International Journal of
Molecular Sciences

MDPI

Article

Predicting Transcription Factor Binding Sites with Deep Learning

Nimisha Ghosh [1,*], Daniele Santoni [2], Indrajit Saha [3] and Giovanni Felici [2]

[1] Department of Computer Science and Information Technology, Institute of Technical Education and Research, Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar 751030, India
[2] Institute for System Analysis and Computer Science "Antonio Ruberti", National Research Council of Italy, 00185 Rome, Italy; daniele.santoni@iasi.cnr.it (D.S.); giovanni.felici@iasi.cnr.it (G.F.)
[3] Department of Computer Science and Engineering, National Institute of Technical Teachers' Training and Research, Kolkata 700106, India; indrajit@nitttrkol.ac.in
* Correspondence: nimishaghosh@soa.ac.in

Abstract: Prediction of binding sites for transcription factors is important to understand how the latter regulate gene expression and how this regulation can be modulated for therapeutic purposes. A consistent number of references address this issue with different approaches, Machine Learning being one of the most successful. Nevertheless, we note that many such approaches fail to propose a robust and meaningful method to embed the genetic data under analysis. We try to overcome this problem by proposing a bidirectional transformer-based encoder, empowered by bidirectional long-short term memory layers and with a capsule layer responsible for the final prediction. To evaluate the efficiency of the proposed approach, we use benchmark ChIP-seq datasets of five cell lines available in the ENCODE repository (A549, GM12878, Hep-G2, H1-hESC, and Hela). The results show that the proposed method can predict TFBS within the five different cell lines very well; moreover, cross-cell predictions provide satisfactory results as well. Experiments conducted across cell lines are reinforced by the analysis of five additional lines used only to test the model trained using the others. The results confirm that prediction across cell lines remains very high, allowing an extensive cross-transcription factor analysis to be performed from which several indications of interest for molecular biology may be drawn.

Keywords: capsule network; deep learning; DNA sequences; transcription factor binding sites (TFBSs)

Citation: Ghosh, N.; Santoni, D.; Saha, I.; Felici, G. Predicting Transcription Factor Binding Sites with Deep Learning. *Int. J. Mol. Sci.* **2024**, *25*, 4990. https://doi.org/10.3390/ijms25094990

Academic Editor: Alexandre G. De Brevern

Received: 20 April 2024
Accepted: 28 April 2024
Published: 3 May 2024

1. Introduction

Transcription Factors (TFs) are proteins that bind to certain genomic sequences and influence a wide range of cellular functions [1,2]. TFs bind to DNA regulatory sequences, which are known as Transcription Factor Binding Sites (TFBSs), typically of size 4–30 bp [3–5], where they modulate the gene transcription while playing an important role in cellular processes [6–8]. The correct prediction of TFBSs is crucial for characterising certain functional aspects of the genome as well as for explaining the organisation of specific sequence expression in complex organisms [9–11]. High-throughput sequencing technology has led to the generation of vast amounts of experimental data about TFBS (e.g., JASPAR [12], TRANSFAC [13], etc.) that now motivate the adoption of methods to identify TFBSs via deep learning approaches.

Many researchers have proposed machine learning methods to identify TFBSs. In this regard, Wong et al. [14] put forth kmerHMM to identify TFBS, where a Hidden Markov Model (HMM) was trained for the underlying motif representation, followed by belief propagation to extract multiple motifs from HMM. To predict DNA binding sites, Ghandi et al. [15] proposed gkm-SVM, which uses a tree for the calculation of the kernel matrix. However, traditional machine learning models usually rely on manual feature extraction, and fail to properly address large-scale datasets. Recently, a number of deep learning models have been specifically developed for computer vision [16,17]

and natural language processing [18]. Models of the same nature have been applied to solve problems in computational biology and bioinformatics as well [19–21]. Deep neural network-based methods such as DeepBind [22] and DeepSEA [23] show competitive or better results compared to traditional methods such as Markov models, support vector machines, hierarchical mixture models, discriminative maximum conditional likelihood, and random forests. DeepBind [22] demonstrates the capabilities of deep learning to assess sequence specificity from experimental data, offering a scalable, adaptable, and integrated calculating technique for finding patterns. DeepBind is the first technology to ever address the demand for precise modelling of protein target binding motifs. A long short-term recurrent convolutional network called DeeperBind [24] is used to anticipate the specificities of how proteins will bind to DNA probes. In order to effectively account for the contributions provided by various sub-regions in DNA sequences, DeeperBind can describe the positional dynamics of probe sequences. It can also be trained and evaluated on datasets with sequences of different lengths. Quang et al. [25] proposed DanQ, which combines CNNs and a bidirectional long short-term memory network (BiLSTM) to predict binding sites. Zeng et al. [26] used multiple CNN architectures for the prediction of DNA sequence binding using an extensive collection of transcription factor datasets. In order to split the DNA binding sequence into overlapping pieces and predict TFBS, Farrel et al. [27] presented an effective pentamer approach. In [28], Qin et al. proposed TFImpute to predict cell-specific TFBS from ChIP-seq data. This method incorporates TFs and cell lines into continuous vectors that are used as inputs to the model. DeepSNR, proposed by Salekin et al. [29], uses a CNN-Deconvolutional model to predict transcription factor binding locations at single-nucleotide resolution. DeepFinder [30] is an enhanced three-stage DNA motif predictor for large-scale pattern analysis; it uses TFBS-associated deep learning neural networks to build the motif model. For data on imbalanced DNA–protein binding sites, Zhang et al. [31] suggested a new prediction approach. Their technique employs a bootstrap algorithm to undersample negative data, while adaptive synthesis is used to oversample positive data. To further capture long-term relationships between DNA sequence motifs, DeepSite [32] uses CNN and BiLSTM. In [32], the authors considered sequence dependencies and addressed the issue of extracting valid information from huge amounts of data while precisely locating motif information in imbalanced data. Yang et al. [33] used deep neural networks along with binomial distribution to enhance motif prediction in the human genome as a way to help with TFBS identification and aid motif prediction accuracy. In [34], Chen et al. used deep learning to develop a TF binding prediction tool known as DeepGRN. The first part of the model is a convolutional layer, while the BiLSTM nodes are recurrent units. Multi-scale convolution along with LSTM (MCNN-LSTM) were chosen in [35] to accurately predict TFBS. The results showed that MCNN-LSTM outperformed several existing TFBS predictors. Zhang et al. [36] combined a convolutional autoencoder with a convolutional neural network (CAE-CNN) to predict TFBS, and used a gated unit to understand the features better. Their primary contribution is in the integration of supervised and unsupervised learning methods to predict TFBS. In [37], Jing et al. used a metalearning-based CNN method (MLCNN) to predict TFBS. The performance of their MLCNN was competitive with or superior to other state-of-the-art CNN methods. A hybrid convolutional recurrent neural network (CNN/RNN) architecture known as CRPTS was proposed in [38] to predict TFBSs by combining DNA sequence and DNA shape features. Cao et al. [39] proposed DeepARC, which combines a convolutional neural network (CNN) and recurrent neural network (RNN) to predict TFBS. DeepARC uses an encoding method combining one-hot encoding and DNA2Vec. This method showed promising results in terms of AUC for benchmark datasets; however, DeepARC lacks efficient encoding policies.

