

# Genome annotation across species using deep convolutional neural networks

**Ghazaleh Khodabandelou** <sup>Corresp., 1</sup>, **Etienne Routhier** <sup>2</sup>, **Julien Mozziconacci** <sup>Corresp. 3</sup>

<sup>1</sup> Laboratoire de Physique Théorique de la Matière Condensée (LPTMC), Sorbonne Université, Paris, France

<sup>2</sup> Laboratoire de Physique Théorique de la Matière Condensée (LPTMC), Sorbonne Université, Paris, France

<sup>3</sup> CNRS UMR 7196 / INSERM U1154 - Sorbonne Universités, Museum national d'Histoire naturelle (MNHN), Paris, France

Corresponding Authors: Ghazaleh Khodabandelou, Julien Mozziconacci

Email address: ghazaleh.khodabandelou@u-pec.fr, julien.mozziconacci@mnhn.fr

Application of deep neural network is a rapidly expanding field now reaching many disciplines including genomics. In particular, convolutional neural networks have been exploited for identifying the functional role of genomic sequences. These approaches rely on gathering a large set of sequences of a given length with known functional role, extracting those sequences from whole-genome. These set is then split into learning, test and validation sets in order to train the network. While the obtained networks perform well on validation sets, they often perform poorly when applied on whole genomes in which the ratio of positive over negative examples can be very different than in the training set. We here address this issue by assessing the genome-wide performance of networks trained with sets exhibiting different ratios of positive to negative examples. As a case study, we use sequences encompassing gene start sites, obtained from the RefGene database, as positive examples and random genomic sequences as negative examples. We also demonstrate that models trained using data from one organism can be used to predict gene start sites in a related species, when using training sets providing good genome-wide performance. This cross-species application of convolutional neural networks provides a new way to annotate any genome from existing high-quality annotations in a related reference species. It also provides a way to determine whether the sequence motifs recognised by chromatin-associated proteins in different species are conserved or not.

# Genome Annotation across Species using Deep Convolutional Neural Networks

Ghazaleh Khodabandelou<sup>1,\*</sup>, Etienne Routhier<sup>1</sup>, and Julien Mozziconacci<sup>1,2,\*</sup>

<sup>1</sup>Laboratoire de Physique Théorique de la Matière Condensée (LPTMC), Sorbonne Université, Paris, France.

<sup>2</sup>CNRS UMR 7196 / INSERM U1154 - Sorbonne Université, Museum national d'Histoire naturelle (MNHN), Paris, France.

\*ghazaleh.khodabandelou@u-pec.fr, julien.mozziconacci@mnhn.fr

## ABSTRACT

Application of deep neural network is a rapidly expanding field, now reaching many disciplines including genomics. In particular, convolutional neural networks have been exploited for identifying the functional role of genomic sequences. These approaches rely on gathering a large set of sequences of a given length with known functional role, extracting those sequences from whole-genome. These set is then split into learning, test and validation sets in order to train the network. While the obtained networks perform well on validation sets, they often perform poorly when applied on whole genomes in which the ratio of positive over negative examples can be very different than in the training set. We here address this issue by assessing the genome-wide performance of networks trained with sets exhibiting different ratios of positive to negative examples. As a case study, we use sequences encompassing gene start sites, obtained from the RefGene database, as positive examples and random genomic sequences as negative examples. We also demonstrate that models trained using data from one organism can be used to predict gene start sites in a related species, when using training sets providing good genome-wide performance. This cross-species application of convolutional neural networks provides a new way to annotate any genome from existing high-quality annotations in a related reference species. It also provides a way to determine whether the sequence motifs recognised by chromatin-associated proteins in different species are conserved or not.

## INTRODUCTION

The improvement of DNA sequencing techniques lead to an explosion in the number and completeness of fully sequenced genomes. One of the major goals in the field is to annotate these DNA sequences, which is to associate a biological function with sequence motifs located at different positions along the genome [1]. In the human genome for instance, while some DNA sequences encode proteins, most sequences do not code for any protein. Many of these non-coding sequences are nevertheless conserved in other species and are necessary for the correct regulation of gene expression. Deciphering the function of these non-coding sequences has been increasingly achieved, notably through improvements in the throughput of next generation sequencing [2]. The 3.2 Billion base pair (bp) long human genome is now annotated with many functional and bio-chemical cues [3, 4]. While these annotations are becoming more numerous and precise, they cannot be determined experimentally for every organism and every cell type. Computational methods are therefore widely used to extract sequence information from known annotations and extrapolate the results to different genomes and/or conditions, e.g. [3, 5].

An related question is to understand the link between these annotations and the underlying DNA sequence. To this end, supervised machine learning algorithms [6] have been particularly successful [7, 8]. Among those, deep Convolution Neural Networks (CNN) are very efficient at detecting sequence features since they rely on the optimisation of convolution filters that can be directly matched to DNA motifs [9]. Stacking several of these convolution layers together can lead to the detection of nested motifs at larger scales. Pioneering studies illustrated this ability of CNN to reliably grasp complex combinations of DNA motifs and their relationship with functional regions of the genome [10, 11, 12, 13, 14, 15].

Min et al. [10] used a CNN to predict enhancers which are specific sequences that regulate gene

expression at a distance. This method reached very good scores ranking it above the state-of-the-art (i.e., support vector machine methods). Similar tools were used in different contexts, aiming at identifying promoters [11, 15] or detecting splicing sites [16, 17]. In these approaches, a sample set is first created by taking all positive class sequences (e.g. enhancers) and adding the same amount of randomly picked negative class examples (e.g. non-enhancers). This sample set is then divided into training, validation and test sets. Balancing the data ensures that the model will be trained on the same number of positive and negative examples, thus giving the same importance to both classes. While these approaches are very successful when assessed on test sets derived from the sample set, we show here that they tend to perform poorly when applied on entire chromosome sequences as required for the task of complete genome annotation. This is due to the fact that the networks are optimised on a similar number of positive and negative examples during training, but that they will usually face very different ratios of negative over positive classes when used on a full chromosome sequence.

Alternative approaches [12] [14] used unbalanced datasets for training (i.e., with more negative than positive examples) to predict DNA-binding sites for proteins and genome accessibility. In these two studies, however, the prediction performance of the model is also assessed on test sets derived from training sets, not on full genomic sequences. The task of genome-wide prediction has been assessed in a more recent study aiming at identifying regulatory elements [18]. The author used long (131 kb) non-overlapping windows covering the whole genome, a procedure which can be effective but requires a lot of memory.

The methodology proposed here is inspired from this last study and presents two novelties for the development and for performance assessment of genome-wide predictions. Firstly, we do not use as a quality measure the classical prediction scores computed on test sets obtained by dividing the sample data into training, validation and test sets as commonly done in machine learning. We rather compute prediction scores that assess the ability of our model to annotate a full chromosome sequence by designing a specific metric (described in Material and Methods). Secondly, we change the ratio between positive and negative examples in order to obtain the highest prediction scores and show that this tuning is has an important effect on the outcome. As a proof of principle, we use in this work gene start sites (GSS) as features. DNA motifs around GSS are recognised by the transcription machinery and indicate the location of the initiation of transcription [19]. The DNA sequence surrounding GSS therefore contains the information that could in principle be used by an algorithm to identify *in silico* the GSS locations.

We then propose a new application of CNN in genomics that leverages the fact that similar organisms tend to have similar regulatory mechanisms, i.e. rely on an homologous molecular machinery and on homologous DNA regulatory motifs. Exploiting these homologies, we first train a model on a dataset corresponding to a given organism and use it to predict the annotation on the genome of a related organism, opening new opportunities for the task of *de-novo* genome annotation. We show that a CNN trained on GSS containing regions in human is able to recover regions containing GSS in the mouse genome and vice versa. We also assess the generalisation of the approach to more distant species, taking as examples a fish and a bird.

## METHODS

### Input Generation

The GSS positions are collected from the reference genomes for human (hg38) and mouse (mm10) species.

Genome sequence datasets are available at

<https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/hg38.fa.gz> and

<https://hgdownload.soe.ucsc.edu/goldenPath/mm10/bigZips/chromFa.tar.gz>.

GSS positions over the entire human and mouse genomes datasets are available at

<http://egg.wustl.edu/>, the gene annotation is taken from RefGene

<https://egg.wustl.edu/d/hg38/refGene.gz> and

<https://egg.wustl.edu/d/mm10/refGene.gz>.

