Deep Learning to Identify Transcription Start Sites from CAGE Data

Hansi Zheng
Department of Electrical Engineering and Computer Science
University of Central Florida
Orlando, United States
hansizheng@knights.ucf.edu

Xiaoman Li
Department of Biomedical Science
University of Central Florida
Orlando, United States
xiaoman@mail.ucf.edu

Haiyan Hu
Department of Electrical Engineering and Computer Science
University of Central Florida
Orlando, United States
haihu@cs.ucf.edu

***Abstract*—Gene transcription start site (TSS) identification is important to understanding transcriptional gene regulation. Cap Analysis Gene Expression (CAGE) experiments have recently become common practice for direct measurement of TSSs. Currently, CAGE data available in public databases created unprecedented opportunities to study gene transcriptional initiation mechanisms under various cellular conditions. However, due to potential transcriptional noises inherent in CAGE data, in-silico methods are required to identify bonafide TSSs from noises further. Here we present a computational approach dlCAGE, an end-to-end deep neural network to identify TSSs from CAGE data. dlCAGE incorporate de-novo DNA regulatory motif features discovered by DeepBind model architecture, as well as existing sequence and structural features. Testing results of dlCAGE in several cell lines in comparison with current state-of-the-art approaches showed its superior performance and promise in TSS identification from CAGE experiments.**

Keywords—cap analysis gene expression, transcription start site, promoter region, deep neural network, gene expression.

I. INTRODUCTION

Cap analysis gene expression (CAGE) has become a standard experimental technique that can directly perform genome-wide measurement of the 5' end of mRNAs in a biological sample [1]. Thus, CAGE has great potential to help discover transcription start sites (TSSs) that are essential to understanding gene regulation [2]. So far, a large number of CAGE datasets have been deposited into public databases [3], providing an unprecedented opportunity to study gene transcriptional regulation. However, CAGE experiments have been shown containing various types of transcriptional noises, such as splicing byproducts and capped molecules [4, 5]. Therefore, computational methods are required to eliminate such transcriptional noises further.

Computational methods have been developed to analyze CAGE data [5-9]. Most of them focus on clustering CAGE tags as TSSs. For example, PARACLU is able to cluster CAGE tags at different scales controlled by a density parameter [8]. The resulted multiple-scaled clusters were used to illustrate the hierarchical organization of promoter structures that may correspond to various gene regulatory processes. RECLU improved PARACLU to identify reproducible clusters across replicas [7]. To that end, they defined the stability criteria for each CAGE cluster, which was subsequently used to perform reproducibility evaluation. CAGEr is an R package tool that can process CAGE data from different resources [5]. It can also cluster CAGE tags as identified TSSs, and provide CAGE data analysis across samples, e.g., performing the differential test on CAGE cluster-represented TSS usage between two samples. The TSS classifier included in TOMTOOLS was developed to separate CAGE-represented TSSs from non-TSSs [9]. TSS classifier is based on a Gaussian mixture model that can capture the relative distribution of 4-mer occurrences surrounding TSSs. The generated 256 vectors of value representative for each sequence and a Random Decision Tree were then used to classify positive and negative sequences. However, studies have frequently shown that both sequence and structural features are important to TSS identification [10-13]. Based on Support Vector Machines (SVM) and Stochastic Gradient Boosting models, the most recent study ADAPT-CAGE attempted accurate identification of TSSs from CAGE data using multiple features [14]. The features include CAGE tag clusters, Polymerase II (Pol II) motifs such as TATA-box, MTE, and CCAAT-Box, and structural features such as duplex disrupt energy, bending stiffness and protein deformation. Nevertheless, many sequence motifs exist in the promoter regions besides Pol II motifs, which can also be useful for TSS identification.

