Towards Learning Regulatory Elements of Promoter Sequences with Deep Learning

A thesis presented
by

Nicasia Beebe-Wang

To
Computer Science
in partial fulfillment of the MBB track requirements
for the degree of
Bachelor of Arts

Harvard College
Cambridge, Massachusetts

March 31, 2017

ABSTRACT

Promoters play a key role in gene regulation. Although progress has been made to understand the elements which make up a promoter, the identification of all of the regulatory elements which comprise the promoter remains challenging due to the high variability of promoter sequences. In this thesis, I aim to identify regulatory elements in promoter regions using deep learning. Specifically, I employ a convolutional neural network (CNN) to predict whether a given genomic sequence contains a promoter versus several null models, i.e. background sequences. I compare the performance of the CNN model for each null model and perform saliency analysis to visualize what the network has learned. The main result I found is that the null model must be carefully selected to avoid learning confounding signals such as nucleotide biases. I found that a dinucleotide shuffle of transcription start sites was able to find known regulatory elements associated with bi-directional promoters.

Author: Nicasia Beebe-Wang
Thesis Advisor: Sean Eddy

ACKNOWLEDGMENTS

I would like to thank my advisor, Sean Eddy, for his guidance and encouragement throughout the process of completing this project and throughout my time with his lab. I am indebted to Peter Koo for his advice and guidance in every aspect of this project. His continual mentorship and support throughout and beyond this project have been invaluable. I would also like to thank Praveen Anand and Tom Jones for their thoughtful comments and suggestions. Finally, I would like to thank my family, friends, and members of the Eddy Lab, for their encouragement throughout the course of this thesis.

INTRODUCTION

Motivation

Promoters, regulatory sequences within genomes, are crucial for gene regulation. Their primary role is to serve as landing pads for specific transcription factors that bind to the DNA upstream of a gene and form a transcription initiation complex to "promote" the recruitment of RNA polymerase. (Lenhard, Sandelin, & Carninci, 2012) Promoters are highly variable, perhaps to differentially regulate various genes across different cell-types; however, they must simultaneously be specific enough for the proper transcription binding factors to identify them (Sandelin et al., 2007).

Due to their importance in gene regulation, it is unsurprising that defects in promoter activity may have large consequences in humans. The root causes of a variety of diseases, including various cancers, can be linked to the dysregulation of the non-coding, regulatory regions of the genome (Maston, Evans, & Green, 2006). In some cases, the disorder is caused by a direct defect in a promoter element for a specific gene (Maston, Evans, & Green, 2006). For example, Bernard-Soulier syndrome is thought to be caused by mutations in the promoter region of *GpIb*β leading to altered expression of the gene (Ludlow et al., 1996). Similarly, researchers have identified that in thalassemia, a heritable blood disorder, a single nucleotide polymorphism creates a promoter-like region that interferes with normal activation of downstream genes (Gobbi et al., 2006). For a comprehensive overview of diseases thought to be caused by specific defects to promoter regions, see Table 1 of (Maston, Evans, & Green 2006).

Using a reporter gene assay system, Buckland et al. (2005) found that of sequence variants that had large impacts on gene expression, there was a strong bias toward location in the transcription start site. However, 33% of functional variants in the transcription start-site regions

are not associated with known binding sites (Buckland et al., 2005), indicating that there may be many unknown functional elements in the promoter region that remain undiscovered. Although reporter-gene assays are one of the most accurate experimental methods available for *verifying* functionality of suspected promoters, several challenges limit their effectiveness in large-scale efforts to identify promoters. Firstly, regulatory elements for a given gene may be widely dispersed and thus it may be difficult to properly encapsulate them in the experiment. Additionally, regulatory elements may have different functions across different cell types. Moreover, a promoter region might operate differently in the experimental cell culture system than in its natural environment.

Another method that has been developed which is more amenable to the genome-wide discovery of promoter regions is cap analysis gene expression (CAGE). This technique captures and isolates the 5' end of RNA sequences (originating at the start of transcription). These short, capped-RNA sequences are then sequenced and mapped onto a reference genome. The frequency of the mapped reads to different genomic coordinates yields a landscape corresponding to active transcription start sites (Shiraki et al., 2003). Thus, for a given cell-type, we can locate the promoter region of each active gene as the upstream region of these transcription start sites.

With the influx of data from high-throughput methods such as CAGE, there has been an effort to apply computational methods to learn how to perform promoter recognition. Early methods tended to rely on known promoter regions and features in order to identify elements with similar motifs. One such method used discriminant functions to learn features, such as CpG islands combined with known promoter elements. By combining the results of these feature detectors into a decision tree, the model was able to perform with relatively high accuracy of 86% (Davuluri, Grosse, & Zhang, 2001). However, such a method requires prior knowledge of

known promoter regions and their regulatory motifs. A challenge faced by computational biologists is the problem of deciding which features to extract in order to most successfully recognize promoters (Zeng, Zhu, & Yan, 2009).