Thus, it can be concluded that despite deep learning being widely applied in the prediction of TFBSs and many aspects of deep learning being well-explored in the context of TFBS prediction, methods based on transformers remain partially unexplored. In addition, capsule

networks have already been used in natural language processing (NLP) for text [40,41] and tweet act classification [42] with competitive results, as well as in bioinformatics [43,44].

Experimental results show that our proposed method performs markedly better than the existing state-of-the-art. Our experiments allow us to make some observations about cross-cell line and cross-transcription factor predictions that are of potential biological interest. In the sections below, we provide an overview of the obtained results and describe our main contributions and findings.

2. Results

DNABERT-Cap was trained to predict TFBSs in DNA sequences and tested on a variety of data. The reported results are the average of multiple runs on 500,000 randomly chosen sequences for the cell lines A549, GM12878, Hep-G2, H1-hESC, and Hela. To show its effectiveness, we compared the results with those of three baselines as well as with other state-of-the-art approaches based on the performance metrics described below. Additional tests were conducted on other cell lines (DnD41, GM12891, GM12892, Huvec, and MCF7) not used in training to test the generalization capabilities of the proposed model.

2.1. Performance Metrics

The different performance parameters used in this work are Accuracy, Recall, Specificity, Mathew's Correlation Coefficient (MCC), and Area Under the Receiver Operating Characteristic Curve (AUC) [39].

2.2. Performance Comparison with Baselines

To show the efficacy of DNABERT-Cap and better evaluate the role of its different components, we compared the results across the following baseline models of increasing complexity:

1. Fine-tuned DNABERT model (Baseline-1): For comparison purposes, the original DNABERT model was fine-tuned with our dataset, where we have added a classification layer on top of DNABERT.
2. DNABERT+CL+BiLSTM+CE (Baseline-2): The capsule layer used to calculate the loss was removed from this baseline, with the categorical cross-entropy loss used as the loss function instead.
3. DNABERT+CL+Capsule Layer (Baseline-3): The BiLSTM layer was removed.

The results in terms of accuracy, recall, specificity, MCC, and AUC are reported in Table 1. As can be observed from Table 1, DNABERT-Cap provides very good results and outperforms the other baselines. When compared to the fine-tuned DNABERT model, DNABERT-Cap shows an improvement of around 5% in terms of accuracy for cell line A549. Similar improvements can be observed for the other metrics and cell lines. Baseline-2 which includes DNABERT with the convolutional and BiLSTM layers but without the advantage of the capsule layer, also shows competitive results compared to baseline-1 for all cell lines considered in this work.

Table 1. Summary of the performance of DNABERT-Cap and the selected baselines. The results indicate that DNABERT with Capsule Network has the best overall performance. Bold indicates the maximum value in each column.

Model	Cell Line	Accuracy (%)	Recall (%)	Specificity (%)	MCC	AUC
Fine-tuned DNABERT model (Baseline-1)	A549	79.14 ± 0.407	78.55 ± 2.511	79.72 ± 0.426	0.609 ± 0.015	0.873 ± 0.004
	GM12878	78.13 ± 0.226	76.87 ± 2.116	78.43 ± 0.246	0.608 ± 0.004	0.854 ± 0.001
	Hep-G2	79.56 ± 0.116	78.20 ± 1.413	79.88 ± 0.118	0.609 ± 0.004	0.876 ± 0.000
	H1-hESC	78.11 ± 0.412	77.14 ± 3.126	78.13 ± 0.426	0.603 ± 0.005	0.858 ± 0.001
	Hela	78.19 ± 0.215	77.98 ± 1.457	78.56 ± 0.219	0.604 ± 0.006	0.873 ± 0.002
DNABERT+CL+BiLSTM +CE (Baseline-2)	A549	80.45 ± 0.310	80.63 ± 2.026	80.01 ± 0.315	0.629 ± 0.004	0.890 ± 0.000
	GM12878	80.32 ± 0.167	78.73 ± 2.551	80.67 ± 0.171	0.625 ± 0.006	0.877 ± 0.001

Table 1. *Cont.*

Model	Cell Line	Accuracy (%)	Recall (%)	Specificity (%)	MCC	AUC
DNABERT+CL+BiLSTM +CE (Baseline-2)	Hep-G2	81.65 ± 0.488	80.66 ± 2.073	81.37 ± 0.483	0.639 ± 0.009	0.896± 0.004
	H1-hESC	80.51 ± 0.334	78.01 ± 2.778	80.59 ± 0.339	0.631 ± 0.004	0.892 ± 0.003
	Hela	81.64 ± 0.145	69.02 ± 1.887	81.55 ± 0.146	0.643 ± 0.002	0.905 ± 0.002
DNABERT+CL+Capsule Layer (Baseline-3)	A549	83.58± 0.331	83.03± 1.734	79.58 ± 0.336	0.681 ± 0.004	0.915 ± 0.001
	GM12878	82.65 ± 0.177	77.57 ± 2.374	82.65 ± 0.181	0.659 ± 0.008	0.903± 0.000
	Hep-G2	84.76 ± 0.414	79.92 ± 1.483	84.68 ± 0.417	0.698 ± 0.001	0.920 ± 0.001
	H1-hESC	82.62 ± 0.357	78.85 ± 2.656	82.59 ± 0.351	0.654 ± 0.003	0.904± 0.001
	Hela	83.09 ± 0.126	80.68 ± 1.913	83.00 ± 0.126	0.662 ± 0.002	0.908± 0.002
DNABERT-Cap	A549	84.66 ± 0.302	81.57 ± 1.711	84.65 ± 0.311	0.696 ± 0.004	0.925 ± 0.001
	GM12878	83.52 ± 0.173	81.11 ± 2.110	83.52 ± 0.173	0.671 ± 0.003	0.913 ± 0.001
	Hep-G2	**85.49 ± 0.157**	**83.43 ± 1.640**	**85.46 ± 0.161**	**0.710 ± 0.003**	**0.930 ± 0.001**
	H1-hESC	83.43 ± 0.350	78.39 ± 2.652	83.50 ± 0.305	0.674 ± 0.003	0.914 ± 0.001
	Hela	83.94 ± 0.112	80.46 ± 1.834	83.89 ± 0.105	0.680 ± 0.002	0.917 ± 0.000

Compared to baseline-2, where the capsule layer is not taken into account, the proposed model shows improvement in terms of accuracy, specificity, MCC, and AUC of around 4% for A549. However, baseline-3 shows better performance in terms of recall. In this regard, AUC is a better parameter for determining which baseline has the best performance [45]. As can be seen from the results, DNABERT-Cap has improved performance in terms of AUC. This improved performance is reflected for all the other cell lines as well. It is worth noting here that the overall improved performance of the proposed model compared to baseline-3 can be attributed to the addition of the BiLSTM layer. All of our experiments were performed considering a confidence level of 95%.

