Boosted Convolutional Decision Trees for Translationally Invariant Pattern Recognition and Transfer Learning

Wooyoung M. Moon[1] & Jun S. Song[1]

[1] Department of Physics, University of Illinois at Urbana-Champaign, Urbana, IL, USA

Correspondence: Jun S. Song, Department of Physics, University of Illinois at Urbana-Champaign, Urbana, IL, USA

Received: December 11, 2018 Accepted: January 9, 2019 Online Published: January 14, 2019

doi:10.5539/ijsp.v8n2p11 URL: https://doi.org/10.5539/ijsp.v8n2p11

Abstract

Decision Tree (DT) models provide a well-known class of interpretable machine learning tools for diverse pattern recognition problems. However, applying DTs to learn floating features in images and categorical data based on their raw representation has been challenging. Convolutional Neural Networks (CNNs) are the current state-of-the-art method for classifying raw images, but have their own disadvantages, including that they are often difficult to interpret, have a large number of parameters and hyperparameters, require a fixed image size, and have only partial translational invariance directly built into its architecture. We propose a novel application of Convolutional Decision Trees (CDTs) and show that our approach is more interpretable and can learn higher quality convolutional filters compared to CNNs. CDTs have full translational invariance built into the architecture and can be trained and make predictions on variable-sized images. Using two independent test cases — protein-DNA binding prediction, and hand-written digit classification — we demonstrate that our GPU-enabled implementation of the Cross Entropy (CE) optimization method for training CDTs learns informative convolutional filters that can both facilitate accurate data classifications in a tree-like pattern and be used for transfer learning to improve CNNs themselves. These results motivate further studies on developing accurate and efficient tree-based models for pattern recognition and computer vision.

Keywords: decision trees, image classification, convolutional neural networks, oblique decision trees, translational invariance

1. Introduction

Decision Tree (DT) models are classification and regression algorithms that can be represented by a tree structure consisting of nodes and branches. The internal nodes are defined by decision rules that determine the final leaf node with which a given sample input is associated, thereby ultimately resulting in an output value or prediction. Typically, DTs are built by iteratively partitioning the feature space into disjoint regions such that some measure of the homogeneity of training data in each region is maximized. To add more distinguishing power, ensemble methods such as boosting and bagging are often applied to DT models to create boosted decision trees (BDT) and random forests (RF), which have both been widely used in diverse fields including biology, medicine, and particle physics (Podgorelec, Kokol, Stiglic, & Rozman, 2002; Roe et al., 2005). The popularity and success of DT models may largely be attributed to their interpretability, speed, high level of performance, and ease of training.

Despite these strengths, however, current DT models have several weaknesses as well. One major shortcoming is that ordinary DT models have difficulty classifying images from their raw pixel representation. Because ordinary DTs split on individual features one at a time (e.g. individual pixel values in the case of raw images), DT-based models have difficulty learning generalizable or translationally invariant patterns required for robust classification rules. For this reason, DT models are generally used only on pre-processed images where the important features of an image have already been hand-curated or learned using some other method (Kolluru, Prabhu, Gharaibeh, Wu, & Wilson, 2018; Maturana, Mery, & Soto, 2010; Wilking & Röfer, 2004).

Convolutional Neural Networks (CNN) attempt to solve the problem of learning translationally invariant patterns by using convolutional and pooling layers. The convolutional layers convolve a learned filter across an image to create a "feature map," while the pooling layers define spatial neighborhoods on the feature map to consolidate and reduce its dimensionality by performing an operation such as keeping the maximum value within the pooling neighborhood. With the use of these principles, CNNs have produced state-of-the-art performance in various computer-vision tasks, such as image classification (Krizhevsky, Sutskever, & Hinton, 2012) and object detection (Girshick, Donahue, Darrell, & Malik, 2014). Despite these strengths in performance, CNNs often remain as black-box models, and developing methods for their interpretation is currently an active area of research (Finnegan & Song, 2017; Zeiler & Fergus, 2014). Furthermore, deep neural networks with many hidden layers are inherently difficult to train (Glorot & Bengio, 2010) and often require a

great amount of expertise and experience to choose an architecture, tune hyperparameters, initialize weights correctly, and avoid the "vanishing gradient" problem. An additional drawback is that a CNN also requires the user to pre-specify the size of images to be modeled; consequently, any images of a different size must be re-sized accordingly to fit the specific pre-defined architecture of the CNN.

In this paper, we introduce a tree-based approach called Boosted Convolutional Decision Trees (BCDT) that addresses many of the weaknesses of both tree-based models and CNNs by using fully translationally invariant convolutional filters in a tree-like structure. We begin by introducing the general theory underlying DT models and reviewing related work that has aimed to improve and extend DT models, followed by a brief introduction to the biological significance of our protein-DNA binding prediction test case. We then introduce our proposed method, BCDT, along with the Cross Entropy (CE) optimization method used to train our model. We demonstrate the power of our approach on two classification problems: protein-DNA binding prediction and MNIST hand-written digit classification. In the former case, we show that BCDT learns informative and interpretable convolutional filters, corresponding to DNA sequence motifs, that can be transferred to CNNs to improve their performance. For hand-written digit classification on the MNIST dataset with no data augmentation, we demonstrate that the BCDT model is more robust against image translations compared to other models.

1.1 Background and Related Work

A DT model is a non-parametric classification/regression framework in which decisions are made by traversing through a series of nodes applying branching rules, ultimately resulting in a final output value determined by the leaf node assigned to a sample. The decision rule at each node is defined by a function $F(\boldsymbol{X})$, which takes a sample $\boldsymbol{X}$ as its argument and outputs a value that determines to which child node the sample gets assigned. For example, in a univariate decision tree, which splits the data at each node using a single feature at a time, one may use $F(\boldsymbol{X}) = X_i$ where X_i is the i-th element of $\boldsymbol{X}$. Thresholding the outcome of $F(\boldsymbol{X})$ then yields the branching rules.