One way to circumvent this challenge is to use machine learning in such a way that the model learns differences between promoter regions and background regions without the help of human feature selectors. One relatively successful approach used a support vector machine using k-mer frequencies as the features (Gangal & Sharma, 2005). This method does not require expert knowledge of exact motifs and patterns, instead depending only on the k-mer frequencies from the DNA sequence itself. Such methods that learn regulatory motifs from the data itself may be useful because, if successful, the model can be used to explore how other mutations in genomic sequences affect the function of these elements.

A Deep Learning Approach

Recently, deep learning has been gaining traction in computational biology. Deep learning methods have demonstrated great potential to learn complex functions by repeatedly applying simple, non-linear transformations to the data. Importantly, features learned at each level are learned directly from the data, rather than carefully selected by domain experts (LeCun, Bengio, & Hinton, 2015). One neural network architecture that has been particularly successful in genomics is convolutional neural networks (CNNs). CNNs have the advantage of being able to capture stationary features within multidimensional data that have high spatial variability; subsequently, they can then combine those features into more complex patterns in higher layers to learn hierarchical feature representations (LeCun, Bengio, & Hinton, 2015). In computer vision, CNNs have been applied to image recognition problems with high levels of success. At

each convolutional layer, filters learned increasingly complex features (e.g., edges in early layers, and then combinations of these edges to form more complex features in the following layers) (Krizhevsky, Sutskever, & Hinton, 2012).

In genomics, DeepBind used a CNN to predict experimentally determined specificities of DNA- and RNA- binding proteins based on new raw sequences (Alipanahi et al., 2015). In addition, DeepSEA, predicted the functional effects, i.e. transcription factor binding, chromatin accessibility, and histone marks, of changes to single nucleotides of non-coding sequences (Zhou & Troyanskaya, 2015). Similarly, Basset employed a CNN to learn chromatin accessibility for genomic sites and predicted differences in accessibility as a result of variation in the sequences (Kelley et al., 2016). In each case, they demonstrated superior performance compared to other support vector machine-based methods.

Leveraging the progress that has been made with deep learning in genomics, I wanted to explore whether I could better understand the key cell-type specific regulatory elements that comprise the promoter region. Thus, I created a CNN model to learn to predict genomic sequences upstream of transcription start sites. In order to learn the important regulatory elements, the CNN must also see examples of background sequences which do not contain the regulatory elements associated with the promoter. In this manuscript, I try various background sequences, i.e. null models, and compare their performances. I then employ saliency analysis to visualize what regulatory elements the CNN model has learned. Interestingly, I found that what sequences I used in the null model significantly altered the feature representations that the deep learning model learned. A CNN trained using short background sequences generated from a dinucleotide shuffle of a corresponding promoter sequence was able to learn known regulatory elements associated with bi-directional promoters. In contrast, a CNN trained using random

shuffle of the same sequences taken over a wider sequence around the promoter sequence learned random genomic signals that were not comprehensible.

METHODS

Data Sources and Processing

To locate cell-type specific promoter sequences, I obtained CAGE data from the ENCODE project (for exact data sources, see Table 1; ENCODE Project Consortium, 2012). For each cell-type, there were multiple experiments that each had multiple files. For each cell-type, we therefore merged the genomic coordinates of the CAGE peaks from the same experiment and then intersected the resulting CAGE peak coordinates for each repeat experiment for a given cell type, keeping all TSS sequences that appeared in at least one experiment. Since a typical read is usually around 20 nucleotides long, a read may map to multiple locations in the genome, especially due to the presence of repeating elements around regulatory regions (de Hoon & Hayashizaki, 2008). To that end, we checked for each purported TSS if its specified region was near a known gene, according to the hg19 annotation. For our analyses, we allowed for a gap of up to 50 nucleotides or an overlap of up to 400 nucleotides between our TSS and a coding region. The TSS data that appear near the start of protein-coding regions constitute our class of a positive promoter region. All TSS that did not fit the above criteria were excluded from our analyses.

If a given TSS coordinate overlapped with a gene, we took the end of the given TSS region (embedded in the gene) as the boundary to our positive sequences. However, if there was a gap, we specified the boundary of our positive sequence as the beginning of the annotated gene (to ensure that we did not miss any of the true transcription start site in our sequence). Then,

given our positive sequence boundary, we extracted either 600 or 250 upstream nucleotides starting from the boundary. The 600 nucleotide sequences comprise the "wide" dataset and the 250 nucleotide sequences comprise the "narrow" dataset. Thus, each of the given TSS regions by CAGE were clipped or augmented to an appropriate size for our "wide" and "narrow" datasets. Because CAGE identifies transcription start sites specific to cell-types, I generated positive promoter sequences for six cell-types: A549, HeLa-S3, K563, GM12878, HepG2, MCF-7 (see Table 1 for dataset sizes and sources).

For a null model, I generated 4 different kinds of background sequences: random shuffle, dinucleotide shuffle, DNase sites, and profile sampling. For random shuffle, we took a positive promoter sequence and randomly shuffled the position of the nucleotides, thus allowing for the same nucleotide composition, but shuffling any motif which may have been present. For the dinucleotide shuffle, we used the MEME suite to generate a dinucleotide shuffle, which shuffles the sequence maintaining the same dinucleotide composition. For DNase sites, we chose regulatory regions that had DNase peaks from the same cell type that did not overlap with the CAGE peaks or a gene (for exact sources, see Table 1; ENCODE Project Consortium, 2012). These regions contain regulatory elements that presumably do not allow for active transcription, and hence should not contain promoter codes, but should contain other cell-type specific regulatory elements. For profile sampling, I created a multiple sequence alignment of the positive promoter sequences and constructed a frequency matrix of each nucleotide. I then sampled from this distribution to generate new sequences, so that it could have the same nucleotide content in a position sensitive manner.