2.3. Performance Comparison with State-of-the-Art Predictors

In order to further analyse the performance of DNABERT-Cap, we compared it with DeepARC [39], DeepTF [35], CNN-Zeng [26], and DeepBind [22] considering the same cell lines as mentioned previously. Table 2 reports the results of these comparisons considering the average of the five cell lines. As is evident from the table, the proposed model has the best predictive performance among all the other state-of-the-art approaches in terms of accuracy, specificity, MCC, and AUC. With respect to recall, DeepARC shows the best performance. Compared to DeepARC, DNABERT-Cap has an improved performance of 1.11%, 0.01%, 2.2%, and 1.10% for accuracy, specificity, MCC, and AUC, respectively. Figure 1 reports the AUC of each method for the five cell lines. It should be noted that in [43,44], the authors used capsule networks for the prediction of transcription factor binding sites. The sequence encoding methods were one-hot encodingand dna2vec, respectively. Thus, to a certain extent, the results are not directly comparable, as these papers did not mention the specific cell lines used in their work. However, the average results reported in these papers are lower than those of DNABERT-Cap.

Table 2. Summary of of DNABERT-CAP performance compared to the state-of-the-art. DNABERT-Cap had the best overall performance among all of the compared prediction models. Bold indicates the maximum value in each column.

Model	Accuracy (%)	Recall (%)	Specificity (%)	MCC
DNABERT-Cap	**84.21**	80.99	**84.20**	**0.686**
DeepARC	83.10	**82.02**	84.19	0.664
DeepTF	80.98	77.44	81.36	0.632

Table 2. *Cont.*

Model	Accuracy (%)	Recall (%)	Specificity (%)	MCC
CNN-Zeng	79.92	72.12	81.96	0.619
DeepBind	79.82	72.64	81.44	0.609

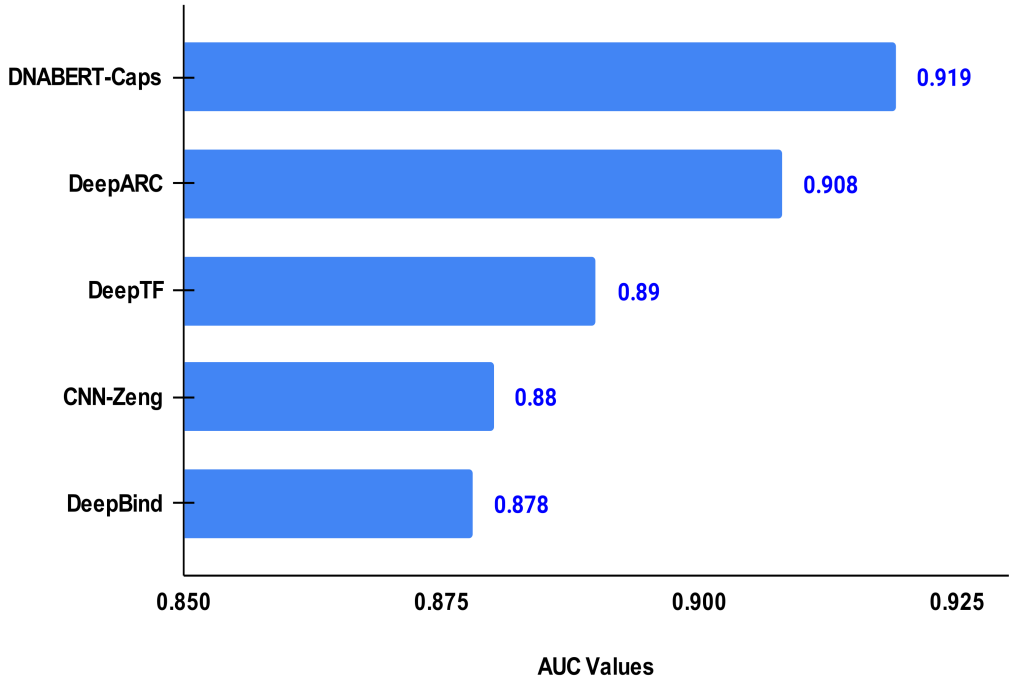

Figure 1. Chart depicting the area under the ROC curve metrics achieved by DNABERT-Cap and the four selected state-of-the-art predictors, with DNABERT-Cap having the best AUC value.

2.4. Cross-Cell Line Prediction

We additionally considered the generalization capability of DNABERT-Cap across the different cell lines. We used five additional cell lines (DnD41, GM12891, GM12892, Huvec, and MCF7) for the purpose of verifying whether models that recognize the binding sites learned from a given cell line can be used to recognize binding sites in a different cell line as well. We restricted our analysis to the value of AUC due to its desirable properties. The overall AUC values across cell lines are reported in Table 3. The general finding from these results is that cross-cell line prediction works well. While the main diagonal in the first five columns still appears to dominate the table, the off-diagonal AUCs are very satisfactory; moreover, when shifting to the five last columns of the table, we find remarkably high values that remain below 0.9 only briefly for GM12891 and GM12892 and partially for MCF7. The binding sites in cell line DnD41 appear to be particularly well-recognised from those appearing in the training cell lines, with the top results being for A549 with 0.956 and H1-hESC with 0.957.

More focused information can be mined from the analysis of results restricted to a single transcription factor of specific biological interest. In Table 4, we report the case of CTCF as an example. Recalling that the binding sites for this factor are among the most frequent in this analysis, appearing more than 17% of the time on average in both the training and test datasets, it can be seen from the table that the very good cross-cell line prediction for DnD41 is confirmed when restricted to only CTCF. A reasonable interpretation of these results is that CTCF is well represented in all of the training cell lines and is very prevalent in DnD41 (more than 95%); indeed, the corresponding values for DnD41 averaged over all TFs are slightly lower, as reported in Table 3.

Table 3. Summary of AUC for cross-cell line prediction. Bold indicates the maximum value in each row.

		Testing									
	Cell Lines	**A549**	**GM12878**	**Hep-G2**	**H1-hESC**	**Hela**	**DnD41**	**GM12891**	**GM12892**	**Huvec**	**MCF7**
Training	A549	0.926	0.860	0.895	0.901	0.908	**0.956**	0.840	0.865	0.920	0.925
	GM12878	0.886	0.914	0.851	0.878	0.878	**0.948**	0.847	0.881	0.908	0.894
	Hep-G2	0.910	0.846	0.932	0.894	0.895	**0.946**	0.826	0.853	0.901	0.899
	H1-hESC	0.899	0.865	0.875	0.917	0.910	**0.957**	0.847	0.872	0.901	0.889
	Hela	0.916	0.865	0.885	0.898	0.917	**0.953**	0.840	0.860	0.920	0.928

Table 4. Summary of AUC for cross-cell line prediction of CTCF. Bold indicates the maximum value in each row.