The most commonly used training algorithms for decision trees learn the decision rules in a greedy manner, choosing the most optimal split at each node by maximizing some measure of homogeneity in the training samples assigned to the child nodes. For classification trees, the Gini index and Shannon entropy are common measures of homogeneity, while for regression trees, the variance within child nodes is often used. Efficient algorithms such as ID3, C4.5, and CART (Breiman, Friedman, Olshen, & Stone, 1984; Quinlan, 1986, 1993) have allowed for fast training of classification and regression trees, largely contributing to the rise in popularity of DT models.

Most common DT algorithms involve univariate trees, which use axis-parallel splits to partition the feature space. In contrast, oblique DTs use a linear combination of available features to split samples at a given node. Specifically, at node k with trained weights $\boldsymbol{\beta}_k$ and threshold T, the decision function F_k is evaluated for each sample $\boldsymbol{X}$ as

$$F_k(\boldsymbol{X}) = \mathbb{1}(\boldsymbol{\beta}_k \cdot \boldsymbol{X} > T) \tag{1}$$

These oblique splits in feature space are better suited for situations where the decision boundaries are not parallel to any individual feature axes, and studies have shown that trees using oblique splits produce smaller and more accurate trees compared to those using axis-parallel splits (Li et al., 2003). However, a major challenge in training an oblique DT is the time complexity of finding the optimal oblique split. For a training set of n samples and p features, the time complexity for an exhaustive search for the best fit at any given node is $O(2^p\binom{n}{p})$ while it is just $O(np)$ for an axis-parallel split (Murthy, Kasif, & Salzberg, 1994).

Much work has been done to date to develop efficient algorithms for finding near-optimal oblique DTs without the need to perform an exhaustive search. Classification and Regression Trees - Linear Combination (CART-LC) (Breiman et al., 1984) was the first major oblique DT induction algorithm, which used a hill-climbing algorithm to search for the best oblique split at a given internal node. A weakness of CART-LC is that there is no built-in mechanism to avoid getting trapped in local minima during optimization. To address this difficulty, Simulated Annealing Decision Tree (SADT) (Heath, Kasif, & Salzberg, 1993) uses the simulated annealing optimization algorithm to search for optimal splitting.

Tree-based methods, including univariate DTs (Sharma, Ghosh, & Joshi, 2013), RFs (Kulkarni & Lowe, 2016), BDTs (Nowakowski, 2015), and oblique DTs (Palaniappan, Zhu, Zhuang, Zhao, & Blanchard, 2000) have been used for image segmentation of remote sensing data where individual pixels are classified based on data from multiple different types of sensors (e.g. infrared cameras, RADAR, etc.). In 2014, the authors of (Laptev & Buhmann, 2014) proposed Convolutional Decision Trees (CDT), a particular instance of an oblique decision tree, as an accelerated method for feature learning and image segmentation. In their method, a quasi-newton optimization algorithm is used to learn convolutional filters that classify individual pixels as belonging to a certain class. Despite the previous work that has applied tree-based

algorithms to image-related tasks, tree-based models have yet to be effectively used for pattern recognition and whole-image classification based on raw pixel values. Our BCDT approach extends the concepts from Laptev's CDT model originally used for image segmentation to classify whole images based solely on their raw pixel values.

Transcription Factors (TF) are proteins that regulate the rate of transcription of genetic information from DNA to RNA. TFs generally bind to DNA in a sequence-specific manner governed by its "DNA-binding domain," which recognizes and binds specific DNA sequences called "binding motifs." The two TFs, GABP and ELF1, belong to the same oncogenic ETS family of TFs and share similar binding motifs characterized by a core 'GGA(A/T)' DNA sequence. Despite the apparent similarity in their binding motifs, GABP and ELF1 can bind different locations in the genome *in vivo*. One particularly important example of this phenomenon involves two proximal point mutations in the promoter of telomerase reverse transcriptase (TERT) gene; these mutations occur at high frequency in multiple cancer types (Bell et al., 2016), and either of them selectively leads to GABP, but not ELF1, binding to the location (Bell et al., 2015). The binding of GABP to the mutant TERT promoter is critical for replicative immortality and is being actively explored as a potential target for cancer therapy (Mancini et al., 2018). Thus, answering the open question of which sequence features determine the binding specificity of GABP vs. ELF1 is of great interest to the cancer research community.

2. Methods

2.1 Boosted Convolutional Decision Trees (BCDT)

Our proposed BCDT model is composed of Adaboosted CDTs (Fig. 1a), which split at non-terminal nodes by scanning data for the presence or absence of learned convolutional filters. More precisely, at node k with a convolutional filter $\boldsymbol{\beta}_k$ of fixed size (w_1,w_2), the following decision function is evaluated for each sample:

$$F_{\boldsymbol{\beta}_k}(\boldsymbol{X}) = \max_{i,j} \mathbb{1}(\boldsymbol{\beta}_k \cdot \boldsymbol{X}_{i:i+w_1, j:j+w_2} > T) \tag{2}$$

where $\boldsymbol{X}$ is the full data matrix encoding the sample, $\boldsymbol{X}_{i:i+w_1, j:j+w_2}$ is the submatrix of $\boldsymbol{X}$ with rows i through $i + w_1 - 1$ and columns j through $j + w_2 - 1$, $\boldsymbol{\beta}_k$ is the convolutional filter at node k, $\mathbb{1}$ is an indicator function, and T is a chosen constant threshold value set by the user. In more intuitive terms, the input $\boldsymbol{X}$ is scanned by the convolutional filter $\boldsymbol{\beta}_k$ to find whether there is a contiguous patch of entries in $\boldsymbol{X}$ resembling the convolutional filter. If a match is found, then $\boldsymbol{X}$ gets mapped to one child node, and if no match is found, then $\boldsymbol{X}$ gets mapped to the other child node.