We generated or sampled an equal number of background sequences as there were positive sequences separately for each cell-type to create our final narrow (250 nt) and wide (600

nt) datasets for each cell-type: A549 (15622 total sequences), HeLa-S3 (15390 total sequences), K562 (14552 total sequences), GM12878 (15000 total sequences), HepG2 (15922 total sequences), MCF-7 (16056 total sequences).

Our test set consisted of all sequences taken from chromosomes 8, 9, and 10 (12-14% of the data, depending on cell-type). Of the remaining data, 10% was randomly separated into a validation set (8-9% of the original dataset), and the remaining sequences (77-79%) were our training data.

Model Implementation

Our model takes as input genomic sequences (250 nucleotides or 600 nucleotides) encoded in a one-hot representation. Our model consists of two 1D convolutional layers followed by an output layer with one output neuron. The first layer contains 16 filters of length 24 (i.e., dimensions 4 by 24), with dropout at probability 0.2, and padding to maintain original size). Next, a rectified linear unit (ReLU) activation function is applied [$f(x) = \max(0, x)$], changing all negative values from the convolution to 0. Using max pooling with pool size 50, the matrix is reduced such that the maximum values for each row (A, C, T, G) in chunks of 50 columns become a single column. Thus, when our model's input has length 250 or 600, our resulting input to the second convolutional layer will be reduced to 5 or 12 columns, respectively. The next convolutional layer is of size (input length)/50 (i.e., 5 for input lengths of 250, and 12 for input lengths of 600). Thus, for this convolution, the 32 filters match the size of the inputs from the previous layer, and no padding is added to the resulting convolution, resulting in a matrix of length 1. Dropout is increased to 0.5 at this layer to prevent overfitting. Again, a ReLU activation is applied. The result is then fed into a final dense layer that uses a sigmoid function to predict the final output.

Training Details

I built the CNN model in tensorflow with custom-written code in python. With the output neuron activation given as a sigmoid, we use a binary cross-entropy cost function for binary classification of whether the sequence is a promoter or not. I trained the model using stochastic gradient descent updates with Nesterov momentum with a learning rate of 0.005 and momentum of 0.95 for 50 epochs. During each iteration, the loss was calculated for the cross-validation set. The model parameters with the lowest cross-validation dataset were used for the test set. All results in Table 2 are from the sequences in the test set only. The model was trained on a NVIDIA GTX 980 graphical processing unit.

Saliency Analysis

To visualize the network, I employed guided backpropagation, developed by (Springenberg et al., 2015). In this guided backpropagation approach, I take the gradient of the output neuron prior to sigmoid activation with respect to the input sequence, thereby getting the sensitivity of each nucleotide variant on the given prediction. In contrast to a typical backpropagation, guided backpropagation only allows positive gradients to flow backwards. To make the resultant saliency map interpretable, I rectify and normalize the saliency map at each nucleotide position, and convert the values into bits. All positive saliency maps are represented as sequence logos using custom code.

RESULTS AND DISCUSSION

DNase Background Sequences

To train my CNN model to predict promoter sequences from other genomic regions, I chose to use DNase hypersensitive sites unrelated to transcription start sites as background sequences. I built this set of background data by starting with our list of DNase hypersensitive sites (ENCODE Project Consortium, 2012) and removing all sites within 600 nucleotides of either one of our positive promoter sequences or a protein-coding region according to the hg19 genome annotation. For each cell-type, the number of remaining DNase hypersensitive background sequences was much larger than our TSS datasets, so I randomly sampled from the hypersensitive background regions to create training sets with equal representation.

Our model performed well across all cell types using the DNase background sequences (Table 2). Interestingly, the performance of the network did not increase when the length of sequences was increased from 250 to 600 nucleotides (Table 2). Since the performance was outstanding, I suspected that the CNN model had easily learned the regulatory elements of the promoter, requiring only the 250 nucleotides closest to the start of transcription. However, upon performing saliency analysis on sequences that yielded the highest predictions, the network seems to have learned random GC content for both the 250nt sequences and 600nt sequences with a bias on the right side (where the corresponding gene is located) for all positive sequences (Figure 1a). Moreover, the saliency plot for the 600nt sequences (Figure 1b) reveals that the network seems to rely much less on the left half of the sequence when making decisions (essentially looking at only the data contained in the 250 nucleotide dataset).

After further inspection, we found that among all training data across cell-types, promoter regions have a higher concentration of G's and C's than our DNase hypersensitive backgrounds.