In this study, we present a deep learning model, named dlCAGE, an end-to-end tool to identify bona fide TSSs from CAGE data. Currently, various deep learning algorithms have been widely practiced in bioinformatics problems with the outstanding performance [15]. The large-scale genome-wide CAGE data's availability makes it possible to take advantage of the deep learning methods. dlCAGE implements the state-of-the-art deep architecture that enables the optimal performance in transcription initiation pattern characterization and TSS identification. Meanwhile, in addition to existing sequence and structural features used for TSS identification based on CAGE data, dlCAGE also incorporates a high-performance protein-binding sequence motif identification model, DeepBind, to identify de-novo regulatory sequence features to facilitate TSS identification [16]. We trained and tested dlCAGE in human embryonic cell lines as well as an additional K562 cell line. dlCAGE has shown good performance overall comparing to recent studies on CAGE-based TSS identification.

The contributions of this work are summarized as follows:

- Present the first attempt to apply deep learning methods to model and predict true TSSs from CAGE experiments
- Incorporate the state-of-the-art DeepBind model for genome-wide discovery of DNA regulatory

This work has been supported by the National Science Foundation [grants 2015838 and 1661414] and the National Institute of Health [grant R15HGM123407]..

motifs as features for TSS identification from CAGE data.

II. Materials and Method

A. Training and testing data

For the training and testing, we downloaded the CAGE samples corresponding to human embryonic H1 and H9 cell lines from FANTOM project (CNhs14067, CNhs14068, CNhs13964, CNhs11917 and CNhs12824), version GRCh38 [3]. We also obtained the CAGE data in the K562 cell line (CNhs11250), version GRCh37. We additionally obtained the corresponding H3K4me3 ChIP-Seq peak data in H1 (ENCFF467XCU), H9 (ENCFF556EHG) and K562 cell line (ENCFF915MJO) from ENCODE. The Pol II ChIP-Seq peaks in the H1 cell line were downloaded from ENCODE (ENCFF574FQP). Annotated TSSs were obtained from GENCODE (Release 32).

We used data from H1 cell line as the training data. The aligned CAGE tags were first filtered by the mapping quality threshold (set as 10). The tags located within 50bps of each other were then grouped as a single cluster, and peaks with tag per million (TPM) less than 0.5 were ignored. The nucleotide position with the highest CAGE tag counts within each CAGE cluster was defined as the CAGE cluster representative (Ccr). The training data was extracted from the 100bp regions surrounding the Ccrs, called Ccr regions. The positive training data was selected from those Ccr regions that have their Ccrs located within the +-500 surrounding regions of the annotated TSSs, and are overlapped with H3K4me3 and Pol II ChIP-Seq peaks. The negative training data was selected from those Ccr regions that have their Ccrs located outside +-500 regions of the annotated TSSs and does not contain H3K4me3 and Polymerase II peaks. To avoid sample imbalance in the training data, we performed random sampling and obtained 12,500 positive and 12,500 negative samples, respectively. We then split the combined 25k dataset into training and validation set using a 4:1 ratio. The resulted training set contains 20,000 Ccr regions, and the validation set includes 5,000 Ccr regions.

For model testing, we similarly obtained 41,314 and 28,608 negative from H9 cell line sample CNhs11917; 33,787 positive and 17,887 negative for H9 cell line sample CNhs12824. For the K562 cell line (CNhs11250), we compiled 12,706 positive and 179,085 negative samples. We then performed random sampling to obtain 13,000 negative K562 samples as part of the final testing set.

B. Network architecture

dlCAGE was developed based on VGG16, one of the best performed convolutional neural network (CNN) model [17]. The model has achieved 92.7% top-5 test accuracy in ImageNet, a dataset of over 14 million images belonging to 1,000 classes with input data as 224x224x3 images. The original VGG16 architecture consists of 13 convolutional layers, five max pooling layers and three dense layers. Considering the smaller size of our encoded input data comparing with ImageNet, we adjusted the size of VGG16 CNN architecture accordingly to fit our input data for better training and testing efficiency. We sampled multiple parameters and different sizes of the deep neural network and kept the dlCAGE architecture that best optimizes the performance and efficiency (Fig. 1). The resulted dlCAGE contains seven convolutional layers and three max pooling layers. We also performed the optimization of the hyperparameters, such as the size of each convolutional kernel and the dropout rate. As a result, we set 0.4 for dropout rate, 64 filters for the first and second convolutional layers, 128 for the third and fourth convolutional layers, and 256 for the rest of the convolutional layers. The kernel size for all of the convolutional layers is all 3x3. A dropout layer follows after each max pooling layers to avoid overfitting. A fully connected layer with 512 units is attached after the flatten operation and followed by another fully connected layer with 128 units. Between the final output layer with two units, there is another dropout layer. dlCAGE then can generate labels include 0 or 1 to indicate the predicted class of the input data, together with a score that shows the probability. We also incorporated the model checkpoint and the early stopping mechanism during the training. Only the model in certain epochs with the highest validation accuracy during the entire training was saved. The training was stopped if the validation accuracy was not improved over 25 iterations, and the last saved model was considered the converged model. We implemented the dlCAGE using Keras with TensorFlow backend.