The main obstacle in training a CDT is finding the optimal convolutional filter $\boldsymbol{\beta}^*$ that maximizes the increase in homogeneity from the parent to child nodes. In our implementation of BCDT, we use Shannon entropy H as our measure of homogeneity for classification, defined as $H = -\sum_{c\in\{L,R\}} \frac{n_c}{n_L+n_R} \sum_y p_{y,c} \log p_{y,c}$, where

$$p_{y,c} = \frac{n_{y,c}}{n_c}, \tag{3}$$

$$n_c = \sum_{y\in Y} n_{y,c}, \tag{4}$$

$$n_{y,c} = \sum_{i\in\{j|C_j=c\}} w_i \, \mathbb{1}(y_i = y), \tag{5}$$

where w_i are the sample weights used for the Adaboost algorithm, $\{j|C_j = c\}$ is the set of all sample indices that are in child node c, y_i are the class labels, and Y is the set of all classes. To search for these optimal convolutional filters, we use the CE method described in the following section.

Finally, each terminal leaf node is assigned a probability value for each class corresponding to the multinomial maximum likelihood estimate (MLE) of the class probabilities according to the training samples that were mapped to that particular terminal leaf node. From this point, it is left as a choice to use either the probabilistic output values or the class label with highest MLE probability $\hat{y}_c$ as the output value for the terminal leaf node.

$$\hat{y}_c = \operatorname*{argmax}_y p_{y,c} \tag{6}$$

In the context of applying CDT to protein-DNA binding prediction, we include two extra biological details to the CDT model. First, we scan both the original sequence and its reverse complement (i.e. $A \leftrightarrow T$, $C \leftrightarrow G$, and sequence reversed) with a given convolutional filter to account for the fact that DNA is double stranded. Second, instead of initializing the CE

method at a random point in parameter space, it was initialized by sampling from evenly distributed points in parameter space in order to more fully and evenly explore the space of all convolutional filters. These evenly distributed points in parameter space corresponded to the one-hot encoding representation of all DNA sequences of length equal to 9. Further details are described below.

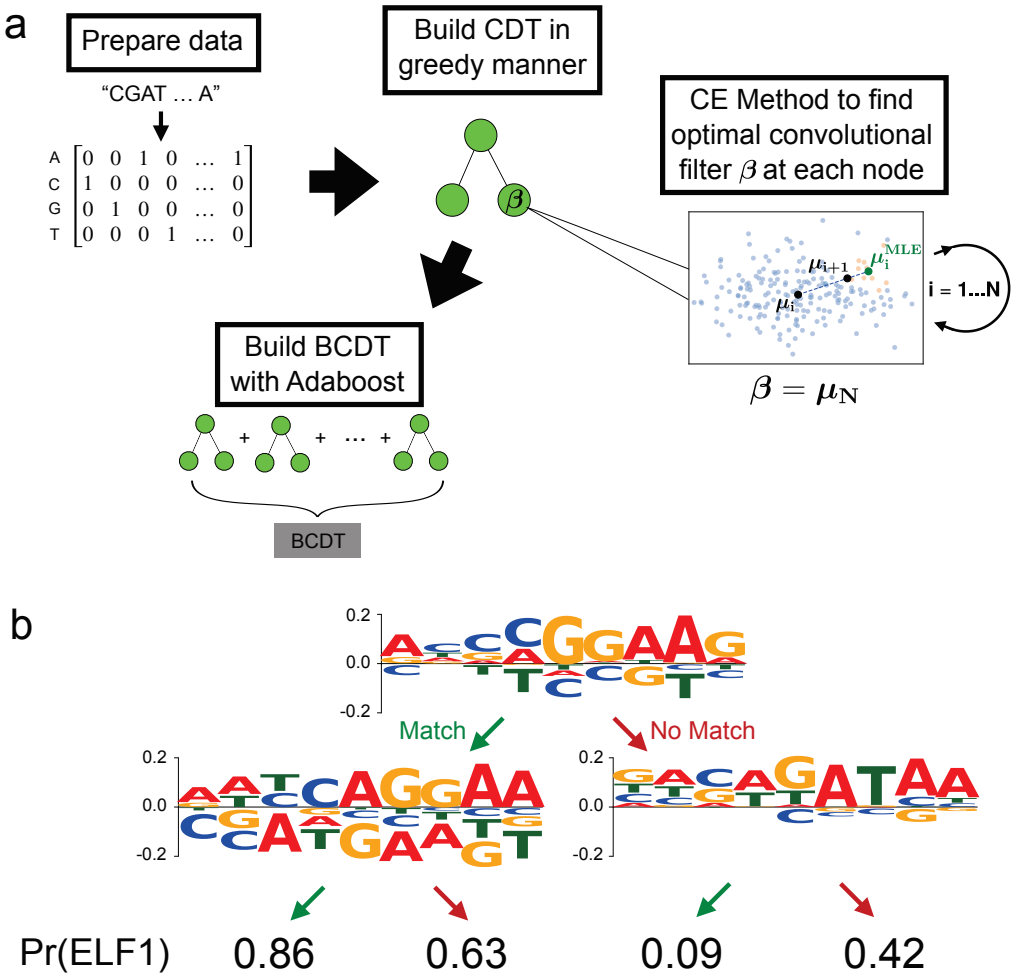

Figure 1. BCDT algorithm summary and structure

Note: (a) Schematic overview of the BCDT algorithm. (b) The first CDT of BCDT trained to classify between ELF1- and GABP-bound DNA sequences. The convolutional filter $\boldsymbol{\beta}$ at each node is represented by a sequence logo where the height of the letters (A, C, G, T) represents the value of the weight corresponding to that nucleotide in the one-hot encoding representation. The green and red arrows point to the child node that an input sample $\boldsymbol{X}$ is mapped to when a sequence pattern matching the convolutional filter of the parent node is found (green) or not found (red) in $\boldsymbol{X}$.

