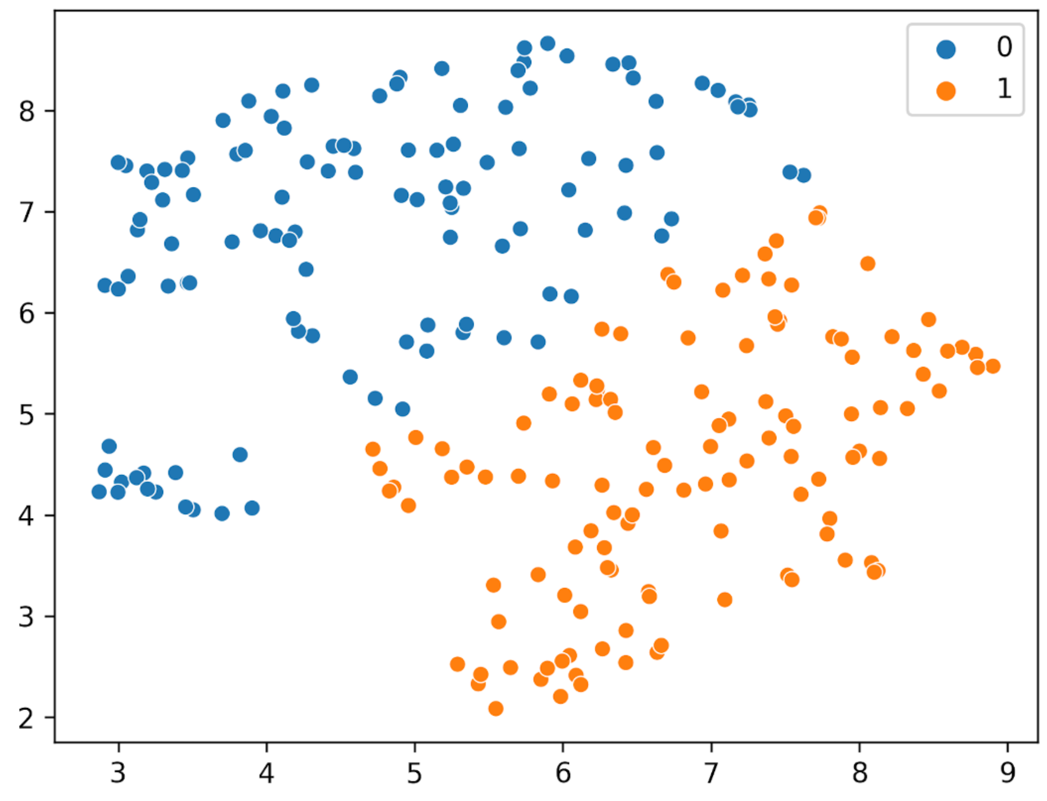

Figure 41: K-means clustering scatter plot of brain tissue. All tissues can be found at chapter 5 Supplements.

3.2.2. Survival analysis

All models were evaluated based on their separation ability of survival. Kaplan-Meier curve [124] estimation indicated separation between patient groups in all seven (7) tissues for two (2) levels of mortality (Figure 42 and Figure 52). For analyzing the survival data two variables are considered. The time for which follow up was available, and the status at the end of the follow-up. The former is a continuous variable (time in days) and the latter a categorical, specifying whether the endpoint was the event of interest. Also, strong statistical significance was discovered for the log rank test between the groups in each cancer.

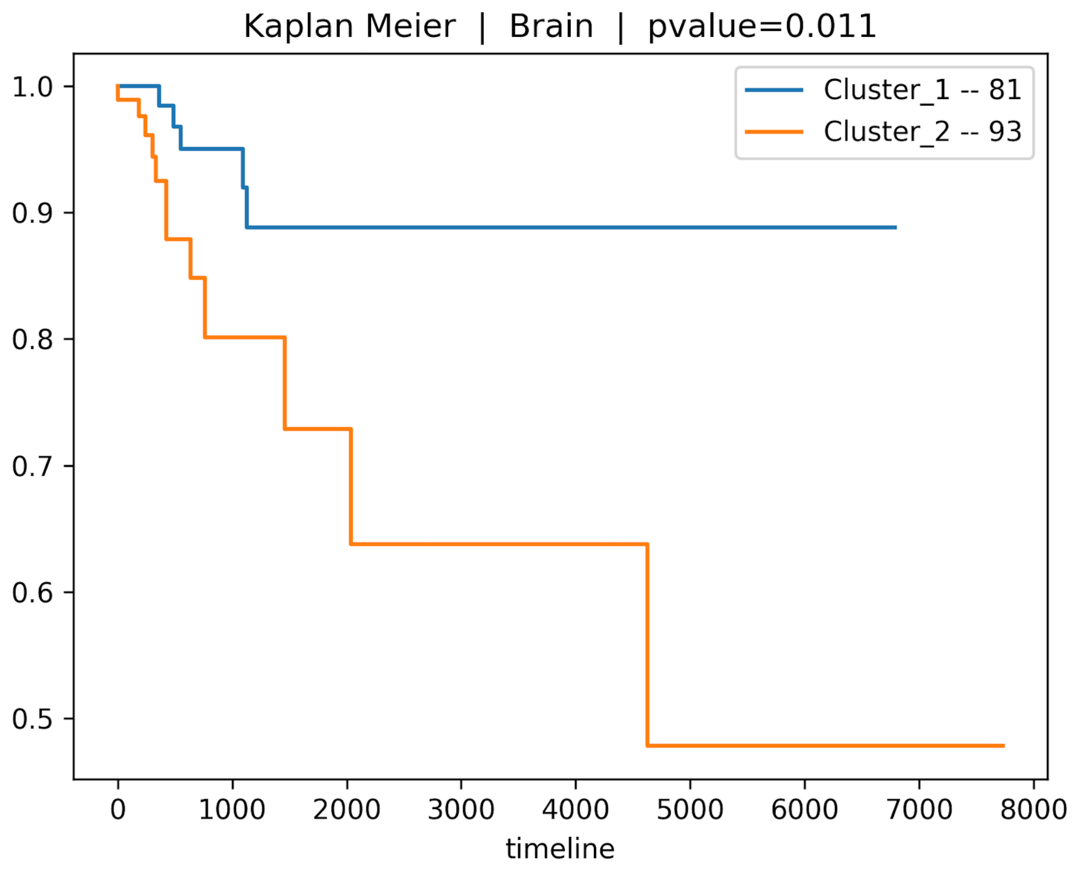

Figure 42: Kaplan-Meir curves of brain tissue. All tissues can be found at chapter 5 Supplements.

3.2.3. Analysis of non-regulatory regions

To assess the predictive power of SNVs residing specifically in promoter regions with respect to patient survival outcome, the pipeline was utilized on a different set of non-coding regions that served as negative (Figure 43). Negative regions consist of 50Kbp length and 10Kbp upstream of the promoters (with respect to strand). Special attention was given so that negative zones do not overlap any upstream gene.

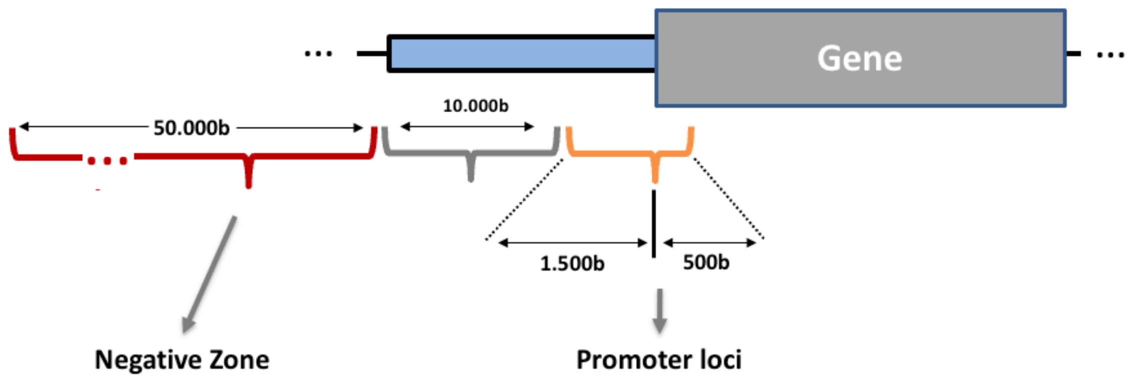

Figure 43: Non-regulatory regions. Negative zones are indicated in red. (Figure has been created for the needs of the current thesis)

3.3. Mutational signature analysis

Each cancer type may present unique underlying mutational processes. To further investigate this matter, for each risk group mutational profiles were extracted and underlying mutational processes were deciphered. In all tissues and for each group, mutational signatures were extracted across the six possible single-nucleotide changes.

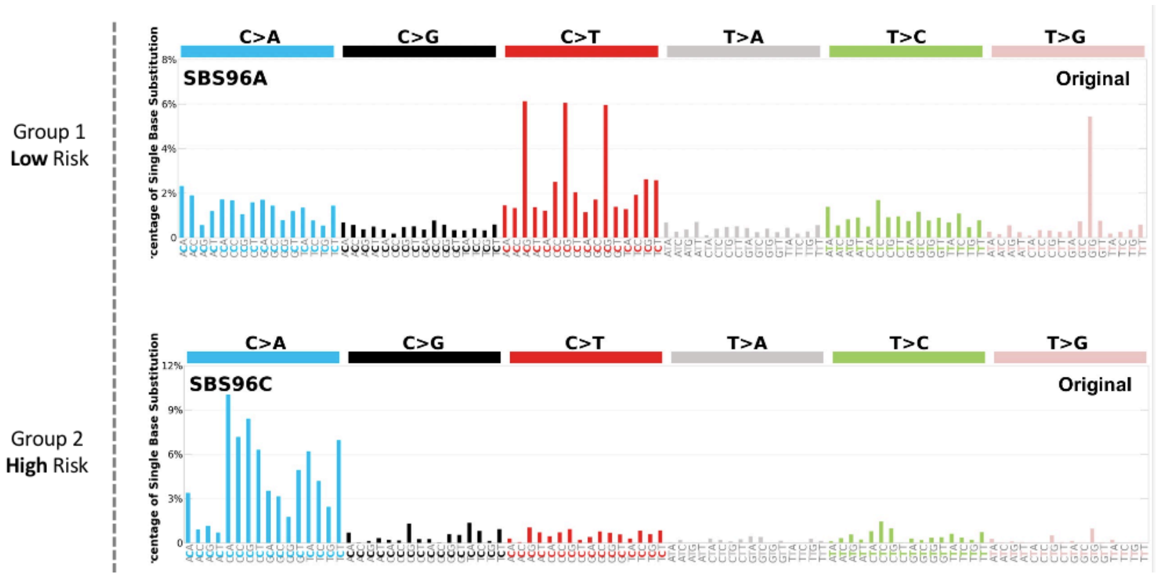

Figure 44: Mutational signatures for brain tissue. Differences are observed between blue, red, and pink SNS categories.