RefGene is a reference for well-characterized genes from the NCBI RNA reference sequences:

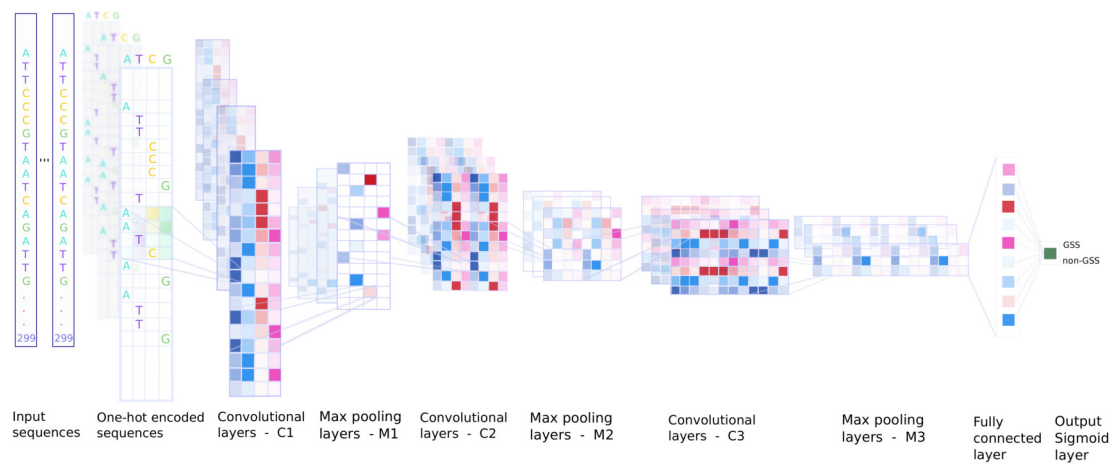
<http://varianttools.sourceforge.net/Annotation/RefGene>.

For the chicken and zebrafish analysis, similar datasets were downloaded from:

<https://hgdownload.soe.ucsc.edu/goldenPath/galGal4/bigZips/galGal4.fa.gz>,

<https://egg.wustl.edu/d/galGal4/refGene.gz>,

<https://hgdownload.soe.ucsc.edu/goldenPath/danRer10/bigZips/danRer10.fa.gz>,



**Figure 1. Overview of the CNN model.** 299 bp-long sequences are one hot encoded into a  $4 \times 299$  input matrix. The first CNN layer performs a convolution on each input matrix to recognise relevant motifs. The next convolutional layers models the interplay among these motifs to grasp higher-level features. Max-pooling layers reduce the dimensions of the layers. The model is trained to correctly label input sequences as GSS or non-GSS. The output layer of the trained network then gives a probability for any 299 bp region to contain a GSS. It can be applied along a full chromosome, i.e. on all 299 bp-long sequences with a 1 bp shift.

<https://egg.wustl.edu/d/danRer10/refGene.gz>.

As a positive input class, we use regions of 299 bp flanking GSS (i.e.,  $\pm 149$  bp around the GSS) which are supposed to contain multiple sequence signals indicating the presence of a GSS to the transcription machinery of the cell. Overall, 31,037 GSS positions are extracted on both DNA strands (15,798 for the positive strand and 15,239 for the negative strand). In a similar fashion, we extract 25,698 GSS positions from the mouse genome (12,938 for positive strand and 12,760 for negative strand). In order to generate the negative class, we select  $31,037 \times Q$  sequences of 299 bp at random positions on a random strand, rejecting regions that do contain a GSS. The odds of getting at random a genomic region containing a GSS are close to 0.28%. For  $Q = 1$ , there is an equal number of negative and positive class examples. Unbalanced datasets are produced using different values of  $Q$  ranging from 1 to 100. For  $Q = 100$ , the negative class encompasses 1Gb, that is one third of the human genome.

### Convolution Neural Network (CNN)

A CNN (see figure 1) is trained in order to predict the presence of a GSS in a DNA sequence of size 299 bp. The shape of the input layer is  $c \times b$  in which  $c = 4$  is the number of different nucleotides and  $b = 299$  is the length of the input sequence. The nucleotide sequences are one hot encoded so that  $A=(1,0,0,0)$ ,  $T=(0,1,0,0)$ ,  $C=(0,0,1,0)$ , and  $G=(0,0,0,1)$ . The training set contains  $N$  samples of labelled pairs  $(X^{(n)}, y^{(n)})$ , for  $n \in \{1, \dots, N\}$ , where  $X^{(n)}$  are matrices of size  $c \times b$  and  $y^{(n)} \in \{0, 1\}$ . Each  $X^{(n)}$  is associated with  $y^{(n)} = 1$  when it corresponds to a region containing a GSS and  $y^{(n)} = 0$  otherwise. The first convolution layer consist in  $k$  kernels which are applied on  $b - s + 1$  successive sequences at positions  $p \in \{1, \dots, (b - s + 1)\}$  to recognise relevant DNA motifs of size  $s$ . This operation generates an output feature map of size  $k \times (b - s + 1)$  for an input  $X^{(n)}$  of size  $c \times b$ . The feature map  $\mathcal{M}$  resulting from the convolution operation is computed as follows:

$$\mathcal{M}_{p,i} = \sum_{j=1}^c \sum_{r=1}^s \mathcal{W}_{i,j,r} X_{p+r-1,j} + \mathcal{B}_i, \quad i \in \{1, \dots, k\} \quad (1)$$

where  $\mathcal{W}$  denotes the network weights with size  $(k \times c \times s)$  and  $\mathcal{B}$  denotes the biases with size  $(k \times 1)$  (see e.g. [6]). After the convolution layer a non-linear function is applied to the output, here a Rectified Linear Unit (ReLU). This activation function computes  $f_{ReLU}(\mathcal{M}) = \max(0, \mathcal{M})$  to incorporate non-linearity by transforming all negative values to zero. In order to reduce the input dimension we

apply a max-pooling process with a pool size  $m$  over the output of  $f_{ReLU}(\mathcal{M})$ . Similar convolution layers followed by ReLU and max-pooling are added sequentially on the input of the first layer to grasp higher order motifs. The output of the last max-pooling layer is then fed into a fully connected layer which output  $x$  is transformed by a sigmoid function ( $\phi = \frac{1}{1+e^{-x}}$ ) in order to give the final output of the CNN. This final score of the input sequence is ideally 0 for non-GSS and 1 for GSS containing sequences. When we need to perform a classification we use a threshold of 0.5 to discriminate between the two classes.

In the training phase, the weights and biases of the convolution layers and the fully connected layer are updated via back-propagation in a way which decreases the loss, which measures the discrepancy between the network predictions and the reality averaged over individual examples. We use here the binary cross-entropy computed as:

$$\mathcal{L} = -1/N \sum_{i=1}^N [y^{(n)} \log(\hat{y}^{(n)}) + (1 - y^{(n)}) \times \log(1 - \hat{y}^{(n)})] \quad (2)$$

where  $\hat{y}^{(n)}$  is the estimated score for the input sample  $X^{(n)}$ .

As data are imbalanced for  $Q > 1$ , the model may reach a local optimum when predicting the non-GSS class for all input sequences. In order to deal with this issue, we attribute different weights to the positive and negative classes. We assign a greater importance to the less represented GSS class by multiplying the associated term in the loss by a weight  $CW = \frac{\text{number of non-GSS}}{\text{number of GSS}} = Q$ .

One of the important issues of any learning algorithm is overfitting. Overfitting occurs when one achieves a good fit of the model on the training and validation data, while it does not generalise well on new, unseen data. To deal with this issue, a regularisation procedure called dropout is usually used [20]. In the training step, some outputs of the pooling layers are randomly masked while the remaining information is fed as inputs for the next layer.

## Implementation

We implement CNN using Keras library and Tensorflow [21] as back-end. Training on a GPU is typically faster than on a CPU. We use here a GTX 1070 Ti GPU. We use Adaptive Moment Estimation (Adam) to compute adaptive learning rates for each parameter [22]. Adam optimiser is an algorithm for first-order stochastic gradient-based optimisation of functions, based on adaptive estimates of lower-order moments. The network architecture (see figure 1) is detailed in Table 1. The models are trained for 150 epochs and they mostly converge rapidly (around 30-35 epochs). Hyper-parameters tuning is detailed in the supplementary materials.

Source codes are available at <https://github.com/StudyTSS/DeepTSS/>.

## Genome wide performance measure

Different measures have been developed in order to assess the performance of different models on conventional test sets, i.e. test sets derived from a subset of the initial data. Such measures are described in details in the corresponding supplementary materials section. In our case, we want to apply our model on all the 299 bp windows spanning a full chromosome and eventually chromosomes from other species. We therefore developed a measure to evaluate the performance of the trained models in this case. This metric, called  $\lambda$ , measures the enhancement of the predicted signal specifically in the regions surrounding the known GSS. We use in the present papers regions of length  $r = 2000$  bp or 400 bp. To compute  $\lambda$ , we first compute the genome-wide Z-score [23]  $Z_g = \frac{y_g - \bar{\mu}}{\sigma}$  from the predictions  $y_g$  where  $g$  denotes positions on the genome, and  $\bar{\mu}$  and  $\sigma$  stand for the prediction mean and standard deviation, respectively. We extract  $\mathcal{Z}_{GSS}$ , the  $Z_g$  signal over 10 kb windows centred on each GSS of the test region, e.g. a full chromosome.  $Z_g$  is a 2D-array whose lines correspond to different genes and columns to different distances to the GSS. We then average element-wise  $\mathcal{Z}_{GSS}$  over all GSS, i.e. along all lines. This gives us  $S$ , the average of the Z-transformed prediction score in a 10 kb window around GSS. In order to measure the signal increase close to the GSS, that we call  $\lambda$ , we compute the average of the curve  $S$  on a region of  $r$  kbp centred on the GSS. A higher value of  $\lambda$  corresponds to a higher signal-to-noise ratio around the GSS.