2.2 Cross Entropy (CE) Method

The Cross Entropy (CE) method (Rubinstein, 1999) was originally proposed for rare-event sampling and was later adapted as an optimization algorithm for finding globally optimal solutions in non-convex optimization problems. Based on the principle of importance sampling, the CE method searches for globally optimal parameter values by iteratively sampling points from a parametric distribution that gradually becomes more sharply centered around the optimal values. More precisely, at the i-th sampling step, M points in parameter space are drawn from a multivariate normal distribution $G(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$ estimated from the previous step. Subsequently, the loss function (for DT models, the loss function is some measure of label homogeneity of nodes) is computed for each of the M sets of parameter values, and only the top p percent of the M points is kept, while the rest is discarded. Finally, the new distribution $G(\boldsymbol{\mu}_{i+1}, \boldsymbol{\sigma}_{i+1})$ for the next iteration is calculated by taking a weighted average of the current parameters ($\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$) and the maximum likelihood estimates ($\boldsymbol{\mu}_i^{\text{MLE}}$ and $\boldsymbol{\sigma}_i^{\text{MLE}}$) obtained from using only the top points:

$$\boldsymbol{\mu}_{i+1} = \alpha\boldsymbol{\mu}_i^{\mathrm{MLE}} + (1-\alpha)\boldsymbol{\mu}_i, \tag{7}$$

$$\boldsymbol{\sigma}_{i+1} = \alpha\boldsymbol{\sigma}_i^{\mathrm{MLE}} + (1-\alpha)\boldsymbol{\sigma}_i, \tag{8}$$

where α is a hyperparameter that takes a value between 0 and 1.

For the case of the BCDT model trained for protein-DNA binding prediction, 8000 samples were drawn from the 36-dimensional parameter space at each iteration of the CE method for a total of 12 iterations. Of the 8000 samples, the top 20 points were kept to calculate the maximum likelihood estimates ($\boldsymbol{\mu}_i^{\mathrm{MLE}}$ and $\boldsymbol{\sigma}_i^{\mathrm{MLE}}$) used for the update step. For the case of the BCDT model trained to classify the MNIST hand-written digit images, 4000 samples were drawn from the 49-dimensional space at each iteration of the CE method with a total of 12 iterations, and the top 20 points were kept to calculate the maximum likelihood estimates used for the update step.

Furthermore, we have modified the initialization step of the CE method for both the protein-DNA interaction and MNIST test cases in order to more fully and evenly explore the parameter space. In the case of predicting protein-DNA interactions, the first sampling step was performed by sampling 8000 unique DNA sequences from all possible 4^9 possible DNA sequences of length 9, corresponding to the width of the convolutional filter, and converting them to their one-hot encoding vector representation. In the case of classifying MNIST hand-written digits, the first sampling step was performed by sampling 4000 unique points from a uniform distribution in a 49-dimensional hypercube with side length equal to one and parameter values between 0 and 1. After the modified initializations of the CE method, the normal CE method was carried out with no restrictions on the possible parameter values. Algorithm 1 summarizes our implementation of the CE method.

Algorithm 1 CE method

function FIND_OPTIMAL_FILTER($N, M, \alpha, \sigma_{\mathrm{initial}}$)
  $\boldsymbol{\sigma} = \sigma_{\mathrm{initial}}\mathbb{1}$
  for $i \leftarrow 1, N$ **do**
    if $i == 1$ **then**
      samples_list = SAMPLE_CUSTOM(M) ▷ Modified sampling described in Methods
      top_samples_list = GET_TOP_SCORING(samples_list, 20)
      $\boldsymbol{\mu}^{\mathrm{MLE}}, \boldsymbol{\sigma}^{\mathrm{MLE}} \leftarrow$ MLE(top_samples_list)
      $\boldsymbol{\mu} = \boldsymbol{\mu}^{\mathrm{MLE}}$
      $\boldsymbol{\sigma} = \alpha\boldsymbol{\sigma}^{\mathrm{MLE}} + (1-\alpha)\boldsymbol{\sigma}$
    else
      samples_list = SAMPLE_MULTIVARIATE_NORMAL($\boldsymbol{\mu}, \boldsymbol{\sigma}, M$)
      top_samples_list = GET_TOP_SCORING(samples_list, 20)
      $\boldsymbol{\mu}^{\mathrm{MLE}}, \boldsymbol{\sigma}^{\mathrm{MLE}} \leftarrow$ MLE(top_samples_list)
      $\boldsymbol{\mu} = \alpha\boldsymbol{\mu}^{\mathrm{MLE}} + (1-\alpha)\boldsymbol{\mu}$
      $\boldsymbol{\sigma} = \alpha\boldsymbol{\sigma}^{\mathrm{MLE}} + (1-\alpha)\boldsymbol{\sigma}$
  return $\boldsymbol{\mu}$ ▷ Final multivariate normal distribution centered at near-optimal point

2.3 BCDT Implementation

The code for the base CDT model was written in Python3 and was made compatible with scikit-learn's implementation of Adaboost in order to create a BCDT. Additionally, the deep learning framework PyTorch was used to run the matrix calculations necessary for the CE optimization method on a GPU-enabled device. The source code for BCDT can be found at https://github.com/jssong-lab/BCDT.

This study is meant as proof of concept, and thus, a full analysis on hyperparameter choice was not carried out. For the example cases presented in this paper, the maximum depth and convolutional filter dimensions for the BCDT models were chosen to maximize classification accuracy on a validation set. Additionally, the CE method parameters were tuned by hand to balance the model's accuracy and training time. In-depth analyses of more optimal hyperparameter choices are left for future studies.

2.4 CNN Architecture and Details

The CNN used in the protein-DNA binding prediction analysis had the following architecture: a convolutional layer consisting of 30 filters of size 4x9 and stride length equal to one, connected to a max-pooling layer with kernel size

1x20 and stride length equal to 10, connected to an output layer of two units corresponding to the two classes (ELF1 and GABP). Batch normalization was used in the convolutional layer and the ReLU activation function was used before the max-pooling operation. The cross entropy loss was used as the loss function and the Adam optimization procedure was used with a learning rate of 10^{-3}. Furthermore, the training set was supplemented with the reverse complement of each DNA sequence in the original training set.

