Deep Learning concepts for genomics: an overview

Merouane Elazami Elhassani, Loic Maisonnasse, Antoine Olgiati, Rey Jerome, Patrice Duroux, Veronique Giudicelli, Sofia Kossida

Abstract

Nowadays, Deep Learning is taking the world by a storm, known as a technology that makes use of Artificial Neural Networks to automatically extrapolate knowledge from a training data set, then uses this knowledge to give predictions for unseen samples. This data driven paradigm gained a widespread adoption in many disciplines, from handwriting recognition, driving an autonomous car to cracking the 50-year-old protein folding problem. With this review, we shed some light on the concepts of Deep Learning and provide some visualizations, skim over the different architectures such as Deep Neural Network (DNN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and touch upon the modern architectures such as Transformers and BERT. We also provide various examples targeting the genomics field, reference utilities, libraries useful for newcomers and disseminate our feedback.

***Keywords*—** Genomics, Artificial Intelligence (AI), Machine Learning (ML), Deep Learning (DL), Deep Neural Network (DNN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Generative Adversarial Network (GAN), Transformers, Attention, Bidirectional Encoder Representations from Transformers (BERT).

Introduction

In few years, Deep Learning (LeCun *et al.*, 2015), a subset of Machine Learning (Figure 1), has become one of the the most successful and promising technologies, as it was able to outperform the countless methods and approaches across many fields and in diversified tasks. At the core of this paradigm shift are the "Artificial Neurons". Initially, they were conceptualized to understand and mimic the physiology and functioning of the human brain, in a computational manner (McCulloch and Pitts, 1943). Interconnecting those neurons produced Artificial Neural Networks (ANNs). Over the past decade, a wide variety of Neural Network architectures were designed (DNN, RNN, CNN, GAN...). So far, they are contentiously flourishing (Transformer, GPT, BERT...). Consequently, new state-of-the-art performances are continuously achieved with endless opportunities.

Figure 1. Machine Leaning techniques fall under Artificial Intelligence umbrella. Deep Learning is a category of Machine Learning that relays heavily on Artificial Neural Networks.

Machine Learning

Machine Learning algorithms are a subset of Artificial Intelligence. They are suitable for problems such as (Computer Vision, Voice Recognition, Face Recognition ...) that traditional programming paradigm may not solve, due to their complexities and high variability. In general, Machine Learning entails four categories of learning: Supervised Learning, Unsupervised Learning, Semi-supervised Learning and Reinforcement Learning (Figure 2).

Figure 2. *Machine Learning comprises four categories of learning: Supervised Learning, Unsupervised Learning, Semi-supervised Learning and Reinforcement Learning.*

Supervised Learning

A Supervised Learning (Kotsiantis *et al.*, 2006) approach consists of using the past experience data to train a Machine Learning Model to predict future ones, with respect to their classes. For instance, DECRES (Li *et al.*, 2018) is an example of Supervised Learning approach for genome-wide prediction of cis-regulatory elements. It delineated locations of 300,000 candidate enhancers genome wide and 26,000 candidate promoters (0.6% of the genome). MPRA-DragoNN (Movva *et al.*, 2019) is another example, used for deciphering regulatory DNA sequences and noncoding genetic variants. It employs a (CNN)-based architecture to predict and interpret the regulatory activity of DNA sequences as measured by MPRAs. For a pedagogical purpose and to explain this concept, let us imagine a fictional system that tries to predict, for a provided DNA sequence as input, it is either an exon or an intron. This kind of problematic is called a "Binary classification problem", in which the model tries to predict from a given input, which class of output it belongs to. With enough labelled data available, one can train a Deep Learning Model in supervised manner to resolve this classification problem, (Figure 3). A Supervised Learning approach relies heavily on the labelled samples within the dataset, they teach the Deep Learning model how to differentiate between the various classes of the output to make accurate predictions.

Figure 3. *An example of Supervised Learning, in which the model is thought to classify the type of sequences (exon or intron). The model is trained using a labelled training dataset, then assessed using the testing dataset to validate the model performance.*

Unsupervised Learning

An Unsupervised Learning (Ghahramani, 2004) occurs when the available data are not labelled at all. Its key concept is to find clues or features to cluster data and reveal their hidden relationships. This category of learning comprises different learning families: Clustering (Figure 4), Dimensionality Reduction and Generative algorithms. For instance, clustering the single-cell RNA-seq data (Kiselev *et al.*, 2019) discusses the challenges and strategies used to cluster the RNA-seq.