**Table 1. Network architecture of the CNN model.** The first column depicts the different layers used consecutively in the network. The "layer shape" column reports the shape of the convolutional kernels, the max-pooling windows and the fully connected layers. The "output shape" column reports the variation of layer shapes at each step.

| Layer name      | Layer shape                         | Output shape             |
|-----------------|-------------------------------------|--------------------------|
| Input           | -                                   | $4 \times 299 \times 1$  |
| Conv2D          | $32 \times 4 \times (4 \times 1)$   | $32 \times 284 \times 1$ |
| Max-pooling     | $2 \times 1$                        | $32 \times 142 \times 1$ |
| Dropout         | -                                   | $32 \times 142 \times 1$ |
| Conv2D          | $64 \times 32 \times (4 \times 1)$  | $64 \times 127 \times 1$ |
| Max-pooling     | $2 \times 1$                        | $64 \times 63 \times 1$  |
| Dropout         | -                                   | $64 \times 63 \times 1$  |
| Conv2D          | $128 \times 64 \times (4 \times 1)$ | $128 \times 48 \times 1$ |
| Max-pooling     | $2 \times 1$                        | $128 \times 24 \times 1$ |
| Dropout         | -                                   | $128 \times 24 \times 1$ |
| Dense           | 128                                 | 128                      |
| Dropout         | -                                   | 128                      |
| Dense (sigmoid) | 1                                   | 1                        |

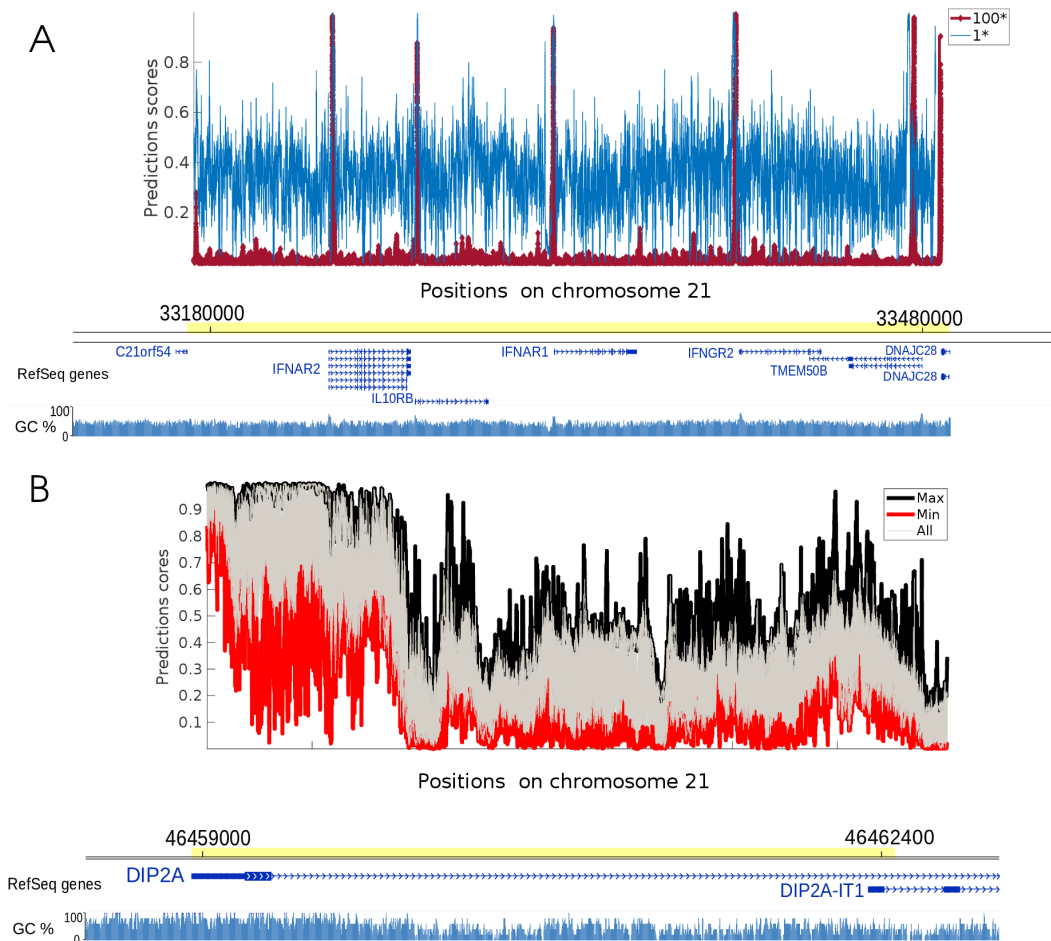
## RESULTS & DISCUSSION

### Training models for genome annotation of GSS

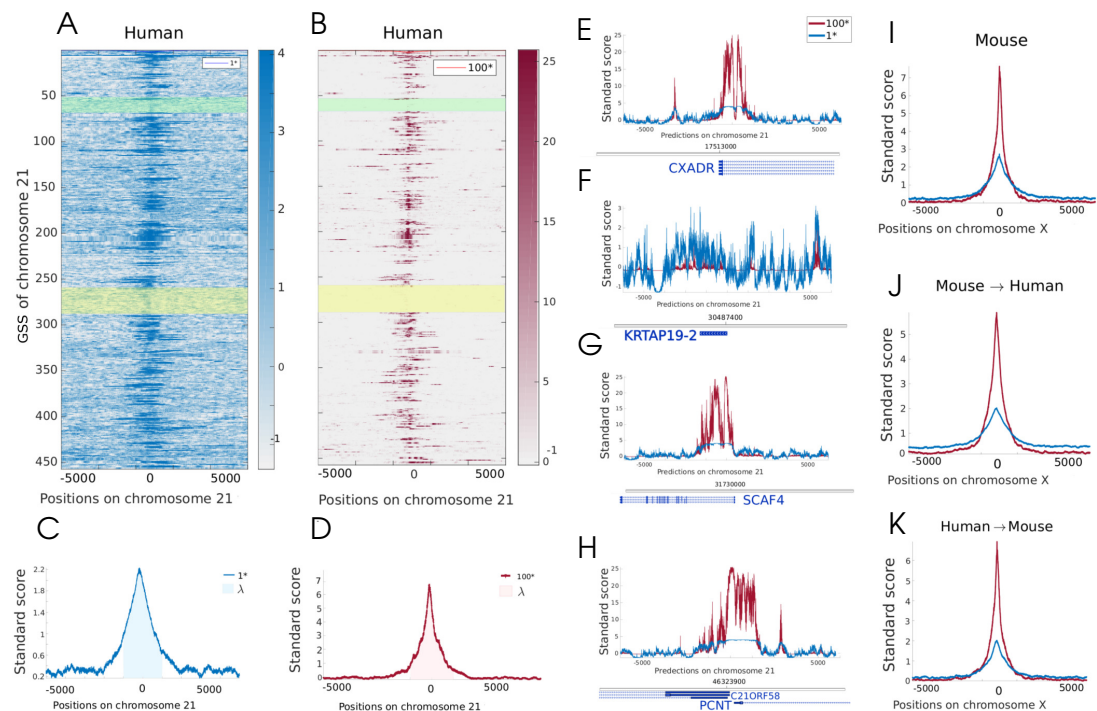
The problem of detecting human GSS using deep neural networks has been tackled in [11]. We first follow a similar approach and use a balanced dataset (see Methods for details). The model is trained/validated on an equal number of 299 bp long positive and negative examples and is evaluated on a test set composed of 15% of the input data. The specificity (Sp), the sensitivity (Sn) and the Matthews Correlation Coefficient (MCC) (see Supplementary materials for definition) were found to be similar to the ones found in [11] which used a similar approach albeit separating the sample data into TATA-containing GSS and non-TATA GSS (Sp = 0.94, Sn = 0.92 and MCC = 0.86).