The CNN used in the MNIST handwritten digit classification analysis had the following architecture: a convolutional layer consisting of 15 filters of size 7x7 and stride length equal to one, connected to a max-pooling layer with kernel size 3x3 and stride length equal to one, connected to a dropout layer, connected to a convolutional layer consisting of 25 filters of size 5x5 and stride length equal to one, connected to a fully connected layer with 225 units, connected to a dropout layer, connected to a fully connected layer with 50 units, connected to a softmax output layer with 10 units corresponding to the 10 different digits. The negative log-likelihood (NLL) loss was used as the loss function and the Adam optimization procedure was used with a learning rate of 10^{-3}.

CNN weights were randomly initialized using the default settings in PyTorch 0.4. Both convolutional and fully connected layer weights were initialized by sampling from a uniform distribution in the range $[\frac{-1}{\sqrt{n}}, \frac{1}{\sqrt{n}}]$, where n is equal to the fan-in for fully connected layers and the kernel size for convolutional layers.

2.5 Data

2.5.1 Protein-DNA Prediction

To test the BCDT model on protein-DNA interaction prediction, ChIP-seq data indicating the binding locations of the transcription factor (TF) proteins ELF1 and GABP in the K562 cell-line were obtained from the publicly available ENCODE database (ENCODE Project Consortium, 2012). Both ELF1 and GABP belong to the ETS family of TFs and share similar binding motifs characterized by a core 'GGA(A/T)' DNA sequence. Of the ChIP-seq binding locations, all overlapping locations were removed; and, balanced labeled training and test sets of 6000 and 2000 DNA sequences of length 200 base pairs (bp) centered around the ELF1 and GABP ChIP-seq peaks were built. Finally, the 200 bp sequences were transformed into one-hot encoding representations. The resulting matrices of dimension 4x200 were treated as images of size 4x200 in the analysis.

2.5.2 MNIST

The well-known MNIST dataset was used to further test and compare the BCDT model against other models. All images were mean-centered and standardized to help with neural network training. Additionally, 15 versions (original plus 14 modified versions, available at https://github.com/jssong-lab/BCDT/translated_MNIST.zip) of the test set with increasing amounts of horizontal and vertical translations applied to the images were used to assess the translational invariance of the various models. For example, the first modified test set contained the test set images where each image was translated either left or right by a single pixel and translated either up or down by a single pixel. Similarly, the 14th modified test set contained the test set images where each image was translated either left or right by 14 pixels and translated either up or down by 14 pixels.

3. Experiments and Results

3.1 Protein-DNA Binding Prediction

3.1.1 BCDT Makes Accurate and Intepretable Predictions on DNA Sequences

A BCDT model consisting of 35 Adaboosted CDTs with maximum depth of two and convolutional filters of length nine was trained to classify ELF1-bound vs. GABP-bound DNA sequences. In addition to the BCDT model, we trained an ordinary DT, a BDT, a fully connected neural network (FC NN) with a single hidden layer of 500 units, and a CNN described in the Methods. The accuracy and AUC scores for these models are presented in Table 1. It is seen that only the two models that make use of convolutional filters (BCDT and CNN) have test set classification accuracies significantly higher than random guessing (i.e. 0.5), with the BCDT classifying with 0.75 accuracy and the CNN classifying with 0.70 accuracy. In contrast, the BDT, DT, and FC NN classifiers have classification accuracies approximately equivalent to random guessing, suggesting that these models were unable to learn meaningful prediction rules. Furthermore, Fig. 1b shows a pictorial representation of the first CDT in the BCDT, in which versions of the known ETS binding motif 'GGAA' can be seen in both the head and left child node. In comparison, Figure S1 shows the convolutional filters learned by the CNN model, none of which display the 'GGAA' ETS motif as clearly as the ones learned by the BCDT. On closer inspection, the convolutional filter of the head node in Fig. 1b contains the 'CCGGAA' hexamer while the convolutional filter of the left child node contains the 'CAGGAA' hexamer, suggesting that variations in the nucleotides flanking the central 'GGAA' ETS binding motif may be important for distinguishing DNA sequences bound by ETS family TFs such as ELF1 and GABP, as proposed in previous studies (Shore et al., 1996; Szymczyna & Arrowsmith, 2000; Wei et al.,

2010). Furthermore, the convolutional filter for the right child node resembles a '(A/T)GATA(A/G)' motif recognized by GATA proteins, which have previously been shown to cooperate with ETS family TFs to regulate the expression of certain genes (Lemarchandel, Ghysdael, Mignotte, Rahuel, & Romeo, 1993; Rothbächer, Bertrand, Lamy, & Lemaire, 2007). Specifically, according to the CDT, the presence of a GATA motif increases the probability that a DNA sequence is bound by GABP compared to ELF1. This can be seen by inspecting the output values associated with the two leaf nodes connected to the right internal child node. In the situation where a DNA sequence is mapped to the right internal child node, the presence of the GATA motif increases the CDT model's output prediction that the sequence is bound by GABP from 0.58 to 0.91.

Table 1. Protein-DNA binding prediction scores

Model	Accuracy	AUC
BCDT	0.75	0.83
BDT	0.50	0.52
DT	0.49	0.49
CNN	0.70	0.78
FC NN	0.53	0.54

3.1.2 BCDT Learns Useful Convolutional Filters for Transfer Learning With CNN

To further study the quality and predictive power of the convolutional filters learned by the BCDT algorithm, CNNs with identical architectures were trained using three different training approaches:

1. **Regular:** Random initialization of all weights; all weights trainable.
2. **Initialization Only:** CNN convolutional filter weights initialized with convolutional filters learned from BCDT and bias values set to zero; random initialization of all other CNN weights; all weights trainable.
3. **Initialization + Freezing:** CNN convolutional layer weights initialized with convolutional filters learned from BCDT and bias values set to zero; random initialization of all other CNN weights; convolutional layer weights frozen, and all other CNN weights trainable.