Whereas our background sequences are made up of 46% and 45% C/G for 250NT and 600NT sequences, respectively, our TSS sequences are 66% and 60% G/C, respectively. Not surprisingly, 5' untranslated regions are enriched for GC content. Thus, the CAGE-based coordinates seem to systematically overshoot the transcription start site in the 5' UTR region of the gene. Moreover, since the network is optimized to reduce classification error, it's likely that all the information needed to minimize error is contained in the second half of the 600 nucleotide sequence, as that is where we find a large discrepancy in G/C content between TSS sequences and our DNase hypersensitive background. In essence, the choice of a DNase background that contains a different distribution of GC content leads to trivially high accuracy that could likely be achieved by a much simpler model that simply calculates frequencies of nucleotides.

Profile Sampling Background Sequences

To disentangle the conflated signals of the GC bias with the regulatory elements within the promoter sequences, we devised a method to generate background sequences by sampling from a probability distribution that has the same spatial nucleotide content, a method we call "profile sampling." This method uses the training set to calculate a distribution over the four nucleotides at each position. Background sequences are then generated via sampling from the distribution of each index (i.e., the probability of assigning the nucleotide at position k to nucleotide "N" is equal to the proportion of the training data that has nucleotide "N" at position k). These backgrounds sequences should therefore have similar G/C and A/T biases as our positive promoter sequences in the training data.

Surprisingly, the performance of the CNN model with profile sampling background sequences improved, almost yielding perfect predictions (Table 2). Once again, however, the

saliency analysis revealed that the network was learning runs of nucleotides as motifs (Figure 2). It is my suspicion that the nucleotide runs are really an artifact of learning features from the samples generated from the profile samples in the negative class and not the positive promoter class. Since, for each cell-type, I generated thousands of samples from the same probability distributions (7276, 8028), the sequences generated from the distribution of our TSS data were likely quite similar to each other. Therefore, it could be the case that in order for the CNN to classify TSS versus background, the network needed only to learn some average sequences from the distribution of TSS as filters. Then, it could simply classify TSS sequences as those that showed large deviations from the expected sequences generated by sampling from the distribution (whereas background sequences were those that matched the distribution well). In this case, the motifs shown in (Figure 2) would indicate sequences that appear more commonly in true TSS cases than are seen in sequences sampled from their combined distribution. In the future, I would like to attempt to modify this profile sampling approach to increase variation in the background sequences – if my suspicions are correct, this may help pressure the network to learn motifs of the positive TSS class rather than the synthesized background.

Random Shuffle Background Sequences

My next approach was designed to prevent the CNN from learning only from G/C frequency differences between our TSS regions and a background region. To that end, we maintained exactly the same proportions of nucleotides in our background sequences as in our TSS sequences by generating data from our positive hits rather than sampling regions from elsewhere in the genome. The most straightforward approach was simply to shuffle the nucleotides

randomly for each TSS sequence such that the frequency of each nucleotide is preserved from the TSS sequence to the corresponding shuffled background sequence.

The performance of the random shuffle was very accurate for the 250nt sequence and improved to near perfect for the 600nt sequences (Table 2). However, saliency analysis reveals that the network continues to exploit the GC bias on the right side of the sequences, especially in the 600nt case (Figure 3). It seems to be the case that in most of our positive sequences, there is a high presence of GC content towards the right side of the sequence. Thus, although the GC content is preserved in a random shuffle through the entire length of the sequence, the shuffling of nucleotides changes the distribution of those CG regions from right-biased in the positive promoter sequences to evenly distributed in the background sequences. Thus, the network seems to have learned more GC motifs on the right half of the sequences and more AT motifs on the left side of the sequence. This pattern is more evident for 600nt sequences.

Dinucleotide Shuffle Background Sequences

Finally, our last method to find regulatory elements was to try dinucleotide shuffled sequences as background samples. This method is similar to the previous random shuffle approach but adds an additional stipulation that the shuffling of nucleotides must maintain not only the same number of nucleotides but also the same frequencies of k-mers of size 2 (Bailey et al., 2009).

Surprisingly, the performance of the CNN model using dinucleotide background samples was the worst method when compared to the other background methods, especially for the 200nt sequences. Strikingly, when performing saliency analysis on the 250nt sequences, we found that the network seems to have learned various sequence motifs that resemble regulatory elements known to be associated with the promoter.

The most commonly found known motif in our saliency analysis, which often appeared several times within a single sequence (Figure 4a), was a "GC box" which binds transcription factor Sp1. Sp1 is a well-known protein that activates transcription of many genes in animal cells (Kadonga et al., 1986). We also found instances of motifs that resembled binding sites of GA-binding protein (GABP), which is a transcription factor that is thought to be a key regulator of genes involved in the cell cycle, protein synthesis, and cellular metabolism (Rosmarin et al., 2004). Interestingly, the learned motifs seem to be quite similar across cell types even though the network was trained separately from random initializations for each cell-type dataset.

In some instances, we are able to show that the network has learned a sequence motif that resembles a binding site of a protein dimer. This is shown as a CGGA motif flanked by a TCCG motif as shown by box 3 in Figure 4a. Upon a search in known literature, we were not able to find a matching motif.