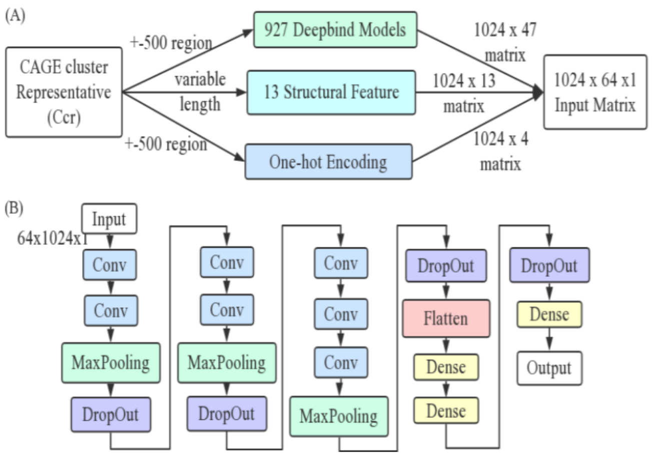

Fig. 1. (A) Feature Encoding Flow Chart; (B) dlCAGE Model Architecture.

C. Control model without DeepBind binding score features

dlCAGE has 927 DeepBind protein binding scores as part of the input. To investigate the DeepBind score's effect on the performance of dlCAGE for CAGE-represented TSS identification, we also created a control model without binding scores as the input features. The architecture of the control model is identical to the dlCAGE. By removing the Binding score features, the control model has a feature input size 17* 1024*1. We use the same 5-fold validation strategy on the same training data. According to the comparison, the dlCAGE with the binding score features achieved better performance overall compared with the control model.

D. Model variation

Besides the VGG16-based dlCAGE approach, we also implemented three additional deep learning models with different architectures for comparison. These include a CNN baseline model, a Long-Term-Short-Term-Memory (LSTM)-based, and a Gated Recurrent Unit (GRU)-based recurrent neural network (RNN) model. (LSTM) and Gated Recurrent Unit (GRU) are two typical RNN models wildly used in sequence prediction problems due to their capacity of learning order dependence [18, 19].

The CNN baseline model contains one convolutional layer with 64 filters followed by a max pooling layer with 2 by 2 pooling size, which is then followed by 64 unit-fully connected layers (FC layer) with ReLu activation, and another 32 units FC layer with the same ReLu activation. At last, a final FC layer with softmax activation. The LSTM-based model consists of 16 units followed by a 0.4 drop out layers and a 2 unit FC layer. The GRU model has an almost identical structure as the above LSTM model, except the LSTM layer was replaced by a GRU layer. Both of the RNN-based models take 64 channels, 1024 time step as input, and generate a single score between zero and one as the prediction probability.

We implemented all the alternative models mentioned above, using Keras with TensorFlow backend. We then trained all three alternative models with the same data and training strategy. We also utilized the same holdout data for performance evaluation and comparison.

E. *Features and feature encoding*

Each Ccr region was encoded using one-hot encoding. We also included 13 DNA Structure features that have been used in literature to identify promoter and TSS regions [12]: bending stiffness, stacking energy, duplex free energy, protein deformation, protein DNA twist, B DNA twist, Z DNA stability, duplex disrupt energy, propeller twist, DNA denaturation, A-philicity, DNA bendability, nucleosome positioning. Each structure feature was encoded as a size-1,024 vector and calculated as described in the recent work [14]. In cases the encoded vector for one structural feature is shorter than the 1,024, we filled the rest of the vector using zeros. Stacking the thirteen encoded vectors together, we have a 1,024x13 structural feature matrix (Fig.1).