		Testing									
	Cell Lines	**A549**	**GM12878**	**Hep-G2**	**H1-hESC**	**Hela**	**DnD41**	**GM12891**	**GM12892**	**Huvec**	**MCF7**
Training	A549	**0.974**	0.965	0.973	0.965	0.967	0.962	0.941	0.943	0.976	0.957
	GM12878	0.963	0.960	0.962	0.949	0.953	0.953	0.929	0.932	**0.967**	0.942
	Hep-G2	0.965	0.958	**0.970**	0.956	0.955	0.951	0.930	0.933	0.969	0.940
	H1-hESC	0.972	0.969	0.974	0.965	0.966	0.962	0.940	0.943	**0.976**	0.955
	Hela	0.969	0.963	0.970	0.958	0.959	0.959	0.933	0.936	**0.972**	0.952

Having ascertained that cross-cell line prediction works reasonably well, we verified the role of the different TFs by testing whether certain TFs are easier to predict than others, whether such difference, if present, might depend on the cell line used in the experiments, and finally, whether (as expected) those TFs that were more frequent in training data would be easier to recognize.

2.5. Frequency of Transcription Factors in Sampling

We investigated the relationship between the frequency of TFs in the training datasets and the ability to recognize them. The latter is expressed by the AUC of the ROC curve associated with the sequences of that TF in a testing cell line dataset. Specific ROC curves and the related AUC were computed for each cell line and for each TF. It is reasonable to assume that when a given TF occurs in the training set with consistent frequency, then it will be possible to better learn how to recognize it, and the model will be better able to recognize sequences associated with that TF. To test this hypothesis, we computed the Pearson correlation value and the related p-value between the average frequency in the training for each TF, averaged on the training datasets corresponding to the five cell lines used for training, along with the AUC averaged on the ten testing datasets associated with different cell lines. To provide consistency to this analysis, we discarded TFs that were scarce in testing (less than 500 positive samples). The results are displayed in the scatter plot in Figure 2. The highly significant p-value ($p < 0.001157$) confirms that patterns are present that depend on specific TFs; indeed, the association between frequency and AUC is definitely not driven by chance and has correlation value of 0.35, suggesting that while frequency has a significant influence, it is not the only factor playing a role in recognition.

In the rightmost stripe of the plot, it can be seen that a number of TFs have AUCs above 95%. While several of these have very high frequency (CTCF with 0.19, Rad21 with 0.07), many with AUCs above 0.9 are supported by a frequency in the training sets that is only a few decimal points above zero. Notably, the TF with largest AUC is not the one with the maximum frequency (Rad21, with AUC 96% and frequency 0.07).

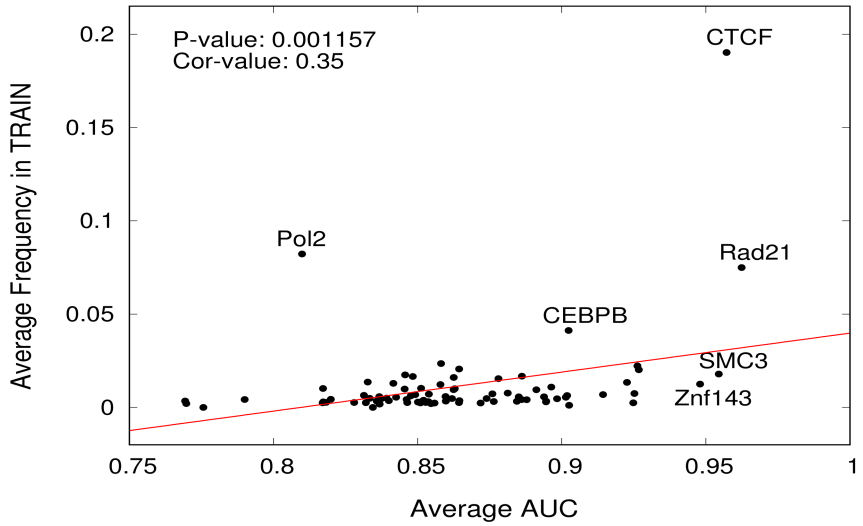

Figure 2. For all the TFs (represented as black dots), the average AUC value (*x*-axis) is reported as a function of the average frequency of the TFs in the training dataset (*y*-axis). Pearson correlation values (Cor-value 0.35) between the two variables and the corresponding *p*-value (0.001157) are reported in the left top corner.

2.6. Transcription Factors Not Appearing in Training

In addition to the behaviour discussed above, there were a large number of TFs that were recognized very well despite not appearing in the training sets at all. For such TFs, by restricting our analysis to those with at least 250 samples in the test sets, we obtain the average AUCs shown in Table 5. In the table, it can be seen that the TFs that appear in the test sets from cell line A549 and do not appear in any of the training sets exhibit an average AUC of 0.875, peaking for training set from Hela with an average AUC of 0.894. Very high values are found for those appearing in the Huvec cell line as well. In general, the values in the table are quite high, and indicate that genomic properties of the binding sites are transferred among transcription factors. In this regard, Table 6 reports the top ten TFs based on AUC that were recognised well during testing despite being absent from the training dataset. For example, although Znf143 is completely absent in A549, it was recognised with an average AUC = 0.968 by models trained on GM12878. One of the most well-recognised TF in this set is SMC3, with an average AUC > 0.95. The complete table is provided as a Supplementary File. Please note that the blanks in Table 5 indicate that the test cell lines contained TFs that were present in the training cell lines as well. For example, DnD41 had no TFs that were absent in GM12878. The diagonals of the five cell lines used for training are blank as well.

Table 5. Summary of AUC for TFs that were well-recognised despite not being present in the training set (limited to TFs with more than 250 samples in testing; Bold indicates the maximum value in each row).

		Testing									
	Cell Lines	**A549**	**GM12878**	**Hep-G2**	**H1-hESC**	**Hela**	**DnD41**	**GM12891**	**GM12892**	**Huvec**	**MCF7**
Training	A549	-	0.839	0.867	0.865	**0.894**	0.788	0.813	0.841	**0.894**	0.869
	GM12878	0.857	-	0.800	0.828	0.836	-	-	-	**0.877**	0.774
	Hepg2	**0.876**	0.821	-	0.839	0.859	-	0.803	0.824	0.860	0.852
	H1-heSC	0.855	0.841	0.841	-	**0.859**	-	0.826	0.843	0.847	0.835
	Hela	**0.893**	0.841	0.858	0.851	-	-	0.832	0.845	0.858	0.793

Table 6. Top ten TFs, based on AUC, that were well-recognised despite not being present in the training set. (limited to TFs with more than 250 samples in testing).

