A Convolutional Deep Learning Approach to identify DNA Sequences for Gene Prediction

Jesús Antonio Motta[1] , Pedro David Gómez[2]

[1]jesus-a.motta.1@ulaval.ca – Laval University (Canada), [2]pdgomez@fucsalud.edu.co – FUCS

Abstract

In this work, we present a highly efficient machine learning method for identifying DNA sequences that code for genes. The learning process is based on Human Genome Build 38 (GRCh38) sequences extracted from various specialized databases. The sequences were then translated into amino acid sequences and used to build matrices that facilitate the extraction of features with the TF*IDF metric for the creation of the training space. The prediction functions are learned using a convolutional neural network (CNN) deep learning model. The training spaces were created using the 24 chromosomes of the human genome and approximately 36,000 genes and pseudogenes whose names were fetched from the HUGO Gene Nomenclature Committee (HGNC). Performance analysis was performed on 24 genes associated with genetic disorders, as well as the surrounding DNA regions. The metrics used were precision, recall, F_score measure, accuracy and ROC curves for the genes of interest. The results achieved exceed all our expectations and place the work at the level of the state of the art for gene prediction.

Keywords: DNA, Amino-Acids, TF×IDF, CNN, Genetic Disorder, Learning Model

1. Introduction

The development of efficient models for gene prediction is a task that presents numerous challenges as well as the development of complex processes that generally consume large resources. It is therefore necessary to undertake certain actions to address problems such as the analysis of very large genomic sequences to identify functional elements such as protein-coding regions and their regulatory regions. For example, in a eukaryotic organism, different situations arise: numerous non-coding regions (introns), a single gene can produce multiple proteins (alternative splicing), several repeated sequences, and the fact that genes evolve differently depending on the species or human groups (allelic frequencies) among others. Let's think for example about some species that live in different habitats or other organisms, they are subject to selective processes that necessarily determine the evolution of a gene (evolutionary genomics) over time [2,8,17]

After the complete sequencing of the human genome was completed in 2003 [4], gene prediction was significantly enhanced, primarily by having a complete genome annotation and the introduction of new techniques based on machine learning. More recently, the use of deep learning-based algorithms running on GPUs has been a key factor in the development of gene prediction.

In this work, we present an efficient method to predict genes from a DNA sequence using the Convolutional Neural Networks learning method and matrices created with the TF*IDF metric from Amino-Acids sequences.

Some methods and techniques for gene prediction

In this part of our work, some developments (in the area of gene prediction) are presented, according to a classification that is related to the way in which DNA sequences are approached and treated for processing.

a. Ab Initio Methods

The methods based on this approach are characterized by the fact that their analysis focuses solely on the DNA sequence itself. Below we mention some important tools based in this approach that are currently used:

GENSCAN

Is a tool developed at Stanford University [5,6], that uses Generalized Hidden Markov Model (GHMM) to identify genes. Other important works that describe and demonstrate the use of HMM for gene prediction can be found in [20,23].

AUGUSTUS

Is used to identify genes in **eukaryotic genomic sequences. It** is trained on over 100 species [31,32]. It is possible to improve predictions by adding RNA-Seq data and protein alignments.

GeneMark

Used to identify protein-coding regions in DNA sequences (genes or gene parts). To analyze the sequences, it uses statistical methods such as Inhomogeneous Three-Periodic Markov Models [3,19]**.** This method has been widely used in both prokaryotic genomes (bacteria, archaea) and eukaryotic genomes (plants, animals, fungi), as well as in metagenomes (mixed microbial communities) and transcriptomes (RNA sequences).

b. Similarity Based

This approach works by aligning an unknown DNA sequence with other known gene or protein sequences that are retrieved from large databases. If there is a good alignment (similarity), the gene or protein is marked as a candidate gene.

Some methods based on this approach are:

BLAST (Basic Local Alignment Search Tool)

Uses heuristic algorithms to find similarities between an input sequence (DNA, RNA, or proteins) and other sequences in a database. It is one of the best-known tools of this genre [1,7]

Exonerate

This method has two ways to traverse the state space: heuristic and exhaustive. It is a good tool to identify genes and its locations by aligning **cDNA or mRNA sequences,** Protein sequences and **DNA to DNA sequences** [30]

PhyloCSF (Phylogenetic Codon Substitution Frequencies)

This method identifies coded and non-coded regions in a DNA sequence, comparing multiple sequences from genomes of different species and analyzing the probability that a sequence has evolved toward a coded or non-coded region [18]. Although this method is not exactly a tool for predicting a gene from a biological sequence (DNA, RNA, etc.), it is a very effective aid in the task of annotating non-coding functional elements. Thus, it could be of great help in the investigation of key regulatory elements of the genome.

c. Unsupervised Machine Learning

In this category we will only mention clustering-based algorithms, taking into account that almost all of the algorithms mentioned in the previous categories use unsupervised methods for their prediction kernels. For example, GenMark uses Markov models, GenScan uses Hidden Markov Models (HMM), Augustus is based on GHMM (Generalized Hidden Markov Models), and others.

Clustering

Clustering groups similar DNA or RNA sequences together based on shared patterns.

Some of the main types of clustering used are: Hierarchical Clustering [35], which builds a tree of clusters; K-means [36], which partitions data into k similar clusters; and Alignment-Free Method, which creates clusters from k-mer frequencies [10].

d. Supervised Machine Learning

In this section of the study, we present the most recent works based on supervised machine learning to predict genes:

In [34], different methods are compared: *regularized regression, instance-based*, *ensemble* and *deep learning* methods. It concludes that their performance is very similar for the particular data set used. In the work presented in [37], a model for the prediction of gene expression is built, using methylation data. It utilizes **adaptive convolutional layers** [38] and **residual blocks** [13] to handle high-dimensional methylation. In [14], a review of supervised and unsupervised learning methods and deep learning is presented to infer gene networks from genes, proteins, or metabolites (molecules that are produced or used during metabolism). Spectral clustering data (eigenvalues and eigenvectors are extracted from data) is used in [26] for feature extraction and to build a **learning model** to predict gene-function associations.

2. Design and Construction of the Model

For the design, implementation, and construction of the prediction functions, all 24 chromosomes of the human genome and a total of approximately 36,000 genes and pseudogenes were considered. The genomic sequences corresponding to these genes and pseudogenes were fetched from the hg38 (GRCh38) builds [21], which are resident in the databases maintained by NCBI [39], Ensembl [40], UCSC [24] and Uniprot [33].

In our study, we were particularly interested in 24 genes (from hundreds of cases) linked to certain diseases due to mutations in the DNA sequence (single gene mutation). In Tables 4-5 we present the list of these genes, which includes a brief description of their function, the effect of the mutation and the chromosome to which they belong. As we have said, the names of the genes (approved symbols) are fetched from HGNC (HUGO Gene Nomenclature Committee) [4,41], which is the entity responsible for assigning symbols and names to human genes. In Table 5, we show the performance of predicting genes from chromosomes and segments (partitions) that contain these genes.

We have divided the design and construction process of our model into the following 7 stages: *DNA sequence fetching, cleaning, partitioning, feature engineering (rebuilding features), training and test set selection, function induction and performance measurement*.

2.A Sequence Fetching

The DNA sequences that form the training space for the learning functions are fetched from reputable specialized databases. We have taken approximately 36,000 gene-coding sequences from each of the 24 chromosomes of the human genome from a list provided by HGNC [41]. The accessed databases were NCBI [21], Ensembl [40], and UCSC [42]

2.B Cleaning

Sequences fetched from databases undergo a cleansing process. It begins by verifying their existence and format, as well as the standardization of upper or lower case. Then, blanks, special characters, numeric characters, and ambiguous bases are removed (only A, T, G, and C are considered).