Figure. 2a shows the average training set and test set accuracy of CNN models using the three different training strategies, as a function of training epoch. The test accuracy of the CNN using convolutional layer initialization and freezing (strategy three) rises the slowest of all three, but gradually becomes the highest while the CNNs using the first and second strategy plateau earlier and even show a slight decrease in accuracy, possibly as a result of overfitting. Furthermore, Table 2 shows, for all three strategies, both the test and training set classification accuracy values at the epoch where the test accuracy was maximal. These values represent a post-hoc look at the model accuracies without needing to specify an early-stopping procedure for training, allowing the "best case scenarios" for the three different training strategies to be compared. It is seen that the difference between the training and test accuracy in Table 2 is smallest for the Initialization + Freezing strategy, suggesting that this method is least prone to overfitting. Additionally, Fig. 2b shows that the difference between the average training and test accuracy of the Initialization + Freezing strategy increases in magnitude at a significantly slower rate compared to the two other strategies, further demonstrating that the Initialization + Freezing strategy may be relatively less prone to overfitting.

Table 2. BCDT-CNN transfer learning accuracy scores

Training Strategy	Training Accuracy	Test Accuracy
Regular	0.757	0.704
Initialization Only	0.800	0.716
Initialization + Freezing	0.765	0.739

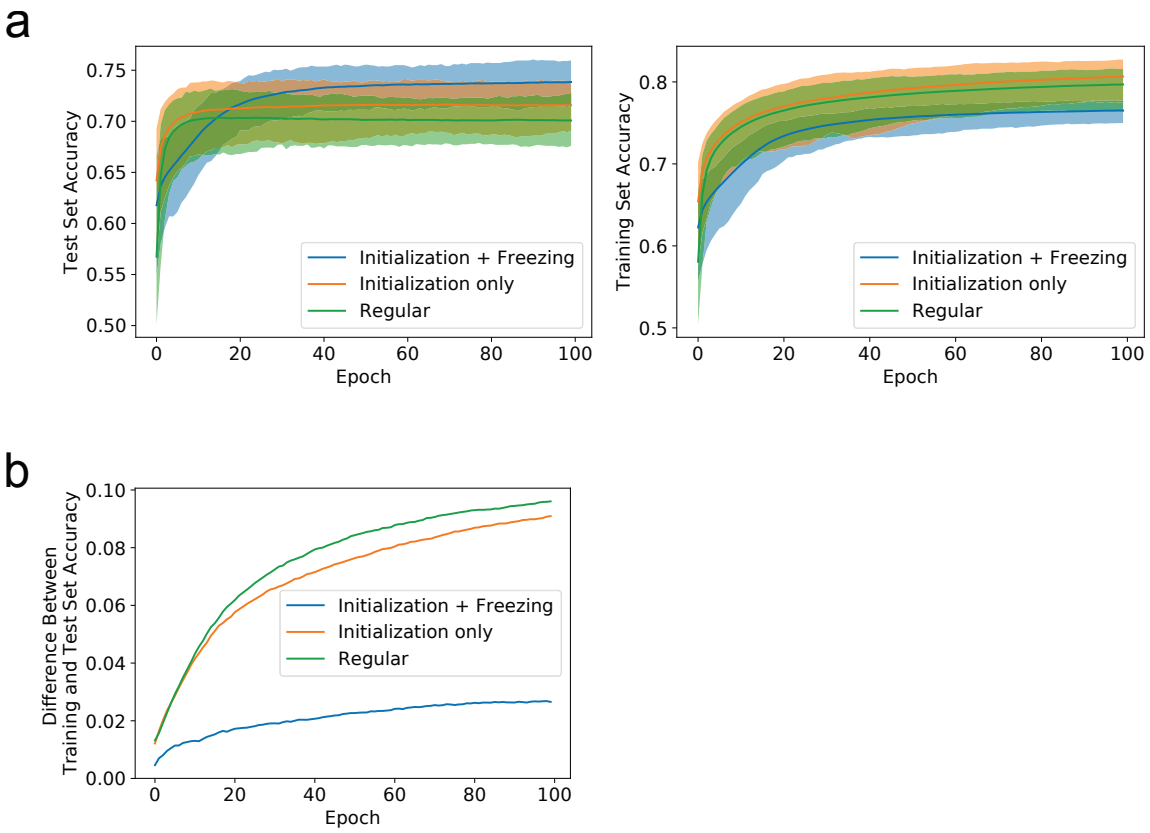

Figure 2. Transfer learning of robust convolutional filters from BCDT to CNN

Note: (a) Training and test set accuracies for ELF1 vs. GABP classification, as a function of training epoch for the three different CNN training strategies (Regular, Initialization Only, and Initialization + Freezing) averaged over 200 iterations with different, randomized splits of training and test sets. The shaded regions represent the 5th to 95th percentile range of classification accuracies over the 200 iterations. (b) Difference between the averaged training and test set accuracies as a function of training epoch for the three CNN training strategies. The significantly smaller difference for the Initialization + Freezing strategy suggests a smaller degree of overfitting.

3.2 MNIST Handwritten Digit Classification

3.2.1 BCDT Classifies MNIST Handwritten Digits With High Accuracy

A BCDT model consisting of 30 Adaboosted CDTs with maximum depth of seven and convolutional filters of size 7x7 was trained to classify the MNIST set of handwritten digit images. In addition to the BCDT model, we trained an ordinary DT, a BDT, a FC NN with a single hidden layer of 500 units, and a convolutional neural network (CNN) described in the Methods. For comparison, Table 3 shows the test set classification accuracies of the various models. The BCDT model (0.934) had a test set classification accuracy comparable to that of the BDT (0.945) and FC NN (0.920), models while the CNN (0.989) had the highest classification accuracy and the ordinary DT (0.879) model had the lowest. The relatively high test accuracy scores of the BDT, DT, and FC NN models compared to that of the protein-DNA binding test case can largely be attributed to the fact that the MNIST images are centered on the hand-written digits. Next, we demonstrate that the test accuracy scores of many of the models in our study are highly sensitive to image translations.