In addition to the above sequence and structural features, DNA regulatory motifs play essential roles in gene transcription and are important features for CAGE-detected TSS identification [14, 20-22]. Instead of specifying a fixed number of known motifs, we incorporate the de-novo motif discovery model, DeepBind, into our dlCAGE pipeline. DeepBind is a CNN-based deep learning model for protein-binding sequence motif identification. Trained on a huge number of sequences, DeepBind provides hundreds of pre-trained models that target different protein-binding sequence motifs. The resulted DeepBind score indicates the binding affinity between a given DNA sequence and a specific regulatory protein. We first applied DeepBind models to our training and testing sequences, and then encoded DeepBind binding scores as part of the input features toward TSS identification. In detail, DeepBind provides 927 pre-trained models that can produce the binding scores for each candidate sequence. In our deep learning models, we included all 927 pre-trained DeepBind models as 927 features. For each sequence in our dataset, we used a moving window strategy to obtain a set of DeepBind binding score feature vector, the size of the window is 50 bps, and the step of the window is 25 bps.

III. Result

A. *dlCAGE performance in comparison to alternative deep learning models*

We performed K-fold (K=5) cross-validation for model performance evaluation based on the compiled training data set corresponding to the H1 cell line (see Material and Method, Section A for details). The TSS identification task is a binary classification problem. We thus included five metrics for the performance evaluation: sensitivity, specificity, precision, F1 score, and accuracy.

The above performance metrics were also used for dlCAGE model architecture selection, the performance comparison with the latest work and the case study in the K562 cell line. The performance of dlCAGE is overall constantly above 90%, suggesting dlCAGE is not likely to suffer from overfitting. To understand how DeepBind score as input features affect model performance, we also performed K-fold cross-validation on dlCAGE architecture with DeepBind model removed from the pipeline (control model without DeepBind). The results show that with the same deep learning architecture, the control model without DeepBind has its performance dropped in terms of all the metrics, and it can have its sensitivity and F1 score dropped below 90% for parts of the data. These results suggest the DeepBind score features contribute to true TSS identification from CAGE data noises, and dlCAGE can suffer from performance loss when working without DeepBind binding score features.

The architecture of a deep neural network is essential to its learning and prediction performance. To investigate the influence of different deep learning model architectures on the TSS prediction accuracy in comparison with dlCAGE, we implemented three alternative deep learning models including a CNN baseline, a LSTM-based, and a GRU-based model (Materials and Method section). All the training procedure was performed on the same 20,000 Ccr training sequences and 5,000 testing sequences in H1 cell line. As shown in Table I, dlCAGE outperformed the three alternative models in terms of specificity, precision, F1 score and accuracy. These deep learning models have achieved sensitivity ranging from approximately 89% to

TABLE I. Comparison of different neural network architecture

Architecture	**Metrics**				
	Sensitivity	***Specificity***	***Precision***	***F1 score***	***Accuracy***
GRU-based	0.8972	0.9042	0.9009	0.8991	0.9008
LSTM-based	0.9022	0.8904	0.8888	0.8954	0.8962
CNN baseline	0.9143	0.9302	0.9271	0.9207	0.9224
dlCAGE	0.9180	0.9543	0.9512	0.9343	0.9364

TABLE II. PERFORMANCE COMPARISON WITH ADAPT-CAGE

Tools	Metrics				
	Sensitivity	*Specificity*	*Precision*	*F1 score*	*Accuracy*
dlCAGE -12626-134E7	0.8789	0.8846	0.9166	0.8974	0.8812
dlCAGE -12724-135G6	0.8863	0.8550	0.9203	0.9030	0.8755
ADAPT-CAGE - 12626-134E7	0.3385	0.6688	0.5961	0.4318	0.4736
ADAPT-CAGE - 12724-135G6	0.3199	0.6745	0.6499	0.4288	0.4426
dlCAGE-overall	0.8823	0.8732	0.9183	0.8999	0.8788
ADAPT-CAGE-overall	0.3302	0.6710	0.6184	0.4305	0.4605

92%, specificity ranging from 89% to 95%, F1 score and accuracy from 89% to 93%. In general, CNN-based deep learning models have better performance than LSTM-based and GRU-based RNN models with the current input data encoding mechanism. dlCAGE achieved 95% precision and specificity, which is much higher than all the alternative models, suggesting dlCAGE is less likely to identify non-TSS CAGE signals and have a higher chance of recovering true CAGE TSSs comparing to the alternative models.