2.C Partitioning

Considering the size of the training space (DNA sequences of 24 chromosomes containing code for approximately 36,000 genes and pseudogenes) and to facilitate efficiency in the different tasks involved, we have decided to use a "divide and conquer" approach: each chromosome *i* is divided into *j* partitions

containing its corresponding genes $g_{i,j}$. Thus, for example, chromosome 1 (which is the largest human chromosome), containing approximately 3500 genes and pseudogenes, has been divided into 12 partitions of approximately 292 genes each. Thus, if **nc** = number of chromosomes and **np** = number of partitions, then the total number of genes in the training space is:

$$\sum_{i=1}^{nc}\sum_{j=1}^{np} g_{i,j}$$

2.D Feature Engineering

The feature engineering stage consists of the following sub-stages:

2.D.1 ORF identification. An ORF (*Open Reading Frame*) is a stretch of DNA that has the potential to create a protein. This sequence always begins with the ATG (methionine) codon and ends with one of these three codons: TAG, TGA, TAA. The National Human Genome Research Institute (NHGRI) defines an ORF as a stretch of DNA that when transcribed into RNA, has no stop codon [43]

Considering that the presence of ORFs in a DNA sequence could indicate the existence of a gene (because it contains coding for amino acids), in our research, we will use ORFs in a different approach. It is not strictly a matter of identifying the presence of a gene in the DNA sequence. We have fetched the genes of each chromosome based on the exact chromosomal location, a single accession ID, or by indicating the range of its location in the DNA sequence that contains it, ensuring that the direction is 5' to 3'.

To identify the ORFs, the following steps are performed:

a. Identification/Extraction of Codons (triplets of nucleotides)

All DNA-based genomes contain 64 types of triplets (codons), created from 4 nucleotides A, T, G, C ($4^3 = 64$). This would result in 61 codons encoding 20 amino acids and 3 codons encoding stop signals (TAG, TGA, TAA). Each gene sequence is examined, and its codons are identified.

b. Building Reading Frames.

The codon sequence is converted into 6 different reading frames (rf): 3 on the forward strand (+1, +2, +3), and 3 on the reverse complement (-1, -2, -3) as shown below: Starting for example with the sequence:

'ATGCGTACGTAGCTAGCTAAATGCGTGA ATGCGTACGTAGCTAGCTAAATGCGTGA', then: **Frame 1**: is the starting sequence with its identified codons: ['ATG', 'CGT', 'ACG', 'TAG', 'CTA', 'GCT', 'AAA', 'TGC', 'GTG', 'AAT', 'GCG', 'TAC', 'GTA', 'GCT', 'AGC', 'TAA', 'ATG', 'CGT']

Frame 2: is formed from the second nucleotide, leaving the first and then grouping again 3 by 3: ['TGC', 'GTA', 'CGT', 'AGC', 'TAG', 'CTA', 'AAT', 'GCG', 'TGA', 'ATG', 'CGT', 'ACG', 'TAG', 'CTA', 'GCT', 'AAA', 'TGC', 'GTG']

Frame 3: is formed from the third nucleotide, leaving the first 2 and then grouping again 3 by 3: ['GCG', 'TAC', 'GTA', 'GCT', 'AGC', 'TAA', 'ATG', 'CGT', 'GAA', 'TGC', 'GTA', 'CGT', 'AGC', 'TAG', 'CTA', 'AAT', 'GCG', 'TGA'

Reverse complement:

'TCACGCATTTAGCTAGCTACGTACGCATT CACGCATTTAGCTAGCTACGTACGCAT'

Now, the same processes of frames 1, 2 and 3 are applied:

Frame -1: ['TCA', 'CGC', 'ATT', 'TAG', 'CTA', 'GCT', 'ACG', 'TAC', GCA', 'TTC', 'ACG', 'CAT', 'TTA', 'GCT', 'AGC', 'TAC', 'GTA', 'CGC']

Frame -2: ['CAC', 'GCA', 'TTT', 'AGC', 'TAG', 'CTA', 'CGT', 'ACG', 'CAT', 'TCA', 'CGC', 'ATT', 'TAG', 'CTA', 'GCT', 'ACG', 'TAC', 'GCA']

Frame -3: ['ACG', 'CAT', 'TTA', 'GCT', 'AGC', 'TAC', 'GTA', 'CGC', 'ATT', 'CAC', 'GCA', 'TTT', 'AGC', 'TAG', 'CTA', 'CGT', 'ACG', 'CAT']

c. Identification of the ORF's

The sequences in the frames are examined to identify those that begin with ATG and end with one of the three termination signals TAG, TGA, TAA. Thus, we find an ORF in Frames 1 and 3(highlighted)

2.D.2 Conversion of ORF stretches to amino acid sequences

The conversion of ORFs to amino acid sequences occurs when the nucleotide T (thymine) is converted into U (uracil) in the physiochemical process of the cell that passes from a DNA sequence to an RNA sequence through the action of the enzyme RNA polymerase.

If we now take the ORF sequences from frames 1 and 3 above, then:

ORF (Frame 1): ['ATG', 'CGT', 'ACG', 'TAG' → AUG CGU ACG UAG → *met arg thr*

ORF (Frame 3): ['ATG', 'CGT', 'GAA', 'TGC', 'GTA', 'CGT', 'AGC', 'TAG'] → AUG CGU GAA UGC GUA CGU AGC → *met arg glu cys val arg ser*

The transfer of an initial DNA sequence to another amino acid sequence in the Training Set (TS) construction pathway to obtain the prediction functions, allows us, firstly, to clearly differentiate *exons* from *introns*, thus enhancing the discriminatory power of the coded part of the DNA sequence, that is, the portion that determines its function in protein production. Secondly, redundancy is reduced, taking into account that many codons produce the same amino acid (e.g., CUU and CUA, which code for leucine). This approach allows us to have more expressive sequences with great discriminatory power, a very important requirement for obtaining good prediction functions.

It is important to note that the above processes are performed automatically using powerful functions provided by different libraries. In our case, we used the Scikit-Learn(sklearn) and Bio-Python libraries.

2.D.3 Construction of TF×IDF matrices

TF×IDF [16] is a classic method for feature extraction, initially used in text mining and Natural Language Processing (NLP). This metric identifies words that are frequent in a specific document but not very common in the entire dataset (corpus). This makes it an excellent resource for determining features and representing text in machine learning (ML) for classification, clustering, or information retrieval tasks.

tf is calculated as follows:

$$tf(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$

where $f_{t,d}$ is the number of times that term t occurs in document d and $\sum_{t' \in d} f_{t',d}$ is the total number of terms in the document.

idf is calculated with the following formula:

$$idf(t,D) = \log \frac{N}{n_t}$$

where D is the set of all documents, $N = D$ is the total of documents and n_t is the number of documents where the term t occurs.

Then,

$$tf \times idf = tf(t,d) \times idf(t,D) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \times \log \frac{N}{n_t}$$

This method has been used to obtain k-mer frequencies (possible DNA or RNA subsequences of length k). An interesting application of this method is presented in the work of [25]. The use of tf×idf-kmer is also found in some other works such as [29], [28] and [15].

In our work, we used the tf×idf method to extract features from amino acid matrices we constructed for each gene on each chromosome. Taking into account that the codon sequences of each gene have already been converted to amino acid sequences, the matrices are constructed as follows: the amino acid strips are taken and 20×20 matrices (corresponding to 20 amino acids) are formed. If any column or row is missing, for example, to construct the final matrix for a given gene, it is completed with the first column or row of the matrix in question. Then, TF-IDF vectors are created by columns of each matrix, thus configuring a TF-IDF matrix.