3.3.1. Deciphering Mutational Signatures

For each signature cosine similarity and correlation metrics were calculated and compared to COSMIC database pre-extracted mutational processes (Figure 45). By deciphering each one of the signatures, mutational processes revealedBER deficiency, Spontaneous deamination of 5-methylcytosine, tobacco smoking and aging as factors of cancer progression. Regarding the brain tissue BER deficiency and tobacco smoking are the two (2) underlying processes between the 2 risk groups. Aging was observed in all tissues, but it is not an environmental factor (like UV light). Also, it is known that as organisms age more mutations are accumulated in the genome. All processes for all tissues can be found in supplementary Table 9.

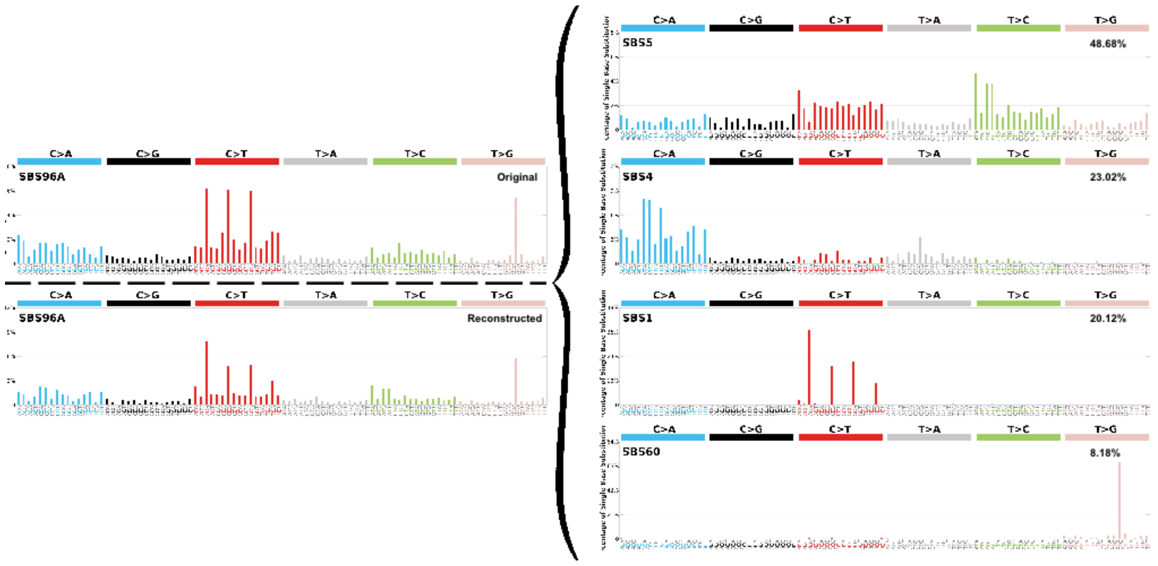

Figure 45: Brain low risk group signatures mutational processes.

3.4. Hotspot analysis

Hotspot analysis, which is a spatial analysis and mapping technique interested in the identification of clustering of spatial phenomena, was utilized for mutations in the promoter loci. For each promoter, mutations across patients with distance below fifty (<50bp) nucleotides are clustered together, and "hotspots" are created (Figure 46). All clusters having less than ten (10) mutations are filtered out. With this type of analysis expression differences in genes can be analyzed in spatial order. Hotspots are highly likely to damage certain parts of the promoter and lead to deregulation of the genes.

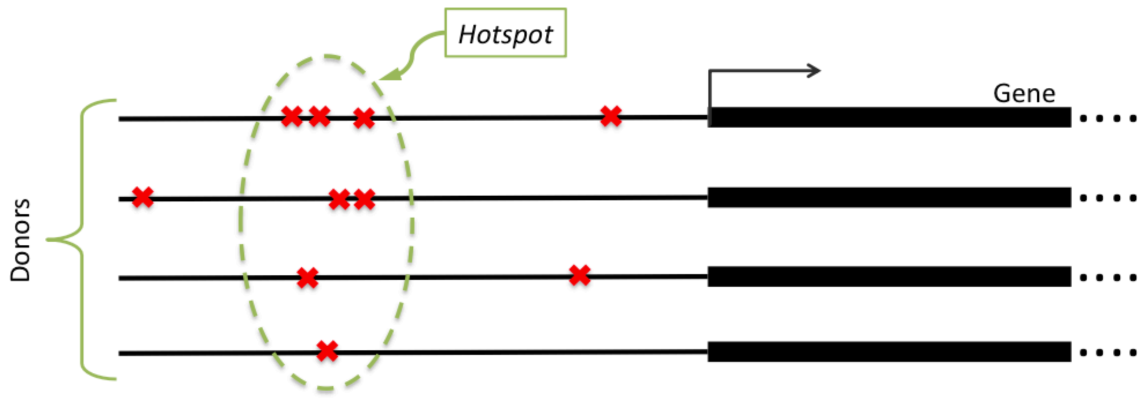

Figure 46: Mutation hotspots in promoter loci of 5 samples/donors.

3.4.1. Abundance of mutations in hotspots

Brain and liver tissues indicated a high abundance of mutations in hotspots, for several promoters, but mean values for all tissues are approximately 7-10 mutations within each hotspot. Overall samples have indicated some mutational activity in their promoter loci.

3.4.2. Expression differences

To further explore the impact of mutations in promoter regions, gene expression analysis was performed for mutated and non-mutated (wild type) promoters. To eliminate biases on gene expression arising from disease onset rather than the underlying mutations, only cancerous samples were included. Due to limitations in the availability of both WGS and RNA-Seq data derived from the same patient, downstream analysis was performed only for cancerous liver tissue with an adequate number of samples (n =295) meeting the criteria. For every gene, two groups were created and compared, one that included samples with the mutated promoter along their gene expression value (measured in TPM) and a second one with the wild type (non-mutated) promoter and gene expression value. The log10 of TPM values in both groups was plotted (Figure 48). MALAT1 was the most frequently mutated promoter in this dataset (Liver). The gene’s downregulation in the mutated samples was found to be statistically significant (p=0.016). Mutations in the promoter of ZNF430 (which encodes for Zinc finger protein 430) were associated with the expression of ZNF430 which was significantly different in wild types and marginally above strong significance (p=0.019). PTGFR had a significant p-value (p=0.001) with downregulated mutated samples, however the dataset lacks abundance of mutated samples. Finally, WDR74 was marginally above significant p-value (p=0.065) and mutated samples tend to have lower expression levels.

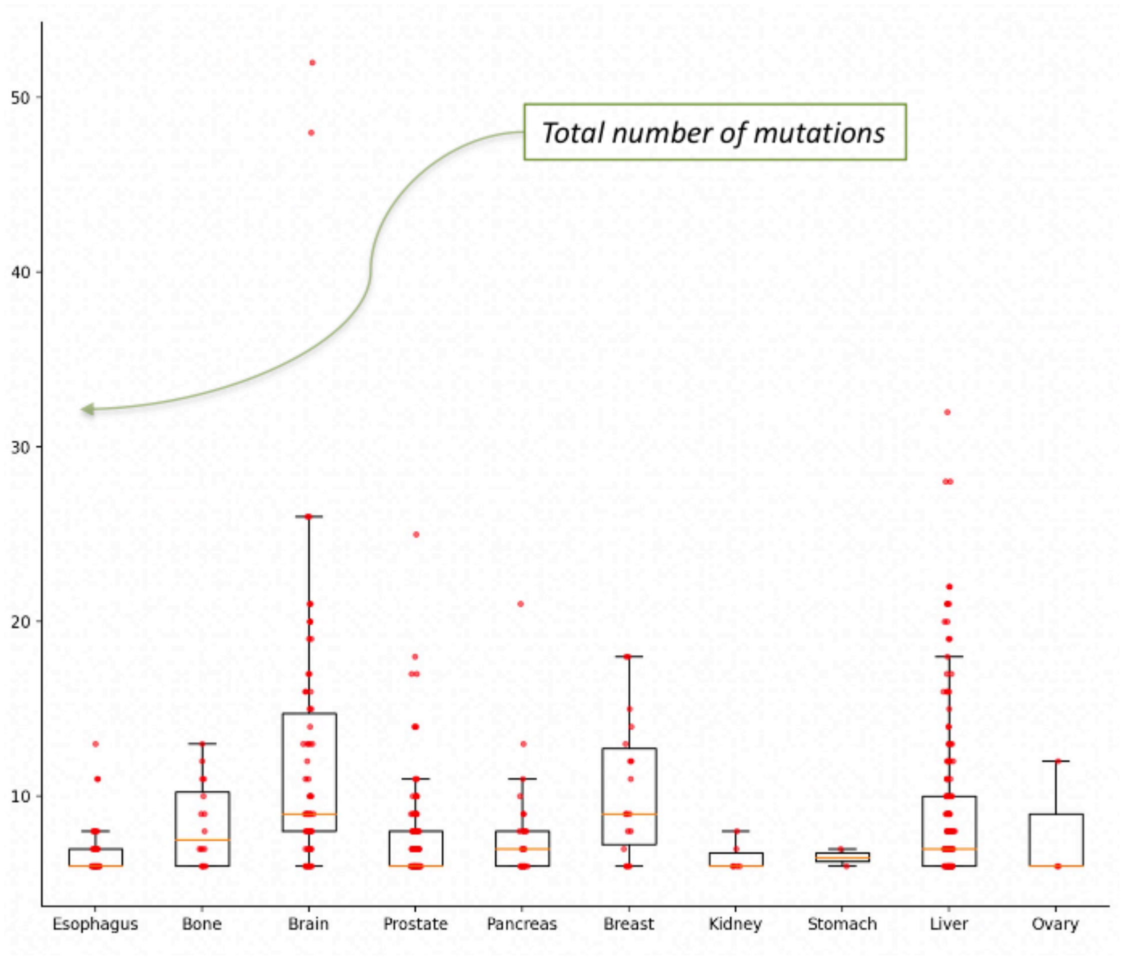

Figure 47: Boxplot of hotspot abundance across tissues in promoters. Each dot stands for a gene's promoter.