In order to assess how this model would perform as a practical tool for detecting GSS on a genome-wide scale, we apply it on all the sequences along chromosome 21 (which has been withdrawn from the training set) obtained using a 299 bp long window sliding with an offset of 1 bp. Figure 2A illustrates the predictions of the CNN model over a typical region of 300 kbp containing 7 out of the 480 GSS of chromosome 21. Although the predictions yield higher scores over GSS positions, they also yield high scores over many non-GSS positions reflecting a low signal-to-noise ratio. This is due to the fact that the reality is biased in the training phase during which the CNN model learns an equal number of examples from the positive and the negative classes [24]. Applied over all the 299-bp sequences of chromosome 21, the model encounters many more examples of the negative class and fails to generalise inductive rules to the new examples.

To address this issue and train a network for genome annotation, we propose a heuristic approach. This approach consists in adding more negative examples into the balanced dataset to alleviating the importance of positive class in training phase and allocating more weight to the negative class. We call such datasets limited unbalanced datasets. We call  $Q$  the ratio between negative and positive training examples and denote as  $Q^*$  models trained with the corresponding ratio. For instance, on Figure 2A the model trained on the balanced data yielding to blue signal predictions is denoted as  $1^*$ . We train our CNN model on a  $100^*$  dataset ( $Q = 100$ ) and assess the efficiency of the trained model. As depicted on Figure 2A by a red signal, the predictions for this model display a much higher signal-to-noise ratio, with significant peaks over each of the 7 GSS (C21orf54, IFNAR2, IL10RB, IFNAR1, IFNGR2, TMEM50B, DNAJC28) and a much weaker signal between these sites. Predicting GSS using the  $100^*$  model is thus expected to generate much less false positives than the  $1^*$  model, regardless the value of the threshold used to identify GSS-containing regions. In order to assess how changing the value of  $Q$  affects GSS classification, we apply a threshold on the prediction and compute the precision and the recall obtained for both models (i.e.  $1^*$  and  $100^*$ ) at 600 bp resolution on a full chromosome. The precision recall curves



**Figure 2. CNN predictions for two regions of chromosome 21.** (A) Prediction scores for balanced model 1\* ( $Q = 1$ ) and unbalanced model 100\* ( $Q = 100$ ), respectively in blue and red on a 300 kb region. The position of genes is indicated below. The GC content (% of C or G bases in a 10 bp window) is indicated as blue bars below the genes. Both models detect effectively 7 GSS positions. The model 1\* returns a noisier prediction. Adding negative examples using the model 100\* mitigates the noise while preserving the high scores over GSS. (B) Application of 30 CNN models, trained on different datasets, over a 3.2 kb region of chromosome 21. At each site, the maximum and minimum prediction scores are respectively displayed in black and red. Other prediction scores are plotted in grey.



**Figure 3. Comparison of the 1\* and 100\* model predictions on chromosome 21.** (A) and (B) Heat maps depict the Z-score of the prediction for the 1\* and 100\* models respectively on 5000 bp flanking each GSS of chromosome 21. (C) and (D) Averaged Z-score of the predictions over each GSS of chromosome 21. (E-H) Zoom on regions around randomly selected GSS. Genes are indicated at the bottom of each plot. Averaged Z-score of the predictions over each GSS of mouse chromosome X for a network trained on the other mouse chromosomes (I) and for networks trained on mouse/human chromosomes (except X) and applied on human/mouse chromosome X (J,K).

confirmed the compromising effect of a lower signal-to-noise ratio on the accuracy of the classification (Supplementary Figure 1). For the sake of exhaustiveness, the performances of more models (1\*, 10\*, 20\*, 30\*, 50\*, 100\*) evaluated using conventional metrics, that is on test sets derived from the initial sample sets, can be found in Supplementary materials.

### Investigating the effect of random selection of the negative examples on predictions

While positive examples are always the same in different sample sets, the negative examples are randomly picked out of the genome. The performance of the model in different regions of chromosome 21 can thus vary for different training sets [25]. To investigate this variation, we set up 30 balanced 1\* datasets and train 30 CNN separately. The 30 models are then applied over human chromosome 21 to study the fluctuations of the predictions. The variation of 30 predictions is depicted in Fig 2B. The first observation is that almost all predictions present a peak over the DIP2A GSS. However, the large gap between the minimum and maximum predictions underlines the variability of predictions obtained with different training datasets. This variability illustrates the uncertainty of the predictions obtained from a single CNN trained on a balanced dataset and highlights the need to use limited unbalanced datasets for the task of genome annotation.

### Comparing 1\* and 100\* models over a full chromosome

Models trained on 1\* and 100\* sets are applied to the full chromosome 21 and the Z-normalized prediction scores around GSS are presented as heat-maps. While the model 1\* (Figure 3A) presents a noisy signal around GSS positions, the model 100\* (Figure 3B) presents a higher signal-to-noise ratio. To investigate the performance of different models on a genome-wide scale we devised a custom metric  $\lambda$  which measures the average signal-to-noise ratio around GSS (see Methods for the definition of  $\lambda$ ).

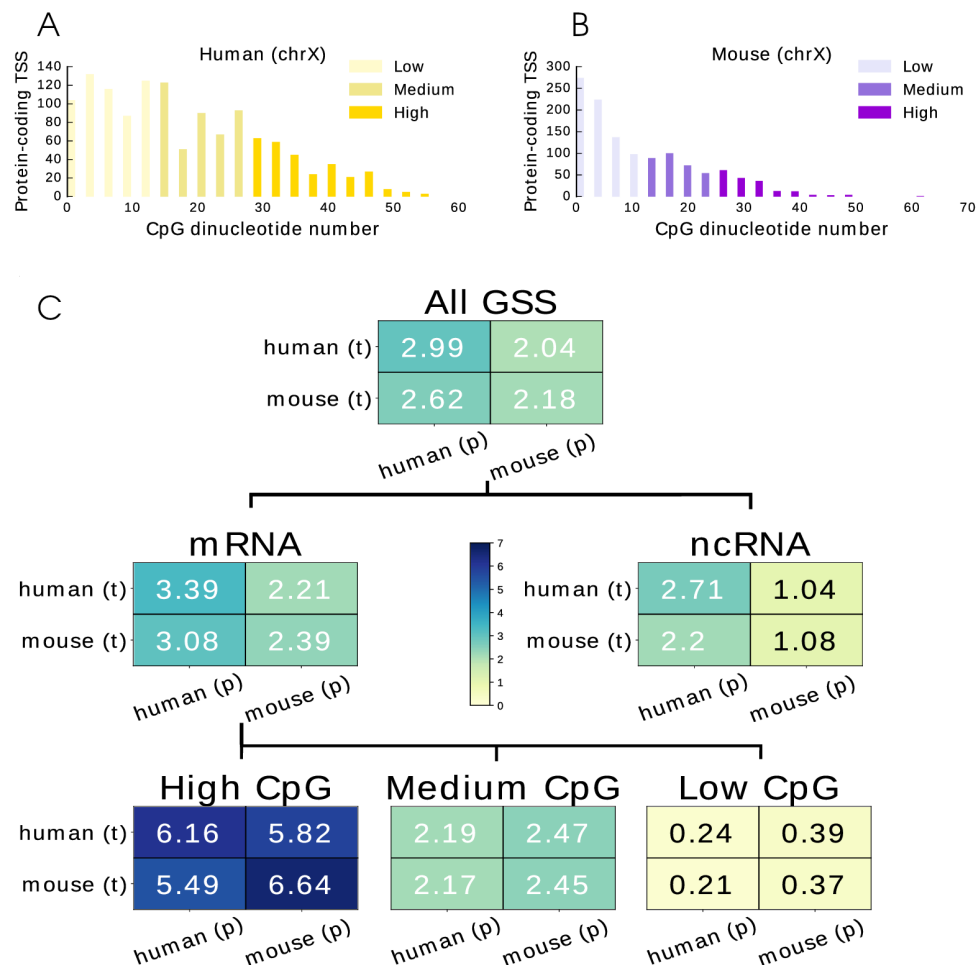
Figure 3C,D illustrate the average of the Z-score over all the GSS of chromosome 21 for the models 1\* and 100\*, respectively and  $\lambda$  denotes the average of this average over a  $r=2$  kb region centred on the GSS. A larger  $\lambda$  score corresponds to a higher signal-to-noise ratio. In this particular case, we find a  $\lambda$  score of 1.49 and 2.99 for the 1\* and 100\* model, respectively.