2.E Training and Test Set selection

The training and test sets are created from the amino acid sequences of each gene on each chromosome partition. The examples are the amino acid chains and their labels are the name of the gene to which the chain corresponds. Each example is a 20x20 TF-IDF matrix. Training sets are created for each P_{ij} partition of each chromosome. The partitions particularly studied contain approximately a total of 23,000 genes and pseudogenes, each corresponding to a "class" within the partition to which it belongs. **Table 1** shows the number of examples per chromosome, and **Table 3** shows the number of examples per chromosome partition. We used an 80/20 approach to split the dataset into training and test sets, as well as a 10% validation set.

Table 1. Number of examples/Genes/Pseudogenes by chromosome

Chromosome	# of Examples	# of genes	# of pseudo-genes	Chromosome	# of Examples	# of genes	# of pseudo-genes
1	50738	2093	1426	13	16616	343	481
2	34472	1200	1080	14	8122	800	550
3	32019	1060	850	15	18752	629	594
4	32442	769	819	16	16978	912	502
5	25540	870	700	17	21335	1199	566
6	32366	1053	911	18	9374	270	300
7	32508	948	933	19	18534	1450	450
8	20234	700	600	20	5119	550	300
9	11080	800	700	21	4743	240	220
10	12432	700	600	22	2175	474	379
11	29540	1314	839	Y	4512	63	388
12	28413	1036	693	X	23419	870	750
Total	341784	13153	7190	Total	149679	7190	5900

2.F Function Induction

As mentioned above, the amino acid sequences of the chromosomes have been divided into partitions containing the gene name (label) and its sequences (examples). Thus, the model will construct prediction functions for each partition of the corresponding chromosome.

Our model's architecture is based on a CNN (Convolutional Neural Network). This is a type of neural network that uses special layers called convolution layers, due to their similarity to the human brain's visual cortex (where everything we see is processed), to detect and extract features from input data.

The network architecture is Sequential Conv2D (2-dimensional convolution). The optimization function is **Adam**, and the activation function is **Softmax**. The general parameters used (hyperparameters) were: **Kernels** (filters): 16 (number of patterns to learn), **Kernel size**: 3 (size of the kernel or filter), **Pooling**: Max Pooling (reduction of the input size, but preserving the most relevant information), **Stride:** 1 (step), **Decay rate**: 0.42 (reduction of the learning rate as training progresses) and **Learning rate**: 0.001 (how much the model updates its weights). The layers number was 3.

We used a number of epochs equal to 120, but introduced the parameter **early_stopping** equal to 6.

Here, we provide a summary of the mathematical foundations of CNN, along with an explanation of how it operates. We will also include a brief application involving the entry [1,1] of the feature map in a tf*idf matrix.

We have a tf*idf matrix of size $m_1 \times m_2$ which will be the tensor T and a kernel K of size $n_1 \times n_2$. The resulting product will be another array F of dimension $(m_1-n_1+1) \times (m_2-n_2+1)$.

Then,

$$F[i,j] = (T * K)_{[i,j]}$$

$$f[i,j] = \sum_{x}^{m_1} \sum_{y}^{m_2} T_{[x,y]} K_{[i-x,j-y]}$$

To calculate the [1,1] entry of the future map using the equation provided with an input tf*idf matrix of order 20, a kernel of size 3 and a stride of 1, we proceed as follows:

$$f[1,1] = \sum_{x=-2}^{20} \sum_{y=-2}^{20} T_{[x,y]} K_{[1-x,1-y]}$$

Then,

$$f[1,1] = \sum_{x=-2}^{20} T_{[x,-2]} K_{[1-x,3]} + \sum_{x=-2}^{20} T_{[x,-1]} K_{[1-x,2]} + \sum_{x=-2}^{20} T_{[x,0]} K_{[1-x,1]}$$

$$+ \sum_{x=-2}^{20} T_{[x,1]} K_{[1-x,0]} + \sum_{x=-2}^{20} T_{[x,2]} K_{[1-x,-1]}$$

This procedure is repeated for all entries in the matrix.

To address the fact that filters are focused on the center of the matrix, ***padding*** is added by including a column and row of zeros on each side of the tensor.

Then, an ***activation function*** is applied to help the network learn non-linear relationships between matrix features for pattern recognition.

Let φ_a be an activation function and b be a bias term, then

$$Conv(T,K) = \varphi_a(c) = \varphi_a(T * K + b)$$

Next, ***pooling P*** is performed. The goal of pooling is to identify the most important features in the convolutional matrix by applying operations that reduce the feature map. We use the Max pooling aggregation function, which selects the maximum value from the feature map.

Let φ_P be a pooling function and $Conv(T,K) = C$, then $P = \varphi_p(C)$

After completion the previous steps, the resulting array is flattened into a single vector, which will serve as the input to a fully connected layer for classification using an activation function

Let g represents the activation function, X be the set of flattened vectors with weights w, pooling equal to P, and d be a bias term, then $X = \sum_i w_i P_i + d$ and $z = g(X)$

Figure 1 presents a block diagram of the function induction processes, from the input of the examples of a chromosome partition to the output of the predictor functions and their performance values.

Here's a general description of the hardware and software used:

Hardware: 24 threads Intel CPU, 24 GB GPU card, 64 GB main memory, 6 TB SSD disks

Software: Linux Ubuntu 24.04, Python 3.10, Scikit-Learn(sklearn) 1.7.0, TensorFlow 2.19.0, CUDA 11.8, BioPython 1.85

2.G Performance Measurement

To evaluate the performance of our model, we used the Precision, Recall, $F_{_score}$ and Accuracy metrics, as well as ROC curves. A description of each is presented below:

Precision(P), Recall(R), F_1 score and Accuracy (ACC)

Let TP be the number of genes correctly classified by our model, FP the number of genes incorrectly classified, FN are the genes that, despite belonging to the class under study, are classified as not belonging to it and TN and TN which are genes that do not belong to the gene in question. Then, precision (P) recall (R) and accuracy (ACC) are defined as follows:

$$P = \frac{TP}{TP + FP} \quad , \quad R = \frac{TP}{TP + FN}$$

$$ACC = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{TP + TN}{TP + TN + FP + FN}$$

The F_β-score [27] is the harmonic mean of precision and recall, where β is a constant that represents the importance of recall on precision. It is defined as:

$$F_\beta \text{ score} = \frac{\beta^2+1}{\frac{\beta^2}{R}+\frac{1}{P}} = \frac{\beta^2+1}{\frac{P.\beta^2+R}{PR}} = \frac{(\beta^2+1).P.R}{P.\beta^2+R}, \text{ if } \beta=1, \text{ then}$$