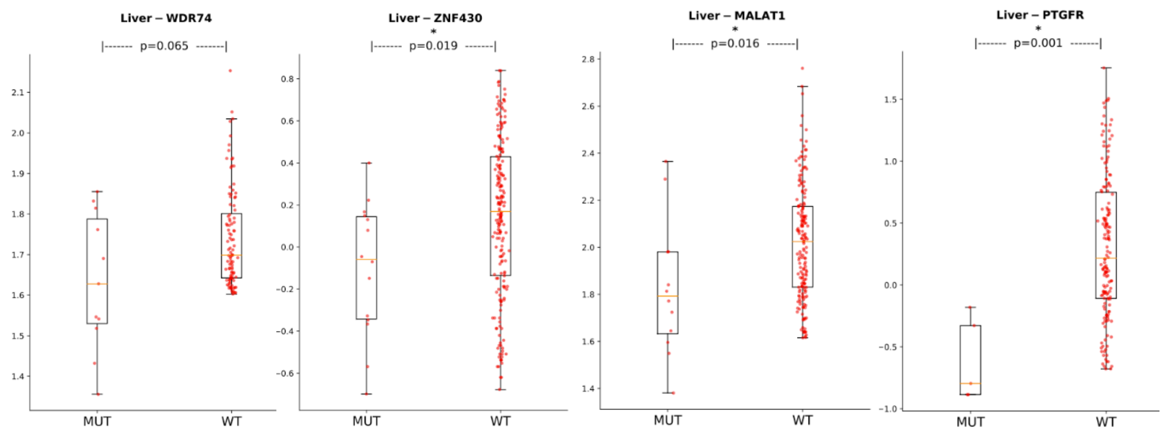

Figure 48: Log10 differences of the expression values (TPM) for mutated and wild type samples.

4. Publications

During the current doctoral thesis several scientific publications were published and are listed below by category and chronological order.

Transcription start site identification algorithms

1. **DiS-TSS: An Annotation Agnostic Algorithm for TSS Identification**. Grigoriadis D., Perdikopanis N., Georgakilas G.K., Hatzigeorgiou A. (2020) Bioinformatics and Biomedical Engineering. IWBBIO 2020. https://doi.org/10.1007/978-3-030-45385-5_55

The above paper was selected by the conference to submit the extended study to the journal "BMC-Bioinformatics".

2. **DeepTSS: multi-branch convolutional neural network for transcription start site identification from CAGE data**. *BMC Bioinformatics* **23** (Suppl 2), 395 (2022). Grigoriadis, D., Perdikopanis, N., Georgakilas, G.K. and A. G. Hatzigeorgiou. https://doi.org/10.1186/s12859-022-04945-y.

Repositories of miRNA transcription start sites and regulatory factors

3. **DIANA-miRGen v4: indexing promoters and regulators for more than 1500 microRNAs. Nucleic acids research,** Perdikopanis, N†., Georgakilas, G. K†., Grigoriadis, D., Pierros, V., Kavakiotis, I., Alexiou, P., & Hatzigeorgiou, A. (2021) https://academic.oup.com/nar/article/49/D1/D151/6007663?login=false

†These authors contributed equally: Nikos Perdikopanis and Georgios K. Georgakilas

Genetic variation in cancer

4. **A deep clustering approach for separating different levels of cancer risk, using genomic variations in regulatory regions.** (Under submission) Dimitris Grigoriadis, Marios Miliotis, Spyros Tastsoglou and Artemis G. Hatzigeorgiou

5. Supplements

Table 4: Performance metrics for all algorithms after applying multiple score cutoffs and combinations of parameters. This table corresponds to the hybrid evaluation presented in Figure 3 for K562 cells (Supplementary Table).

Algorithm	Threshold	Predictions	TP	FN	FP	TN	Specificity	Sensitivity	precision	accuracy
DeepTSS	0	9117	6663	36	765	1653	0,684	0,995	0,897	0,912
DeepTSS	0,01	9140	6645	54	573	1868	0,765	0,992	0,921	0,931
DeepTSS	0,02	9150	6634	65	496	1955	0,798	0,990	0,930	0,939
DeepTSS	0,03	9155	6629	70	447	2009	0,818	0,990	0,937	0,944
DeepTSS	0,04	9162	6624	75	424	2039	0,828	0,989	0,940	0,946
DeepTSS	0,05	9169	6618	81	400	2070	0,838	0,988	0,943	0,948
DeepTSS	0,1	9188	6599	100	342	2147	0,863	0,985	0,951	0,952
DeepTSS	0,2	9209	6581	118	288	2222	0,885	0,982	0,958	0,956
DeepTSS	0,3	9224	6567	132	260	2265	0,897	0,980	0,962	0,958
DeepTSS	0,4	9238	6555	144	232	2307	0,909	0,979	0,966	0,959
DeepTSS	0,6	9263	6530	169	195	2369	0,924	0,975	0,971	0,961
DeepTSS	0,7	9278	6517	182	176	2403	0,932	0,973	0,974	0,961
DeepTSS	0,8	9300	6489	210	159	2442	0,939	0,969	0,976	0,960
DeepTSS	**0.9 (default)**	9354	**6398**	301	**123**	2532	0,954	**0,955**	**0,981**	0,955
DeepTSS	0,91	9364	6386	313	119	2546	0,955	0,953	0,982	0,954
DeepTSS	0,92	9370	6376	323	118	2553	0,956	0,952	0,982	0,953
DeepTSS	0,93	9376	6352	347	114	2563	0,957	0,948	0,982	0,951
DeepTSS	0,94	9391	6325	374	112	2580	0,958	0,944	0,983	0,948
DeepTSS	0,95	9407	6299	400	107	2601	0,960	0,940	0,983	0,946
DeepTSS	0,96	9432	6267	432	100	2633	0,963	0,936	0,984	0,944
DeepTSS	0,97	9471	6220	479	97	2675	0,965	0,928	0,985	0,939
DeepTSS	0,98	9539	6102	597	83	2757	0,971	0,911	0,987	0,929
DeepTSS	0,99	9694	5753	946	55	2940	0,982	0,859	0,991	0,897
DeepTSS	0,995	9694	5753	946	55	2940	0,982	0,859	0,991	0,897
DeepTSS	0,996	9694	5753	946	55	2940	0,982	0,859	0,991	0,897
DeepTSS	0,997	9694	5753	946	55	2940	0,982	0,859	0,991	0,897
ADAPT-CAGE	0,01	9170	6617	82	753	1718	0,695	0,988	0,898	0,909
ADAPT-CAGE	0,02	9198	6587	112	581	1918	0,768	0,983	0,919	0,925
ADAPT-CAGE	0,03	9227	6545	154	454	2074	0,820	0,977	0,935	0,934
ADAPT-CAGE	0,04	9247	6537	162	408	2140	0,840	0,976	0,941	0,938
ADAPT-CAGE	0,05	9251	6525	174	380	2172	0,851	0,974	0,945	0,940
ADAPT-CAGE	0,1	9281	6491	208	304	2278	0,882	0,969	0,955	0,945
ADAPT-CAGE	0,2	9315	6444	255	245	2371	0,906	0,962	0,963	0,946
ADAPT-CAGE	0,3	9337	6415	284	215	2423	0,918	0,958	0,968	0,947
ADAPT-CAGE	0,4	9384	6353	346	187	2498	0,930	0,948	0,971	0,943
ADAPT-CAGE	**0.5 (default)**	9421	**6294**	405	**172**	2550	0,937	**0,940**	**0,973**	0,939
ADAPT-CAGE	0,6	9466	6214	485	155	2612	0,944	0,928	0,976	0,932
ADAPT-CAGE	0,7	9524	6102	597	140	2685	0,950	0,911	0,978	0,923
ADAPT-CAGE	0,8	9612	5906	793	125	2788	0,957	0,882	0,979	0,904
ADAPT-CAGE	0,9	9778	5495	1204	107	2972	0,965	0,820	0,981	0,866
ADAPT-CAGE	0,91	9817	5328	1371	99	3019	0,968	0,795	0,982	0,850
ADAPT-CAGE	0,92	9835	5279	1420	99	3037	0,968	0,788	0,982	0,846
ADAPT-CAGE	0,93	9860	5220	1479	99	3062	0,969	0,779	0,981	0,840
ADAPT-CAGE	0,94	9939	4950	1749	89	3151	0,973	0,739	0,982	0,815
ADAPT-CAGE	0,95	9964	4852	1847	86	3179	0,974	0,724	0,983	0,806
ADAPT-CAGE	0,96	10040	4591	2108	75	3266	0,978	0,685	0,984	0,783
ADAPT-CAGE	0,97	10099	4338	2361	64	3336	0,981	0,648	0,985	0,760
ADAPT-CAGE	0,98	10209	3942	2757	55	3455	0,984	0,588	0,986	0,725
ADAPT-CAGE	0,99	10373	3150	3549	49	3625	0,987	0,470	0,985	0,653
ADAPT-CAGE	0,995	10569	2227	4472	40	3830	0,990	0,332	0,982	0,573
TOMETOOLS	0,1	9544	5924	775	592	2253	0,792	0,884	0,909	0,857
TOMETOOLS	0,15	9644	5890	809	309	2636	0,895	0,879	0,950	0,884
TOMETOOLS	**0.228 (default)**	9819	**5765**	934	**228**	2892	0,927	**0,861**	**0,962**	0,882
TOMETOOLS	0,25	10024	5482	1217	165	3160	0,950	0,818	0,971	0,862
TOMETOOLS	0,3	10178	5015	1684	126	3353	0,964	0,749	0,975	0,822
TOMETOOLS	0,4	10427	3570	3129	70	3658	0,981	0,533	0,981	0,693
TOMETOOLS	0,5	10673	1635	5064	31	3943	0,992	0,244	0,981	0,523

Table 5: Performance metrics based on multiple score cutoffs and combinations of parameters for the annotated protein coding gene-oriented evaluation that is presented in Figure 3D in K562 cells (Supplementary Table).