Figure 4*. An example of Unsupervised Learning (Clustering), in which the model does not have any clues about the input, but it was able to distinguish 3 distinct clusters of sequence data. The* *samples from each group share a certain number of features, thus they appear close to each other.*

Semi-supervised Learning

Semi-supervised Learning lays in the middle between Supervised and Unsupervised Learning. It is often used when the available data are partially labelled. For instance, the Semi-supervised Learning was employed to classify microRNA in order to maximize the utility of both labelled and unlabelled data (Sheikh Hassani and Green, 2019). The results outperformed the state-of-the-art miRDeep2 (Friedländer *et al.*, 2012) and miPIE (Peace *et al.*, 2018) methods, with an improved performance of 8.3% and 4.2% in average AUPRC.

Reinforcement Learning

In contrast to the other learning algorithms, the Reinforcement Learning (RL) (Sutton and Barto, 2018) has the particularity to be suited for unknown environments. RL was used for drug design (Popova *et al.*, 2018), genome Assembly (Xavier *et al.*, 2020), protein interaction network constructing (F *et al.*, 2015), where an infinite number of possibilities exist. RL formalizes the problem as a learning agent that interacts with its environment through a series of selected actions. Each action generates a new state that may impact the environment. The agent tries to learn how to interact as well as possible with this environment by selecting the best possible actions, which are rewarded as good or bad, depending on the newly generated state and their impacts on this environment, (Figure 5).

Figure 5. *Reinforcement Learning agent interacting with its environment. As the agent emits an action (A_i), the environment produces a new state ($S_i + 1$) and a new reward ($R_i + 1$).*

Deep Learning

Deep Learning is a subset of Machine Learning that demonstrated a great potential in several areas. In genomics for instance, it was used to learn and represent the hierarchical organization of yeast transcriptomics machinery (Chen *et al.*, 2016), to understand gene regulation (Singh *et al.*, 2017), to predict enhancer-promoter interaction from genomic sequence (Singh *et al.*, 2019), to create artificial human genomes using generative models (Yelmen *et al.*, 2019), to predict cell type specific transcription factor binding from nucleotide-resolution sequential data (Quang and Xie, 2019), to model and design protein structure (Gao *et al.*, 2020) and so on. In essence, DL is characterized by the use of Artificial Neurons that serve as the building blocks for the different Neural Network architectures.

A brief history

Deep Learning is not new, many of its concepts date back to more than half a century. Initially, (McCulloch and Pitts, 1943) put the first brick and proposed a simplified version of a neuron, as an attempt to understand how the brain could produce very complex patterns using only simple interconnected cells (biological neurons). (Rosenblatt, 1957) proposed the

Perceptron, a simplified neuron which had true learning abilities for doing a binary classification. Afterwards, came the first Feedforward Multilayered Neural Networks (Ivakhnenko and Lapa, 1966) (interconnected layered neurons). A breakthrough was made by (Le Cun *et al.*, 1989), who was able to train a Convolutional Neural Network (CNN) named LeNet to recognize handwritten digits. Decades later, a Deep Convolutional Neural Networks model "AlexNet" (Krizhevsky *et al.*, 2012) made a quantum leap in computer vision. It won the ImageNet Large Scale Visual Recognition Challenge 2012 with a phenomenally great margin. Also, it demonstrated for the first time, the automatic feature learning aspect. Recently, DeepMind (Senior *et al.*, 2020) was able to crack the 50-year old Protein Folding Prediction problem, which is another milestone for understanding biology using Artificial Intelligence.

Underlying Concepts

Artificial Neurons

Artificial Neurons or simply neurons are the backbone of the Deep Learning technology. A neuron is comprised of a single input layer and one output node, the input layer contains n nodes that transmit n features. The output y is calculated using the inputs and their weights (i.e. the weighted sum), where each x_i from the input vector X = $[x_1, \dots, x_n]$ is multiplied respectively with its w_i from the learning weight vector W = $[w_1, \dots, w_n]$, an additional bias variable b is added to capture the invariant part of the prediction. An activation function A is applied to the weighted sum which decides whether the output of this neuron should be activated or not, (Figure 6).

$$\hat{y} = A(W.X + b) = A\left(\sum_{i=1}^{n} w_i \, . \, x_i + b\right)$$

Figure 6. *A single neuron comprised of inputs and their respective weights, the bias, the arithmetic operation, the activation function and the output, mathematically expressed with the Formula ([formula:neuron]).*

Activation function

The main idea behind the activation function (Nwankpa *et al.*, 2018) is adding a non-linearity to the neural network, which allows the network to capture more complex pattern inherent within the data. The choice of this function impacts profoundly the design of a neural network, it is a task-specific. For instance, a binary classification will definitely have a different activation function then a multi-class probability classification. A multitude of activation functions exist, (Nwankpa *et al.*, 2018) compiles majority of them and outlines the current trends in the applications and usage of these functions in practical deep learning deployments against the state-of-the-art research results.