$$F_1 \text{ score} = 2\frac{P.R}{P+R}$$

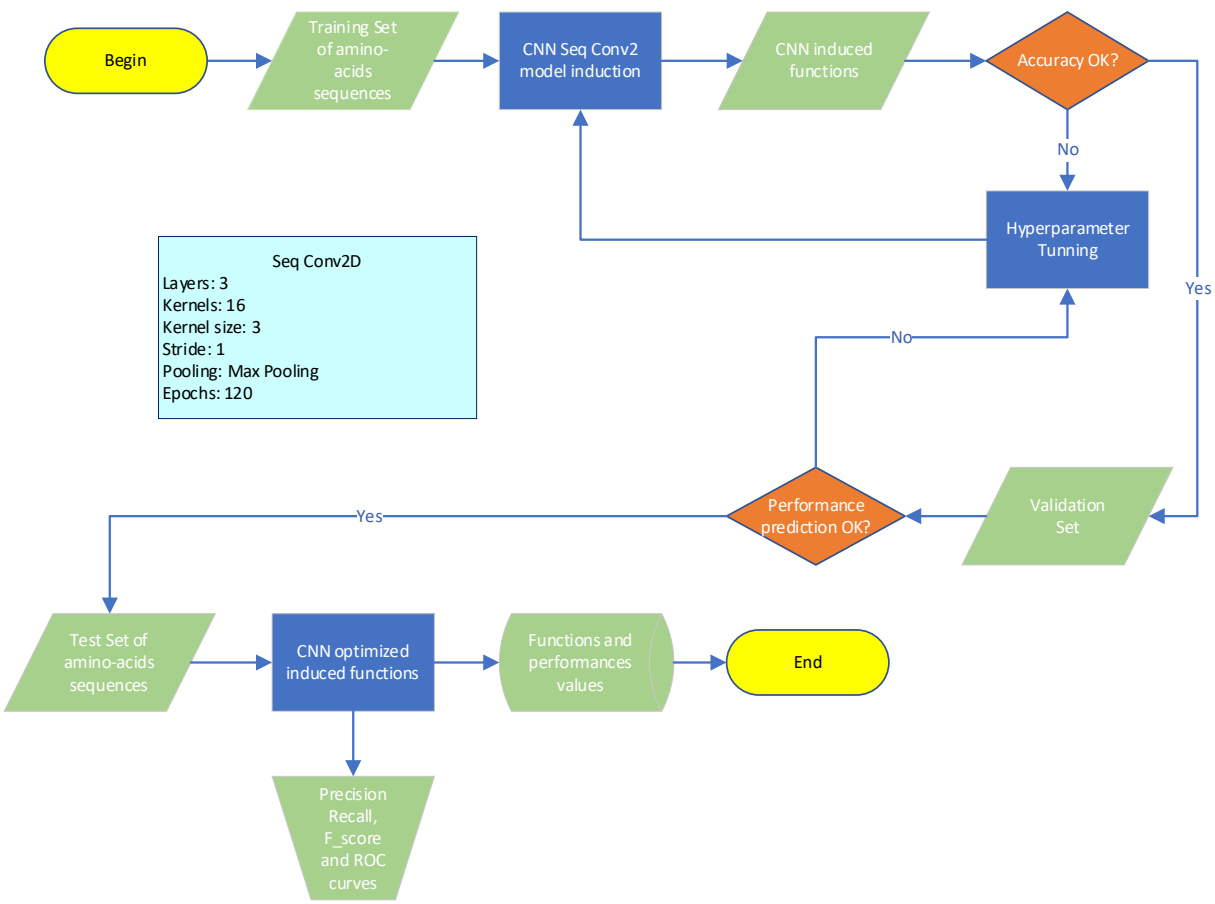

Figure 1. Function Induction Process

ROC curves[11]

Let N be the number of genes to be classified, TN the number of genes that do not belong to the class of the gene examined, TPR be the rate of genes that belong to the class examined. If TN is the number of genes that do not belong to the class examined, we define the following metrics:

$$sensitivity = TPR = \frac{TP}{TP+FN},$$

$$specificity = TNR = \frac{TN}{N}\frac{TN}{TN+FP} = 1 - FPR$$

Then,

$$FPR = 1 - specificity, \Rightarrow FPR = \frac{FP}{FP+TN}$$

We used **sensitivity** and **specificity** measures to create a **ROC** plot and assess the quality of the prediction by examining the point where the maximum TPR value is obtained and which corresponds to the minimum FPR.

The Area Under the Curve (AUC) of the graph indicates the probability of success of the function to accurately classify a gene vs. the probability of being wrong in this task.

2.H Discussion

The metrics applied to both, the learning functions to evaluate the model's ability to fit the data on which it was trained and its ability to learn patterns, as well as the metrics applied to the model's predictive ability (ability to generalize) on unseen examples from the test set, present excellent results as discussed below.

Table 2 shows the metrics **precision (P), recall (R), F_1 score** and **accuracy (ACC)** for each of the partitions containing the genes of interest and on which we focused our attention for the performance evaluation. These partitions correspond to chromosomes 1, 4, 6, 7, 11, 12, 13, 15, 16, 17, 19, 22, X, Y and each one is associated with the predictor function learned from the sequences (examples) they contain.

It can be observed that the **Accuracy** for all partitions was equal to 1.0, which indicates the great power of the function to learn the patterns found in DNA (amino acid) sequences and, consequently, their ability to fit these same sequences.

Consistent with these ideas, we found that the **precision (P)** of the gene prediction for chromosome partitions is above 95%, with the sole exception of partition 7_0 (94%), which contains the CFTR gene. We observed that in 67% of cases, this metric was between 97% and 100%. The P value for partitions 16_2, 17_3, 19_3, 22_4, and Y, which contain the HNF1B, SRY, COMT, LDLR, FTO, and APOE genes, was remarkably high, ranging from 99% to 100%. The average across all partitions was 97%. The behavior of **recall (R)** was similar to precision, noting that its average was reduced by one point with respect to this, but at the same time the **F_1 score** was significantly high with a value of 97%, which indicates a very good balance between false positives and false negatives of the prediction.

Analyzing the ROC curves in **figures 2** and **3**, we observe that in all of them the maximum value for predictions with "true positive" results corresponds to the minimum value of "false positives", which indicates the great power of the classifier to truly identify the class (gene) in question. Likewise, we notice that in 71% of cases the AUC value is equal to or greater than 0.95. We also find that in 29% of these values they are between 0.90 and 0.95. Values less than 0.90 were found in only one case, which corresponds to the CFTR gene (0.88). We also observe that the function is extraordinarily efficient in identifying the genes: ACE, APOE, BRCA2, COL2A1, FBN1, HBA1, HBA2, HBB, HFE, HLA-DRB1, HNF1B, LDLR, MYBPC3 and SRY, whose AUC values are equal to 1.0.

Considering that the partitions specially studied contain approximately 23000 genes and pseudogenes, the ROC curves of each of these genes have been constructed by comparing them with 6 other ROC curves of genes randomly extracted from the partition to which the gene belongs, thus also showing the quality of the prediction for additional genes. For example, the ROC curve for the MYBPC3 gene on chromosome 11 (see the graph) was compared with the genes PDE3B, OSBPL5, OOSP4B, PDE3B, PARVA, and PGM2L1

3 Conclusion

In this work, we have presented an approach for building prediction functions of genes based on convolutional neural networks from real human genome sequences. The resulting model presents performances with average values of 97%, which places it above other works on the same topic and positions it at the state of the art in developments for gene prediction based on machine learning, and in particular based on neural networks.

Taking into account the size of the training space (all coded sequences of the human genome), in an approach divide and conquer, we divided each chromosome into partitions containing

sequences (examples) labeled with the name of the gene to which they belong. To construct the training set, we then implemented a novel method based on tf×idf matrices from the DNA sequences converted to amino acids sequences to identify and extract features for the creation of training examples. We evaluated the quality of the prediction of the induced functions using sequences of genes related to diseases caused by single-gene mutations. The excellent results confirm its potential use in medical applications and genomic research.