Table 3. MNIST classification accuracy scores

Model	Accuracy
BCDT	0.934
BDT	0.945
DT	0.879
CNN	0.989
FC NN	0.920

3.2.2 BCDT Decision Rules Are Robust Against Image Translations

In order to quantify the translational invariance of the various models trained for digit classification, we performed an experiment where we compared the classification accuracies of BCDT, CNN, FC NN, BDT, and single DT models on 15 different versions of the MNIST test dataset with increasing degrees of linear translations applied to the images (Methods and Fig. 3). Fig. 3a shows the classification accuracies for each model as a function of the magnitude of translation of the test set images. Although the CNN model initially has the highest classification accuracy (0.989), as the test set images are translated further from their original positions, its classification accuracy decreases quickly, particularly once the test set images have been translated by more than two pixels. In contrast, the BCDT model starts with a lower classification accuracy (0.934), but its accuracy decays much more slowly compared to the other models and becomes the most accurate classifier for all test sets with images translated in both the horizontal and vertical directions by two or more pixels.

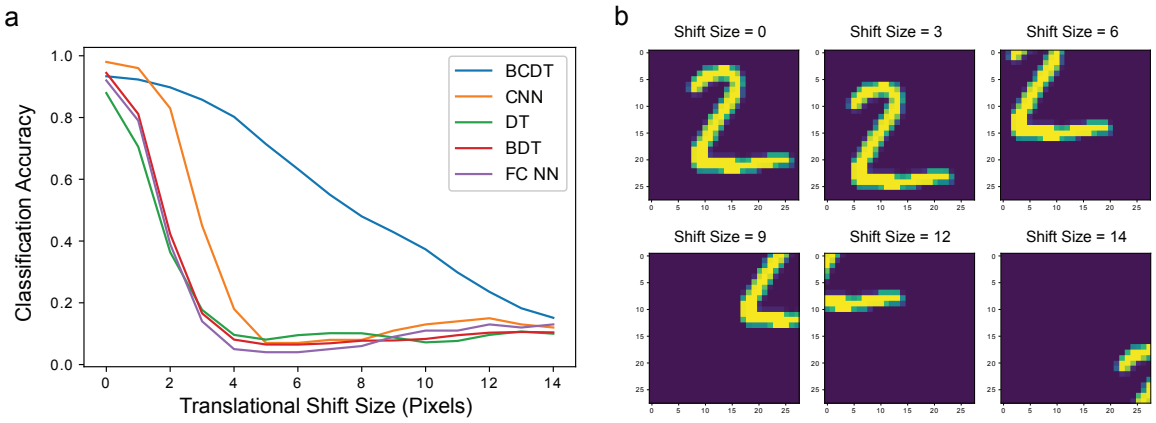

Figure 3. BCDT is relatively invariant under translation

Note: (a) Line plot showing the classification accuracies of various models on increasingly translated versions of the MNIST test set. In particular, the BCDT model's classification accuracy decays substantially more slowly than the other models tested. (b) Examples of an MNIST test set image of a handwritten "2" with increasing amounts of translation.

4. Discussion and Conclusion

We have introduced a novel application of BCDTs using the CE optimization method and applied this BCDT model to protein-DNA interaction prediction and hand-written digit classification. Our results demonstrated several advantages of BCDTs over other models often used in image classification problems: First, we showed that BCDTs are able to learn biologically interpretable convolutional filters to accurately predict the binding-specificity of two proteins belonging to the same family. Due to the relative simplicity of the binary tree structure of CDTs, it is feasible to interpret how individual convolutional filters are used by the BCDT by tracking the various paths and output values down a given CDT. In contrast, not only does the CNN model learn less biologically interpretable convolutional filters (Fig. 1 vs. Fig. S1), but it is also much more difficult to interpret how the individual convolutional filters are being used by the CNN to make its final predictions. These distinct advantages of the BCDT model over the CNN model are particularly important in situations such as protein-DNA interaction prediction where extracting the biologically relevant information from the model is just as important as maximizing prediction accuracy. Along this line, we have recently demonstrated the utility of our method in the context of learning biological sequence features that can help distinguish the *in-vivo* binding patterns of two important oncogenic transcription factors in the same protein family (Hejna et al., 2019).

Second, we demonstrated that the convolutional filters learned by BCDT can be transferred to CNNs to allow the CNN to make more accurate protein-DNA interaction predictions. The fact that freezing the CNN convolutional layer weights with those learned by BCDT leads to greater performance compared to the fully trained CNN suggests that the convolutional filters learned by the BCDT through the CE method may be more biologically relevant than the ones learned by the CNN through gradient descent and backpropagation. Furthermore, the use of transfer learning in genomics has most commonly been carried out by training a neural network to do a given task for one particular species and then sharing many of its learned parameters with another neural network trained for another species (Plekhanova, Nuzhdin, Utkin, & Samsonova, 2019). In contrast, our method of transfer learning does not rely on using a separate training set. Instead, we present a BCDT method for learning high quality convolutional filters directly from the dataset of interest. To our knowledge, we provide the first example of using a tree-based model to directly learn convolutional filters for transfer learning with CNNs in DNA sequence classification.