Cell Train	Cell Test	TF	Summation of Positive Test Data	Mean AUC
A549	GM12878	Znf143	384	0.968
A549	Hela	SMC3	1176	0.966
A549	Hela	JunD	886	0.963
A549	Hep-G2	SMC3	584	0.962
A549	Hela	c-Fos	299	0.956
H1hesc	Hela	SMC3	1176	0.955
A549	GM12878	SMC3	584	0.955
A549	MCF7	c-Fos	6801	0.954
H1hesc	GM12878	SMC3	584	0.953
Hela	A549	FOSL2	1254	0.952

3. Discussion

In this work, we present DNABERT-Cap, a transformer-based capsule network to predict TFBSs. Our results show that the combination of these two powerful deep learning methods significantly improves prediction performance. Moreover, we performed ablation studies in order to report the utility of applying DNABERT with a capsule network for such prediction. In this regard, the improved performance of the proposed model can be attributed to the ability of DNABERT embeddings to generate rich bidirectional contextual representations, thanks to multiple attention heads concurrently focusing on various input sections. Moreover, the superior ability of the capsule network to retain information about the location of an object represents an advantage over traditional convolutional networks, which can lose track of such information due to the pooling layers only extracting the most important information from the data. The proposed model further benefits from joint optimisation of DNABERT and capsule layers along with the convolutional and BiLSTM layers, allowing it to learn important attributes and features of TFBSs. Similar to how spatial correlation is crucial for correctly identifying objects in images, the ordering of *k*-mers and their semantic representations is important for DNA sequences. The proposed model seems to be able to identify such relationships, allowing it to make better predictions. In future research work, attempts could be made to further improve the performance of the model by considering other parameters for feature embedding in addition to DNA sequences. Moreover, DNABERT-Cap could be tested on other prediction problems in bioinformatics, such as predicting RNA–protein and DNA–protein binding sites from sequences.

4. Materials and Methods

In this section, data preparation is elaborated, followed by the discussion of the pipeline of the proposed work.

4.1. Data Preparation

The benchmark dataset from Encyclopedia of DNA Elements (ENCODE) [46] was used to acquire the TFBS data analysed by the ChIPseq method. These data were used to train and test the proposed model. Data preprocessing was the same as considered in [26], where the positive samples with 101 bps were generated in the centre of each ChIP-seq peak and the negative samples were obtained by recombining the positive sequence conserving dinucleotide frequencies. The positive and negative samples were then distinguished based on the presence or absence of TFBSs in a sequence. For experimental purposes, five cell lines, viz., A549, GM12878, Hep-G2, H1-hESC, and Hela, encompassing 352 datasets from the TF-690 Chip-Seq dataset, were considered for non-cross-cell line predictions. Based

on availability of resources, 500,000 sequences for each of the five cell lines were selected randomly from these datasets. This random selection ensured a fair balance between positive and negative sequences. Each dataset was divided into 70% training, 20% test, and 10% validation sets. For cross-cell line predictions, the ten cell lines A549, GM12878, Hep-G2, H1-hESC, Hela, DnD41, GM12891, GM12892, Huvec, and MCF7 were considered, encompassing 404 datasets (352 + 52) from the TF-690 dataset. All of our experiments were conducted on machines with NVIDIA GA100 GPUs.

4.2. DNABERT and Capsule Network

Before delving into the pipeline of the work, a brief discussion of the DNABERT model and capsule networks is provided.

4.2.1. DNABERT

DNABERT [47] is a pretrained bidirectional encoder representation which captures the intricacies of DNA sequences based on both the upstream and downstream nucleotide contexts. A set of sequences divided into k-mer tokens of appropriate sizes is provided as input to DNABERT. Each sequence is represented as a matrix $\mathcal{X}$, where the tokens are embedded into numerical vectors. This matrix captures the contextual information of a sequence by executing a multi-head self-attention mechanism on $\mathcal{X}$:

$$multihead\mathcal{X} = Concatenation(head_1, \cdots, head_h)\mathcal{W}^O \quad (1)$$

where

$$head_i = softmax\left(\frac{\mathcal{X}\mathcal{W}_i^Q \mathcal{X}\mathcal{W}_i^K T}{\sqrt{d_k}}\right).\mathcal{X}\mathcal{W}_i^V. \quad (2)$$

Here, all $\mathcal{W}$s are parameters learned during training of the model. Equations (1) and (2) are performed T times, where T is the number of layers.

4.2.2. Capsule Networks

In order to derive local patterns from a vector sequence, CNNs build convolutional feature detectors [42]. The most noticeable patterns are then chosen using max-pooling. However, CNNs may lose many important information during the pooling process, resulting in poor performance on problems characterised by positional invariance. In contrast, approaches that do not take spatial relationships into account perform flawlessly when making inferences for local patterns; however, they cannot encode the rich structures that may be present in a sequence. In this regard, capsule networks [48] can help to improve efficiency when encoding spatial patterns by including knowledge about the relationships between parts and the whole. Each capsule is a group of neurons, in which the input and output are both vectors. These groups of neurons work together to recognise specific features or patterns. An iterative dynamic routing algorithm [49] helps to determine the most important features from lower to higher layers. As a result, capsule networks generalise a particular class instead of memorising every viewpoint variant of the class, thereby becoming invariant to the viewpoint and showing improved performance compared to CNNs.

4.3. Pipeline of this Work

The pipeline of this work is depicted in Figure 3. Initially, the DNA sequences for each cell line are fed to the DNABERT model in the form of k-mer tokens (k = 6 in this case, as [47] have reported that *6*-mers show the best performance). For each input sequence of tokens, DNABERT returns an embedding. Let $W_A \in \mathbb{R}^{d \times l}$ be the weight matrix for each such embedding, where l is the length of a sequence and d is the dimension of each token representation. Each weight matrix is then passed through a series of layers to obtain the best possible sequence representation for the classification of such sequences as Transcription Factor Binding Sites. The subsequent layers are as follows:

1. **Convolutional Layer:** The output from DNABERT is provided as an input to the convolutional layer. This is represented as

$$\alpha_i = W_b * d_i + b, \tag{3}$$

where the output feature map α_i is produced by a kernel d_i with bias b by applying convolution and η such feature maps are then combined to form a η-channel layer, as follows:

$$A = [\alpha_1, \alpha_2, \cdots, \alpha_\eta]. \tag{4}$$

This layer helps in understanding the importance of a k-mer token in a sequence by concentrating on the feature map.

2. **Bidirectional LSTM Layer:** In order to learn semantic dependencies, the feature vector obtained from the previous layer is passed through a bidirectional Long Short-Term Memory (BiLSTM) network. The long-term dependencies in a sequence are captured using the BiLSTM by sequentially encoding the feature maps into hidden states [42]. In this regard, the η-channel feature vector A is passed through the BiLSTM as follows:

$$\overrightarrow{h}_t = \overrightarrow{LSTM}(\alpha_\eta, h_{t-1}), \tag{5}$$

$$\overleftarrow{h}_t = \overleftarrow{LSTM}(\alpha_\eta, h_{t+1}), \tag{6}$$

with each feature map mapped to forward and backward hidden states. This helps to retain the context-sensitive nature of the tokens. The final hidden state matrix is defined as

$$HS = [h_1, h_2, \cdots, h_t], \tag{7}$$

where $HS \in \mathbb{R}^{t \times 2dim}$ and dim is the number of hidden state. Thus, BiLSTM is important for capturing the context of k-mers in a DNA sequence.

3. **Primary Capsule Layer:** The primary capsule layer was originally introduced to handle the drawbacks of conventional CNNs by replacing the scalar outputs with vector-output capsules, in order to preserve the local order and semantic representations of tokens. Keeping this context in mind, the features obtained from the previous layers in the form of vectors are fed into the primary capsule layer. By sliding over the hidden states HS generated in the previous layer, each kernel d_i generates a sequence of capsules cap_i of dimension dim, thereby creating a channel C_i:

$$C_i = S(d_i * HS + b) \tag{8}$$

where S is the squash function and b represents the capsule's bias weight parameter. This layer captures the local ordering of k-mers in a sequence and its corresponding semantic representations.

4. **Dynamic Routing Between Capsules:** The primary idea behind dynamic routing [49] is to iteratively build a nonlinear map, ensuring that a suitable capsule in the next layer is strongly connected to a lower-level capsule. Moreover, the pooling function of the traditional convolution layer, which normally removes the location information, is replaced with a dynamic routing technique, leading to a more robust network. To ensure that the length of a capsule is within [0, 1], a nonlinear squashing function is applied, as shown in Equation (9):

$$v_y = \frac{||s_y||^2}{1 + ||s_y||} \frac{s_y}{|||s_y||^2} \tag{9}$$

$$\hat{u}_{x|y} = W_{xj} u_x, \tag{10}$$

$$s_y = \sum_x c_{xy} \hat{u}_{y|x}, \tag{11}$$

where v_y is the output vector of capsule y, s_y is its total input, s_y is a weighted sum over all prediction vectors $\hat{u}_{y|x}$ calculated by capsule x and transferred to capsule y, and $\hat{u}_{y|x}$ is calculated by multiplying previous layer capsule output u_x by W_{xy} (weight matrix). This process helps the capsule network to capture the relationship between a subpart and the entire sequence, as detailed in Equations (10) and (11). c_{xy} represents the coupling coefficients calculated by the dynamic routing algorithm; c_{xy} is computed as a softmax b_{xy}, which represents the log-prior probabilities between capsules x and y and is given by:

$$c_{xy} = \frac{exp(b_{xy})}{\sum_k exp(b_{xk})}. \tag{12}$$

The initial coupling coefficients are refined iteratively based on b_{xy}, measuring the agreement between v_y and $\hat{u}_{y|x}$; the agreement can be calculated as $a_{xy} = v_y.\hat{u}_{y|x}$, where a_{xy} is a scalar product and b_{xy} is updated as follows:

$$b_{xy} = b_{xy} + \hat{u}_{y|x}.v_y. \tag{13}$$

This entire procedure reflects the dynamic routing for all capsules s in layer P and capsules y in layer $P+1$. In this work, a dynamic routing algorithm helps to determine the importance and agreement of tokens for a specific task by learning the importance of k-mer tokens in a sequence.

5. **TFBS Capsule Layer:** In this layer, the TFBS capsules are responsible for detecting the TFBS of a given DNA sequence. The sequence vector of the primary capsule is carried forward to the TFBS capsule layer, which generates one vector for each of the TFBS class capsules: one for the class which exhibits the presence of TFBS, and another depicting the absence of the same.
6. **Output:** The output of the TFBS capsule layer has two class capsules (denoted in Figure 3 by the two blue circles), generating outputs with two vectors encoding various properties of features; the lengths are the probabilities of the corresponding class being present in the input data. In order to improve the separation between the two class capsules, the separate margin loss [49] L_b is used in this work:

$$L_b = G_b max(0, m^+ - ||v_k||)^2 + \lambda(1 - G_b)max(0, m^- - ||v_b||)^2 \tag{14}$$

where v_b is the capsule for class b, $G_b = 1$ iff class b is the ground truth, $m^+ = 0.9$, $m^- = 0.1$, and λ is used to tune the weight of an absent class.

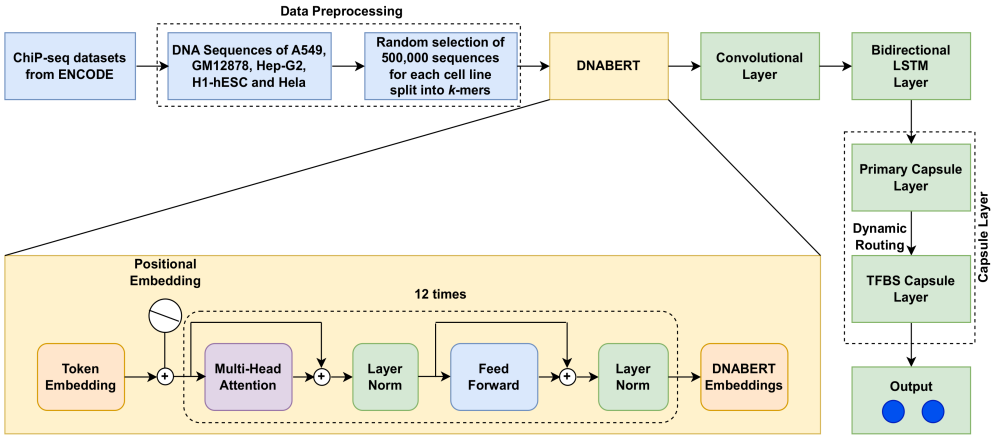

Figure 3. Depiction of the pipeline of the methodology of DNABERT-Cap. The first row represents the main sequences of the components, while the yellow box below shows an exploded view of the DNABERT sequence. The last column of boxes, in green, shows where the capsule layer is in force (the two capsules are represented by blue dots in the output layer).

4.4. Hyperparameters

The pretrained DNABERT [47] model has 12 transformer layers and 768 hidden units, along with 12 attention heads in each layer. The number of units for both the convolutional and BiLSTM layers is 64, while the kernel size for the convolutional layer is 2. Furthermore, a dropout value of 0.3 and a capsule length of 16 are considered along with a batch size of 64. The dynamic routing algorithm with three iterations provided the optimum results, and the Adam optimiser [50] was used for all our experiments. All of these parameters were arrived at after conducting thorough experiments.