4 Future Research

In order to further improve the predictive capacity of our models, we have set a goal for the near future to develop algorithms for concurrent prediction (Ensemble Learning), combining deep learning CNN and Markovian methods as well as methods based on conditional probabilities

Table *2*. Performance of partitions of Genes of Interest

Interest Gene	Chr. Partition	Precision	Recall	F_1	Accuracy
1. HTT	4_2	0.95	0.94	0.95	1.0
2. TLR2	4_6	0.98	0.97	0.98	1.0
3. HLA-DRB1	6_2	0.97	0.96	0.96	1.0
4. COL2A1	12_0	0.95	0.94	0.95	1.0
5. BRCA2	13_0	0.98	0.98	0.98	1.0
6. APOE	19_0	0.99	0.98	0.99	1.0
7. PTPN22	1_7	0.96	0.95	0.96	1.0
8. FTO	16_2	0.99	0.99	0.99	1.0
9. BRCA1	17_0	0.97	0.96	0.96	1.0
10. LEPR	1_4	0.97	0.95	0.96	1.0
11. HFE	6_2	0.97	0.96	0.96	1.0
12. LDLR	19_3	0.99	0.98	0.99	1.0
13. CFTR	7_0	0.94	0.93	0.94	1.0
14. FBN1	15_1	0.98	0.97	0.97	1.0
15. HBB	11_2	0.95	0.94	0.95	1.0
16. FMR1	X_1	0.95	0.94	0.95	1.0
17. HBA1	16_3	0.98	0.97	0.98	1.0
18. HBA2	16_3	0.98	0.96	0.97	1.0
19. ACE	17_0	0.97	0.96	0.96	1.0
20. COMT	22_4	0.99	0.97	0.98	1.0
21. FOXP2	7_1	0.95	0.95	0.95	1.0
22. DMD	X_1	0.95	0.94	0.94	1.0
23. SRY	Y_4	1.0	0.97	0.98	1.0
24. HNF1B	17_3	0.99	0.97	0.98	1.0
$\bar{x}$		0.97	0.96	0.97	1.0