Interestingly, the network did not learn motifs of TATA boxes, which are known to be associated with promoters. Rather, the network has learned more GC boxes, which have been associated with bi-directional transcription. Researchers have found that bi-directionally transcribed promoters have a significantly lower presence of TATA boxes, and a higher proportion of CG boxes than uni-directional promoters (Trinklein et al., 2004). This suggests that many of our sequences may contain bi-directional promoters, or our data may otherwise be somewhat biased towards GC containing regions.

Interestingly, the saliency analysis shows that the performance for the 600nt sequences was more unreliable with some cell types revealing the same GC bias as the other background sequences (Figure 4b). It could be the case that as the length of the sequences increases, the discrepancy between the location of CG for shuffled and unshuffled sequences becomes

apparent; thus the network is able to rely on GC content to classify sequences, rather than the more specific promoter motifs.

CONCLUSIONS

Our assessments of the four background sequences described above relied on both accuracy and saliency analyses. Based on these separate metrics, we found that good performance does not imply useful results. Although we had high accuracy rates for our DNase background and randomly shuffled sequences, saliency analysis revealed that the networks behaved as detectors of CG bias, recognizing differences in frequency and location, respectively. Similarly, when we attempted to sample from the distribution of our positive promoter sequences, we found that the network classified test sequences with motifs that were not interpretable (possibly because the network was able to easily classify background sequences due to their similarity to each other). In contrast, when our CNNs were trained with our dinucleotide shuffled background sequences, the test performance of the network was worse than the other background methods; however, the network seems to have learned known promoter sequences.

These results suggest that neural networks should not be treated as black boxes where methods that produce a better prediction accuracy are considered the best method. Instead, it's important to consider what the network has learned, and whether those results are meaningful as they relate to the problem. CNNs are trained to find features that best discriminate between classes. Thus, if a true signal is conflated with some other confounding factor (such as GC content) and that confounding factor provides more discriminative power, then the network must only learn to classify based on the confounding signal. Thus, in order to produce a network that performs reliable inference based on a specific type of signal, the method of selecting

background data must be carefully chosen to disentangle confounding factors. In the case of this study, the goal was to learn regulatory sequences as they relate to promoter regions, and using a dinucleotide shuffled background (with sequences of length 250nt) turned out to be the method that most successfully performed that task, despite having the lowest accuracy rates.

When a suitable background was selected, our CNN was able to learn some known regulatory motifs, especially those that correspond to Sp1 and GABP. Interestingly, no TATA-box motifs were identified. This is consistent with a genome-wide analysis of transcription start sites (found using CAGE) that found that transcription start sites with a classical TATA-box promoter are a minority in mammals, and that the majority of transcription start sites found with CAGE contain many CpG islands and no TATA-box (Carninci et al., 2006). Additionally, we found a CGGA motif flanked by a TCCG motif that appeared multiple times across different cell-types, which to our knowledge does not correspond to any known binding proteins. This motif may be of interest in future studies.

Results of this study show that the choice of an appropriate background sequence is crucial for the identification of promoters within transcription start sites. Although using a dinucleotide shuffle allowed the CNN to learn some known promoters, further efforts should be made to improve generation of background sequences, especially since this method fails for longer sequences. One clear possibility is to improve the profile sampling method such that the background class is as variable as the promoter containing sequences. Given improvements in background sequence selection, CNNs may prove to be useful in identifying important promoter elements within genomes.

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Zheng, X. (2015). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. Software available from tensorflow.org.

Alipanahi, B., Delong, A., Weirauch, M. T., & Frey, B. J. (2015). Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning. *Nature Biotechnology*, *33*(8), 831-838.

Bailey, T. L., Boden, M., Buske, F. A., Frith, M., Grant, C. E., Clementi, L., ... & Noble, W. S. (2009). MEME SUITE: Tools for motif discovery and searching. *Nucleic Acids Research*, 37:W202-W208

Buckland, P. R., Hoogendoorn, B., Coleman, S. L., Guy, C. A., Smith, S. K., & O'donovan, M. C. (2005). Strong bias in the location of functional promoter polymorphisms. *Human Mutation*, *26*(3), 214-223.

Carninci, P., Sandelin, A., Lenhard, B., Katayama, S., Shimokawa, K., Ponjavic, J., ... & Hayashizaki, Y. (2006). Genome-wide analysis of mammalian promoter architecture and evolution. *Nature Genetics*, *38*(6), 626-635.

Davuluri, R. V., Grosse, I., & Zhang, M. Q. (2001). Computational identification of promoters and first exons in the human genome. *Nature Genetics*, *29*(4), 412-417.

de Hoon, M., & Hayashizaki, Y. (2008). Deep cap analysis gene expression (CAGE): Genome-wide identification of promoters, quantification of their expression, and network inference. *Biotechniques*, *44*(5), 627.

ENCODE Project Consortium. (2012). An integrated encyclopedia of DNA elements in the human genome. *Nature*, *489*(7414), 57-74.

Gangal, R., & Sharma, P. (2005). Human pol II promoter prediction: time series descriptors and machine learning. *Nucleic acids research*, *33*(4), 1332-1336.