5. Conclusions

In this work, we propose DNABERT-Cap, a deep learning mmethod based on DNABERT, CNN, BiLSTM and capsule networks, for the identification of transcription factor binding sites in DNA sequences. The proposed model performs very well when compared with state-of-the-art approaches, showing an accuracy of more than 83% and an AUC of more than 0.91 for all of the five cell lines considered in this work for non-cross-cell line prediction. Additionally, the results for cross-cell line prediction show the robustness of the proposed model, making the model highly useful for verification of TFs across cell lines. As the results shown in this work provide evidence that such a model is able to capture intrinsic TFBS patterns, we plan to extend this work by designing specific models able to identify TFBS patterns for a given TF. The results presented in this paper and in the Supplementary Materials could be of interest for molecular biology studies, as they can allow researchers to verify several hypothesis concerning the behaviour of TFs without the need to resort to in vitro experiments. For example, specific similarities between TFs can be deduced based on their cross-prediction rate, identifying patterns and substructures that would be missed otherwise; alternatively, binding sites for factors of different kinds may be located.

Supplementary Materials: The supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/ijms25094990/s1.

Author Contributions: Conceptualization, N.G.; methodology, N.G.; software, N.G.; validation, N.G., D.S., I.S. and G.F.; formal analysis, N.G. and G.F.; investigation, D.S. and I.S.; resources, N.G. and G.F.; data curation, N.G., D.S. and I.S.; writing—original draft preparation, N.G.; writing—review and editing, D.S., I.S. and G.F.; visualization, N.G. and D.S.; supervision, I.S. and G.F.; project administration, N.G. and I.S.; funding acquisition, N.G. and G.F. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the SERB-SURE scheme (SUR/2022/002836) funded by Government of India and by the PNRR MUR project PE0000013-FAIR, Next Generation EU program, funded by the Italian Government.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets and code used in this work are available at https://github.com/NimishaGhosh/DNABERT-Cap/tree/main, accessed on 27 April 2024.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study, in the collection, analysis, or interpretation of data, in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

BERT	Bidirectional Encoder Representations from Transformers
BiLSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Network
TF	Transcription Factors
TFBS	Transcription Factor Binding Site

References

1. Latchman, D.S. Transcription factors: An overview. *Int. J. Biochem. Cell Biol.* **1997**, *29*, 1305–1312. [CrossRef]
2. Karin, M. Too many transcription factors: Positive and negative interactions. *New Biol.* **1990**, *2*, 126–131. [PubMed]
3. Tompa, M.; Li, N.; Bailey, T.L.; Church, G.M.; De Moor, B.; Eskin, E.; Favorov, A.V.; Frith, M.C.; Fu, Y.; Kent, W.J.; et al. Assessing computational tools for the discovery of transcription factor binding sites. *Nat. Biotechnol.* **2005**, *23*, 137–144. [CrossRef] [PubMed]
4. Tan, G.; Lenhard, B. TFBSTools: An R/bioconductor package for transcription factor binding site analysis. *Bioinformatics* **2016**, *32*, 1555–1556. [CrossRef]
5. Qu, K.; Wei, L.; Zou, Q. A Review of DNA-binding Proteins Prediction Methods. *Curr. Bioinform.* **2019**, *14*, 246–254. [CrossRef]
6. Alexandrov, B.S.; Gelev, V.; Yoo, S.W.; Alexandrov, L.B.; Fukuyo, Y.; Bishop, A.R.; Rasmussen, K.O.; Usheva, A. DNA dynamics play a role as a basal transcription factor in the positioning and regulation of gene transcription initiation. *Nucleic Acids Res.* **2010**, *38*, 1790–1795. [CrossRef]
7. Li, J.; Ou, J.-H. Differential Regulation of Hepatitis B Virus Gene Expression by the Sp1 Transcription Factor. *J. Virol.* **2001**, *75*, 8400–8406. [CrossRef]
8. Wilkinson, A.C.; Nakauchi, H.; Göttgens, B. Mammalian Transcription Factor Networks: Recent Advances in Interrogating Biological Complexity. *Cell Syst.* **2017**, *5*, 319–331. [CrossRef]
9. Lambert, S.A.; Jolma, A.; Campitelli, L.F.; Das, P.K.; Yin, Y.; Albu, M.; Chen, X.; Taipale, J.; Hughes, T.R.; Weirauch, M.T.; et al. The Human Transcription Factors. *Cell* **2018**, *172*, 650–665. [CrossRef]
10. Basith, S.; Manavalan, B.; Shin, T.H.; Lee, G. iGHBP: Computational identification of growth hormone binding proteins from sequences using extremely randomised tree. *Comput. Struct. Biotechnol. J.* **2018**, *16*, 412–420. [CrossRef] [PubMed]
11. Shen, Z.; Lin, Y.; Zou, Q. Transcription factors–DNA interactions in rice: Identification and verification. *Briefings Bioinform.* **2019**, *21*, 946–956. [CrossRef]
12. Fornes, O.; Castro-Mondragon, J.A.; Khan, A.; van der Lee, R.; Zhang, X.; Richmond, P.A.; Modi, B.P.; Correard, S.; Gheorghe, M.; Baranasic, D.; et al. JASPAR 2020: Update of the open-access database of transcription factor binding profiles. *Nucleic Acids Res.* **2019**, *48*, D87–D92. [CrossRef] [PubMed]
13. Matys, V.; Kel-Margoulis, O.V.; Fricke, E.; Liebich, I.; Land, S.; Barre-Dirrie, A.; Reuter, I.; Chekmenev, D.; Krull, M.; Hornischer, K.; et al. TRANSFAC and its module TRANSCompel: Transcriptional gene regulation in eukaryotes. *Nucleic Acids Res.* **2006**, *34*, D108–D110. [CrossRef] [PubMed]
14. Wong, K.C.; Chan, T.K.; Peng, C.; Li, Y.; Zhang, Z. DNA motif elucidation using belief propagation. *Nucleic Acids Res.* **2013**, *41*, e153. [CrossRef]
15. Ghandi, M.; Lee, D.; Mohammad-Noori, M.M.; Beer, M.A. Enhanced Regulatory Sequence Prediction Using Gapped k-mer Features. *PLoS Comput. Biol.* **2014**, *10*, 1–15. [CrossRef] [PubMed]
16. He, K.; Zhang, X.; Ren, S.; Sun, J. Identity Mappings in Deep Residual Networks. In Proceedings of the Computer Vision—ECCV 2016, Amsterdam, The Netherlands, 11–14 October 2016; pp. 630–645.
17. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
18. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2019**, arXiv:1810.04805.
19. Zhao, H.; Tu, Z.; Liu, Y.; Zong, Z.; Li, J.; Xiong, F.; Zhan, J.; Hu, X.; Xie, W. PlantDeepSEA, a deep learning-based web service to predict the regulatory effects of genomic variants in plants. *Nucleic Acids Res.* **2021**, *49*, W523–W529. [CrossRef] [PubMed]
20. Min, S.; Kim, H.; Lee, B.; Yoon, S. Protein transfer learning improves identification of heat shock protein families. *PLoS ONE* **2021**, *16*, 1–14. [CrossRef]
21. Liu, Y.; Zhu, Y.H.; Song, X.; Yu, D.J. Why can deep convolutional neural networks improve protein fold recognition? A visual explanation by interpretation. *Briefings Bioinform.* **2021**, *22*, bbab001. [CrossRef]
22. Alipanahi, B.; Delong, A.; Weirauch, M.T.; Frey, B.J. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat. Biotechnol.* **2015**, *33*, 831–838. [CrossRef]
23. Zhou, J.; Troyanskaya, O.G. Predicting effects of noncoding variants with deep learning–based sequence model. *Nature Methods* **2015**, *12*, 931–934. [CrossRef] [PubMed]
24. Hassanzadeh, H.R.; Wang, M.D. DeeperBind: Enhancing Prediction of Sequence Specificities of DNA Binding Proteins. In Proceedings of the 2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Shenzhen, China, 15–18 December 2016; pp. 178–183.
25. Quang, D.; Xie, X. DanQ: A hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences. *Nucleic Acids Res.* **2016**, *44*, e107. [CrossRef] [PubMed]
26. Zeng, H.; Edwards, M.D.; Liu, G.; Gifford, D.K. Convolutional neural network architectures for predicting DNA–protein binding. *Bioinformatics* **2016**, *32*, i121–i127. [CrossRef] [PubMed]
27. Farrel, A.; Guo, J.-T. An efficient algorithm for improving structure-based prediction of transcription factor binding sites. *BMC Bioinform. Vol.* **2017**, *18*, 1–11. [CrossRef] [PubMed]
28. Qin, Q.; Feng, J. Imputation for transcription factor binding predictions based on deep learning. *PLoS Comput. Biol.* **2017**, *13*, 1–20. [CrossRef] [PubMed]