Third, we demonstrated the intrinsic translational invariance of BCDTs by showing that the BCDT test classification accuracy decays at a significantly slower rate compared to other models when the MNIST test set images were translated. Although data augmentation methods such as adding random translation to the training set are sometimes used to help CNNs perform better in these settings, we have demonstrated that BCDTs are able to learn relatively translationally invariant decision rules without the need to augment the training set. Related to this property is the capability of BCDTs to be trained and make predictions on images of different sizes. For example, the fact that BCDTs are agnostic to image size would allow a BCDT model to be trained to classify DNA sequences of almost any length (as long as it is greater than the length of a single convolutional filter). In contrast, most other model types would require that every DNA sequence in both the training and test set be of the same exact length. This extra flexibility of BCDT models may be useful in settings where commonly used methods such as cropping or rescaling of images are either impossible or undesirable, as in many biomedical applications.

The BCDT algorithm represents a step forward in applying DT models to image classification and a useful alternative to CNNs in situations where model interpretability, translational invariance, and image size flexibility are particularly important. Directions for future research may include better understanding the effects of the BCDT model hyperparameters, developing more advanced methods of BCDT model interpretation, and developing more efficient BCDT training algorithms.

Acknowledgements

This research was supported by grants from the National Brain Tumor Society and the National Institutes of Health (R01CA163336).

References

Bell, R. J., Rube, H. T., Kreig, A., Mancini, A., Fouse, S. D., Nagarajan, R. P., ... Costello, J. F. (2015). The transcription factor GABP selectively binds and activates the mutant TERT promoter in cancer. *Science*, *348*(6238), 1036–1039.

Bell, R. J., Rube, H. T., Xavier-Magalhães, A., Costa, B. M., Mancini, A., Song, J. S., & Costello, J. F. (2016). Understanding TERT promoter mutations: a common path to immortality. *Molecular Cancer Research*, *14*(4), 315–323.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth & Brooks/Cole Advanced Books & Software.

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

Finnegan, A., & Song, J. S. (2017). Maximum entropy methods for extracting the learned features of deep neural networks. *PLoS Computational Biology*, *13*(10), e1005836.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580–587).

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 249–256).

Heath, D., Kasif, S., & Salzberg, S. (1993). Induction of oblique decision trees. *Journal of Artificial Intelligence Research*, *2*(2), 1–32.

Hejna, M., Moon, W. M., Cheng, J., Kawakami, A., Fisher, D. E., & Song, J. S. (2019). Local genomic features predict the distinct and overlapping binding patterns of the bHLH-zip family oncoproteins MITF and MYC-MAX. *Pigment Cell & Melanoma Research*, *32*.

Kolluru, C., Prabhu, D., Gharaibeh, Y., Wu, H., & Wilson, D. L. (2018). Voxel-based plaque classification in coronary intravascular optical coherence tomography images using decision trees. In *Medical Imaging 2018: Computer-aided Diagnosis* (Vol. 10575, p. 105752Y).

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

Kulkarni, A. D., & Lowe, B. (2016). Random forest algorithm for land cover classification. *International Journal on Recent and Innovation Trends in Computing and Communication*, *4*(3), 58–63.

Laptev, D., & Buhmann, J. M. (2014). Convolutional decision trees for feature learning and segmentation. In *German Conference on Pattern Recognition* (pp. 95–106).

Lemarchandel, V., Ghysdael, J., Mignotte, V., Rahuel, C., & Romeo, P. (1993). GATA and Ets cis-acting sequences mediate megakaryocyte-specific expression. *Molecular and Cellular Biology*, *13*(1), 668–676.

Li, X.-B., Sweigart, J. R., Teng, J. T., Donohue, J. M., Thombs, L. A., & Wang, S. M. (2003). Multivariate decision trees using linear discriminants and tabu search. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, *33*(2), 194–205.

Mancini, A., Xavier-Magalhaes, A., Woods. W. S., Nguyen, K.-T., Amen, A. M., Hayes, J. L., ... Costello, J. F. (2018). Disruption of the β1L isoform of GABP reverses glioblastoma replicative immortality in a TERT promoter mutation-dependent manner. *Cancer Cell*, *34*(3), 513–528.

Maturana, D., Mery, D., & Soto, A. (2010). Face recognition with decision tree-based local binary patterns. In *Asian Conference on Computer Vision* (pp. 618–629).

Murthy, S. K., Kasif, S., & Salzberg, S. (1994). A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research*, *2*, 1–32.

Nowakowski, A. (2015). Remote sensing data binary classification using boosting with simple classifiers. *Acta Geophysica*, *63*(5), 1447–1462.

Palaniappan, K., Zhu, F., Zhuang, X., Zhao, Y., & Blanchard, A. (2000). Enhanced binary tree genetic algorithm for automatic land cover classification. In *IGARSS 2000. IEEE 2000 International Geoscience and Remote Sensing Symposium, 2000.* (Vol. 2, pp. 688–692).

Plekhanova, E., Nuzhdin, S. V., Utkin, L. V., & Samsonova, M. G. (2019). Prediction of deleterious mutations in coding regions of mammals with transfer learning. *Evolutionary Applications*, *12*, 18–28.

Podgorelec, V., Kokol, P., Stiglic, B., & Rozman, I. (2002). Decision trees: an overview and their use in medicine. *Journal of Medical Systems*, *26*(5), 445–463.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Roe, B. P., Yang, H.-J., Zhu, J., Liu, Y., Stancu, I., & McGregor, G. (2005). Boosted decision trees as an alternative to artificial neural networks for particle identification. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, *543*(2-3), 577–584.

Rothbächer, U., Bertrand, V., Lamy, C., & Lemaire, P. (2007). A combinatorial code of maternal GATA, Ets and β-catenin-TCF transcription factors specifies and patterns the early ascidian ectoderm. *Development*, *134*(22), 4023–4032.

Rubinstein, R. (1999). The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, *1*(2), 127–190.

Sharma, R., Ghosh, A., & Joshi, P. (2013). Decision tree approach for classification of remotely sensed satellite data using open source support. *Journal of Earth System Science*, *122*(5), 1237–1247.

Shore, P., Whitmarsh, A. J., Bhaskaran, R., