Squeeze and Learn: Compressing Long Sequences with Fourier Transformers for Gene Expression Prediction

Vittorio Pipoli[1,2][0009-0008-5749-6007], Giuseppe Attanasio[3][0000-0001-6945-3698],

Marta Lovino[1][0000-0001-7124-8319] and Elisa Ficarra[1][0000-0002-8061-2124]

[1] University of Modena, Modena, DIEF 41125, Italy
name.surname@unimore.it
[2] University of Pisa, Pisa, 56126, Italy
name.surname@phd.unipi.it
[3] Instituto de Telecomunicações, Universidade de Aveiro Campus Universitário de
R. Santiago, 3810-193 Aveiro, Portugal
name.surname@lx.it.pt

Abstract. Genes regulate fundamental processes in living cells, such as the synthesis of proteins or other functional molecules. Studying gene expression is hence crucial for both diagnostic and therapeutic purposes.
State-of-the-art Deep Learning techniques such as Xpresso have been proposed to predict gene expression from raw DNA sequences. However, DNA sequences challenge computational approaches because of their length, typically in the order of thousands, and sparsity, requiring models to capture both short- and long-range dependencies. Indeed, the application of recent techniques like transformers is prohibitive with common hardware resources.
This paper proposes FNetCompression, a novel gene-expression prediction method. Crucially, FNetCompression combines Convolutional encoders and memory-efficient Transformers to compress the sequence up to 95% with minimal performance tradeoffs. Experiments on the Xpresso dataset show that FNetCompression outscores our baselines and the margin is statistically significant. Moreover, FNetCompression is 88% faster than a classical transformer-based architecture with minimal performance tradeoffs. Code and data are available at https://github.com/vittoriopipoli/FNetCompression.

Keywords: DNA Sequences, Gene Expression, Transformer.

1 Introduction

Gene Expression [2] regulates the existence of every living organism. It consists of the fundamental mechanisms the cells exploit to gather information from the deoxyribonucleic acid (DNA) and synthesize functional molecules (e.g., proteins) according to inherent regulatory mechanisms. Recent work has proposed to use of Deep Learning (DL) models to predict gene expression directly from raw DNA sequences sampled and sequenced from living organisms, e.g., human tissues [3]. However, DNA sequences often count thousands of elements, and the signal within them is sparse (*e.g.* functional

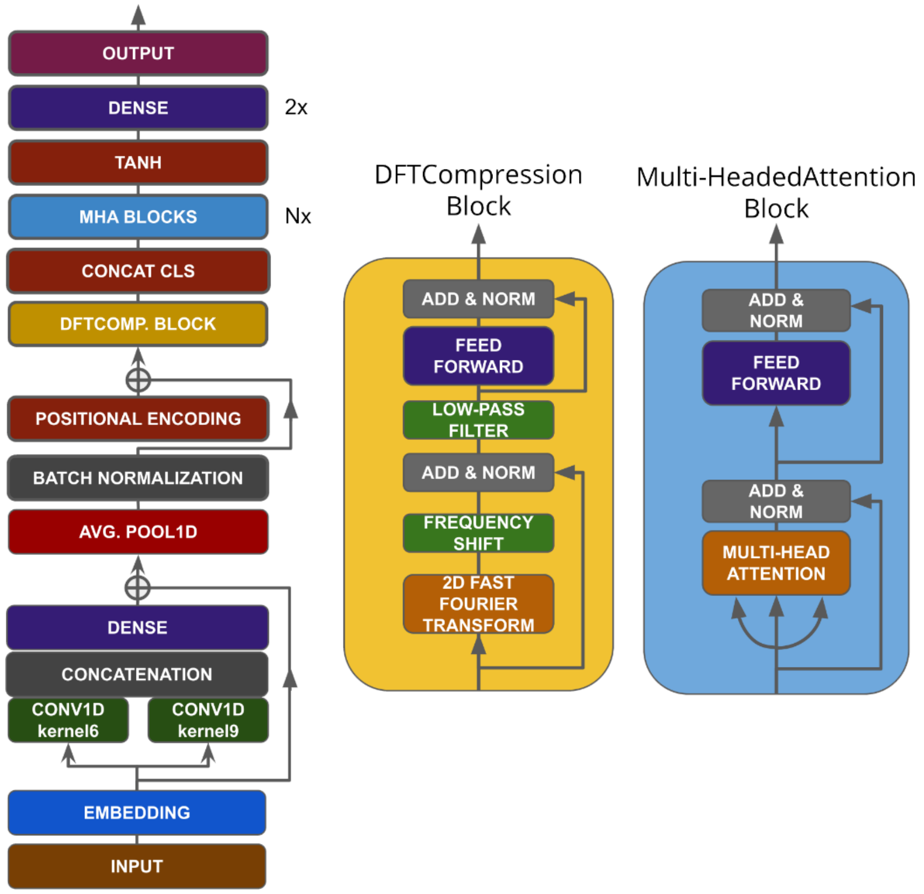

Fig. 1. FNetCompression overview (left). After sequence embedding and pooling, DFTCompression (center) and MHA (right) layers compress and route information from the input sequence. Similar functional blocks share background colors.

coding regions alternate with long non-coding parts). Length and sparsity make such sequences impractical for modern DL models, motivating increasing interest in compression for efficiency and noise reduction. Therefore, recent methods encode the sequence of original base pairs (bp) into shorter sequences, where each new token "represents" several bp. 1D Convolution layers [18], Long Short-Term Memory [13], and Transformer-based networks [24] have been adopted for the task [3,25,5,16].

The nature of such DNA sequences requires gene expression prediction algorithms to learn from both local- and long-range interactions. For example, recent evidence found interactions among DNA elements at several kilobase pairs (kbp) of distance [22]. Transformers models [24] provide a suitable method to learn from both short- and long-range dependencies: the Multi-Headed Attention (MHA) mechanism. A typical MHA layer connects every input item with every other item and learns how to weigh every

pair. By contrast, Convolutional Neural Networks (CNNs) [19] need a deep structure with many layers to enlarge the receptive field to distant elements.
However, MHA's memory footprint grows quadratically with the sequence length, motivating recent research efforts on efficient transformers [10,8]. FNet [20] is a prominent example: the network substitutes MHA with a Discrete Fourier Transform (DFT), a non-parametric, linearithmic token mixing strategy.

Contributions. This work introduces FNetCompression, a novel approach to gene expression prediction from long DNA sequences. FNetCompression uses convolution kernels, DFT-based transformers, and low-pass filters to compress input sequences and a final MHA layer for improved information routing. Results on gene-expression datasets [3] show that FNetCompression significantly outperforms the baseline solution, reaching up to 93% of performances of less efficient standard transformers despite compressing inputs by 95% of their length. Moreover, we conducted a qualitative analysis on FNetCompression and discovered that i) attention weights are stronger on low-frequency components of the sequence, and ii) all elements contribute to the prediction.

2 Related Work

Recent advances in sequence modeling and compression have motivated new neural gene expression models to learn from raw DNA sequences.

Xpresso [3] (Agarwal et al., 2020) is a state-of-the-art Deep Convolutional Neural Network [19] in the field of gene expression. The network predicts the steady-state gene expression levels in human and mouse organisms, exploiting DNA sequences and features associated with mRNA stability. The authors claim that Xpresso explains 59% of the variation (measured with R^2) in steady-state mRNA levels in humans. Xpresso handles sequences of several thousand base pairs. The best-reported range is 7,000 bps and 3,500 bps, respectively, upstream and downstream Transcription Start Site (TSS). Note that the information around the TSS is an important proxy for gene expression [15]. We build on Xpresso and use initial Convolutional layers for input summarization. However, we differ on the embedding of the nitrogenous basis, the type of pooling layers, and the transformer encoder.

Expecto [25] (Zhou et al., 2018) is a Convolutional Deep Neural Network for predicting tissue-specific gene expression levels in humans. Unlike Xpresso, it requires additional biological information related to chromatin, defining different experimental conditions.

Enformer [5] (Avsec et al., 2021) is a state-of-the-art transformer-based architecture for encoding even longer DNA sequences. Although Enformer and FNetCompression share several architectural parts, e.g., pooling and transformer blocks, the former was devised to predict sequences of biological tracks.

FNet [20] (Lee-Thorp et al., 2021) replaces the self-attention sublayers, which pay a quadratic complexity, with a standard, non-parametrized and linearithmic two-dimensional Fast Fourier Transform achieving 92-97% accuracy of BERT [11], but training

80% faster on GPU and 70% faster on TPU. We build on FNet to introduce FNetCompression and add further compression layers to enhance efficiency.

3 Datasets

Aiming for a fair comparison, we test FNetCompression in existing gene-expression prediction setups. Specifically, we use the dataset of sequences introduced in Xpresso [3], which counts 18,377 genes. For each gene, Xpresso releases: 1. the DNA sequence (20,000 bp long); 2. the half-life features (that estimate the time required for degrading 50% of the existing mRNA molecules [3]) which are embedded in a vector of 8 real numbers for each gene; 3. the expression value, which is the label to be predicted. Moreover, both the validation and test set are obtained by sampling at random 1000 genes from the train set. These DNA sequences are arrays of nitrogenous bases extracted from the human reference genome. Additionally, the neighborhood of the Transcription Start Site (TSS) contains the most useful information for the prediction of gene expression [15]. Therefore, all the sequences are extracted and centered with respect to the TSS and contain the 10kbp upstream and downstream of it.

The locations of about 15k TSS have been downloaded from the FANTOM5 consortium's UCSC data hub (Lizio et al.) [21]. For the remaining genes, Xpresso is considered as TSS, among all the transcripts for each gene, the start coordinate of the one with the longest Open Reading Frame [23], followed by the longest 5' Untranslated Gene Region, followed by the longest 3' Untranslated Gene Region [7].

The gene expression values were retrieved from the Epigenomics Roadmap Consortium [9]. In particular, the values were retrieved in a tabular format of normalized expression values for protein-coding genes across 56 tissues and cell lines obtained by RNA-seq data. The preprocessing foresees the averaging among the tissues, ending up with one expression value per gene. After the aggregation steps, the values are then processed with a log transformation as follows:

$$\hat{y} = log_{10}(y + pseudocount) \tag{1}$$

to reduce the right skew of the labels' distribution. The *pseudocount* value is set to 0.1.

In addition to Xpresso's dataset, we perform experiments on a Controlled Test Bench (CTB) that removes the half-life features but relies on longer sequences, i.e., 65,536 bp. The TSS locations are downloaded by the FANTOM5 consortium's UCSC data hub (Lizio et al.) [21], and for the genes that are not covered, we decided to take the start coordinate of the longest transcript [1].

CTB uses the same Xpresso target labels but different splits, built as follows. Chromosomes 8 and 10 were used for the test and validation splits, respectively, and the remaining chromosomes were used for the training set. Genes have different lengths, and extracting a fixed-size window of base pairs can result in extracting the information of multiple genes. Stratifying on chromosomes prevents any overlap between training and test sequences. The resulting CTB training, validation, and test sets count 16,832, 683, and 618 sequences, respectively.

4 Proposed Method

This paper presents FNetCompression, a novel method for gene expression level prediction. FNetCompression uses a convolutional sequence embedding and a transformer encoder. The latter is composed of a non-parametric 2D Discrete Fourier Transform [6,20], a subsequent low-pass filter to reduce the sequence length up to 95%, and an MHA layer for optimizing the final information routing.[1] The model takes as input DNA sequences tens of thousands of nitrogenous bases long (and optionally the half-life features vector, concatenated after the “Tanh pooler”) and gives as output a real number that quantifies the gene expression level.

4.1 Sequence Embedding

As standard transformers handle sequences shorter than a thousand items [24], we first need to embed the input into a shorter sequence.

Unlike prior work using one-hot encoding [3,5], we use an initial embedding layer to represent DNA bases as dense vectors. Next, two 1D convolutional layers with different kernel sizes (kernel1=6, kernel2=9) transform the sequence. Note that different kernels capture different local patterns from the sequence. Convolutional outputs are concatenated and projected via a dense layer to recombine the information, and a skip-connection [12] is used to facilitate gradient backpropagation. Next, we apply a 1D Average Pooling, i.e., the first compression step, Batch Normalization [14], and sum absolute sinewave Positional Encodings [24]. Note that empirical experiments revealed Batch Normalization to be crucial for sequence embedding. We hypothesize this layer improves numeric stabilization before the addition of positional information, ensuring proper weighing of semantic and positional information.

4.2 DFTCompression

The output of the sequence embedding stage is fed to the *DFTCompression* block, which learns long-range patterns and further compresses the input sequences.

First, we apply a 2D DFT and retain only the real part [20]. By a first approximation, the resulting sequence represents the same signal in the “frequency” domain. Using this time-frequency intuition, we apply a low-pass filter (i.e., the second and most prominent compression step) as follows. We shift the zero-frequency component of the sequence to the center of the sequence and cut out symmetrically the outermost positions.

[1] Code and data are available at https://github.com/vittoriopipoli/FNetCompression.

Our results have shown that we can push this compression to remove up to 95% of the sequence while retaining most of the prediction accuracy. The output of the compression block is prepended with a special token and fed to a MHA layer. Using starting special tokens is commonplace in Computer Vision and Natural Language Processing, as such tokens are often used to summarize sequences. The final part of the network consists of a “Tanh pooler”, composed of a linear layer and a Tanh activation function that takes in input the special token, two dense layers with a ReLU activation function each, and a final linear layer with one neuron that represents the output of our regression model.

5 Results

We evaluated the learning capability of FNetCompression compared to Xpresso's model. Then, we tested generalization to longer sequences by reducing the pooling size on Xpresso's dataset and using the long sequences of our CTB. We compared FNetCompression to four different baseline configurations: 1) the sequence embedder without any transformer encoder block, 2) *FNet_1_0* which has one DFT block and no MHA blocks, 3) *FNet_1_1* which has one DFT block and one MHA (i.e., with no compression, like [20]), and 4) a Transformer with two encoder blocks. All these models are obtained by removing the DFTCompression block from the backbone depicted on the left in Figure 1 and modifying the blocks of the totem pole that follow the concatenation of the special token. Moreover, we provide the study of the computational complexity paid by the models, the attention maps, and gradient x input analysis.

Confidence intervals have been computed with 14 runs per experiment, a confidence level of 0.95, the unbiased standard deviation estimator, and t-student distribution.

5.1 Training details

All the methodologies have been fitted employing the Adam optimizer [17] exploiting a warm-up step scheduler [24]. The loss metric adopted is Mean Squared Error (MSE) and the test metric is R^2. The compression rate of FNetCompression is always set to 95%. All the MHA blocks have four heads. Refer to our GitHub for the rest of the hyperparameters. We adopted Google's Tesla T4 and TPU as hardware resources.

5.2 Performances on Xpresso Dataset and CTB

Here, we compare FNetCompression (§4.2) with Xpresso's model [3] on their dataset. Xpresso's gene prediction values have been obtained with the authors' code [4]. As shown in Table 1 FNetCompression and FNet_1_1 provide the best results even if FNetCompression reduces the input sequence length by 95%. Experiments on the CTB dataset show that FNetCompression outperforms FNet_1_1 with sequences long three times Xpresso's.

Table 1. Gene expression R^2 on the test set of the Xpresso's dataset and CTB (0.95 confidence levels).

Dataset	Method	Low_CI	Mean_CI	Upp_CI
Xpresso	Xpresso	0.5593	0.5668	0.5743
	Seq. Emb.	0.5343	0.5422	0.5501
	FNet_1_0	0.5567	0.5604	0.5641
	FNet_1_1	0.6121	0.6183	0.6254
	FNetComp.	0.6076	0.6133	0.6190
CTB	FNet_1_1	0.5786	0.5859	0.5931
	FNetComp.	**0.5944**	**0.6006**	**0.6068**

5.3 Computational Complexity

We studied the computational complexity of the tested models. Table 2 reports the results. FNetCompression's speed-up over FNet_1_1 increases with the input sequence length, as an expected result of our compression stages. Moreover, FNetCompression performance remains stable, unlike FNet_1_1.

We do not report the comparison of execution times on the CTB dataset due to out-of-memory errors in the testing environment. Preliminary tests on TPU hardware proved FNetCompression as the fastest model but by a smaller margin.

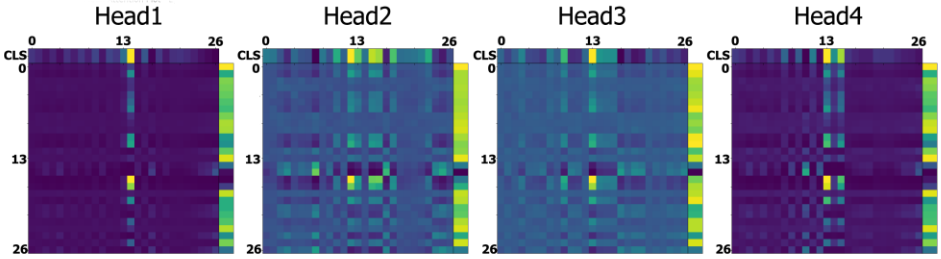

Fig. 2. Attention weights in FNetCompression trained on CTB. Attention values expressed (first row) and received (last column) by the special token are magnified and min-max normalized.

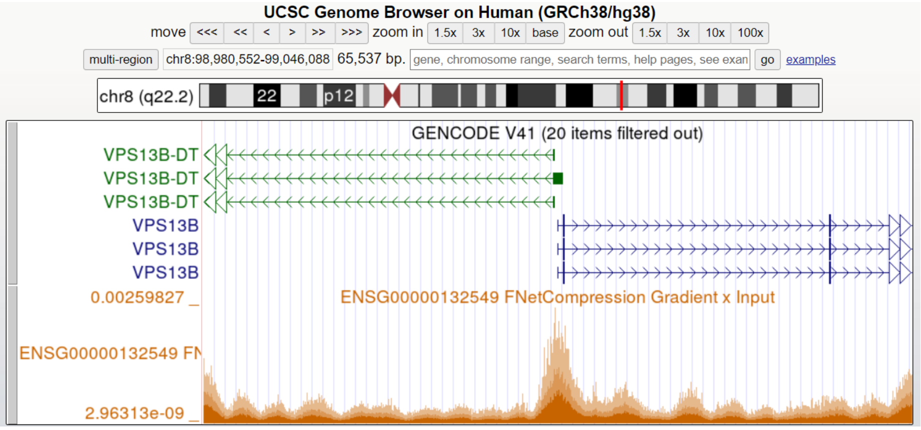
UCSC Genome Browser on Human (GRCh38/hg38)
move <<< << < > >> >>> zoom in 1.5x 3x 10x base zoom out 1.5x 3x 10x 100x
multi-region chr8:98,980,552-99,046,088 65,537 bp. gene, chromosome range, search terms, help pages, see exan go examples
chr8 (q22.2) 22 p12
GENCODE V41 (20 items filtered out)
VPS13B-DT
VPS13B-DT
VPS13B-DT
VPS13B
VPS13B
VPS13B
0.00259827 _
ENSG00000132549 FNetCompression Gradient x Input
ENSG00000132549 FN
2.96313e-09 _

Fig. 3. Gradient x Input attribution obtained by feeding FNETCOMPRESSION with the CTB test gene VPS13B.

Table 2. Speed up and performance comparisons of FNETCOMPRESSION and FNet_1_1 with a classic Transformer architecture on Xpresso's dataset using a Tesla T4 GPU.

Method	Pool Size	2DFFT O(nlogn)	MHA $O(n^2)$	Relative Perf.	Time per Batch [s]	Speed up
Transformer	128	-	156x2	-	36	-
FNet_1_0		156	0	85%	32	+11%
FNet_1_1		156	156	93%	34	+6%
FNetComp.		156	8	**93%**	32	**+11%**
Transforrmer	32	-	626x2	-	60	-
FNet_1_1		626	626	89%	48	25%
FNetComp.		**626**	**34**	**93%**	**32**	**+87.5%**

5.4 Attention and Gradient x Input Analysis

Attention plots can reveal some interesting patterns in transformer architectures' data modeling. In particular, the attention patterns of the Multi-Headed Attention block that follows the *DFTCompression block* of our model can be examined. As we can see in Figure 2, it is possible to spot vertical patterns in the middle of the matrix. Vertical patterns occur when all the elements of a sequence are paying attention to the same location. Therefore, most of the elements are paying attention to the regions that embed the lowest frequencies.