Gene oriented TP/FP

Algorithm	**parameters**	**length**	**count TP**	**count FP**
DeepTSS	0	500	3216	503
DeepTSS	0,01	500	3210	390
DeepTSS	0,02	500	3206	346
DeepTSS	0,03	500	3205	313
DeepTSS	0,04	500	3202	297
DeepTSS	0,05	500	3199	283
DeepTSS	0,1	500	3191	240
DeepTSS	0,2	500	3185	204
DeepTSS	0,3	500	3179	188
DeepTSS	0,4	500	3175	173
DeepTSS	0,6	500	3163	141
DeepTSS	0,7	500	3160	130
DeepTSS	0,8	500	3150	120
DeepTSS	**0.9 (default)**	500	**3122**	**91**
DeepTSS	0,91	500	3119	88
DeepTSS	0,92	500	3117	88
DeepTSS	0,93	500	3112	87
DeepTSS	0,94	500	3107	87
DeepTSS	0,95	500	3102	85
DeepTSS	0,96	500	3097	81
DeepTSS	0,97	500	3084	79
DeepTSS	0,98	500	3064	70
DeepTSS	0,99	500	2971	52
DeepTSS	0,995	500	2971	52
DeepTSS	0,996	500	2971	52
DeepTSS	0,997	500	2971	52
ADAPT-CAGE	0,01	500	3202	512
ADAPT-CAGE	0,02	500	3189	404
ADAPT-CAGE	0,03	500	3175	323
ADAPT-CAGE	0,04	500	3172	290
ADAPT-CAGE	0,05	500	3169	273
ADAPT-CAGE	0,1	500	3155	225
ADAPT-CAGE	0,2	500	3135	181
ADAPT-CAGE	0,3	500	3129	162
ADAPT-CAGE	0,4	500	3112	140
ADAPT-CAGE	**0.5 (default)**	500	**3091**	**125**
ADAPT-CAGE	0,6	500	3071	114
ADAPT-CAGE	0,7	500	3049	106
ADAPT-CAGE	0,8	500	3015	93
ADAPT-CAGE	0,9	500	2923	81
ADAPT-CAGE	0,91	500	2891	75
ADAPT-CAGE	0,92	500	2881	75

ADAPT-CAGE	0,93	500	2868	75
ADAPT-CAGE	0,94	500	2818	67
ADAPT-CAGE	0,95	500	2790	66
ADAPT-CAGE	0,96	500	2723	62
ADAPT-CAGE	0,97	500	2636	55
ADAPT-CAGE	0,98	500	2523	49
ADAPT-CAGE	0,99	500	2251	47
ADAPT-CAGE	0,995	500	1818	41
TOMETOOLS	0,1	500	3086	454
TOMETOOLS	0,15	500	3069	237
TOMETOOLS	**0.228 (default)**	500	**3016**	**174**
TOMETOOLS	0,25	500	2867	131
TOMETOOLS	0,3	500	2612	99
TOMETOOLS	0,4	500	1899	62
TOMETOOLS	0,5	500	1035	29

Table 6: Table corresponding to all experimental-based evaluations. Columns (b) correspond to the number of predictions for each algorithm. These numbers are identical to Supplementary Table 1. In the remaining columns (from c to f) we can see the number of predictions overlapping various experimental data enriched regions of the genome. The summation of columns E and G for H9 and F and H for K562 does not add up to the numbers shown C for H9 and D for K562 because certain predictions do not overlap with any of the ChromHMM chromatin states annotated regions and some overlaps with more than one regions (Supplement table).

Algorithm (a)	Cluster Of TSS (predictions) (b)	Overlap with												
		positive predictions				negative predictions								
		Active transcription	%	**transcription**	%	**Active transcription**	%	**transcription**	**%**	**(e)**	**%**	**(f)**	**%**
DeepTSS	**31443**	30376	96,66	1047	3,33	4576	28,74	11343	71,25	30898	98,27	28939	92,04
ADAPT-CAGE	**31177**	29885	95,90	1276	4,09	5067	31,31	11114	68,68	30466	97,72	28409	91,12
CAGER	**14465**	12195	84,62	2215	15,37	13643	57,63	10029	42,36	12681	87,67	11732	81,11
PARACLU	**9453**	8868	94,01	565	5,98	18522	61,25	11718	38,75	9005	95,26	8702	92,06
RECLU	**11558**	10773	93,35	767	6,64	27147	69,72	11790	30,27	10902	94,32	10397	89,96
TOMETOOLS	**30689**	28395	92,59	2272	7,40	6609	39,42	10155	60,57	29296	95,46	27044	88,12
iTiSS	**1762**	721	41,58	1013	58,41	34405	26,05	12121	26,05	848	48,12	641	36,37

Table 7: Evaluation algorithms default values (Supplement table).

Algorithm	Default values
ADAPT-CAGE	Score threshold=0.5
PARACLU	Number of reads=30
RECLU	tpm=0.1
CAGER	threshold = 0.5 (tpm filter)
TOMETOOLS	Score threshold=0.228
iTiSS	modeType=DENSE_PEAK

Table 8: DeepTSS hyper parameters (Supplementary table)

Training metrics				Evaluation metrics				Hyper parameters		
Acc	loss	precision	recall	Val accuracy	Val loss	Val precision	Val recall	Learning rate	Batch size	Drop out
0,938	0,187	0,929	0,948	0,960	0,137	0,973	0,945	0,01	64	0,2
0,950	0,149	0,944	0,957	0,961	0,118	0,953	0,969	0,001	64	0,2
0,953	0,139	0,947	0,960	0,963	0,122	0,975	0,950	0,0005	64	0,2
0,957	0,125	0,951	0,963	0,951	0,140	0,927	0,979	0,0001	64	0,2
0,959	0,120	0,953	0,966	0,957	0,135	0,943	0,972	0,01	128	0,2
0,960	0,115	0,954	0,967	0,960	0,117	0,947	0,973	0,001	128	0,2
0,961	0,112	0,955	0,967	0,960	0,119	0,964	0,955	0,0005	128	0,2
0,955	0,133	0,949	0,962	0,962	0,121	0,968	0,955	0,0001	128	0,2
0,957	0,125	0,951	0,963	0,962	0,119	0,954	0,970	0,01	256	0,2
0,978	0,061	0,975	0,981	0,977	0,085	0,971	0,982	0,001	256	0,2
0,953	0,142	0,946	0,960	0,954	0,140	0,935	0,975	0,0005	256	0,2
0,956	0,133	0,949	0,963	0,960	0,129	0,971	0,948	0,0001	256	0,2
0,953	0,140	0,946	0,961	0,949	0,158	0,959	0,937	0,01	512	0,2
0,955	0,133	0,949	0,962	0,955	0,136	0,948	0,963	0,001	512	0,2
0,957	0,126	0,951	0,964	0,954	0,136	0,940	0,969	0,0005	512	0,2
0,960	0,127	0,953	0,964	0,952	0,131	0,940	0,960	0,0001	512	0,2
0,948	0,156	0,939	0,958	0,953	0,143	0,941	0,967	0,01	64	0,25
0,951	0,147	0,943	0,959	0,944	0,160	0,915	0,979	0,001	64	0,25
0,952	0,143	0,945	0,960	0,954	0,134	0,943	0,966	0,0005	64	0,25
0,954	0,137	0,947	0,962	0,953	0,137	0,945	0,963	0,0001	64	0,25
0,955	0,134	0,948	0,962	0,957	0,130	0,960	0,954	0,01	128	0,25
0,956	0,130	0,949	0,963	0,954	0,141	0,961	0,946	0,001	128	0,25
0,954	0,137	0,948	0,961	0,959	0,126	0,972	0,945	0,0005	128	0,25
0,956	0,131	0,950	0,963	0,960	0,132	0,969	0,949	0,0001	128	0,25
0,957	0,127	0,951	0,963	0,959	0,128	0,960	0,956	0,01	256	0,25
0,958	0,124	0,952	0,964	0,963	0,120	0,968	0,956	0,001	256	0,25
0,959	0,121	0,953	0,965	0,965	0,119	0,956	0,964	0,0005	256	0,25
0,959	0,118	0,953	0,966	0,966	0,118	0,955	0,965	0,0001	256	0,25
0,960	0,116	0,955	0,966	0,963	0,119	0,952	0,964	0,01	512	0,25
0,961	0,114	0,955	0,968	0,966	0,118	0,963	0,961	0,001	512	0,25
0,963	0,104	0,957	0,970	0,950	0,117	0,942	0,926	0,0005	512	0,25
0,965	0,101	0,959	0,971	0,958	0,118	0,974	0,940	0,0001	512	0,25
0,966	0,119	0,960	0,972	0,960	0,117	0,960	0,950	0,01	64	0,3
0,966	0,181	0,961	0,972	0,950	0,191	0,955	0,921	0,001	64	0,3
0,967	0,171	0,962	0,971	0,953	0,184	0,945	0,952	0,0005	64	0,3
0,961	0,128	0,964	0,974	0,959	0,124	0,961	0,958	0,0001	64	0,3
0,957	0,126	0,951	0,964	0,961	0,125	0,963	0,958	0,01	128	0,3
0,957	0,129	0,951	0,963	0,965	0,122	0,968	0,961	0,001	128	0,3
0,961	0,113	0,954	0,968	0,965	0,116	0,962	0,967	**0,0005**	**128**	**0,3**
0,959	0,118	0,953	0,965	0,966	0,117	0,958	0,961	0,0001	128	0,3
0,949	0,155	0,943	0,955	0,944	0,157	0,916	0,977	0,01	256	0,3
0,953	0,139	0,947	0,960	0,953	0,136	0,935	0,974	0,001	256	0,3
0,957	0,125	0,951	0,964	0,959	0,122	0,953	0,966	0,0005	256	0,3
0,958	0,121	0,952	0,965	0,961	0,119	0,956	0,966	0,0001	256	0,3
0,960	0,118	0,955	0,965	0,959	0,127	0,959	0,960	0,01	512	0,3