De Gobbi, M., Viprakasit, V., Hughes, J. R., Fisher, C., Buckle, V. J., Ayyub, H., ... & Higgs, D. R. (2006). A regulatory SNP causes a human genetic disease by creating a new transcriptional promoter. *Science*, *312*(5777), 1215-1217.

Kadonaga, J. T., Carner, K. R., Masiarz, F. R., & Tjian, R. (1987). Isolation of cDNA encoding transcription factor Sp1 and functional analysis of the DNA binding domain. *Cell*, *51*(6), 1079-1090.

Kelley, D. R., Snoek, J., & Rinn, J. L. (2016). Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks. *Genome Research*, *26*(7), 990-999.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems,*1097-1105.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436-444.
Lenhard, B., Sandelin, A., & Carninci, P. (2012). Metazoan promoters: Emerging characteristics and insights into transcriptional regulation. *Nature Reviews Genetics*, *13*(4), 233-245.

Ludlow, L. B., Schick, B. P., Budarf, M. L., Driscoll, D. A., Zackai, E. H., Cohen, A., & Konkle, B. A. (1996). Identification of a mutation in a GATA binding site of the platelet glycoprotein Ibβ promoter resulting in the Bernard-Soulier syndrome. *Journal of Biological Chemistry*, *271*(36), 22076-22080.

Maston, G. A., Evans, S. K., & Green, M. R. (2006). Transcriptional regulatory elements in the human genome. *Annual Review of Genomics and Human Genetics*, *7*, 29-59.

Rosmarin, A. G., Resendes, K. K., Yang, Z., McMillan, J. N., & Fleming, S. L. (2004). GA-binding protein transcription factor: a review of GABP as an integrator of intracellular signaling and protein–protein interactions. *Blood Cells, Molecules, and Diseases*, *32*(1), 143-154.

Sandelin, A., Carninci, P., Lenhard, B., Ponjavic, J., Hayashizaki, Y., & Hume, D. A. (2007). Mammalian RNA polymerase II core promoters: Insights from genome-wide studies. *Nature Reviews Genetics*, *8*(6), 424-436.

Shiraki, T., Kondo, S., Katayama, S., Waki, K., Kasukawa, T., Kawaji, H., ... & Hayashizaki, Y. (2003). Cap analysis gene expression for high-throughput analysis of transcriptional starting point and identification of promoter usage. *Proceedings of the National Academy of Sciences*, *100*(26), 15776-15781.

Springenberg, J. T., Dosovitskiy, A., Brox, T., & Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv:1412.6806*.

Zeng, J., Zhu, S., & Yan, H. (2009). Towards accurate human promoter recognition: a review of currently used sequence features and classification methods. *Briefings in Bioinformatics*, *10*(5), 498-508.

Zhou, J., & Troyanskaya, O. G. (2015). Predicting effects of noncoding variants with deep learning-based sequence model. *Nature Methods*, *12*(10), 931-934.

TABLES

Table 1. The number of promoter contain sequences that meet our criteria for each cell-type, given along with the experimental ENCODE accession numbers used to access the data. Note that an equal number of background sequences as positive sequences are used in training and testing, so for each cell type, the number of sequences that are split into training, validation, and test sets is double the number of promoter containing sequences.

Cell-type	# of processed promoter containing regions	CAGE source (ENCODE GEO accession numbers)	DNase peaks source (ENCODE GEO accession numbers)
A549	7811	ENCFF953KFV, ENCFF569VNO, ENCFF335MBE, ENCFF672BLL, ENCFF887MUK, ENCFF203UCM	GSM736580
GM12878	7500	ENCFF945DZC, ENCFF140PCA, ENCFF335DEQ, ENCFF601LZG, ENCFF792GOR, ENCFF358CEV	GSM736620
HeLa-S3	7695	ENCFF255XNM, ENCFF722BNB, ENCFF315EKL, ENCFF966XAJ, ENCFF240UGX, ENCFF924EBE	GSE90432
HepG2	7961	ENCFF556KQM, ENCFF597GDN, ENCFF769REJ, ENCFF809XAS, ENCFF991ASD, ENCFF892NYI	GSE90300
562	7276	ENCFF061ZFD, ENCFF268LKW, ENCFF744UZC, ENCFF095ZKV, ENCFF983UJK, ENCFF172WEH	GSM736629
MCF-7	8028	ENCFF748CKE, ENCFF322KJE, ENCFF144UNV, ENCFF185OIG, ENCFF674KUW, ENCFF677JCJ	GSM1024767

Table 2. Average test set accuracy, area under the ROC curve, and area under the precision recall curve for the trained CNN on (a) 200nt sequences, and (b) 600nt sequences. Each value is shown as an average ± standard deviation over the 6 cell-types.