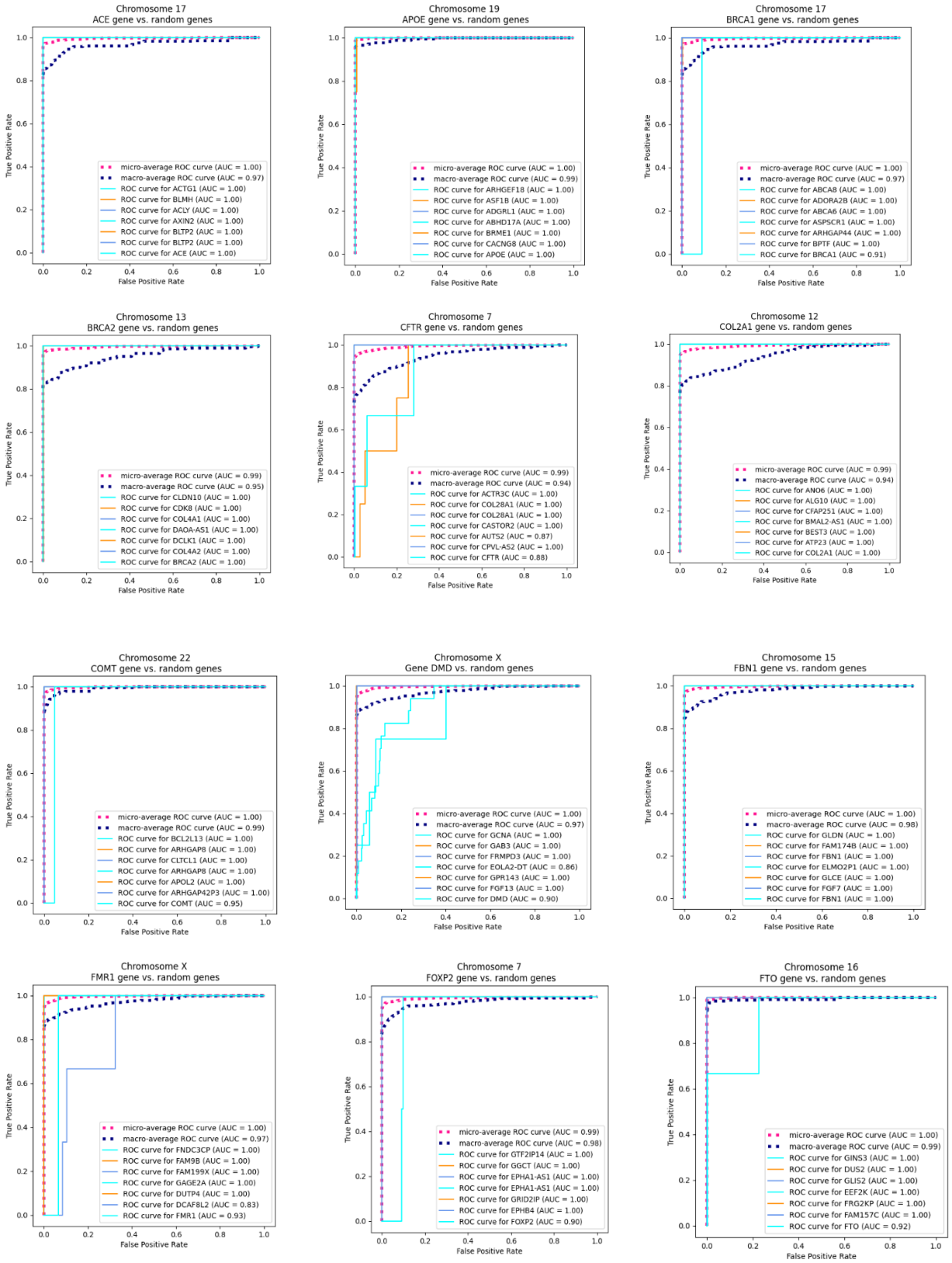

Figure 2 Genes of Interest vs. random genes

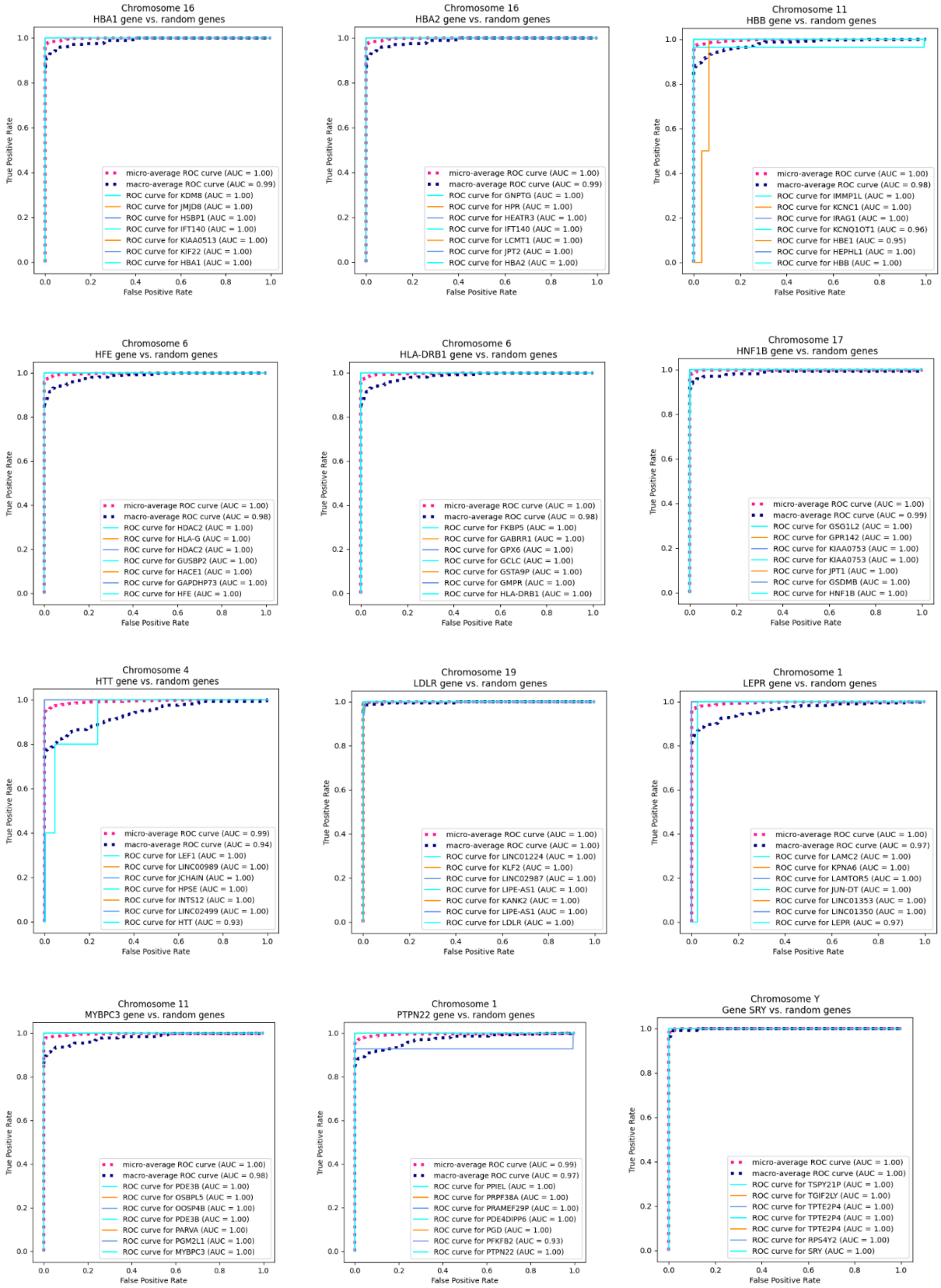

Figure 3 Genes of Interest vs. random genes (continued)

Table 3. Interest Genes (Genetics Disorders)

GENE	CHROMOSOME	GRAL. FUNCTION	MUTATION'S EFFECT
1. HTT[22,39]	4	Encodes the **huntingtin muscle protein,** which plays a crucial role in nerve cells, as cell **signaling, protein interactions, and DNA repair**	Huntington's disease, a neurodegenerative disorder
2. MYBPC3[9,12]	4	**Encodes cardiac myosin-binding protein C (cMyBP-C), a structural and regulatory protein in the heart's contractile function**	Left ventricular hypertrophy. Arrhythmias, sudden cardiac death. Ventricular dilation, systolic disfunction, stiff ventricles
3. HLA-DRB1 [9,22]	6	Plays a role in **autoimmune diseases**, like **rheumatoid arthritis, multiple sclerosis, and autoimmune Addison disease**	Affects immune responses differently. Par example, specific variants are associated with **autoimmune Addison disease,** where the immune system mistakenly attacks the adrenal glands
4. COL2A1 [9,22,39]	12	A very important protein for the formation of **cartilage, joints, eyes, and the inner ea**	**Stickler syndrome (affecting vision, hearing). Spondyloepiphyseal dysplasia congenita (SEDC) (a skeletal disorder). Kniest dysplasia (leading to short stature and joint problems). Achondrogenesis type 2 (a severe skeletal disorder)**
5. BRCA2 [22,39]	13	Is a **tumor suppressor.** Plays a crucial role in **DNA repair.**	Increase the risk of **several cancers,** most notably **breast and ovarian cancer**, but also **prostate and pancreatic cancer**
6. APOE [9,22,39]	19	A protein that helps transport cholesterol and other fats through the bloodstream	**The variant APOE ε4** increases the risk of developing **Alzheimer's disease a**nd is linked to more severe forms of the condition
7. PTPN2 [9,22]	1	Encodes **protein tyrosine phosphatase non-receptor type 22**, which plays a essential role in **immune system regulation**	Have been linked to an increased risk of **autoimmune diseases**, including: **Type 1 diabetes, Rheumatoid arthritis, Systemic lupus erythematosus, Graves' disease, Hashimoto's thyroiditis**
8. FTO [9,22,39]	16	Is linked to **body weight regulation and metabolism**	Variants of the **FTO gene** have been associated mainly with **higher body mass index (BMI) and obesity risk**
9. LEPR [9,22,39]	1	Encodes the **leptin receptor**, a protein that plays a crucial role in **appetite regulation, metabolism, and energy balance**	**Leptin receptor deficiency**, which is associated with **severe obesity, excessive hunger, and hormonal imbalances**
10. HFE[9,22,39]	6	Plays a definitive role in **iron metabolism** by regulating how the body absorbs iron from food	**Hereditary hemochromatosis,** where excess iron builds up in organs like the **liver, heart, and pancreas**, potentially leading to serious health issues such as **liver disease, diabetes, and heart problems**
11. LDR [9,22]	19	Produces a lipoprotein receptor (LDLR) that helps remove **LDL (bad cholesterol)** from the bloodstream by binding to LDL particles and transporting them into cells for processing	**Familial hypercholesterolemia (FH)**: leads to **high cholesterol levels**, increasing the risk of **heart disease and early-onset heart attacks**

Table 4. Interest Genes (Genetics Disorders) (continued)

GENE	CHROMOSOME	GRAL. FUNCTION	MUTATION'S EFFECT
12. CFTR [9,22]	7	Encodes a **chloride ion channel** that plays a crucial role in maintaining the balance of salt and water on cell surfaces, particularly in the **lungs, pancreas, and digestive system**	**Cystic fibrosis (CF)**, that causes thick, sticky mucus to build up in the lungs and other organs
13. FBN1[9,22]	15	Encodes **fibrillin-1**, a very important protein in the **extracellular matrix** that helps form **microfibrils**, which provide structural support to tissues like **skin, blood vessels, and bones**	**Marfan syndrome,** affecting the **heart, eyes, and skeleton**. **Weill-Marchesani syndrome**, which leads to **short stature and joint stiffness**. **Ectopia lentis**, where the **lens of the eye becomes displaced**, affecting vision
14. HBB [9,22]	11	Encodes **hemoglobin subunit beta**, a essential component of hemoglobin—the protein responsible for carrying oxygen in red blood cells	**Sickle Cell Disease**: leads to abnormal hemoglobin that distorts red blood cells into a sickle shape, causing **pain, anemia, and organ damage**
15. FMR1 [9,22]	X	Plays a fundamental role in **brain development and cognitive function**. It encodes the **FMRP protein**, which helps regulate **synaptic plasticity**, influencing learning and memory	**Fragile X Syndrome (FXS)**: Caused by an expansion of the **CGG nucleotide repeats,** leading to intellectual disability and developmental delays **Fragile X-associated primary ovarian insufficiency (FXPOI)**: Affects ovarian function and fertility **Fragile X-associated tremor/ataxia syndrome (FXTAS)**: A neurodegenerative disorder
16. HBA1 [9,12]	16	**HBA1** encodes **hemoglobin subunit alpha 1**, a key component of hemoglobin—the protein responsible for carrying oxygen in red blood cells Works with **HBA2**, gene that produces alpha-globin	**Alpha Thalassemia**: Caused by deletions or mutations in **HBA1 and HBA2**, leading to reduced hemoglobin production **Hemoglobin H Disease**: Results from the loss of three alpha-globin alleles, causing **anemia, jaundice, and enlarged spleen** **Hb Bart Syndrome: The most severe form, leads to hydrops fetalis (severe fluid buildup before birth)**
17. HBA2 [9,12]	16	Encodes **hemoglobin subunit alpha 2**, which is a key component of hemoglobin—the protein responsible for carrying oxygen in red blood cells	**Alpha Thalassemia** **Hemoglobin H Disease** **Hb Bart Syndrome**
18. BRCA1 [22,39]	17	Is a **tumor suppressor gene** that plays a vital role in **DNA repair** and maintaining genomic stability	Mutations in **BRCA1** significantly increase the risk of **breast and ovarian cancer**, as well as other cancers like **pancreatic and prostate cancer**
19. ACE[12,22]	17	Encodes the angiotensin-converting enzyme, which is a very important regulator of blood pressure, fluid balance, and electrolyte homeostasis	Impaired kidney development, low blood pressure, anuria, hypertension, stroke, diabetic nephropathy, late-onset Alzheimer's disease