B. *Performance comparison with the most recent work*

We also compared dlCAGE with the most recent work on TSS identification from CAGE data, ADAPT-CAGE. ADAPT-CAGE was trained on ten Pol II motif binding information, 13 structure features and a TPM-based CAGE cluster expression score. The method involves Gradient Boosting Machines (GBMs) trained with Pol II motif affinity scores, and SVM to model the structural features. The combined output of the trained GBMs and SVMs was then used as the input of the next GBM for an overall structural feature score. Then the overall structural feature and the CAGE cluster expression scores were combined as the input of the final GBM that will produce a final classification score. Applying a cutoff on the final score, the user can determine the ADAPT-CAGE prediction result. The comparison was performed on two samples of the H9 cell line (12724-135G6 and 12626-134E7), respectively (see Material and method). We run ADAPT-CAGE with the default parameters. The minimum TPM value of accepted CAGE clusters was set as 0.5. The CAGE tags with a mapping quality lower than ten were filtered out, and the maximum required distance between discovered CAGE clusters was set to 50. The maximum size of the cluster was limited to 1 kbp. This setting resulted in a total of 121,596 Ccr regions and TSS predictions (69,922 from the sample 12626-134E7 and 51,674 from 12724-135G6, respectively), including 40,094 positive and 81,502 negative predictions. We applied dlCAGE on the same set of Ccr regions and predicted 72,155 positive and 49,441 negative TSSs. Table II shows the performance of dlCAGE in comparison with ADAPT-CAGE. The results show that dlCAGE has a significant performance increase across all five metrics. For example, dlCAGE achieved a 88% sensitivity in contrast to the 33% sensitivity of ADAPT-CAGE.

Meanwhile, dlCAGE obtained an 87% specificity value while ADAPT-CAGE had its specificity as 67%. This result suggests dlCAGE is more likely to pick up true TSSs and true noises from CAGE data. Similarly, dlCAGE achieved nearly 90% F1 score and 88% accuracy comparing to the 43% F1 score and 46% accuracy of ADAPT-CAGE. Further investigation revealed individual cases dlCAGE had predicted correctly. For example, Ccr regions surrounding chr20: 1185670, chr15: 82680533 and chr8 66870785 were predicted as positive TSSs by dlCAGE, but negative TSSs by ADAPT-CAGE. We found all these Ccrs are located within the +/-500bp surrounding regions of genes Transmembrane Protein 74B, Adaptor Related Protein Complex 3 Subunit Beta 2 and Minichromosome Maintenance Domain Containing 2, respectively. They all have H3K4me3 peaks in their surrounding 200bp regions. Therefore, they are most likely to be true TSSs. On the other hand, Ccr regions surrounding chr1: 214646966; chr2: 38231464 and chr5: 83522984 were predicted as false TSSs by dlCAGE, but true TSSs by ADAPT-CAGE. We checked their genomic locations and found they are not close to any annotated TSSs. In the meantime, there are not H3K4me3 signals in these regions. They are thus most likely to be noises.

C. *Case study(K562 cell line)*

For a more comprehensive comparison, we performed an additional test on the K562 cell line (sample: CNhs11250). We identified in total 25,707 Ccr regions in the K562 sample, from which dlCAGE predicted 7,935 as positive TSSs and 17,772 as negative TSSs. Table III shows the overall performance. We observed that dlCAGE achieved a 75% F1 score and 80% accuracy. In contrast, ADAPT-CAGE obtained a 4% F1 score and 50% accuracy. In fact, out of 7,935 positive TSS predictions made by dlCAGE, 7,770 (98%) contains H3K4me3 and are located within +/-500 bp surrounding regions of annotated TSSs. Only 165 (2%) do not contain H3K4me3 peaks and are located outside the +/- 500 bp region of annotated TSSs, and thus are likely false predictions. In the meantime, out of the 17,772 negative TSSs predicted by dlCAGE, 12,728 do not contain H3K4me3 peaks and are located outside +/-500 bp surrounding regions of annotated TSSs. On the other hand, ADAPT-CAGE predicted 553 positive TSSs and 25,154 as negative TSSs. Only 280 of the 553 positive-predicted Ccr