Loss function

One of the most important questions in designing a Neural Network architecture is how to gauge its performance. The loss or cost function is the answer. It quantifies how the Neural Network model is performing by calculating the difference between the output y and the predicted output ŷ. In other words, it measures how far or close the predictions are from the excepted values. Many loss functions exist, selecting a particular one impacts profoundly the learning process of the Neural Network (Hennig and Kutlukaya). Advanced analyses were conducted by (Wang *et al.*, 2020) that summarize and analyse 31 classical loss functions in Machine Learning.

Optimization

At the outset, most of the Neural Network models will not have the best performances (a poor prediction and very high Loss). During the training, an optimizer algorithm is configured with the model. Its goal is to minimize the loss function and maximize the prediction by tuning parameters of the model in response to the output of the loss function, (Figure 7). Several optimization algorithms exist, (Choi *et al.*, 2020) empirically compared them.

Figure 7. *The optimizer looks to minimize the loss function by tuning the weights of the neurons.*

Neural Networks

A single neuron may not be enough to learn all the necessary features for complex tasks. Interconnecting those neurons produces advanced architectures called Neural Networks, as they are capable of learning more complex patterns than a single neuron, (Figure 8). A Neural Network is organized in form of layers, each layer contains a number of neurons. The layers in between the input and output layers are called the hidden layers. A Neural Network that has only one or two hidden layers is named a "Shallow Neural Network" as opposed to a "Deep Neural Network" which has several hidden layers. DeeplyEssential (Hasan and Lonardi, 2020) is an example of the Deep Neural Network, conceived to predict the critical genes for the survival and reproduction of bacteria and microbes. It consists of an input followed by six hidden layers, in which the rectified linear unit (ReLU) is used as the activation function. The output has two classes (binary classification) where Sigmoid is used as the activation function. The binary cross-entropy is chosen as the loss function.

Figure 8. *An example of a Shallow Neural Network comprised of an input layer (two neurons $[x_{11}, x_{12}]$), a hidden layer in the middle, (three neurons $[x_{21}, x_{22}, x_{23}]$) and the output layer (2 neurons $[y_1, y_2]$).*

Deep Learning architectures

Convolutional Neural Network

Convolutional Neural Network (CNN) is one of the most successful Neural Network architectures. Originally, it was inspired from the work of (Hubel and Wiesel, 1962) to understand the cat's visual cortex. CNN became so famous and has a wide variety of applications. For instance, it was successfully implemented to detect COVID-19, given a chest X-ray image. It predicts if the patient has the COVID-19 or not with 98.92% average accuracy (Irmak, 2020). CNN was also used to learn the functional activity of DNA sequences from genomics data. It was trained on a compendium of accessible genomic sites mapped in 164 cell types by DNase-seq and demonstrated greater predictive accuracy than previous methods (Kelley *et al.*).

CNN architecture consists of an input layer, followed by a convolution layer 2.3.1 that produces feature maps, then hidden layers (the Pooling layers 2.3.2, Normalization) and finally the output, (Figure 9).

Figure 9. *A example of a Convolution Neural Network comprised of the input, a Convolution layer, Feature maps, Pooling layer, Fully connected layer and the output.*

Convolution layer

The outstanding capacity of CNN is owed to its ability to analyse spatial information and automatically extract features with the help of a convolution operation. The convolution operation is a mathematical operation between the input and the kernels, (Figure 10). Numerous convolution operations exist, such as standard convolution, dilated convolution, transposed convolution and separable convolution.

Figure 10. *An example of a standard convolution calculation. The window size and kernel is 4. The output c_i is calculated based on the input and the kernel.*

Pooling layer

The Pooling layer aims to reduce the resolution of the feature maps produced by the Convolution layer. The most famous pooling layers are: Average and Max Pooling layers, (Figure 11). **Average Pooling** takes the average of a range of numbers. **Max Pooling** takes the maximum number within a range of numbers.

Figure 11. (a) Average Pooling operation calculates the average value for a given range. (b) Max Pooling operation looks for the maximum number in a given range.

Padding

Padding is simply the process of adding layers of zeros to the input, as to avoid the problem of losing values on corners or shrinking the input, (Figure 12).

Figure 12. *An example of zero padding, where zeros are appended and prepended to the original input to avoid losing x_1 and x_n values.*

Stride

While convolving, the stride describes how the convolution window moves over the input. By default, it slides by one at each step, (Figure 13).

Figure 13. *An example of a convolutional filter that convolves over the input. Its window size is four and it strides by one at each step.*

Fully connected layer

This layer often appears at the end of the CNN architectures to sum the features produced by the previous layers and make predictions for the output, (Figure 9).

Recurrent Neural Network

Recurrent Neural Network (RNN) architecture is specially designed for sequential data, such as genomics or text. It is able to model space-temporal structures. Thanks to its hidden states that serve as a memory and keep track of the previous state. They provide a context for the current prediction, (Figure 14). For instance, RNN-VirSeeker (Liu *et al.*, 2020) successfully used RNN to outperform three widely used methods: VirSorter, VirFinder and DeepVirFinder in identifying short viral sequences, by obtaining 92% precision for sequences of 500bp.

Figure 14. *An unfold RNN cell. It has an input x_t, a hidden state h_t and an output y_t at a timeslot t. V, U and W are respectively the hidden state, input and weight matrices.*

One drawback of the vanilla RNNs is the long-term dependencies problem, formulated as Exploding 4.4.4 and Vanishing Gradients 4.4.3. As a remedy, special variant of RNNs named respectively, ‘‘Long Short Term Memory’’(LSTM) and ‘‘Gated Recurrent Unit’’(GRU) are introduced.

Long Short Term Memory

Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) is a variant of the RNN architecture that tries to solve the Vanishing Gradient Problem. Internally, the LSTM cell differs from the RNN cell. It has three control gates: forget, update and output gates, (Figure 15). The **Forget** gate allows the cell to forget information in the cell state. The **Update** gate allows the cell to place a new value in the memory (cell state). The **Output** gate applies the Sigmoid activation function and produces the output of the current timestep. ProLanGO (Cao *et al.*, 2017) is an example of LSTM that predicts protein functions. It converts the protein sequences into a language space "ProGO" based on the frequency of k-mers (around 500,000 protein sequences were employed). The Gene Ontology terms are encoded into a language space "LanGO", as well as a neural machine translation model is built based on Recurrent Neural Networks that translates "ProLan" language to "GOLan" language.

Figure 15. *The LSTM cell internal structure. At time t, the cell reads the input x_t, updates the cell state c_t and the hidden state h_t using three gates that control the signal workflow.*

Gated Recurrent Unit

Gated Recurrent Unit (GRU) (Cho *et al.*, 2014) is another variant of the RNN architecture. It simplifies the LSTM by having only two gates: the Update and Reset gates, (Figure 16). The GRU was successfully used for pan-specific prediction of HLA-I-binding peptides (Human leukocyte antigens) (Heng *et al.*, 2020). The model performance was very good after 31 epochs. The prediction accuracies of the training and validation sets are, respectively, 87% and 85%.

Figure 16. *The GRU cell simplifies the architecture of the LSTM. It has only two gates: the Update and Reset gates.*

Generative Adversarial Network

The particularity of the Generative Adversarial Network (GAN) (Goodfellow *et al.*) architecture is the competition aspect between two Neural Networks. One agent is the Generator (G), the other one is the Discriminator (D), they contest with each other. The Generative tries to synthesize fake data that resemble to real ones, whereas the Discriminative tries to distinguish between the real and the fake ones, (Figure 17). G is trained in a such way to maximize the probability of D making a mistake. For instance, GAN was used for gene expression inference to approximate the joint distribution of landmark for the target genes and to learn their conditional distribution given the landmark gene (Ghasedi Dizaji *et al.*, 2018).

Figure 17. *An illustrative example of how the GAN architecture is trained. The Generator tries to synthesize sequences, while the Discriminator tries to distinguish between the real and fake ones.*

Transformers

One of the modern Deep Learning architectures are Transformers (Vaswani *et al.*, 2017). They are game changers as they have outperformed the previous architectures in many tasks. Initially, they were designed for textual data. Recently, it was shown that they can work with any kind of data. The transformer model relies heavily on the Attention mechanism.

Basically, the Attention mechanism tries to learn and score the part of the data that is more important within a context. A transformer is comprised of two parts, Encoders and Decoders, (Figure 18). The stacked encoders are responsible for encoding the information while the stacked decoders decode it. The size of the stack is related to the architecture design. For instance, Transformers are used for improving the compound–protein interaction prediction by sequence-based Deep Learning with self-attention mechanism and label reversal experiments (Chen *et al.*, 2020). Transformers were also used to learn the protein language (facebookresearch/esm, 2021), in which the Unsupervised Learning was employed to train a deep contextual language model on 86 billion amino acids across 250 million protein sequences spanning evolutionary diversity. The resulting model contained information about biological properties. The model learned the representation space in a multi-scale organization reflecting structure from the level of biochemical properties of amino acids to remote homology of proteins.