Table 3. Number of examples by chromosome partition

No.	Chr.	Part.	Exs.	No.	Chr.	Part.	Exs.	No.	Chr.	Part.	Exs.	No.	Chr.	Part.	Exs.
1	4	4_0	5905	50	17	17_3	1488	99	21	21_1	969	148	14	14_5	1960
2	4	4_1	5551	51	17	17_4	1637	100	21	21_2	276	149	14	14_6	1827
3	4	4_3	4114	52	17	17_5	1652	101	21	21_3	1117	150	9	9_0	1574
4	4	4_4	3193	53	17	17_6	2019	102	21	21_4	193	151	9	9_1	1449
5	4	4_5	4072	54	13	13_0	2961	103	21	21_5	1002	152	9	9_2	1233
6	4	4_6	3942	55	13	13_1	3469	104	5	5_0	4874	153	9	9_3	1515
7	6	6_0	4380	56	13	13_2	2359	105	5	5_1	3664	154	9	9_4	790
8	6	6_1	3982	57	13	13_3	3865	106	5	5_2	4614	155	9	9_5	1698
9	4	4_2	5665	58	13	13_4	1480	107	5	5_3	3994	156	9	9_6	1015
10	16	16_0	1595	59	13	13_5	2482	108	5	5_4	3613	157	9	9_7	1806
11	16	16_1	2107	60	19	19_0	1859	109	5	5_5	1530	158	15	15_0	3384
12	16	16_2	1656	61	19	19_1	1755	110	5	5_6	3251	159	15	15_1	3085
13	16	16_3	1608	62	19	19_2	1874	111	8	8_0	4292	160	15	15_2	3063
14	16	16_4	1638	63	19	19_3	1955	112	8	8_1	2188	161	15	15_3	3277
15	16	16_5	1376	64	19	19_4	1879	113	8	8_2	3166	162	15	15_4	2552
16	16	16_6	2323	65	19	19_5	1839	114	8	8_3	3190	163	15	15_5	3391
17	16	16_7	1140	66	19	19_6	1745	115	8	8_4	2033	164	20	20_0	569
18	16	16_8	1514	67	19	19_7	1913	116	8	8_5	1872	165	20	20_1	644
19	16	16_9	2021	68	19	19_8	1885	117	8	8_6	3493	166	20	20_2	600
20	6	6_2	3165	69	19	19_9	1830	118	Y	Y_0	555	167	20	20_3	403
21	6	6_3	3162	70	12	12_0	4814	119	Y	Y_1	632	168	20	20_4	709
22	6	6_4	2923	71	12	12_1	3625	120	Y	Y_2	914	169	20	20_5	1195
23	6	6_5	4127	72	12	12_2	3376	121	Y	Y_3	555	170	20	20_6	999
24	6	6_6	2464	73	12	12_3	4528	122	Y	Y_4	541	171	22	22_0	310
25	6	6_7	1866	74	12	12_4	4559	123	Y	Y_5	742	172	22	22_1	356
26	6	6_8	3365	75	12	12_5	2716	124	2	2_0	3786	173	22	22_2	521
27	6	6_9	2932	76	12	12_6	4795	125	2	2_1	3016	174	22	22_3	355
28	X	X_0	3895	77	7	7_0	7137	126	2	2_2	3755	175	22	22_4	104
29	X	X_1	4537	78	7	7_1	5236	127	2	2_3	1535	176	22	22_5	529
30	X	X_2	3392	79	7	7_2	5248	128	2	2_4	3195	177	3	3_0	2836
31	X	X_3	3286	80	7	7_3	3751	129	2	2_5	4003	178	3	3_1	3068
32	X	X_4	2450	81	7	7_4	3387	130	2	2_6	2613	179	3	3_2	2727
33	X	X_5	2884	82	7	7_5	4594	131	2	2_7	3315	180	3	3_3	2100
34	X	X_6	2975	83	7	7_6	3155	132	2	2_8	1480	181	3	3_4	4087
35	1	1_0	5189	84	11	11_0	3634	133	2	2_9	772	182	3	3_5	2891
36	1	1_1	4481	85	11	11_1	4760	134	2	2_10	3322	183	3	3_6	2728
37	1	1_2	4782	86	11	11_2	3626	135	2	2_11	3680	184	3	3_7	2363
38	1	1_3	3719	87	11	11_3	4189	136	10	10_0	2109	185	3	3_8	1161
39	1	1_4	4611	88	11	11_4	4067	137	10	10_1	2416	186	3	3_9	1809
40	1	1_5	4181	89	11	11_5	2652	138	10	10_2	1248	187	3	3_10	3625
41	1	1_6	4232	90	11	11_6	3427	139	10	10_3	1499	188	3	3_11	2624
42	1	1_7	4764	91	11	11_7	3185	140	10	10_4	1962	189	Y	Y_6	573
43	1	1_8	3527	92	18	18_0	2431	141	10	10_5	1894	190	17	17_7	1737
44	1	1_9	2932	93	18	18_1	1679	142	10	10_6	1304	191	17	17_8	1302
45	1	1_10	4106	94	18	18_2	1577	143	14	14_0	966	192	17	17_9	1787
46	1	1_11	4214	95	18	18_3	1736	144	14	14_1	1041	193	17	17_10	2084
47	17	17_0	2471	96	18	18_4	426	145	14	14_2	1462	194	17	17_11	1667
48	17	17_1	1705	97	18	18_5	1525	146	14	14_3	209				
49	17	17_2	1786	98	21	21_0	1186	147	14	14_4	657				