0,956	0,129	0,950	0,962	0,959	0,123	0,943	0,976	0,001	512	0,3
0,957	0,125	0,952	0,963	0,961	0,118	0,952	0,970	0,0005	512	0,3
0,958	0,123	0,952	0,964	0,955	0,131	0,939	0,973	0,0001	512	0,3
0,959	0,119	0,953	0,965	0,960	0,121	0,945	0,976	0,01	64	0,35
0,959	0,117	0,954	0,965	0,958	0,122	0,943	0,974	0,001	64	0,35
0,960	0,116	0,954	0,966	0,961	0,118	0,949	0,974	0,0005	64	0,35
0,960	0,117	0,954	0,966	0,959	0,128	0,962	0,955	0,0001	64	0,35
0,963	0,106	0,957	0,968	0,963	0,119	0,956	0,970	0,01	128	0,35
0,956	0,130	0,949	0,963	0,954	0,141	0,961	0,946	0,001	128	0,35
0,959	0,120	0,952	0,965	0,961	0,130	0,972	0,949	0,0005	128	0,35
0,960	0,116	0,954	0,966	0,959	0,131	0,949	0,969	0,0001	128	0,35
0,948	0,156	0,939	0,958	0,953	0,143	0,941	0,967	0,01	256	0,35
0,960	0,116	0,954	0,967	0,961	0,129	0,965	0,956	0,001	256	0,35
0,960	0,118	0,955	0,965	0,959	0,127	0,959	0,960	0,0005	256	0,35
0,960	0,117	0,954	0,966	0,959	0,128	0,962	0,955	0,0001	256	0,35
0,960	0,116	0,954	0,966	0,959	0,131	0,949	0,969	0,01	512	0,35
0,961	0,112	0,956	0,966	0,957	0,125	0,940	0,975	0,001	512	0,35
0,961	0,112	0,955	0,966	0,960	0,127	0,966	0,952	0,0005	512	0,35
0,962	0,109	0,957	0,968	0,962	0,119	0,954	0,971	0,0001	512	0,35

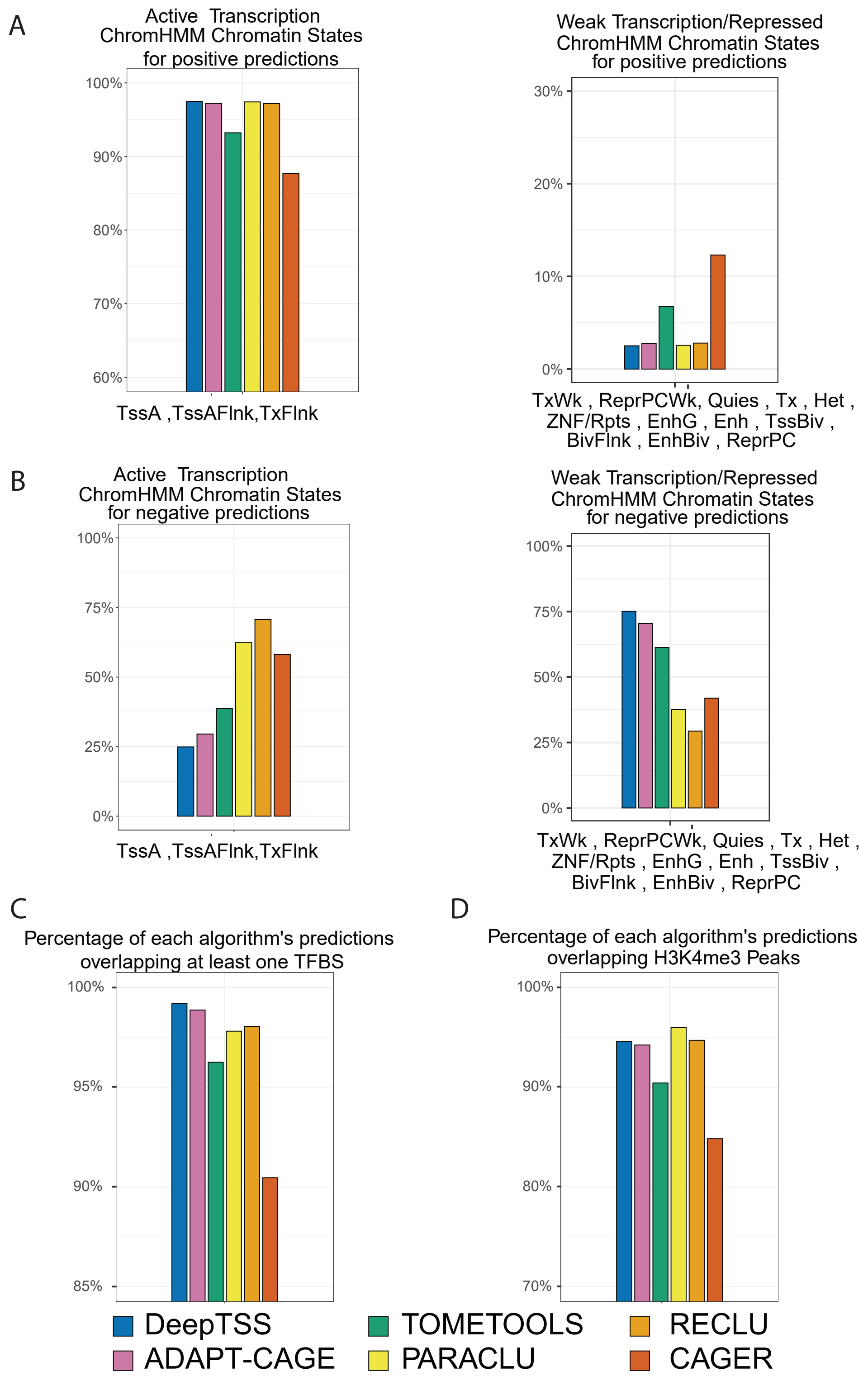

Figure 49: Evaluation results of all algorithms for chromosome 15 (supplement figure).

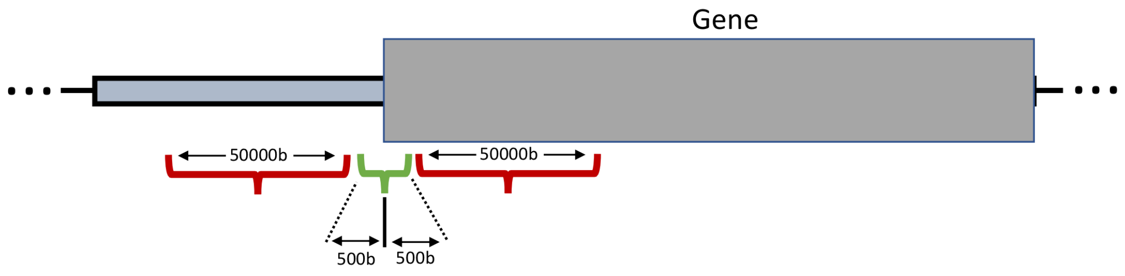

Figure 50: Positive and negative zones around annotated genes (supplement figure).

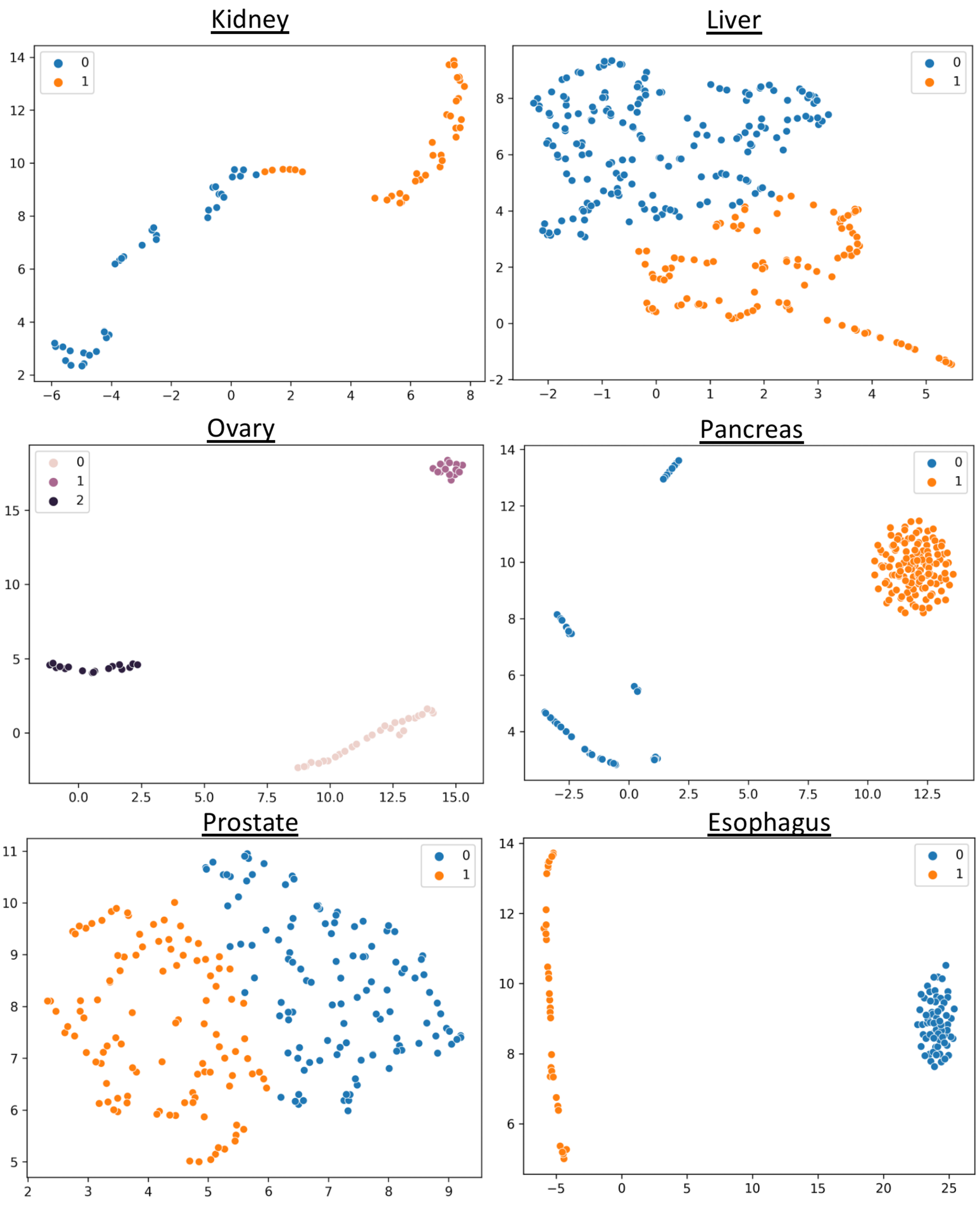

Figure 51: K-means clustering of tissues Kidney, Liver, Ovary, Pancreas, Prostate and Esophagus. (Supplement figure)