29. Salekin, S.; Zhang, J.M.; Huang, Y. Base-pair resolution detection of transcription factor binding site by deep deconvolutional network. *Bioinformatics* **2018**, *34*, 3446–3453. [CrossRef] [PubMed]
30. Lee, N.K.; Azizan, F.L.; Wong, Y.S.; Omar, N. DeepFinder: An integration of feature-based and deep learning approach for DNA motif discovery. *Biotechnol. Biotechnol. Equip.* **2018**, *32*, 759–768. [CrossRef]
31. Zhang, Y.; Qiao, S.; Ji, S.; Han, N.; Liu, D.; Zhou, J. Identification of DNA–protein binding sites by bootstrap multiple convolutional neural networks on sequence information. *Eng. Appl. Artif. Intell.* **2019**, *79*, 58–66. [CrossRef]
32. Zhang, Y.; Qiao, S.; Ji, S.; Li, Y. DeepSite: Bidirectional LSTM and CNN models for predicting DNA–protein binding. *Int. J. Mach. Learn. Cybern. Vol.* **2020**, *11*, 841–851. [CrossRef]
33. Yang, J.; Ma, A.; Hoppe, A.D.; Wang, C.; Li, Y.; Zhang, C.; Wang, Y.; Liu, B.; Ma, Q. Prediction of regulatory motifs from human Chip-sequencing data using a deep learning framework. *Nucleic Acids Res.* **2019**, *47*, 7809–7824. [CrossRef]
34. Chen, C.; Hou, J.; Shi, X.; Yang, H.; Birchler, J.A.; Cheng, J. DeepGRN: Prediction of transcription factor binding site across cell-types using attention-based deep neural networks. *BMC Bioinform.* **2021**, *22*, 38. [CrossRef] [PubMed]
35. Bao, X.R.; Zhu, Y.H.; Yu, D.J. DeepTF: Accurate Prediction of Transcription Factor Binding Sites by Combining Multi-scale Convolution and Long Short-Term Memory Neural Network. In *Intelligence Science and Big Data Engineering. Big Data and Machine Learning*; Springer: Nanjing, China, 2019; pp. 126–138.
36. Zhang, Y.; Qiao, S.; Zeng, Y.; Gao, D.; Han, N.; Zhou, J. CAE-CNN: Predicting transcription factor binding site with convolutional autoencoder and convolutional neural network. *Expert Syst. Appl.* **2021**, *183*, 115404. [CrossRef]
37. Jing, F.; Zhang, S.W.; Zhang, S. Prediction of the transcription factor binding sites with meta-learning. *Methods* **2022**, *203*, 207–213. [CrossRef] [PubMed]
38. Wang, S.; Zhang, Q.; Shen, Z.; He, Y.; Chen, Y.-H.; Chen, Z.-H.; Li, J.; Huang, D.-S. Predicting transcription factor binding sites using DNA shape features based on shared hybrid deep learning architecture. *Mol. Ther. Nucleic Acids* **2021**, *24*, 154–163. [CrossRef] [PubMed]
39. Cao, L.; Liu, P.; Chen, J.; Deng, L. Prediction of Transcription Factor Binding Sites Using a Combined Deep Learning Approach. *Front. Oncol.* **2022**, *12*, 893520. [CrossRef] [PubMed]
40. Kim, J.; Jang, S.; Park, E.; Choi, S. Text classification using capsules. *Neurocomputing* **2020**, *376*, 214–221. [CrossRef]
41. Chen, B.; Xu, Z.; Wang, X.; Xu, L.; Zhang, W. Capsule Network-Based Text Sentiment Classification. *IFAC-PapersOnLine* **2020**, *53*, 698–703. [CrossRef]
42. Saha, T.; Jayashree, S.R.; Saha, S.; Bhattacharyya, P. BERT-Caps: A Transformer-Based Capsule Network for Tweet Act Classification. *IEEE Trans. Comput. Soc. Syst.* **2020**, *7*, 1168–1179. [CrossRef]
43. Zhang, Q.; Yu, W.; Han, K.; Nandi, A.K.; Huang, D.-S. Multi-Scale Capsule Network for Predicting DNA-Protein Binding Sites. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2021**, *18*, 1793–1800. [CrossRef]
44. Cheng, J.; Wang, Z.; Liu, Y.; Huang, W. CapBind: Prediction of Transcription Factor Binding Sites Based on Capsule Network. In Proceedings of the 2021 6th International Conference on Computational Intelligence and Applications (ICCIA), Xiamen, China, 11–13 June 2021; pp. 31–35. [CrossRef]
45. Ghosh, N.; Banerjee, I. IoT-based freezing of gait detection using grey relational analysis. *Internet Things* **2021**, *13*, 100068. [CrossRef]
46. The ENCODE Project Consortium. An integrated encyclopedia of DNA elements in the human genome. *Nature* **2012**, *489*, 57–74. [CrossRef] [PubMed]
47. Ji, Y.; Zhou, Z.; Liu, H.; Davuluri, R.V. DNABERT: Pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome. *Bioinformatics* **2021**, *37*, 2112–2120. [CrossRef] [PubMed]
48. Hinton, G.E.; Sabour, S.; Frosst, N. Matrix capsules with EM routing. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
49. Sabour, S.; Frosst, N.; Hinton, G.E. Dynamic Routing between Capsules. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 3859–3869.
50. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015.