Total number of examples: 491463

Chr.: Chromosome, Part.: Partition, Exs. : Number of Examples

REFERENCES

1. S.F Altschul et al. 1990. Basic local alignment search tool. *Journal of Molecular Biology* 215, 3: 403–410. https://doi.org/10.1016/S0022-2836(05)80360-2
2. C.T Bergstrom and L.A Dugatkin. 2023. *EVOLUTION*. WW Norton & Company, New York.
3. M. Borodovsky and J. McIninch. 1993. Computers & Chemistry GeneMark: Parallel gene recognition for both DNA strands. *Computers & Chemistry* 17, 2: 123–133. https://doi.org/10.1016/0097-8485(93)85004-V
4. E.A. Bruford, M.J. Lush, M.W. Wright, T.P. Sneddon, S., and E. Birney. 2020. Guidelines for human gene nomenclature. *Nature Genetics* 52: 754–758. https://doi.org/10.1038/s41588-020-0669-3
5. C. B. Burge. 1998. Modeling dependencies in pre-mRNA splicing signals. *New Comprehensive Biochemistry* 32, 1998: 129–164. https://doi.org/10.1016/S0167-7306(08)60465-2
6. C. Burge and S. Karlin. 1997. Prediction of complete gene structures in human genomic DNA. *Journal of Molecular Biology* 268, 1: 78–94. https://doi.org/10.1006/jmbi.1997.0951
7. C. Camacho, G. Colouris, V. Avagyan, N. Ma, J. Papadopoulos, K. Bealer, and T.L. Madden. 2009. BLAST+: architecture and applications. *BMC Bioinformatics* 10, Article 421. https://doi.org/10.1186/1471-2105-10-421
8. B Charlesworth and D Charlesworth. 2010. *Elements of Evolutionary Genetics*. Roberts & Company, Greenwood Village, CO, USA.
9. Weizmann Institute of Science in Crown Human Genome Center. 1997. GeneCards. *GeneCards. The Human Gene DataBase*. Retrieved from www.genecards.org
10. M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD)*, 226–231.
11. T. Fawcett. 2006. An Introduction to ROC Analysis. *Pattern Recognition Letters* 27, 8: 861–874. https://doi.org/doi:10.1016/j.patrec.2005.10.010
12. J.F Griffits et al. 2020. *Introduction To Genetic Analysis*. USA.
13. K. He, Zhang, Xiangyu, Ren, Shaoqing, and J. Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770-778. Retrieved from https://ieeexplore.ieee.org/xpl/conhome/7776647/proceeding
14. A. Hedge, T. Nguyen, and J. Cheng. Machine Learning Methods for Gene Regulatory Network Inference. 2025.*arXiv*: 1–40. ttps://doi.org/10.48550/arXiv.2504.12610
15. W. Hu, M. Li, H. Xiao, and L. Guan. 2024. "Essential Genes Identification Model Based on Sequence Feature Map and Graph Convolutional Neural Network. *BMC Genomics* 25, 47. https://doi.org/10.1186/s12864-024-09958-w
16. K. Spärck Jones. 2004. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation* 60: 493–502.
17. T. Lefevre et al., M. Raymond, and F. Thomas. 2016. *Biologie évolutive*. De Boeck Superieur, Louvain-la-Neuve, Belgium.
18. M. Lin, I. Jungreis, and M. Kellis. 2011. PhyloCSF: a comparative genomics method to distinguish protein coding and non-coding regions. *Bioinformatics* 27, 13: i275–i282. https://doi.org/10.1093/bioinformatics/btr209
19. Zhenxin Liu and Di Lu. 2023. Ergodicity of Inhomogeneous Markov Processes Under General Criteria. *Frontiers of Mathematics*. https://doi.org/10.48550/arXiv.2307.13064

20. M. A. Mohamed and P. D. Gader. 2002. Generalized Hidden Markov Models—Part I: Theoretical Frameworks. *IEEE Transactions on Fuzzy Systems,* 8, 1: 67–81. https://doi.org/10.1109/91.824772
21. NCBI GRC. 2007. Genome Reference Consortium. GRCh38.p14 Primary Assembly. NCBI. Retrieved from https://www.ncbi.nlm.nih.gov/grc/human/data
22. NIH. 2000. MedlinePlus. *MedlinePlus*. Retrieved from https://medlineplus.gov/
23. L. Pachter, M. Alexandersson, and S. Cawley. Applications of generalized pair hidden Markov models to alignment and gene finding problems. *Journal of Computational Biology,* 9, 3: 389–399. https://doi.org/10.1089/1066527025293552
24. G. Perez, G.P. Barber, A. Bennet-Pages, J. Casper, H. Clawson, A. Hinrichs, C.M. Lee, L.R. Nassar, B.J. Raney, M. Speir, J. Marijke, C.J. Vaske, D. Haussler, W.J Kent, M. Haeussler, M. Diekhans, C. Fischer, and J. Navarro Gonzalez. 2024. The UCSC Genome Browser database: 2025 update. *Nucleic Acids Research* 53(D1): D1243–D1249. https://doi.org/10.1093/nar/gkae974
25. J.A.M Rexie and et al. 2023. K-mer based prediction of gene family by applying multinomial naïve bayes algorithm in DNA sequence. In *AIP Conference Proceedings*, 7. https://doi.org/10.1063/5.0175878
26. M. Romero, O. Ramirez, J. Finke, and C. Rocha. Supervised gene function prediction using spectral clustering on gene co-expression network. *Complex Networks & Their Applications X, Studies in Computational Intelligence* 1073: 652–663. https://doi.org/10.1007/978-3-030-93413-2_54
27. M. Saritha, M. Lavanya, and M. Narendra Reddy. 2022. Methods to Predict the Performance Analysis of Various Machine Learning Algorithms. *Bayesian Reasoning and Gaussian Processes for Machine Learning Applications,*. Retrieved from https://taylorandfrancis.com/knowledge/Engineering_and_technology/Engineering_support_and_special_topics/F-score
28. J. Shaw and Y. Yun William. 2024. Fairy: fast approximate coverage for multi-sample metagenomic binning. *Microbiome* 12, 151. https://doi.org/10.1186/s40168-024-01861-6
29. N. Shenker-Tauris and J. Gehrig. 2022. IMPROVED METAGENOMIC BINNING WITH TRANSFORMERS. *bioRxiv*: 21. https://doi.org/10.1101/2022.02.12.479459
30. G.S.C Slater and E Birney. 2005. Automated generation of heuristics for biological sequence comparison. *BMC Bioinformatics* 6, 31: Article 31.
31. M. Stanke, M. Diekhans, R. Baertsch, and D. Haussler. 2008. Using native and syntenically mapped cDNA alignments to improve de novo gene finding. *Bioinformatics* 24, 5: 637–644. https://doi.org/10.1093/bioinformatics/btn013
32. M Stanke and et al. 2006. AUGUSTUS: ab initio prediction of alternative transcripts. Nucleic Acids Research. *Nucleic Acids Research* 34, Web Server issue: W435–W439. https://doi.org/10.1093/nar/gkl200
33. The UniProt Consortium. 2021. UniProt: the universal protein knowledgebase in 2021. *Nucleic Acids Research* 49, D1: D480-D489,. https://doi.org/10.1093/nar/gkaa1100
34. Lourenço Vanda M and et al. 2024. Genomic prediction using machine learning: a comparison of the performance of regularized regression, ensemble, instance-based and deep learning methods on synthetic and empirical data. *BMC Genomics*. https://doi.org/10.1186/s12864-023-09933-x
35. D. Wei, Q. Jiang, Y. Wei, and S. Wang. 2012. A novel hierarchical clustering algorithm for gene sequences. *BMC Bioinformatics* 13, Article 174. https://doi.org/10.1186/1471-2105-13-174
36. F.-X. Wu, J. W., and A.J. Kusalik. 2003. A genetic K-means clustering algorithm applied to gene expression data. *Lecture*

Notes in Artificial Intelligence 2671: 520–526. https://doi.org/10.1007/3-540-4488
37. Yuyao Yang, Ch. Xinyi, W Li, W. Tao, L. Jiajun, and S. Wang. 2025. DeepMethyGene: a deep-learning model to predict gene expression using DNA methylations. *BMC Bioinformatics* 26, no. 99. https://doi.org/10.1186/s12859-025-06115-2
38. J. Zamora-Esquivel, A. Cruz Vargas, P. Lopez, and Omesh Tickoo. 2025. Adaptive Convolutional Kernels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*. Retrieved from https://openaccess.thecvf.com/content_ICCVW_2019/papers/NeurArch/Esquivel_Adaptive_Convolutional_Kernels_ICCVW_2019_paper.pdf
39. National Library of Medicine - National Institutes of Health. Retrieved July 9, 2025 from https://www.nlm.nih.gov/
40. Ensembl genome browser. Retrieved July 9, 2025 from https://www.ensembl.org/index.html?redirect=no
41. HUGO Gene Nomenclature Committee. Retrieved July 9, 2025 from https://www.genenames.org/
42. UCSC Genome Browser Home. Retrieved July 18, 2025 from https://genome.ucsc.edu/
43. National Human Genome Research Institute Home | NHGRI. Retrieved July 11, 2025 from https://www.genome.gov/