To illustrate the variability of prediction scores achieved around different GSS, we randomly selected four GSS within the chromosome. The first GSS corresponds to the gene CXADR, shown in Figure 3E. While the prediction of model 1\* results in a low averaged Z-scores over all positions, the averaged Z-score of model 100\* strongly peaks around the GSS position and shows low variations over non-GSS positions. Figure 3F depicts the second selected GSS corresponding to the KRTAP19-2 gene. This gene is part of a cluster of similar genes belonging to the family of Keratin Associated Proteins (highlighted by a yellow rectangle on Figure 3A,B). For this particular cluster, the predictions are poor for both 1\* and 100\*, probably reflecting a specific GSS signature that has not been grasped by the model. Another example of gene cluster with a poor prediction score for GSS is the t-RNA cluster, highlighted in green in Figure 3A,B. Figure 3G,H displays the predictions around the GSS of the SCAF4 and, PCNT and C21ORF58 genes, respectively. On these more typical GSS the 100\* model shows a higher signal-to-noise ratio than the 1\* and regions containing GSS are detected. These regions often stretch over 1 kb while our training sequence centred on each GSS is only 299bp long. This could indicate the presence either of alternative GSS close to the annotated GSS or of similar sequence patterns in broader regions surrounding the GSS [26, 27].

### Learning and predicting in human and mouse

To show the potential of our annotation method in a different context, we replicate a similar GSS analysis in mouse. Models with values of  $Q$  ranging from 1 to 100 trained on mouse chromosomes (except X) are applied over the mouse chromosome X to assess the model performance (see Figure 3I, Supplementary Figure 2 and Supplementary Figure 3a,d,g). The averaged Z-score of  $\lambda$  reaches values of 1.47 and 2.18 respectively for the 1\* and 100\* models in quantitative agreement with the model performance in human.

Mammals show a substantial degree of homologies in the DNA sequence found at GSS and earlier computational models were trained to recognise transcription start site in any mammalian species [28]. Following this line, we next determine the possibility of predicting GSS in one organism with a network trained on a related organisms. This possibility has previously been shown to be effective for sequence variants calling [29] To this end, the mouse trained model is applied on human chromosome X and the human trained model is applied on mouse chromosome X. The two chromosomes carry homologous genes, the number of annotated GSS varies with a total of 4,968 GSS in human and 2,005 GSS in mouse.



**Figure 4. Evaluation of the model performance for different RNA classes.** (A) CG di-nucleotide (CpG) number in 299bp regions centred on mRNA-GSS in test chromosomes (X). Three populations (low, medium and high) can be identified. (B) Each box contains the  $\lambda$  values obtained when training the network on the organism corresponding to each line and predicting the GSS on the chromosome X of the organism corresponding to the column;  $\lambda$  scores are also computed for mRNA- and ncRNA-GSS separately and finally for mRNA-GSS divided into three classes based on the CpG density populations identified in (A).

While the model trained and applied on mouse shows a better signal-to-noise ratio, the same model applied to human chromosome X still captures most of the GSS and gives a  $\lambda$  score of 2.28 for the 100\* model (see Figure 3J and Supplementary Figure 3b,e,h). Similarly, the models trained on human capture most of GSS on the mouse X chromosome as shown in Figure 3K and Supplementary Figure 3c,f,i and reaches a  $\lambda$  score of 2.04 for the 100\* model. In all cases, the signal-to-noise ratio is improved in the 100\* models with respect to the 1\* models. The values of  $\lambda$  for the cross-species comparison on chromosome X are summed up in Supplementary Figure 2. The human model applied on human provides the highest scores for both 1\* and 100\* models probably a signature of an overall better GSS annotation. In all cases,  $\lambda$  gradually increases for 10\* up to 100\* models. Since the performance of models 30\* and 100\* varies slightly, the 30\* model can be used instead of 100\* to perform cost-effective computations.

## Evaluation of the prediction for different GSS classes

The potential of our trained networks to recover GSS containing regions along the human and mouse genomes is assessed in the previous parts without any distinction between different GSS classes. Since we find that some GSS are better predicted than others (Figure 3), we compute the  $\lambda$  score independently for the two main classes of GSS: mRNA-GSS and ncRNA-GSS. While  $\lambda$  is higher for the mRNA-GSS class, the model is versatile and is also able to predict the ncRNA-GSS (Figure 4B). In human and mouse, mRNA-GSS are found in different classes, that can be derived from the CG di-nucleotide (CpG) content of the region flanking the GSS. High CpG regions, also called "CpG island" can be methylated and play an important role in gene regulation [30]. Figure 4A displays the distribution of the CpG number in 299 bp regions surrounding the all mRNA-GSS for the mouse and human X chromosome. From this distribution, we identify three classes of mRNA-GSS with respectively a high, medium and low CpG content. High CpG GSS correspond to genes regulated by DNA methylation and have been shown to exhibit a different pattern of chromatin modifications [31]. Assessing the performance of the model for the three different classes, we find that stronger scores are obtained for CpG richer GSS (Figure 4B). The worst performing GSS are low CpG content GSS which are hardly recovered by our model.

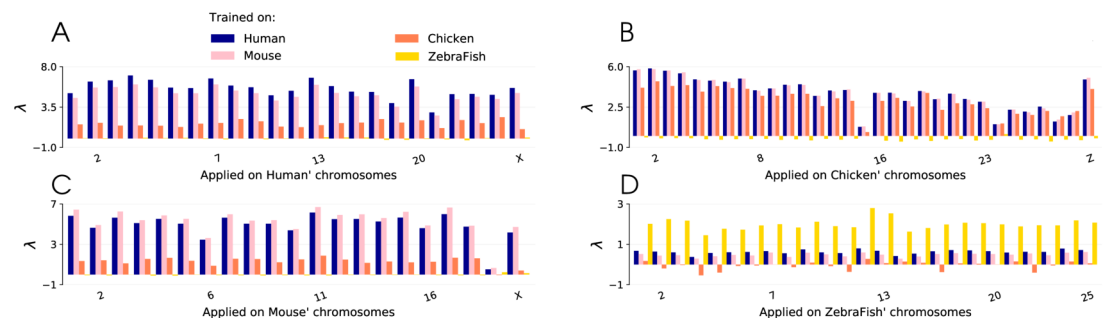
## Application of the approach to other vertebrates

The performance of a CNN trained on human GSS to recover mouse GSS is not surprising given the similarity between the two mammalian genomes [32]. We next set out to apply the same methodology on more diverse species, including a bird and a fish (Figure 5). Four CNN are trained on all the GSS of the Human, Mouse, Chicken and Danio rerio (Zebrafish) genomes which provide the most comprehensive GSS annotations for mammals, birds and fishes. These four CNN are then applied genome wide on each of the four species and the  $\lambda$  metric is computed for each chromosome independently, using a  $r$  value of 400 bp (see Methods). Using a smaller value for  $r$  leads to higher absolute values for  $\lambda$  so that one should only compare values computed using the same  $r$ . The conclusions drawn however do not depend on the specific value chosen.

The results for the human and mouse genomes are very similar, with only a slightly better performance when the model trained on a species is applied on the same species. The model trained on the Chicken genome performs less well when applied on the mammalian genomes and the model trained on the Zebrafish genome is not able to recover the mammalian GSS as shown by a  $\lambda$  value of 0.

When applied on the Chicken genome, the mouse and human models surprisingly outperform the chicken model, probably because the GSS annotation is better in the two mammals so that the training phase is more efficient. This result highlights the potential of the method when used across different species when the genome of one species is more precisely annotated.

When applied on the Zebrafish genome on the other hand, the human, mouse and chicken models all show poor performances while the Zebrafish model performs well. This is in line with the fact that the CpG composition of Zebrafish regions around GSS is very different than in birds and mammals. CpG islands, which are high density CpG regions, are found upstream many GSS for coding genes in birds and mammals while they are absent in fishes. All together, these results suggest that the molecular machinery that is able to interpret the genome sequence in order to find the start sites of genes has a similar activity in human, mouse and chicken but a different activity in fishes.



**Figure 5. Lambda scores obtained with CNN trained on four different species: Human, Mouse, Chicken and Zebrafish.** Lambda scores are computed from GSS predictions done on (A) human, (B) mouse, (C) Chicken and (D) Zebrafish chromosomes. The size of the window  $r$  on which  $\lambda$  is computed is equal to 400 bp.

## CONCLUSIONS

With the surge of DNA sequencing technologies, one million of genome datasets are now available and millions of gigabases are sequenced every year to annotate these datasets with functional marks [33]. It has not escaped the notice of many computational biologists that deep neural networks are a key tool to deal with this exponentially increasing amount of data [33]. One possible application is to leverage datasets with good annotations in order to train neural networks and to predict annotations on other datasets. One of the practical issues when applying neural networks on genomic sequences is the unbalanced data, a well-known issue in the machine learning literature [24, 34, 35]. In the present paper, we address this problem using GSS as a case study. Indeed, GSS occupy only a few locations on the genome (31,037 GSS for human) leading to extreme unbalances in datasets (i.e., the ratio of GSS-containing 299 bp windows to non-GSS in the human genome is 1/400). In this case, the lack of examples of the minority class (i.e., true GSS) deteriorates the learning process as conventional machine learning algorithms usually measure the model performance on the majority class (i.e., non-GSS) leading to biased or inaccurate prediction of the minority class. To deal with this disparity, we adopt a weighting strategy to decrease the importance of the majority class samples (non-GSS) during the learning process, improving thereby identification of the rare minority class samples (GSS). Using this approach, we show that learning on imbalanced datasets can be performed effectively and that a ratio of 1 to 30 positive over negative examples is usually sufficient to achieve a good signal to noise ratio in the prediction. This approach can be easily extended to identify other functional regions in any annotated genome.