Figure 18. *The transformer inner architecture comprised of stacked encoders in which the input is encoded, followed by stacked decoders that decode the information. The size of the stack is related to the architecture design.*

BERT

Bidirectional Encoder Representations from Transformers (BERT) (Devlin *et al.*, 2019) is based on the Transformer architecture. It caused a stir in the Deep Learning community by achieving new state-of-the-art results on several tasks and in diversified disciplines. BERT is trained in two steps: pre-training and fine-tuning. During the pre-training phase, Unsupervised Learning is employed to train the model on unlabelled data. Consequently, a general purpose pre-trained model is obtained that can be fine-tuned for a specific problem using its labelled data. BERT uses two learning strategies : Masked Language Model (MLM) and Next Sentence Prediction (NSP). Regarding the **MLM**, the model randomly masks some tokens from the input sequence, then it tries to predict them based on the information provided by unmasked tokens in the sequence. For the **NSP**, the model needs to predict whether a given sentence is the subsequent sentence to the current one or not.

BERT technology is brand new. In genomics, for instance, it is used to decipher the language of non-coding DNA (Ji *et al.*, 2020). It was able to simultaneously achieve the state-of-the-art performance on many sequence predictions tasks, such as: identifying the transcription factor binding sites also predicting the proximal and the core promoter regions.

Genomic data

Deep Learning requires huge amount of data. Luckily, genomic data were collected, organized and stored in open access databases for several decades. For instance, GenBank® (Benson *et al.*, 2013) is the NIH genetic sequence database, an annotated collection of all publicly available DNA sequence. Gene database (Ostell, 2013) is another example that integrates information from a wide range of species where it contains over 17 million entries.

Such databases can serve as data sources to extract genomic data to train Deep Learning models in order to solve specific problems. Conventionally, datasets can be split into three subsets : the training set, test set and the validation set.

The **training set**: is subset of the original data used to train the model.

The **test set** : is a subset of the original data used to test the trained model.

The **validation set** : is used to optimize the model during the development process.

Training

Training a Deep Learning model refers to the process of searching the best parameters that fit the model to the data set. While being trained, a Neural Network looks for the best values to tune its weights and obtain the best performances with the help of an optimization algorithm 2.1.3. The role of the optimizer is to minimize the loss function 2.1.2 by reaching global minima. Neural Networks are trained iteratively. Each training iteration consists of a Forward propagation 4.1 and Backward propagation 4.2 passes in which a subset of the dataset named "mini-batch" is passed to the network. When the entire training set is consumed, it is called "Epoch". The performance of a Deep Learning model depends on a multitude of hyperparameters. Hyperparameters refer to the parameters whose values are used to control the learning process. Tuning those hyperparameters refers to the process of deciding about their values that determine the network structure such as the number of hidden layers ..., also those that determine how the network is trained, such as the learning rate, epochs ... Setting them is one of the difficulties of the Deep Learning approach due to their considerable numbers and their empirical attitudes (Makwe and Rathore, 2021).

Forward propagation

Forward propagation pass is when data flow from the input towards the output. It comprises all the calculations of the Neural Network weights for the prediction in a forward direction.

Backward propagation

Historically, training the Neural Networks was one of the big challenges. Hence, the Backpropagation (Rumelhart and McClelland, 1987) algorithm was introduced to fulfil this duty. The backward propagation pass is when data flow from the output to the input with the purpose to tune the model. The Neural Network gradients are calculated in the backward direction. The Backprop abstracts the extensive computations used to calculate and update the weights of the network in a backward propagation.

Hyperparameters

This section will discuss some important hyperparameters.

Initialization

Initialization refers to the strategy selected to initialize the weights of the network when it starts the learning process. A good initialization strategy may reduce training time and computational costs.

Regularization

Regularization represents the techniques put in place to fight the overfitting 4.4.1. They are concerned with adjusting the prediction function.

Dropout

Dropout is a method where the not needed neurons are dropped from the network. It is used to reduce the overfitting 4.4.1.

Normalization

Normalization is concerned with feature scaling techniques for data adjustment.

Commun problems

Overfitting

Overfitting is the situation when the model learns too much on the used dataset, thus it gives good accuracy on the training data but does not generalize well on new data. It often appears when working with finite samples or limited datasets.

Underfitting

Underfitting is the scenario when the model has not learn enough from the data, thus it was not able to generalize.

Vanishing gradient

The Vanishing Gradient problem (Hochreiter, 1998) may happen when the network is comprised from several neural layers, in particular the Recurrent Neural Networks. In essence, Vanishing Gradient occurs when gradients are very small or zero. Thus, little to no training can take place and a poor predictive performance is noticed.

Exploding gradient

Exploding gradient is a problem where large error gradients accumulate resulting very large updates to Neural Network model weights during training. Consequently, the model becomes unstable and unable to learn from the training data.