Table 2a. Average CNN Test Set Performance for 200nt Sequences

Background type	Accuracy	Area under ROC Curve	Area Under PR curve
DNase Background	0.94 ± 0.007	0.975 ± 0.004	0.975 ± 0.005
Profile Sampling	0.953 ± 0.004	0.991 ± 0.001	0.991 ± 0.001
Random Shuffle	0.932 ± 0.007	0.982 ± 0.002	0.982 ± 0.002
Dinucleotide Shuffle	0.873 ± 0.007	0.945 ± 0.005	0.945 ± 0.005

Table 2b. Average CNN Test Set Performance for 600nt Sequences

Background type	Accuracy	Area under ROC Curve	Area Under PR curve
DNase Background	0.946 ± 0.007	0.98 ± 0.004	0.98 ± 0.004
Profile Sampling	0.994 ± 0.002	1.0 ± 0.0	1.0 ± 0.0
Random Shuffle	0.985 ± 0.002	0.999 ± 0.0	0.999 ± 0.0
Dinucleotide Shuffle	0.945 ± 0.004	0.988 ± 0.002	0.988 ± 0.002

FIGURES

Figure 1. DNase Background Saliency

Saliency plot of positive test sequences that were correctly classified (with a probability >0.99) by a CNN trained with DNase background sequences. Plots are displayed for (a) a 250nt sequence from cell-type GM12878, (b) a 600nt sequence from cell-type GM12878. In the top row, motifs that contributed more weight to the class assignment appear larger, and the bottom row represents the true sequence from the test set. Note that for 1b, the sequence has been split in half so that the top segment shows the first 300nts and the bottom segment displays the last 300nts.

Figure 1a

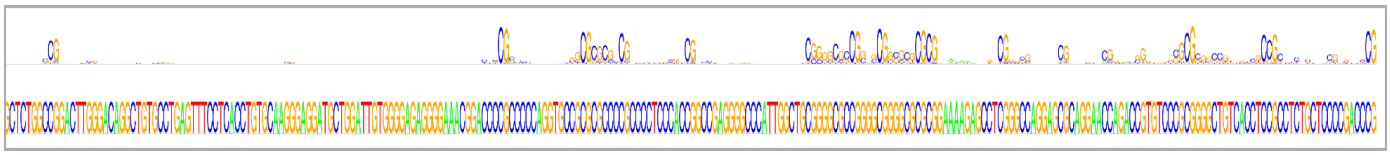

Figure 1b

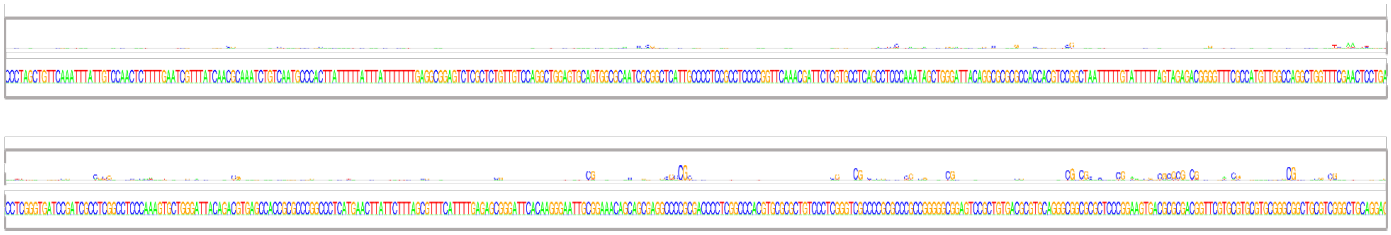

Figure 2. Profile Sampling Saliency

Saliency plot of positive test sequences that were correctly classified (with a probability >0.99) by a CNN trained with background sequences created with profile sampling. Plots are displayed for (a) a 250nt sequence from cell-type A549, (b) a 600nt sequence from cell-type A549. In the top row, motifs that contributed more weight to the class assignment appear larger, and the bottom row represents the true sequence from the test set. Note that for 2b, the sequence has been split in half so that the top segment shows the first 300nts and the bottom segment displays the last 300nts.

Figure 2a.

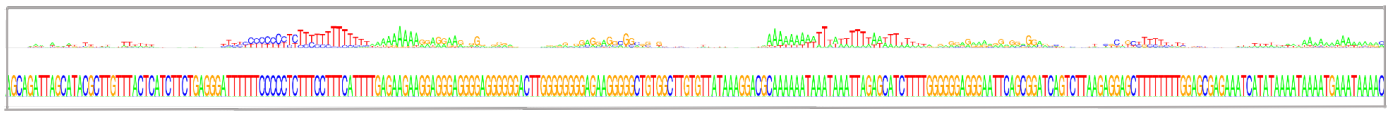

Figure 2b.

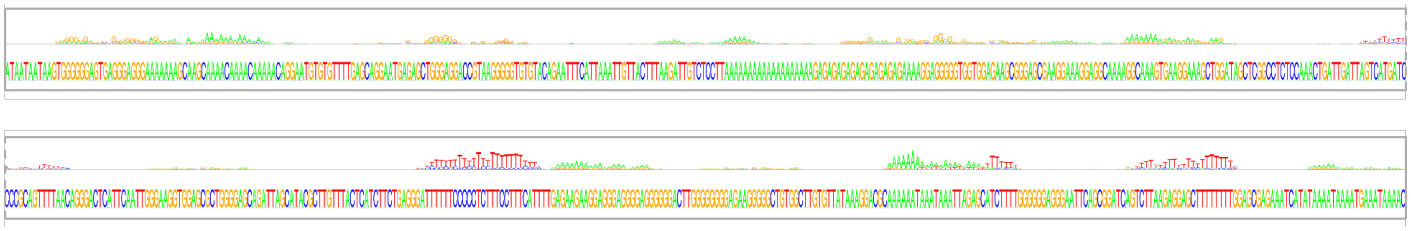

Figure 3. Random Shuffle Saliency

Saliency plot of positive test sequences that were correctly classified (with confidence >0.99) by a CNN trained with background sequences that were created by randomly shuffling positive sequences. Plots are displayed for (a) a 250nt sequence from cell-type HeLa-S3, (b) a 600nt sequence from cell-type HeLa-S3. In the top row, motifs that contributed more weight to the class assignment appear larger, and the bottom row represents the true sequence from the test set. Note that for 3b, the sequence has been split in half so that the top segment shows the first 300nts and the bottom segment displays the last 300nts.

Figure 3a.

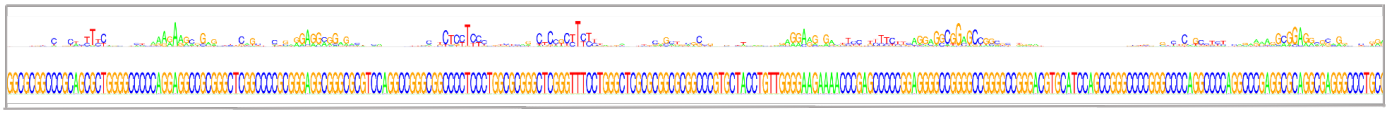

Figure 3b.

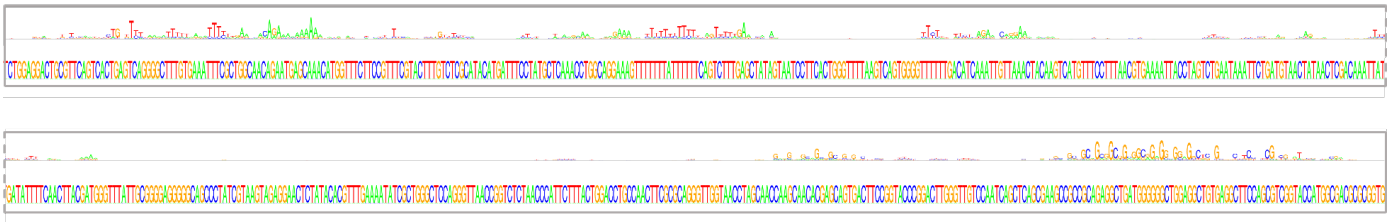

Figure 4. Dinucleotide Shuffle Saliency

Saliency plot of positive test sequences that were correctly classified (with a probability >0.99) by a CNN trained with background sequences that were created by performing a dinucleotide shuffle of positive sequences. For part (a), the first row is a sequence from GM12878, rows 2 and 3 are both sequences from HepG2, and row 4 is a sequence from MCF-7. For clarity, the original sequences are not displayed. Motifs of interest discussed in the results section are highlighted with a purple box. These highlighted motifs are also shown under the sequences for easier comparison. Note that the binding motif for Sp1 actually appears many times throughout the sequences, but for clarity, a purple box does not appear around all appearances. For part (b) the top two rows represent the original sequence and CNN's contributing motifs for a sequence from HeLa-S3, and the bottom two rows represent the CNN's contribution motifs and original sequence for a sequence from GM12878. Note that for 4b, the sequence has been split in half so that the top segment shows the first 300nts and the bottom segment displays the last 300nts.

Figure 4a

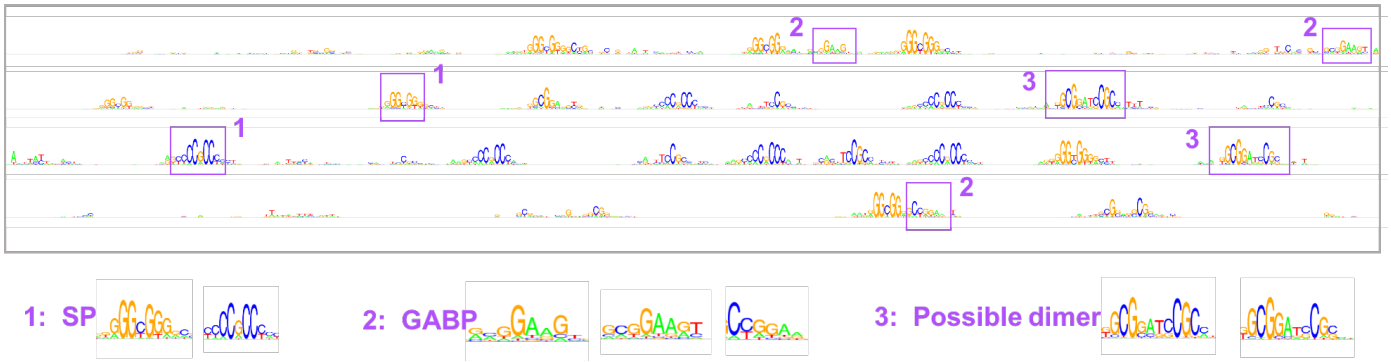

Figure 4b