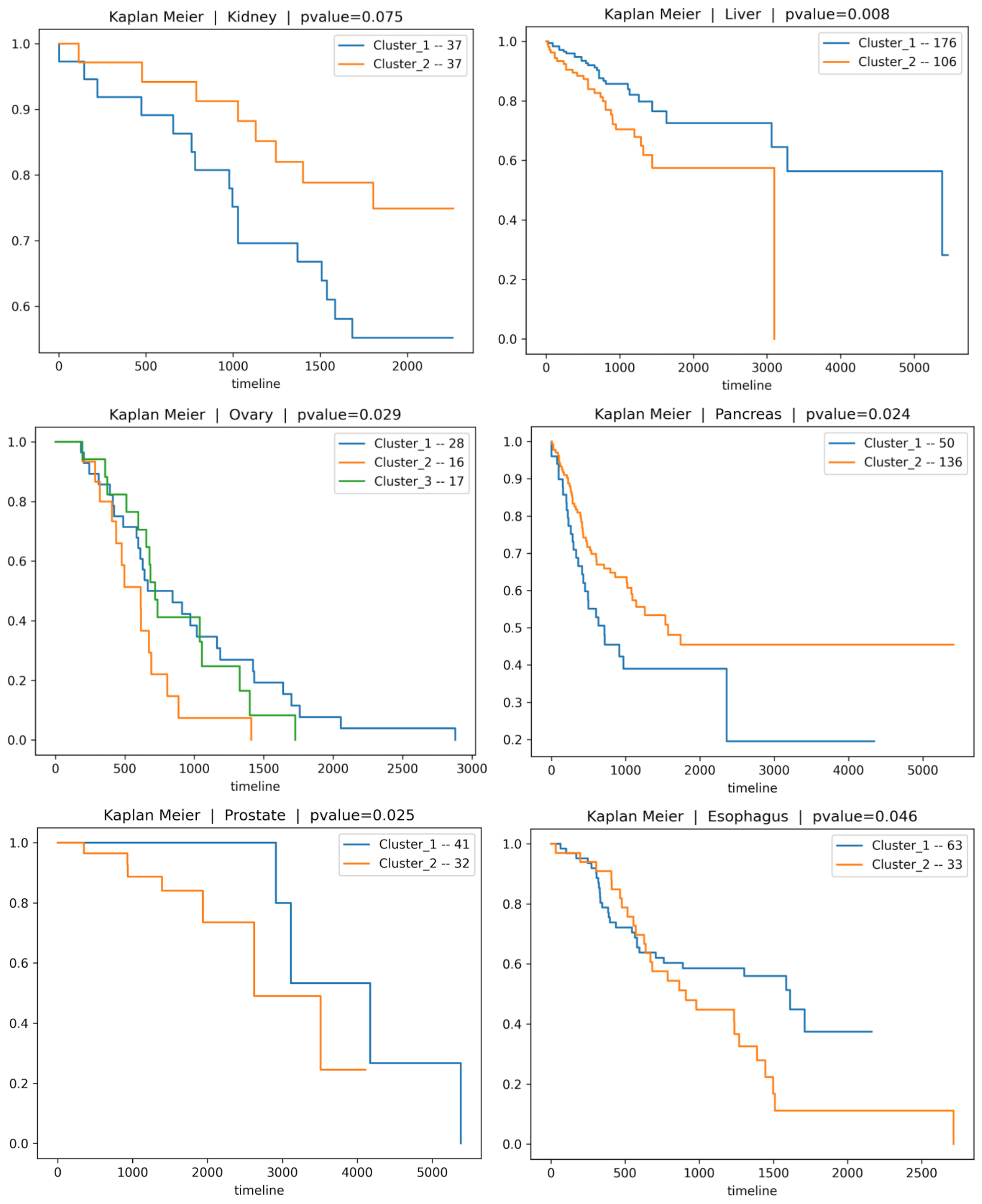

Figure 52: Kaplan-Meir curves of tissues Kidney, Liver, Ovary, Pancreas, Prostate and Esophagus. Ovary has 3 mortality levels, p value is created between groups 1 & 2. (Supplement figure)

Table 9:Mutational processes of each group in all tissues. For ovary group 1 and group 2 are used (highest and lowest risk groups).

Signature	Mutational process	Group 1	Group 2	Tissue
SBS1	Spontaneous deamination of 5-methylcytosine	1	2,98	Brain
SBS4	Tobacco smoking	0	58,86	Brain
SBS5	Aging / Tobacco smoking / NER deficiency	14,04	1,86	Brain
SBS36	BER deficiency	0	22,96	Brain
SBS49	Possible sequencing artefact	0	13,34	Brain
SBS50	Possible sequencing artefact	71,8	0	Brain
SBS60	Possible sequencing artefact	13,16	0	Brain

SBS1	Spontaneous deamination of 5-methylcytosine	8,28	4,82	Esophagus
SBS2	APOBEC activity	21,78	11,72	Esophagus
SBS3	Defective homologous recombination-based DNA damage repair [...]	0	46,06	Esophagus
SBS5	Aging / Tobacco smoking / NER deficiency	52,02	24,2	Esophagus
SBS13	APOBEC activity	16,5	13,2	Esophagus

SBS1	Spontaneous deamination of 5-methylcytosine	0,88	8,28	Kidney
SBS5	Aging / Tobacco smoking / NER deficiency	18,38	18,06	Kidney
SBS22	Aristolochic acid exposure	80,74	0	Kidney
SBS40	(Unknown) correlated with patients' ages for some types of human cancer	0	73,66	Kidney

SBS1	Spontaneous deamination of 5-methylcytosine	10,6	13,34	Ovary
SBS3	Defective homologous recombination-based DNA damage repair [...]	65,2	0	Ovary
SBS5	Aging / Tobacco smoking / NER deficiency	24,2	21,72	Ovary
SBS40	(Unknown) correlated with patients' ages for some types of human cancer	0	64,94	Ovary

SBS5	Spontaneous deamination of 5-methylcytosine	0,58	0	Pancreas
SBS36	BER deficiency (somatic MUTYH mutations)	87,86	100	Pancreas
SBS53	Possible sequencing artefact	11,56	0	Pancreas

SBS1	Spontaneous deamination of 5-methylcytosine	17,56	18,46	Prostate
SBS4	Tobacco smoking	0	71,98	Prostate
SBS5	Aging / Tobacco smoking / NER deficiency	18,82	51,38	Prostate
SBS15	Defective DNA mismatch repair (MMR)	13,26	12,18	Prostate
SBS40	(Unknown) correlated with patients’ ages for some types of human cancer	50,36	0	Prostate

SBS1	Spontaneous deamination of 5-methylcytosine	16,3	11,7	Liver
SBS5	Aging / Tobacco smoking / NER deficiency	39,24	31,74	Liver
SBS15	Defective DNA mismatch repair (MMR)	21,3	17,82	Liver
SBS19	Unknown	23,16	0	Liver
SBS21	DNA mismatch repair deficiency	0	15,82	Liver
SBS44	Defective DNA mismatch repair (MMR)	0	22,92	Liver

6. References

[1] G. Giannakakis, D. Grigoriadis, K. Giannakaki, O. Simantiraki, A. Roniotis, and M. Tsiknakis, “Review on psychological stress detection using biosignals,” *IEEE Trans. Affect. Comput.*, vol. 13, no. 1, pp. 440–460, Jan. 2022.

[2] M. B. Kursa, A. Jankowski, and W. R. Rudnicki, “Boruta--a system for feature selection,” *Fund. Inform.*, vol. 101, no. 4, pp. 271–285, 2010.

[3] T. M. Phuong, Z. Lin, and R. B. Altman, “Choosing SNPs using feature selection,” *J. Bioinform. Comput. Biol.*, vol. 4, no. 2, pp. 241–257, Apr. 2006.

[4] V. Fonti and E. Belitser, “Feature selection using lasso,” *VU Amsterdam research paper in business analytics*, vol. 30, pp. 1–25, 2017.

[5] G. C. McDonald, “Ridge regression,” *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 1, no. 1, pp. 93–100, Jul. 2009.

[6] Norvig and Intelligence, “A modern approach,” *Prentice Hall Upper Saddle River, NJ*, 2002.

[7] L. P. Kaelbling, M. L. Littman, and A. W. Moore, “Reinforcement Learning: A Survey,” *jair*, vol. 4, pp. 237–285, May 1996.

[8] B. J. Pandian and M. M. Noel, “Control of a bioreactor using a new partially supervised reinforcement learning algorithm,” *J. Process Control*, vol. 69, pp. 16–29, Sep. 2018.

[9] Hinton and Roweis, “Stochastic neighbor embedding,” *information processing systems*, 2002.

[10] R. Xu and D. Wunsch 2nd, “Survey of clustering algorithms,” *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, May 2005.

[11] H. E. Driver and A. L. Kroeber, *Quantitative expression of cultural relationships*, vol. 31. Berkeley: University of California Press, 1932.

[12] C. Cortes and V. Vapnik, “Support-vector networks,” *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[13] D. Grigoriadis, N. Perdikopanis, G. K. Georgakilas, and A. Hatzigeorgiou, “DiS-TSS: An Annotation Agnostic Algorithm for TSS Identification,” in *Bioinformatics and Biomedical Engineering*, 2020, pp. 613–623.

[14] J. H. Friedman, “Greedy function approximation: A gradient boosting machine,” *Ann. Stat.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.

[15] T. K. Ho, “Random decision forests,” in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, Montreal, Que., Canada, 2002.

[16] T. K. Ho, “The random subspace method for constructing decision forests,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, 1998.

[17] E. M. Kleinberg, “On the algorithmic implementation of stochastic discrimination,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 5, pp. 473–490, May 2000.

[18] E. ;. Fix and J. L. Hodges, *Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties (PDF) (Report). USAF School of Aviation Medici*. 1951.

[19] D. A. Freedman, *Statistical models: theory and practice*. cambridge university press, 2009.

[20] J. Tolles and W. J. Meurer, “Logistic Regression: Relating Patient Characteristics to Outcomes,” *JAMA: the journal of the American Medical Association*, vol. 316, no. 5. jamanetwork.com, pp. 533–534, 02-Aug-2016.

[21] J. R. Quinlan, “Induction of decision trees,” *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, Mar. 1986.

[22] D. H. Moore, “Classification and regression trees, by Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. Brooks/Cole Publishing, Monterey, 1984,358 pages, $27.95,” *Cytometry*, vol. 8, no. 5, pp. 534–535, Sep. 1987.

[23] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1943.

[24] J. Hopfield, “Hopfield network,” *Scholarpedia J.*, vol. 2, no. 5, p. 1977, 2007.

[25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[26] H. Sak, A. W. Senior, and F. Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” 2014.

[27] J. Xiao and Z. Zhou, “Research progress of RNN language model,” in *2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, Dalian, China, 2020.

[28] G. Bebis and M. Georgiopoulos, “Feed-forward neural networks,” *IEEE Potentials*, vol. 13, no. 4, pp. 27–31, Oct. 1994.

[29] D. L. Prados and S. C. Kak, “Neural network capacity using delta rule,” *Electron. Lett.*, vol. 25, no. 3, p. 197, 1989.

[30] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[31] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, “A novel connectionist system for unconstrained handwriting recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 855–868, May 2009.

[32] M. V. Valueva, N. N. Nagornov, P. A. Lyakhov, G. V. Valuev, and N. I. Chervyakov, “Application of the residue number system to reduce hardware costs of the convolutional neural network implementation,” *Math. Comput. Simul.*, vol. 177, pp. 232–243, Nov. 2020.

[33] I. Tabian, H. Fu, and Z. S. Khodaei, “A convolutional neural network for impact detection and characterization of complex composite structures,” *Sensors (Basel)*, vol. 19, no. 22, p. 4933, Nov. 2019.

[34] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[35] T. Wolf *et al.*, “Transformers: State-of-the-art natural language processing,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online, 2020.

[36] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

[37] J. Jumper *et al.*, “Highly accurate protein structure prediction with AlphaFold,” *Nature*, vol. 596, no. 7873, pp. 583–589, Aug. 2021.

[38] D. Silver *et al.*, “Mastering the game of Go without human knowledge,” *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.

[39] H. Kashyap, H. A. Ahmed, N. Hoque, S. Roy, and D. K. Bhattacharyya, “Big data analytics in bioinformatics: architectures, techniques, tools and issues,” *Network Modeling Analysis in Health Informatics and Bioinformatics*, vol. 5, no. 1, p. 28, Sep. 2016.

[40] J. D. Watson and F. H. Crick, “Molecular structure of nucleic acids; a structure for deoxyribose nucleic acid,” *Nature*, vol. 171, no. 4356, pp. 737–738, Apr. 1953.

[41] P. Cramer, “Rosalind Franklin and the advent of molecular biology,” *Cell*, vol. 182, no. 4, pp. 787–789, Aug. 2020.

[42] M. Z. Barciszewska, P. M. Perrigue, and J. Barciszewski, “tRNA--the golden standard in molecular biology,” *Mol. Biosyst.*, vol. 12, no. 1, pp. 12–17, Jan. 2016.

[43] F. Sanger, S. Nicklen, and A. R. Coulson, “DNA sequencing with chain-terminating inhibitors. 1977,” *Biotechnology*, vol. 24, pp. 104–108, 1992.

[44] Z. G. Chidgeavadze, R. S. Beabealashvilli, A. M. Atrazhev, M. K. Kukhanova, A. V. Azhayev, and A. A. Krayevsky, “2′, 3′ -Dideoxy-3′ amlnonudeo 5′ .triphosphates are the terminators of DNA synthesis catalyzed by DNA polymerases, ” *Nucleic Acids Res.*, vol. 12, no. 3, pp. 1671‐1686, 1984.

[45] J. Shendure *et al.*, “Accurate multiplex polony sequencing of an evolved bacterial genome,” *Science*, vol. 309, no. 5741, pp. 1728–1732, Sep. 2005.

[46] L. Liu *et al.*, “Comparison of next-generation sequencing systems,” *J. Biomed. Biotechnol.*, vol. 2012, p. 251364, Jul. 2012.

[47] I. Braslavsky, B. Hebert, E. Kartalov, and S. R. Quake, “Sequence information can be obtained from single DNA molecules,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 100, no. 7, pp. 3960–3964, Apr. 2003.

[48] T. Shiraki *et al.*, “Cap analysis gene expression for high-throughput analysis of transcriptional starting point and identification of promoter usage,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 100, no. 26, pp. 15776–15781, Dec. 2003.

[49] C. Plessy *et al.*, “Linking promoters to functional transcripts in small samples with nanoCAGE and CAGEscan,” *Nat. Methods*, vol. 7, no. 7, pp. 528–534, Jul. 2010.

[50] R. Kodzius *et al.*, “CAGE: cap analysis of gene expression,” *Nat. Methods*, vol. 3, no. 3, pp. 211–222, Mar. 2006.

[51] A. M. Maxam and W. Gilbert, “A new method for sequencing DNA. 1977,” *Biotechnology*, vol. 24, pp. 99–103, 1992.

[52] F. Sanger *et al.*, “Nucleotide sequence of bacteriophage ϕX174 DNA,” *Nature*, vol. 265, no. 5596, pp. 687–695, Feb. 1977.

[53] I. Dunham *et al.*, “The DNA sequence of human chromosome 22,” *Nature*, vol. 402, no. 6761, pp. 489–495, Dec. 1999.

[54] M. D. Adams *et al.*, “The genome sequence of Drosophila melanogaster,” *Science*, vol. 287, no. 5461, pp. 2185–2195, Mar. 2000.

[55] Arabidopsis Genome Initiative, “Analysis of the genome sequence of the flowering plant Arabidopsis thaliana,” *Nature*, vol. 408, no. 6814, pp. 796–815, Dec. 2000.

[56] J. C. Venter *et al.*, “The sequence of the human genome,” *Science*, vol. 291, no. 5507, pp. 1304–1351, Feb. 2001.

[57] Mouse Genome Sequencing Consortium *et al.*, “Initial sequencing and comparative analysis of the mouse genome,” *Nature*, vol. 420, no. 6915, pp. 520–562, Dec. 2002.

[58] C. Gilissen *et al.*, “Genome sequencing identifies major causes of severe intellectual disability,” *Nature*, vol. 511, no. 7509, pp. 344–347, Jul. 2014.

[59] T. H. Kim and J. Dekker, “Formaldehyde cross-linking,” *Cold Spring Harb. Protoc.*, vol. 2018, no. 4, p. db.prot082594, Apr. 2018.

[60] T. H. Kim and J. Dekker, “Preparation of cross-linked chromatin for ChIP,” *Cold Spring Harb. Protoc.*, vol. 2018, no. 4, p. db.prot082602, Apr. 2018.

[61] P. J. Park, “ChIP-seq: advantages and challenges of a maturing technology,” *Nat. Rev. Genet.*, vol. 10, no. 10, pp. 669–680, Oct. 2009.

[62] Y. Chu and D. R. Corey, “RNA sequencing: platform selection, experimental design, and data interpretation,” *Nucleic Acid Ther.*, vol. 22, no. 4, pp. 271–274, Aug. 2012.

[63] C. A. Maher, C. Kumar-Sinha, X. Cao, S. Kalyana-Sundaram, B. Han, and X. Jing, “Transcriptome sequencing to detect gene fusions in cancer,” *Bibcode:2009Natur.458...97M*, vol. 458, pp. 97–101, 2009.

[64] M. Griffith, J. R. Walker, N. C. Spies, B. J. Ainscough, and O. L. Griffith, “Informatics for RNA sequencing: A web resource for analysis on the cloud,” *PLoS Comput. Biol.*, vol. 11, no. 8, p. e1004393, Aug. 2015.

[65] R. Morin *et al.*, “Profiling the HeLa S3 transcriptome using randomly primed cDNA and massively parallel short-read sequencing,” *Biotechniques*, vol. 45, no. 1, pp. 81–94, Jul. 2008.

[66] Q. Sun, Q. Hao, and K. V. Prasanth, “Nuclear long noncoding RNAs: Key regulators of gene expression,” *Trends Genet.*, vol. 34, no. 2, pp. 142–157, Feb. 2018.

[67] B. Sigurgeirsson, O. Emanuelsson, and J. Lundeberg, “Sequencing degraded RNA addressed by 3’ tag counting,” *PLoS One*, vol. 9, no. 3, p. e91851, Mar. 2014.

[68] P. Moll, M. Ante, A. Seitz, and T. Reda, “QuantSeq 3′ mRNA sequencing for RNA quantification,” *Nat. Methods*, vol. 11, no. 12, pp. i–iii, Nov. 2014.

[69] A. Mortazavi, B. A. Williams, K. McCue, L. Schaeffer, and B. Wold, “Mapping and quantifying mammalian transcriptomes by RNA-Seq,” *Nat. Methods*, vol. 5, no. 7, pp. 621–628, Jul. 2008.

[70] K. R. Kukurba and S. B. Montgomery, “RNA sequencing and analysis,” *Cold Spring Harb. Protoc.*, vol. 2015, no. 11, pp. 951–969, Apr. 2015.

[71] A. P. Boyle *et al.*, “High-resolution mapping and characterization of open chromatin across the genome,” *Cell*, vol. 132, no. 2, pp. 311–322, Jan. 2008.

[72] O. Chumsakul *et al.*, “High-resolution mapping of in vivo genomic transcription factor binding sites using in situ DNase I footprinting and ChIP-seq,” *DNA Res.*, vol. 20, no. 4, pp. 325–338, Aug. 2013.

[73] G. Ling and D. J. Waxman, “Isolation of nuclei for use in genome-wide DNase hypersensitivity assays to probe chromatin structure,” *Methods Mol. Biol.*, vol. 977, pp. 13–19, 2013.

[74] N. Perdikopanis *et al.*, “DIANA-miRGen v4: indexing promoters and regulators for more than 1500 microRNAs,” *Nucleic Acids Res.*, vol. 49, no. D1, pp. D151–D159, Jan. 2021.

[75] H. Ohmiya *et al.*, “RECLU: a pipeline to discover reproducible transcriptional start sites and their alternative regulation using capped analysis of gene expression (CAGE),” *BMC Genomics*, vol. 15, p. 269, Apr. 2014.

[76] M. C. Frith, E. Valen, A. Krogh, Y. Hayashizaki, P. Carninci, and A. Sandelin, “A code for transcription initiation in mammalian genomes,” *Genome Res.*, vol. 18, no. 1, pp. 1–12, Jan. 2008.

[77] V. Haberle, A. R. R. Forrest, Y. Hayashizaki, P. Carninci, and B. Lenhard, “CAGEr: precise TSS data retrieval and high-resolution promoterome mining for integrative analyses,” *Nucleic Acids Res.*, vol. 43, no. 8, p. e51, Apr. 2015.

[78] G. K. Georgakilas, N. Perdikopanis, and A. Hatzigeorgiou, “Solving the transcription start site identification problem with ADAPT-CAGE: a Machine Learning algorithm for the analysis of CAGE data,” *Sci. Rep.*, vol. 10, no. 1, p. 877, Jan. 2020.

[79] M. E. Samuels and J. M. Friedman, “Genetic mosaics and the germ line lineage,” *Genes (Basel)*, vol. 6, no. 2, pp. 216–237, Apr. 2015.

[80] I. Martincorena and P. J. Campbell, “Somatic mutation in cancer and normal cells,” *Science*, vol. 349, no. 6255, pp. 1483–1489, Sep. 2015.

[81] M. H. Bailey *et al.*, “Comprehensive characterization of cancer driver genes and mutations,” *Cell*, vol. 174, no. 4, pp. 1034–1035, Aug. 2018.

[82] E. Rheinbay *et al.*, “Analyses of non-coding somatic drivers in 2,658 cancer whole genomes,” *Nature*, vol. 578, no. 7793, pp. 102–111, Feb. 2020.

[83] W. Jiao *et al.*, “A deep learning system accurately classifies primary and metastatic cancers using passenger mutation patterns,” *Nat. Commun.*, vol. 11, no. 1, p. 728, Feb. 2020.

[84] S. A. Forbes *et al.*, “COSMIC: somatic cancer genetics at high-resolution,” *Nucleic Acids Res.*, vol. 45, no. D1, pp. D777–D783, Jan. 2017.

[85] L. B. Alexandrov *et al.*, “The repertoire of mutational signatures in human cancer,” *Nature*, vol. 578, no. 7793, pp. 94–101, Feb. 2020.

[86] D. D. Lee and H. S. Seung, “Learning the parts of objects by non-negative matrix factorization,” *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999.

[87] L. B. Alexandrov, S. Nik-Zainal, D. C. Wedge, P. J. Campbell, and M. R. Stratton, “Deciphering signatures of mutational processes operative in human cancer,” *Cell Rep.*, vol. 3, no. 1, pp. 246–259, Jan. 2013.

[88] L. B. Alexandrov *et al.*, “Clock-like mutational processes in human somatic cells,” *Nat. Genet.*, vol. 47, no. 12, pp. 1402–1407, Dec. 2015.

[89] J. E. Kucab *et al.*, “A compendium of mutational signatures of environmental agents,” *Cell*, vol. 177, no. 4, pp. 821-836.e16, May 2019.

[90] ICGC/TCGA Pan-Cancer Analysis of Whole Genomes Consortium, “Pan-cancer analysis of whole genomes,” *Nature*, vol. 578, no. 7793, pp. 82–93, Feb. 2020.

[91] S. A. Forbes *et al.*, “COSMIC: exploring the world’s knowledge of somatic mutations in human cancer,” *Nucleic Acids Res.*, vol. 43, no. Database issue, pp. D805-11, Jan. 2015.

[92] L. Ségurel and C. Bon, “On the evolution of lactase persistence in humans,” *Annu. Rev. Genomics Hum. Genet.*, vol. 18, pp. 297–319, Aug. 2017.

[93] D. Hughes and D. I. Andersson, “Evolutionary trajectories to antibiotic resistance,” *Annu. Rev. Microbiol.*, vol. 71, pp. 579–596, Sep. 2017.

[94] R. Taylor Raborn, V. P. Brendel, and K. Sridharan, “TSRchitect: Promoter identification from large-scale TSS profiling data.” .

[95] Q. Li, J. B. Brown, H. Huang, and P. J. Bickel, “Measuring reproducibility of high-throughput experiments,” *Ann. Appl. Stat.*, vol. 5, no. 3, pp. 1752–1779, Sep. 2011.

[96] D. N. Grigoriadis, “NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS SCHOOL OF SCIENCE DEPARTMENT OF INFORMATICS & TELECOMMUNICATIONS POSTGRADUATE PROGRAM ‘INFORMATION TECHNOLOGIES IN MEDICINE AND BIOLOGY’ MASTER THESIS A method for identifying TSS from CAGE data using a genomic signal processing approach,” 2019. [Online]. Available: http://itmb.di.uoa.gr/bulletin/Grigoriadis.pdf. [Accessed: 22-Dec-2022].

[97] D. P. Morgan and C. L. Scofield, “Signal Processing and Feature Extraction,” in *Neural Networks and Speech Processing*, D. P. Morgan and C. L. Scofield, Eds. Boston, MA: Springer US, 1991, pp. 163–201.

[98] R. Telgarsky, “Dominant Frequency Extraction,” *arXiv [cs.NA]*, 01-Jun-2013.

[99] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, “Gene Selection for Cancer Classification using Support Vector Machines,” *Mach. Learn.*, vol. 46, no. 1, pp. 389–422, Jan. 2002.

[100] B. E. Bernstein *et al.*, “Methylation of histone H3 Lys 4 in coding regions of active genes,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 99, no. 13, pp. 8695–8700, Jun. 2002.

[101] H. Santos-Rosa *et al.*, “Active genes are tri-methylated at K4 of histone H3,” *Nature*, vol. 419, no. 6905, pp. 407–411, Sep. 2002.

[102] M. G. Guenther, S. S. Levine, L. A. Boyer, R. Jaenisch, and R. A. Young, “A chromatin landmark and transcription initiation at most promoters in human cells,” *Cell*, vol. 130, no. 1, pp. 77–88, Jul. 2007.

[103] ENCODE Project Consortium, “An integrated encyclopedia of DNA elements in the human genome,” *Nature*, vol. 489, no. 7414, pp. 57–74, Sep. 2012.

[104] FANTOM Consortium and the RIKEN PMI and CLST (DGT) *et al.*, “A promoter-level mammalian expression atlas,” *Nature*, vol. 507, no. 7493, pp. 462–470, Mar. 2014.

[105] D. Grigoriadis, N. Perdikopanis, G. K. Georgakilas, and A. G. Hatzigeorgiou, “DeepTSS: multi-branch convolutional neural network for transcription start site identification from CAGE data,” *BMC Bioinformatics*, vol. 23, no. Suppl 2, p. 395, Dec. 2022.

[106] K. S. Pollard, M. J. Hubisz, K. R. Rosenbloom, and A. Siepel, “Detection of nonneutral substitution rates on mammalian phylogenies,” *Genome Res.*, vol. 20, no. 1, pp. 110–121, Jan. 2010.

[107] G. K. Georgakilas, A. Grioni, K. G. Liakos, E. Chalupova, F. C. Plessas, and P. Alexiou, “Multi-branch Convolutional Neural Network for Identification of Small Non-coding RNA genomic loci,” *Sci. Rep.*, vol. 10, no. 1, p. 9486, Jun. 2020.

[108] D. R. Zerbino *et al.*, “Ensembl 2018,” *Nucleic Acids Res.*, vol. 46, no. D1, pp. D754–D761, 2017.

[109] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier nonlinearities improve neural network acoustic models,” 2013. [Online]. Available: http://robotics.stanford.edu/~amaas/papers/relu_hybrid_icml2013_final.pdf. [Accessed: 15-Nov-2022].

[110] R. Zhang and C. T. Zhang, “Z curves, an intutive tool for visualizing and analyzing the DNA sequences,” *J. Biomol. Struct. Dyn.*, vol. 11, no. 4, pp. 767–782, Feb. 1994.

[111] J. A. Berger, S. K. Mitra, M. Carli, and A. Neri, “Visualization and analysis of DNA sequences using DNA walks,” *J. Franklin Inst.*, vol. 341, no. 1, pp. 37–53, Jan. 2004.

[112] M. Akhtar, J. Epps, and E. Ambikairajah, “On DNA Numerical Representations for Period-3 Based Exon Prediction,” in *2007 IEEE International Workshop on Genomic Signal Processing and Statistics*, 2007, pp. 1–4.

[113] B. D. Silverman and R. Linsker, “A measure of DNA periodicity,” *J. Theor. Biol.*, vol. 118, no. 3, pp. 295–300, Feb. 1986.

[114] M. A. el Hassan and C. R. Calladine, “Propeller-twisting of base-pairs and the conformational mobility of dinucleotide steps in DNA,” *J. Mol. Biol.*, vol. 259, no. 1, pp. 95–103, May 1996.

[115] I. Brukner, R. Sánchez, D. Suck, and S. Pongor, “Trinucleotide models for DNA bending propensity: comparison of models based on DNaseI digestion and nucleosome packaging data,” *J. Biomol. Struct. Dyn.*, vol. 13, no. 2, pp. 309–317, Oct. 1995.

[116] D. W. Ussery, “DNA Structure: A-, B-and Z-DNA Helix Families,” *e LS*, 2001.

[117] Y. Gan, J. Guan, and S. Zhou, “A comparison study on feature selection of DNA structural properties for promoter prediction,” *BMC Bioinformatics*, vol. 13, p. 4, Jan. 2012.

[118] J. Ernst and M. Kellis, “ChromHMM: automating chromatin-state discovery and characterization,” *Nat. Methods*, vol. 9, no. 3, pp. 215–216, Feb. 2012.

[119] A. Reyes and W. Huber, “Alternative start and termination sites of transcription drive most transcript isoform differences across human tissues,” *Nucleic Acids Res.*, vol. 46, no. 2, pp. 582–592, Jan. 2018.

[120] A. Sendoel *et al.*, “Translation from unconventional 5’ start sites drives tumour initiation,” *Nature*, vol. 541, no. 7638, pp. 494–499, Jan. 2017.

[121] S. A. Dugger, A. Platt, and D. B. Goldstein, “Drug development in the era of precision medicine,” *Nat. Rev. Drug Discov.*, vol. 17, no. 3, pp. 183–196, Mar. 2018.

[122] E. N. Bergstrom *et al.*, “SigProfilerMatrixGenerator: a tool for visualizing and exploring patterns of small mutational events,” *BMC Genomics*, vol. 20, no. 1, p. 685, Aug. 2019.

[123] S. M. A. Islam *et al.*, “Uncovering novel mutational signatures by de novo extraction with SigProfilerExtractor,” *Cell Genomics*, vol. 2, no. 11, p. 100179, Nov. 2022.

[124] J. Ranstam and J. A. Cook, “Kaplan–Meier curve,” *Br. J. Surg.*, vol. 104, no. 4, pp. 442–442, Feb. 2017.