Frameworks

Recently, Deep Learning approach has witnessed a wide adoption. One of the many reasons is the plethora of available libraries and frameworks that emerged to support this trend. They save time and offer the necessary toolkits for rapid prototyping of new concepts. The Tables 1 and 2 summarize some of the widely used ones.

Limitations

Deep Learning faces many challenges in genomics, from which interesting to note :

The curse of dimensionality

Genomics is considered as a Big Data science. Taking in account the volume of the available datasets (Gigabytes), their heterogeneity (sequencing of coding or non-coding genes, gene variants...) and their variety, which can pose challenges for this approach.

Lack of data

Deep Learning requires a huge amount of data. Sometimes and for a specific problem, the available data are not enough to obtain good performance.

Imbalanced classes

Usually, the collected genomics data suffer from imbalanced ratio of instances per class. Thus, the DL model may fail to generalize about certain classes.

Model interpretation

Generally, it is considered to be a major problem of the Deep Learning approach. Sometimes, it is difficult for the model designer to understand and interpret the learned patterns. This problem is known as the black box.

Conclusion

With the advancement in processing power, availability of toolbox for practitioners and abundance of genomics data, Deep Learning is delivering impressive results in various fields including genomics. In this work, we introduced the different concepts of this technology and supplied various use case examples, also pointed out some of its advantages, difficulties and challenges. Deep Learning can be a real opportunity for researchers to tackle various genomics problems in a data driven approach.

Key Points

- Deep Learning is a subset of Machine Learning methods that makes use of interconnected Artificial Neurons "Neural Networks" to automatically learn features from raw data.
- Learning can be supervised, semi-supervised, unsupervised or reinforcement.
- A wide variety of Deep Learning architectures exist such as CNN, RNN, GAN, Transformers…
- Training a Deep Learning model may require a huge amount of data and fine-tuning the hyperparameters certainly impacts the learning process and the performance.
- Deep Learning is a promising technology that demonstrated its supremacy in various tasks and has various application in different domains including the genomics.

Acknowledgements

IMGT® was funded in part by the BIOMED1 (BIOCT930038), Biotechnology BIOTECH2 (BIO4CT960037), 5th PCRDT Quality of Life and Management of Living Resources (QLG2-2000-01287), and 6th PCRDT Information Science and Technology (ImmunoGrid, FP6 IST-028069) programmes of the European Union (EU). IMGT® received financial support from the GIS IBiSA, the Agence Nationale de la Recherche (ANR) Labex MabImprove (ANR-10-LABX-53-01), the Région Occitanie Languedoc-Roussillon (Grand Plateau Technique pour la Recherche (GPTR), BioCampus Montpellier. IMGT® is currently supported by the Centre National de la Recherche Scientifique (CNRS), the Ministère de l'Enseignement Supérieur, de la Recherche et de l'Innovation (MESRI), the University of Montpellier, and the French Infrastructure Institut Français de Bioinformatique (IFB) ANR-11-INBS-0013. IMGT® is a registered trademark of CNRS. IMGT® is member of the International Medical Informatics Association (IMIA) and a member of the Global Alliance for Genomics and Health (GA4GH). IMGT is granted access to the High Performance Computing (HPC) resources of Meso@LR and of Centre Informatique National de l'Enseignement Supérieur (CINES) and to Très Grand Centre de Calcul (TGCC) of the Commissariat à l'Energie Atomique et aux Energies Alternatives (CEA) under the allocation 036029 (2010-2021) made by GENCI (Grand Equipement National de Calcul Intensif).

We are grateful to Fotis Psomopoulos for helpful scientific discussions.

References

Benson DA, Cavanaugh M, Clark K, Karsch-Mizrachi I, Lipman DJ, *et al.* (2013) GenBank. *Nucleic Acids Res.* **41** (Database issue), D36-42. http://dx.doi.org/10.1093/nar/gks1195

Cao R, Freitas C, Chan L, Sun M, Jiang H, *et al.* (2017) ProLanGO: Protein Function Prediction Using Neural Machine Translation Based on a Recurrent Neural Network. *Molecules* **22** (10), 1732. http://dx.doi.org/10.3390/molecules22101732

Chen L, Cai C, Chen V, and Lu X (2016) Learning a hierarchical representation of the yeast transcriptomic machinery using an autoencoder model. *BMC Bioinformatics* **17** (1), S9. http://dx.doi.org/10.1186/s12859-015-0852-1

Chen L, Tan X, Wang D, Zhong F, Liu X, *et al.* (2020) TransformerCPI: improving compound–protein interaction prediction by sequence-based deep learning with self-attention mechanism and label reversal experiments. *Bioinformatics* **36** (16), 4406–4414. http://dx.doi.org/10.1093/bioinformatics/btaa524