We also show that our method can be efficiently used across genomes of different species, i.e. training the model on one genome and applying it to another genome. We use human and mouse GSS as case study and first apply both models on chromosome X of each organism. While the sequence of this chromosome has evolved differently in both species, many genes are homologous [36]. The fact that we are able to recover GSS in mouse/human with a model trained in the other organism, suggests that the machinery capable of recognising GSS in each organism is overall conserved. We also show that this methodology can be applied to more distant species, and use as examples a bird and a fish species. Our results point toward a higher similarity between mammal and bird species while fish species GSS cannot be efficiently predicted from mammal and bird sequences. While the genome sequence conservation can be computed directly from DNA sequences, further developments of our method may provide a new tool to address the conservation of the activity of the nuclear machinery that interprets the DNA sequences *in vivo*.

## ACKNOWLEDGEMENTS

We would like to thank Léopold Carron for helping us with datasets, Hugues Roest Croeluis for discussions and Michel Quaggetto for technical support.

# 342 ADDITIONAL INFORMATION

## 343 Data availability statement

344 Genome sequences data and gene annotation are available at <http://hgdownload.soe.ucsc.edu/> and  
345 <http://egg.wustl.edu/>.

## 346 Conflict of interest statement

347 The authors declare no competing interests.

## 348 Funding

349 This work was supported by the Agence Nationale pour la Recherche [HiResBac ANR-15-CE11-0023-03].

# 350 REFERENCES

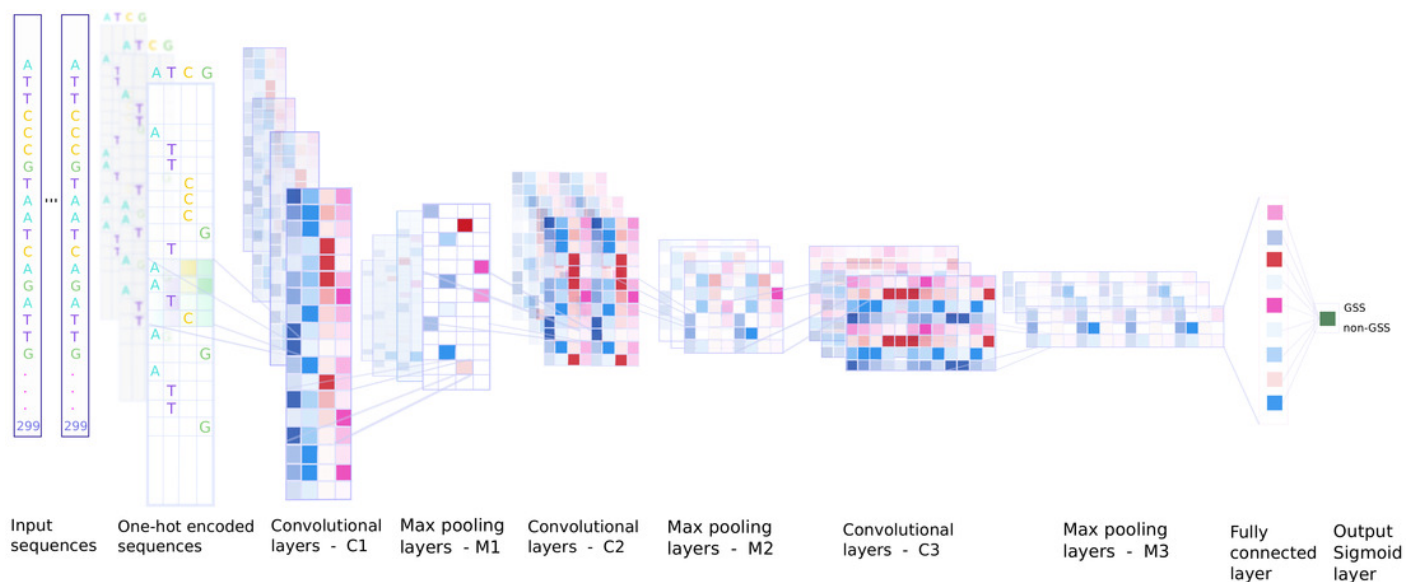
- 351 [1] Lincoln Stein. Genome annotation: from sequence to biology. *Nature reviews genetics*, 2(7):493–503,  
352 2001.
- 353 [2] Chloe M Rivera and Bing Ren. Mapping human epigenomes. *Cell*, 155(1):39–55, 2013.
- 354 [3] Anshul Kundaje, Wouter Meuleman, Jason Ernst, Misha Bilenky, Angela Yen, Alireza Heravi-  
355 Moussavi, Pouya Kheradpour, Zhizhuo Zhang, Jianrong Wang, Michael J Ziller, et al. Integrative  
356 analysis of 111 reference human epigenomes. *Nature*, 518(7539):317, 2015.
- 357 [4] ENCODE Project Consortium et al. An integrated encyclopedia of dna elements in the human  
358 genome. *Nature*, 489(7414):57, 2012.
- 359 [5] Timothy J Durham, Maxwell W Libbrecht, J Jeffry Howbert, Jeff Bilmes, and William Stafford  
360 Noble. Predictd parallel epigenomics data imputation with cloud-based tensor decomposition. *Nature*  
361 *communications*, 9(1):1–15, 2018.
- 362 [6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.
- 363 [7] James Zou, Mikael Huss, Abubakar Abid, Pejman Mohammadi, Ali Torkamani, and Amalio Telenti.  
364 A primer on deep learning in genomics. *Nature genetics*, page 1, 2018.
- 365 [8] Christof Angermueller, Tanel Pärnamaa, Leopold Parts, and Oliver Stegle. Deep learning for  
366 computational biology. *Molecular systems biology*, 12(7):878, 2016.
- 367 [9] Travers Ching, Daniel S Himmelstein, Brett K Beaulieu-Jones, Alexandr A Kalinin, Brian T Do,  
368 Gregory P Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M Hoffman, et al.  
369 Opportunities and obstacles for deep learning in biology and medicine. *Journal of The Royal Society*  
370 *Interface*, 15(141):20170387, 2018.
- 371 [10] Xu Min, Ning Chen, Ting Chen, and Rui Jiang. Deepenhancer: Predicting enhancers by convolutional  
372 neural networks. In *Bioinformatics and Biomedicine (BIBM), 2016 IEEE International Conference*  
373 *on*, pages 637–644. IEEE, 2016.
- 374 [11] Ramzan Kh Umarov and Victor V Solovyev. Recognition of prokaryotic and eukaryotic promoters  
375 using convolutional deep learning neural networks. *PloS one*, 12(2):e0171410, 2017.
- 376 [12] Babak Alipanahi, Andrew Delong, Matthew T Weirauch, and Brendan J Frey. Predicting the sequence  
377 specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831,  
378 2015.
- 379 [13] Jian Zhou and Olga G Troyanskaya. Predicting effects of noncoding variants with deep learning–based  
380 sequence model. *Nature methods*, 12(10):931, 2015.
- 381 [14] David R Kelley, Jasper Snoek, and John L Rinn. Basset: learning the regulatory code of the accessible  
382 genome with deep convolutional neural networks. *Genome research*, 26(7):990–999, 2016.
- 383 [15] Stepan Pachganov, Khalimat Murtazalieva, Aleksei Zarubin, Dmitry Sokolov, Duane R Chartier,  
384 and Tatiana V Tatarinova. Transprise: a novel machine learning approach for eukaryotic promoter  
385 prediction. *PeerJ*, 7:e7990, 2019.
- 386 [16] Michael KK Leung, Hui Yuan Xiong, Leo J Lee, and Brendan J Frey. Deep learning of the tissue-  
387 regulated splicing code. *Bioinformatics*, 30(12):i121–i129, 2014.
- 388 [17] Kishore Jaganathan, Sofia Kyriazopoulou Panagiotopoulou, Jeremy F McRae, Siavash Fazel Darbandi,  
389 David Knowles, Yang I Li, Jack A Kosmicki, Juan Arbelaez, Wenwu Cui, Grace B Schwartz, et al.  
390 Predicting splicing from primary sequence with deep learning. *Cell*, 2019.
- 391 [18] David R Kelley, Yakir Reshef, Maxwell Bileschi, David Belanger, Cory Y McLean, and Jasper Snoek.  
392 Sequential regulatory activity prediction across chromosomes with convolutional neural networks.  
393 *Genome research*, pages gr–227819, 2018.