TABLE III. CASE STUDY IN THE K562 CELL LINE

Tools	Metrics				
	Sensitivity	*Specificity*	*Precision*	*F1-score*	*Accuracy*
dlCAGE	0.6115	0.9873	0.9792	0.7529	0.8016
ADAPT -CAGE	0.0220	0.9790	0.5063	0.0422	0.5060

regions (0.506) are located within the +/-500 bp surrounding regions of annotated TSSs and contain H3K4me3 peaks. We found that there were 273 Ccr regions predicted as positive TSSs by ADAPT-CAGE and negative TSSs by dlCAGE that do not have H3K4me3 peaks and are not located close to annotated TSSs. For example, the Ccr regions surrounding the genomic locations: chr7: 120889267, chr18: 9744585, and chr7, 26224223. In the meantime, we found 7,588 Ccr regions predicted as negative TSSs by ADAPT-CAGE were identified as positive TSSs by dlCAGE. All of these Ccr regions are located within the +/-500bp regions of annotated TSSs and also overlap with H3K4me3 peaks in the K562 cell line. For example, the Ccr regions surrounding genomic locations: chr1: 32797313, chr19: 51226649, and chr12: 95467315, overlap with three annotated TSSs of genes: Histone Deacetylase 1, C-Type Lectin Domain Containing 11A and nuclear receptor subfamily 2 group C member 1, respectively.

IV. Conclusion

In this article, we introduced dlCAGE, a deep learning model, to distinguish true TSSs from transcriptional noises in CAGE data. Identifying TSSs is essential to the discovery of transcription initiation mechanisms and further understanding of gene transcriptional regulation. Deep learning methods have been widely applied in many bioinformatics problems and achieved outstanding performance, but have not been seen in TSS identification from CAGE data. In the meantime, current work on TSS identification from CAGE data is mostly based on a limited number of sequence and structural features. dlCAGE was developed to exploit the current development of deep learning and also take advantage of the genome-wide motif discovery methods for feature identification. With VGG16-based deep learning architecture, dlCAGE integrated scores from 927 DeepBind motif binding models as part of sequence features to identify TSSs in addition to the one-hot encoded sequence surrounding the Ccrs and 13 Structure features. Applying dlCAGE to several human embryonic cell lines and K562, we showed dlCAGE is able to achieve outstanding performance in TSS prediction in comparison with the most recent work. We also shown integrating de-novo protein binding motif models such as DeepBind into the TSS prediction pipelines can improve the overall prediction performance.

V. References

[1] R. Kodzius *et al.*, "CAGE: cap analysis of gene expression," (in eng), *Nat Methods,* vol. 3, no. 3, pp. 211-22, Mar 2006. [Online]. Available: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve&db=PubMed&dopt=Citation&list_uids=16489339

[2] S. Wang, A. Talukder, M. Cha, X. Li, and H. Hu, "Computational annotation of miRNA transcription start sites," *Brief Bioinform,* Jan 31 2020, doi: 10.1093/bib/bbz178.

[3] M. Lizio *et al.*, "Gateways to the FANTOM5 promoter level mammalian expression atlas," *Genome Biol,* vol. 16, p. 22, Jan 5 2015, doi: 10.1186/s13059-014-0560-6.

[4] H. Takahashi, T. Lassmann, M. Murata, and P. Carninci, "5' end-centered expression profiling using cap-analysis gene expression and next-generation sequencing," *Nat Protoc,* vol. 7, no. 3, pp. 542-61, Feb 23 2012, doi: 10.1038/nprot.2012.005.