Cho K, van Merrienboer B, Bahdanau D, and Bengio Y (2014) On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *ArXiv14091259 Cs Stat*

Choi D, Shallue CJ, Nado Z, Lee J, Maddison CJ, et al. (2020) On Empirical Comparisons of Optimizers for Deep Learning. ArXiv191005446 Cs Stat

Devlin J, Chang M-W, Lee K, and Toutanova K (2019) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv181004805 Cs*

F Z, Q L, X Z, and B S (2015) Protein interaction network constructing based on text mining and reinforcement learning with application to prostate cancer. *IET Syst Biol* **9** (4), 106–112. http://dx.doi.org/10.1049/iet-syb.2014.0050

facebookresearch/esm (2021) Facebook Research,.

Friedländer MR, Mackowiak SD, Li N, Chen W, and Rajewsky N (2012) miRDeep2 accurately identifies known and hundreds of novel microRNA genes in seven animal clades. *Nucleic Acids Res.* **40** (1), 37–52. http://dx.doi.org/10.1093/nar/gkr688

Gao W, Mahajan SP, Sulam J, and Gray JJ (2020) Deep Learning in Protein Structural Modeling and Design. *Patterns N* **1** (9), 100142. http://dx.doi.org/10.1016/j.patter.2020.100142

Ghahramani Z (2004) Unsupervised Learning. In: BousquetO, von LuxburgU, and RätschG (eds) *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2 - 14, 2003, Tübingen, Germany, August 4 - 16, 2003, Revised Lectures*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, Berlin, Heidelberg,pp. 72–112

Ghasedi Dizaji K, Wang X, and Huang H (2018) Semi-Supervised Generative Adversarial Network for Gene Expression Inference. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD ’18. Association for Computing Machinery, New York, NY, USA, New York, NY, USA,pp. 1435–1444

Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, *et al.* Generative Adversarial Nets. , 9.

Hasan MA and Lonardi S (2020) DeeplyEssential: a deep neural network for predicting essential genes in microbes. *BMC Bioinformatics* **21** (14), 367. http://dx.doi.org/10.1186/s12859-020-03688-y

Heng Y, Kuang Z, Huang S, Chen L, Shi T, *et al.* (2020) A Pan-Specific GRU-Based Recurrent Neural Network for Predicting HLA-I-Binding Peptides. *ACS Omega* **5** (29), 18321–18330. http://dx.doi.org/10.1021/acsomega.0c02039

Hennig C and Kutlukaya M Some thoughts about the design of loss functions. *REVSTAT–Statistical J.*, 2007.

Hochreiter S (1998) The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *Int J Unc Fuzz Knowl Based Syst* **06** (02), 107–116. http://dx.doi.org/10.1142/S0218488598000094

Hochreiter S and Schmidhuber J (1997) Long Short-Term Memory. *Neural Comput* **9** (8), 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735

Hubel DH and Wiesel TN (1962) Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *J Physiol* **160**, 106–154. http://dx.doi.org/10.1113/jphysiol.1962.sp006837

Irmak E (2020) Implementation of convolutional neural network approach for COVID-19 disease detection. *Physiol. Genomics* **52** (12), 590–601. http://dx.doi.org/10.1152/physiolgenomics.00084.2020

Ji Y, Zhou Z, Liu H, and Davuluri RV (2020) DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome. *bioRxiv*, 2020.09.17.301879. http://dx.doi.org/10.1101/2020.09.17.301879

Kelley DR, Snoek J, and Rinn JL Basset: Learning the regulatory code of the accessible genome with deep convolutional neural networks. , 35.

Kiselev VY, Andrews TS, and Hemberg M (2019) Challenges in unsupervised clustering of single-cell RNA-seq data. *Nat Rev Genet* **20** (5), 273–282. http://dx.doi.org/10.1038/s41576-018-0088-9

Kotsiantis SB, Zaharakis ID, and Pintelas PE (2006) Machine learning: a review of classification and combining techniques. *Artif Intell Rev* **26** (3), 159–190. http://dx.doi.org/10.1007/s10462-007-9052-3

Krizhevsky A, Sutskever I, and Hinton GE (2012) ImageNet Classification with Deep Convolutional Neural Networks. In: PereiraF, BurgesCJC, BottouL, and WeinbergerKQ (eds) *Advances in Neural Information Processing Systems*. Curran Associates, Inc., Vol25,pp. 1097–1105

Le Cun Y, Boser B, Denker JS, Henderson D, Howard RE, *et al.* (1989) Handwritten digit recognition with a back-propagation network. In: *Proceedings of the 2nd International Conference on Neural Information Processing Systems*. NIPS'89. MIT Press, Cambridge, MA, USA, Cambridge, MA, USA,pp. 396–404

LeCun Y, Bengio Y, and Hinton G (2015) Deep learning. *Nature* **521** (7553), 436–444. http://dx.doi.org/10.1038/nature14539

Li Y, Shi W, and Wasserman WW (2018) Genome-wide prediction of cis-regulatory regions using supervised deep learning methods. *BMC Bioinformatics* **19** (1), 202. http://dx.doi.org/10.1186/s12859-018-2187-1

Liu F, Miao Y, Liu Y, and Hou T (2020) RNN-VirSeeker: a deep learning method for identification of short viral sequences from metagenomes. *IEEEACM Trans Comput Biol Bioinform* **PP** http://dx.doi.org/10.1109/TCBB.2020.3044575

Makwe A and Rathore AS (2021) An Empirical Study of Neural Network Hyperparameters. In: BhatejaV, PengS-L, SatapathySC, and ZhangY-D (eds) *Evolution in Computational Intelligence*. Advances in Intelligent Systems and Computing. Springer, Singapore, Singapore,pp. 371–383

McCulloch WS and Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.* **5** (4), 115–133. http://dx.doi.org/10.1007/BF02478259

Movva R, Greenside P, Marinov GK, Nair S, Shrikumar A, *et al.* (2019) Deciphering regulatory DNA sequences and noncoding genetic variants using neural network models of massively parallel reporter assays. *bioRxiv*, 393926. http://dx.doi.org/10.1101/393926

Nwankpa C, Ijomah W, Gachagan A, and Marshall S (2018) Activation Functions: Comparison of trends in Practice and Research for Deep Learning. *ArXiv181103378 Cs*Ostell J (2013) What's in a Genome at NCBI? National Center for Biotechnology Information (US),.

Peace RJ, Hassani MS, and Green JR (2018) miPIE: NGS-based Prediction of miRNA Using Integrated Evidence. *bioRxiv*, 405357. http://dx.doi.org/10.1101/405357

Popova M, Isayev O, and Tropsha A (2018) Deep reinforcement learning for de novo drug design. Sci. Adv. 4 (7), eaap7885. http://dx.doi.org/10.1126/sciadv.aap7885

Press TM Reinforcement Learning \textbar The MIT Press.

Quang D and Xie X (2019) FactorNet: A deep learning framework for predicting cell type specific transcription factor binding from nucleotide-resolution sequential data. *Methods* **166**, 40–47. http://dx.doi.org/10.1016/j.ymeth.2019.03.020

Rosenblatt F (1957) The Perceptron, a Perceiving and Recognizing Automaton Project Para Cornell Aeronautical Laboratory,.

Rumelhart DE and McClelland JL (1987) Learning Internal Representations by Error Propagation. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*. MIT Press, pp. 318–362

Senior AW, Evans R, Jumper J, Kirkpatrick J, Sifre L, *et al.* (2020) Improved protein structure prediction using potentials from deep learning. *Nature* **577** (7792), 706–710. http://dx.doi.org/10.1038/s41586-019-1923-7

Sheikh Hassani M and Green JR (2019) A semi-supervised machine learning framework for microRNA classification. *Hum Genomics* **13** (Suppl 1), 43. http://dx.doi.org/10.1186/s40246-019-0221-7

Singh R, Lanchantin J, Sekhon A, and Qi Y (2017) Attend and Predict: Understanding Gene Regulation by Selective Attention on Chromatin. *ArXiv170800339 Cs*

Singh S, Yang Y, Póczos B, and Ma J (2019) Predicting enhancer-promoter interaction from genomic sequence with deep neural networks. *Quant Biol* **7** (2), 122–137. http://dx.doi.org/10.1007/s40484-019-0154-0

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, *et al.* (2017) Attention Is All You Need. *ArXiv170603762 Cs*

Wang Q, Ma Y, Zhao K, and Tian Y (2020) A Comprehensive Survey of Loss Functions in Machine Learning. *Ann Data Sci* http://dx.doi.org/10.1007/s40745-020-00253-5

Xavier R, de Souza KP, Chateau A, and Alves R (2020) Genome Assembly Using Reinforcement Learning. In: KowadaL and de OliveiraD (eds) *Advances in Bioinformatics and Computational Biology*. Lecture Notes in Computer Science. Springer International Publishing, Cham, Cham,pp. 16–28

Yelmen B, Decelle A, Ongaro L, Marnetto D, Tallec C, *et al.* (2019) Creating Artificial Human Genomes Using Generative Models. *bioRxiv*, 769091. http://dx.doi.org/10.1101/769091