- 394 [19] Jennifer F Kugel and James A Goodrich. Finding the start site: redefining the human initiator element.  
395 *Genes & development*, 31(1):1–2, 2017.
- 396 [20] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov.  
397 Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning*  
398 *Research*, 15(1):1929–1958, 2014.
- 399 [21] Martín Abadi, Ashish Agarwal, and Paul Barham. TensorFlow: Large-scale machine learning on  
400 heterogeneous systems, 2015. Software available from tensorflow.org.
- 401 [22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint*  
402 *arXiv:1412.6980*, 2014.
- 403 [23] Erwin Kreyszig. Advanced engineering mathematics, 10th eddition, 2009.
- 404 [24] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge*  
405 *and data engineering*, 21(9):1263–1284, 2009.
- 406 [25] Agata Wesolowska-Andersen, Grace Zhuo Yu, Vibe Nylander, Fernando Abaitua, Matthias Thurner,  
407 Jason M Torres, Anubha Mahajan, Anna L Gloyn, and Mark I McCarthy. Deep learning models  
408 predict regulatory variants in pancreatic islets and refine type 2 diabetes association signals. *eLife*,  
409 9:e51503, 2020.
- 410 [26] Piero Carninci, Albin Sandelin, Boris Lenhard, Shintaro Katayama, Kazuro Shimokawa, Jasmina  
411 Ponjavic, Colin AM Semple, Martin S Taylor, Pär G Engström, Martin C Frith, et al. Genome-wide  
412 analysis of mammalian promoter architecture and evolution. *Nature genetics*, 38(6):626–635, 2006.
- 413 [27] Albin Sandelin, Piero Carninci, Boris Lenhard, Jasmina Ponjavic, Yoshihide Hayashizaki, and  
414 David A Hume. Mammalian rna polymerase ii core promoters: insights from genome-wide studies.  
415 *Nature Reviews Genetics*, 8(6):424–436, 2007.
- 416 [28] Thomas A Down and Tim JP Hubbard. Computational detection and location of transcription start  
417 sites in mammalian genomic dna. *Genome research*, 12(3):458–461, 2002.
- 418 [29] Ryan Poplin, Pi-Chuan Chang, David Alexander, Scott Schwartz, Thomas Colthurst, Alexander Ku,  
419 Dan Newburger, Jojo Dijamco, Nam Nguyen, Pegah T Afshar, et al. A universal snp and small-indel  
420 variant caller using deep neural networks. *Nature biotechnology*, 36(10):983–987, 2018.
- 421 [30] Aimée M Deaton and Adrian Bird. CpG islands and the regulation of transcription. *Genes &*  
422 *development*, 25(10):1010–1022, 2011.
- 423 [31] Tanya Vavouri and Ben Lehner. Human genes with cpG island promoters have a distinct transcription-  
424 associated chromatin organization. *Genome biology*, 13(11):R110, 2012.
- 425 [32] Robert H Waterston, Kerstin Lindblad-Toh, Ewan Birney, Jane Rogers, Josep F Abril, Pankaj Agarwal,  
426 Richa Agarwala, Rachel Ainscough, Marina Alexandersson, Peter An, et al. Initial sequencing and  
427 comparative analysis of the mouse genome. *Nature*, 420(6915):520–562, 2002.
- 428 [33] Michael Wainberg, Daniele Merico, Andrew Delong, and Brendan J Frey. Deep learning in  
429 biomedicine. *Nature biotechnology*, 36(9):829, 2018.
- 430 [34] Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Special issue on learning from  
431 imbalanced data sets. *ACM Sigkdd Explorations Newsletter*, 6(1):1–6, 2004.
- 432 [35] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of  
433 several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*,  
434 6(1):20–29, 2004.
- 435 [36] Amit U Sinha and Jaroslaw Meller. Cinteny: flexible analysis and visualization of synteny and  
436 genome rearrangements in multiple organisms. *BMC bioinformatics*, 8(1):82, 2007.

# Figure 1

Overview of the CNN model.

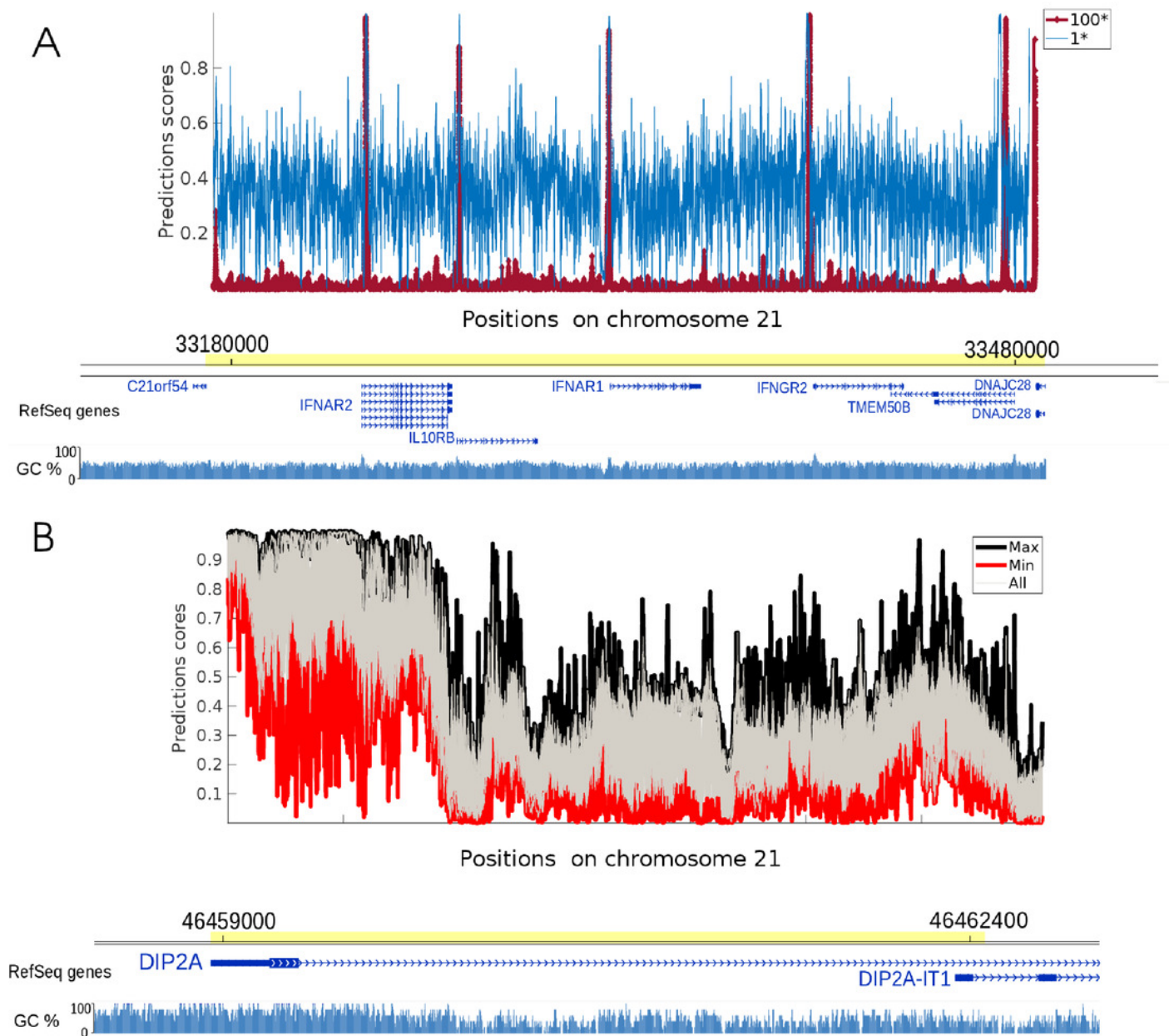
299 bp-long sequences are one hot encoded into a  $4 \times 299$  input matrix. The first CNN layer performs a convolution on each input matrix to recognise relevant motifs. The next convolutional layers models the interplay among these motifs to grasp higher-level features. Max-pooling layers reduce the dimensions of the layers. The model is trained to correctly label input sequences as GSS or non-GSS. The output layer of the trained network then gives a probability for any 299 bp region to contain a GSS. It can be applied along a full chromosome, i.e. on all 299 bp-long sequences with a 1 bp shift.



# Figure 2

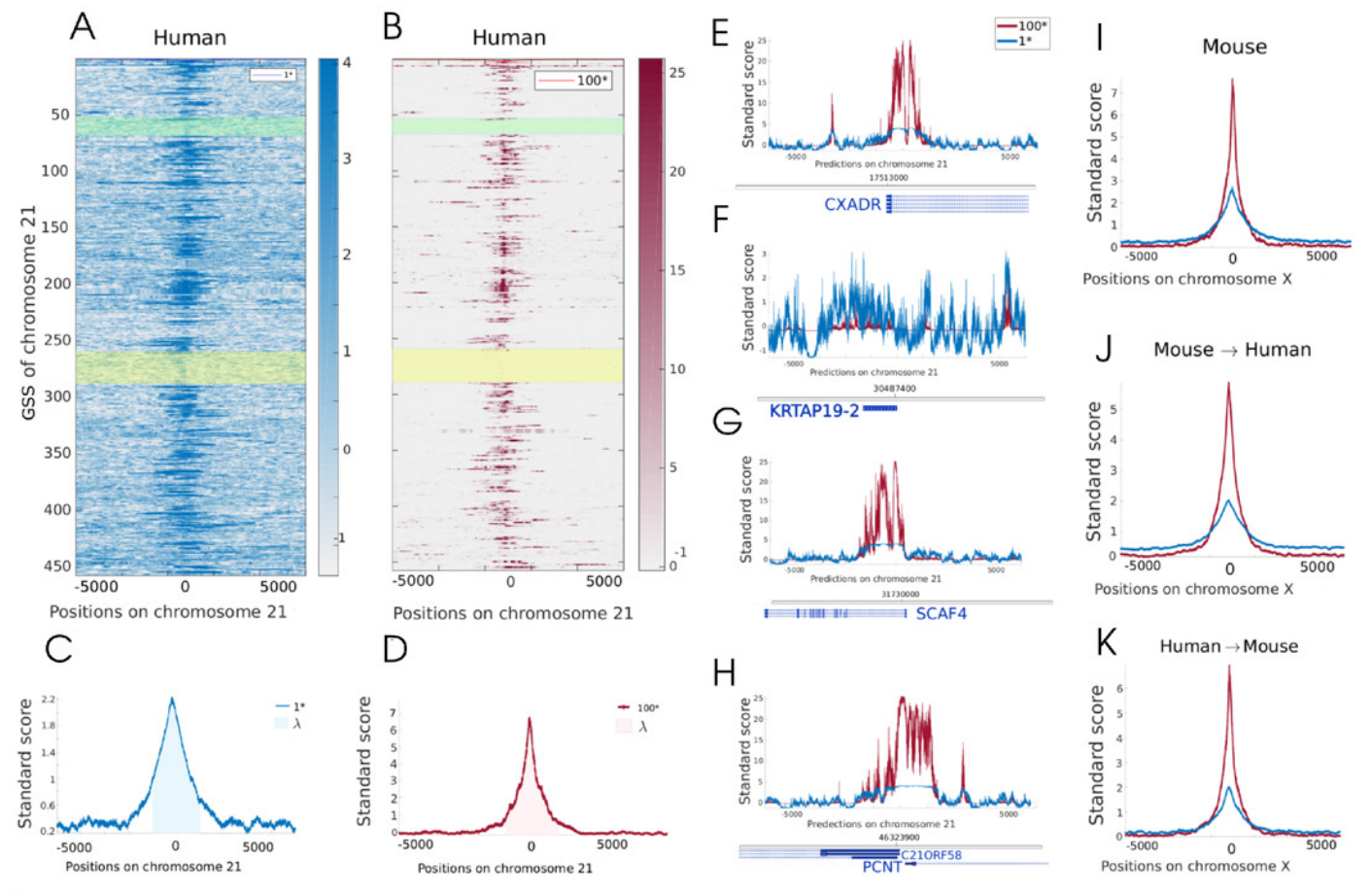
CNN predictions for two regions of chromosome 21

(A) Prediction scores for balanced model 1\* ( $Q=1$ ) and unbalanced model 100\* ( $Q=100$ ), respectively in blue and red on a 300 kb region. The position of genes is indicated below. The GC content (% of C or G bases in a 10 bp window) is indicated as blue bars below the genes. Both models detect effectively 7 GSS positions. The model 1\* returns a noisier prediction. Adding negative examples using the model 100\* mitigates the noise while preserving the high scores over GSS. (B) Application of 30 CNN models, trained on different datasets, over a 3.2 kb region of chromosome 21. At each site, the maximum and minimum prediction scores are respectively displayed in black and red. Other prediction scores are plotted in grey.



# Figure 3

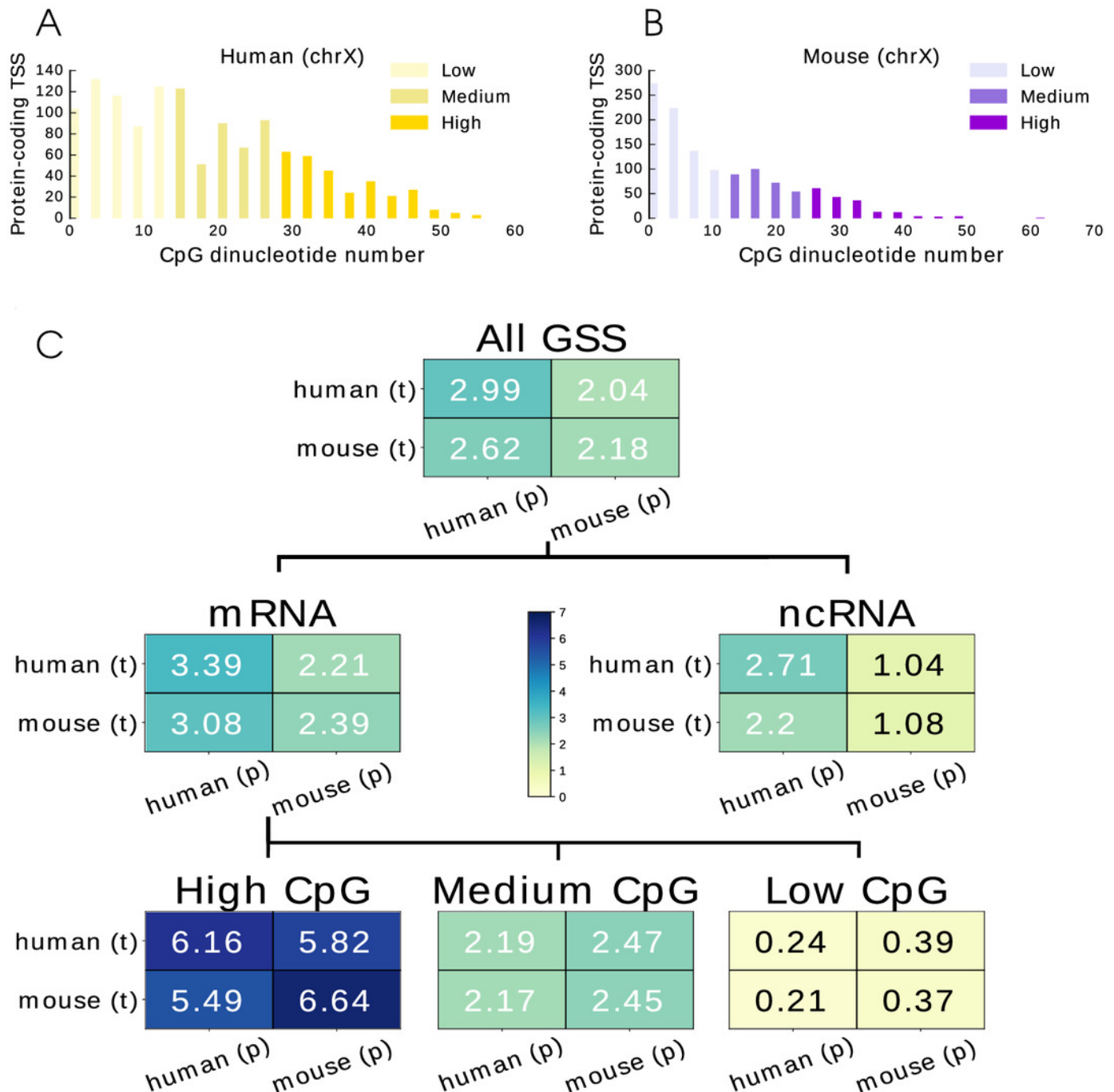
Comparison of the 1\* and 100\* models predictions over chromosome 21



# Figure 4

Evaluation of the model performance for different RNA classes

(A) CG di-nucleotide (CpG) number in 299bp regions centred on mRNA-GSS in test chromosomes (X). Three populations (low, medium and high) can be identified. (B) Each box contains the  $\lambda$  values obtained when training the network on the organism corresponding to each line and predicting the GSS on the chromosome X of the organism corresponding to the column;  $\lambda$  scores are also computed for mRNA- and ncRNA-GSS separately and finally for mRNA-GSS divided into three classes based on the CpG density populations identified in (A)



# Figure 5

Lambda scores obtained with CNN trained on four different species: Human, Mouse, Chicken and Zebrafish

Lambda scores are computed from GSS predictions done on (A) human, (B) mouse, (C) Chicken and (D) Zebrafish chromosomes. The size of the window  $r$  on which  $\lambda$  is computed is equal to 400 bp.