When FNetCompression applies a compression factor of 95%, only 5% of the sequences in processed by the subsequent Multi-Headed Attention layer. Hence, we computed the Gradient x Input to prove that all the elements of the original sequence take part in the loss contribution. Results are shown in Figure 3, and it is possible to notice that all the nitrogenous bases have a significant contribution, and the signal follows a sinusoidal pattern.

5.5 Reproducibility

To ensure full reproducibility we created a GitHub repository that contains the code to run the experiments and instructions to download the dataset. The link to the repository is the following: https://github.com/vittoriopipoli/FNetCompression. The data loader and the model classes can be found in the *"Classes"* folder. The hyperparameters of the proposed approach are in the folder *"Hyperparameters"* (one file for each dataset). The notebooks for running the experiments are in folders *"Workflow_GPU"* and *"Workflow_TPU"*, so that the user can choose the hardware that he wishes between GPU and TPU. Such notebooks must be downloaded and uploaded on Google Colab to be used. All the notebooks contain initial commands that download the dataset from the official repository and instantiate the data loader (the first two commands of each script). The uploaded notebooks are already run so that the user can already have a preview of a possible experiment. The remaining details can be found in the *"README.md"* file.

6 Conclusion

This work presented a transformer-based [24] model, called FNetCompression, for predicting gene expression levels from raw DNA sequences exploiting a crucial sequence compression. The main challenge of this work is to deal with the quadratic complexity of the attention mechanism by designing a transformer-based architecture that exploits a 2D DFT that can analyze and compress long DNA sequences even with few computational resources.

Results proved that FNetCompression (§4.2) outperforms Xpresso on their dataset. Hence, Xpresso's authors claim to explain up to 59% of the variation of gene expression levels, while FNetCompression explains up to 61%.

The comparison between FNetCompression and FNet_1_1 shows that FNetCompression is capturing all the useful information even if it is discarding 95% of the sequences. On the other hand, FNet_1_1 becomes unstable when its input length grows. Finally, FNetCompression is the fastest algorithm of these experiments. For future works, we suggest finding better ways to exploit the 2D DFT and compression strategies.

References

1. Kevin L Howe, Premanand Achuthan, James Allen, Jamie Allen, Jorge Alvarez-Jarreta, M Ridwan Amode, Irina M Armean, Andrey G Azov, Ruth Bennett,Jyothish Bhai, Konstantinos Billis, Sanjay Boddu, Mehrnaz Charkhchi, CarlaCummins, Luca Da Rin Fioretto, Claire Davidson, Kamalkumar Dodiya, Bilal ElHoudaigui, Reham Fatima, Astrid Gall, Carlos Garcia Giron, Tiago Grego, CristinaGuijarro-Clarke, Leanne Haggerty, Anmol Hemrom, Thibaut Hourlier, Osagie GIzuogu, Thomas Juettemann, Vinay Kaikala, Mike Kay, Ilias Lavidas, Tuan Le, Di-ana Lemos, Jose Gonzalez Martinez, Jos´e Carlos Marug´an, Thomas Maurel, AoifeC McMahon, Shamika Mohanan, Benjamin Moore, Matthieu Muffato, Denye NOheh, Dimitrios Paraschas, Anne Parker, Andrew Parton, Irina Prosovetskaia,Manoj P Sakthivel, Ahamed I Abdul Salam, Bianca M Schmitt, Helen Schuilen-burg, Dan Sheppard, Emily Steed, Michal Szpak, Marek Szuba, Kieron Taylor,Anja Thormann, Glen Threadgold, Brandon Walts, Andrea Winterbottom, MarcChakiachvili, Ameya Chaubal, Nishadi De Silva, Bethany Flint, Adam Frankish,Sarah E Hunt, Garth R IIsley, Nick Langridge, Jane E Loveland, Fergal J Mar-tin, Jonathan M Mudge, Joanella Morales, Emily Perry, Magali Ruffier, John Tate,David Thybert, Stephen J Trevanion, Fiona Cunningham, Andrew D Yates, DanielR Zerbino, Paul Flicek Ensembl 2021. Nucleic Acids Res. 2021, vol. 49(1):884–891PubMed PMID: 33137190. doi:10.1093/nar/gkaa942Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016)
2. Adams, J.U.: Differential Control of Transcription and Translation UnderliesChanges in Cell Function. MA: NPG Education, Cambridge (2010),Author, F.: Contribution title. In: 9th International Proceedings on Proceedings, pp. 1–2. Publisher, Location (2010)
3. Agarwal, V., Shendure, J.: Predicting mrna abundance directly from genomic se-quence using deep convolutional neural networks. Cell reports31(7), 107663 (2020)
4. Agarwal V, S.J.: Predicting mrna abundance directly from genomic sequence usingdeep convolutional neural networks, Xpresso Colab
5. Avsec,ˇZ., Agarwal, V., Visentin, D., Ledsam, J.R., Grabska-Barwinska, A., Taylor,K.R., Assael, Y., Jumper, J., Kohli, P., Kelley, D.R.: Effective gene expressionprediction from sequence by integrating long-range interactions. Nature methods18(10), 1196–1203 (2021)
6. Bailey, D.H., Swarztrauber, P.N.: A fast method for the numerical evaluation ofcontinuous fourier and laplace transforms. SIAM Journal on Scientific Computing15(5), 1105–1110 (1994)
7. Barrett, L.W., Fletcher, S., Wilton, S.D.: Untranslated gene regions and othernon-coding elements (2013), springer. ISBN 978-3-0348-0679-4
8. Beltagy, I., Peters, M.E., Cohan, A.: Longformer: The long-document transformer.arXiv preprint arXiv:2004.05150 (2020)
9. Bernstein, B.E., Stamatoyannopoulos, J.A.: The NIH roadmap epigenomics map-ping consortium. Nature Biotechnology28(10), 1045–1048 (Oct 2010)
10. Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T.,Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al.: Rethinking attention withperformers. arXiv preprint arXiv:2009.14794 (2020)
11. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidi-rectional transformers for language understanding (2018).
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recog-nition (2015).

13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation9,1735–80 (12 1997).
14. offe, S., Szegedy, C.: Batch normalization: Accelerating deep network trainingby reducing internal covariate shift. In: Bach, F., Blei, D. (eds.) Proceedings ofthe 32nd International Conference on Machine Learning. Proceedings of MachineLearning Research, vol. 37, pp. 448–456. PMLR, Lille, France (7 2015)
15. Kapranov, P.: From transcription start site to cell biology. Genome Biology10(4), 217 (2009)
16. Kelley, D.R., Reshef, Y.A., Bileschi, M., Belanger, D., McLean, C.Y., Snoek, J.:Sequential regulatory activity prediction across chromosomes with convolutionalneural networks. Genome research28(5), 739–750 (2018)
17. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint-arXiv:1412.6980 (2014)
18. Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., Inman, D.J.: 1dconvolutional neural networks and applications: A survey (2019)
19. LeCun, Y., Bengio, Y.: Convolutional Networks for Images, Speech, and TimeSeries, p. 255–258. MIT Press, Cambridge, MA, USA (1998)
20. Lee-Thorp, J., Ainslie, J., Eckstein, I., Ontanon, S.: Fnet: Mixing tokenswith fourier transforms (2021).
21. Lizio, M., Harshbarger, J., Shimoji, H., Severin, J., Kasukawa, T., Sahin, S.,Abugessaisa, I., Fukuda, S., Hori, F., Ishikawa-Kato, S., et al.: Gateways to thefantom5 promoter level mammalian expression atlas. Genome biology16, 1–14(2015)
22. Sanyal, A., Lajoie, B.R., Jain, G., Dekker, J.: The long-range interaction landscapeof gene promoters. Nature489(7414), 109–113 (Sep 2012)
23. Sieber Patricia, Platzer Matthias, S.S.: The definition of open reading frame revis-ited. [PMID]. PubMed34(3), 167–170 (2018)
24. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser,L., Polosukhin., I.: Attention is all you need doi. arXiv 1706.03762 (2017)
25. Zhou, J., Theesfeld, C.L., Yao, K., Chen, K.M., Wong, A.K., Troyanskaya, O.G.:Deep learning sequence-based ab initio prediction of variant effects on expressionand disease risk. Nature genetics50(8), 1171–1179 (2018)