[5] V. Haberle, A. R. Forrest, Y. Hayashizaki, P. Carninci, and B. Lenhard, "CAGEr: precise TSS data retrieval and high-resolution promoterome mining for integrative analyses," *Nucleic Acids Res,* vol. 43, no. 8, p. e51, Apr 30 2015, doi: 10.1093/nar/gkv054.

[6] C. Barham, M. Cha, X. Li, and H. Hu, "Application of Deep Learning Models to MicroRNA Transcription Start Site Identification," in *IEEE 7th International Conference on Bioinformatics and Computational Biology*, 2019: IEEE, pp. 22-28.

[7] H. Ohmiya *et al.*, "RECLU: a pipeline to discover reproducible transcriptional start sites and their alternative regulation using capped analysis of gene expression (CAGE)," *BMC Genomics,* vol. 15, p. 269, Apr 25 2014, doi: 10.1186/1471-2164-15-269.

[8] M. C. Frith, E. Valen, A. Krogh, Y. Hayashizaki, P. Carninci, and A. Sandelin, "A code for transcription initiation in mammalian genomes," *Genome Res,* vol. 18, no. 1, pp. 1-12, Jan 2008, doi: 10.1101/gr.6831208.

[9] F. Consortium *et al.*, "A promoter-level mammalian expression atlas," *Nature,* vol. 507, no. 7493, pp. 462-70, Mar 27 2014, doi: 10.1038/nature13182.

[10] A. Kanhere and M. Bansal, "Structural properties of promoters: similarities and differences between prokaryotes and eukaryotes," *Nucleic Acids Res,* vol. 33, no. 10, pp. 3165-75, 2005, doi: 10.1093/nar/gki627.

[11] Y. Fukue, N. Sumida, J. Nishikawa, and T. Ohyama, "Core promoter elements of eukaryotic genes have a highly distinctive mechanical property," *Nucleic Acids Res,* vol. 32, no. 19, pp. 5834-40, 2004, doi: 10.1093/nar/gkh905.

[12] Y. Gan, J. Guan, and S. Zhou, "A comparison study on feature selection of DNA structural properties for promoter prediction," *BMC Bioinformatics,* vol. 13, p. 4, Jan 7 2012, doi: 10.1186/1471-2105-13-4.

[13] T. Abeel, Y. Saeys, E. Bonnet, P. Rouze, and Y. Van de Peer, "Generic eukaryotic core promoter prediction using structural features of DNA," *Genome Res,* vol. 18, no. 2, pp. 310-23, Feb 2008, doi: 10.1101/gr.6991408.

[14] G. K. Georgakilas, N. Perdikopanis, and A. Hatzigeorgiou, "Solving the transcription start site identification problem with ADAPT-CAGE: a Machine Learning algorithm for the analysis of CAGE data," *Sci Rep,* vol. 10, no. 1, p. 877, Jan 21 2020, doi: 10.1038/s41598-020-57811-3.

[15] A. Talukder, C. Barham, X. Li, and X. Hu, "Interpretation of Deep Learning in Genomics and Epigenomics," *Briefings in Bioinformatics,* 2020.

[16] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning," *Nat Biotechnol,* vol. 33, no. 8, pp. 831-8, Aug 2015, doi: 10.1038/nbt.3300.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[18] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 2014.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory " *Neural Computation,* vol. 9, no. 8, pp. 1735–1780, 1997.

[20] Y. Wang, S. Goodison, X. Li, and H. Hu, "Prognostic cancer gene signatures share common regulatory motifs," *Sci Rep,* vol. 7, no. 1, p. 4750, Jul 6 2017, doi: 10.1038/s41598-017-05035-3.

[21] J. Ding, V. Dhillon, X. Li, and H. Hu, "Systematic discovery of cofactor motifs from ChIP-seq data by SIOMICS," *Methods,* vol. 79-80, pp. 47-51, Jun 2015, doi: 10.1016/j.ymeth.2014.08.006.

[22] S. M. Ruppert, M. Chehtane, G. Zhang, H. Hu, X. Li, and A. R. Khaled, "JunD/AP-1-mediated gene expression promotes lymphocyte growth dependent on interleukin-7 signal transduction," *PLoS One,* vol. 7, no. 2, p. e32262, 2012, doi: 10.1371/journal.pone.0032262.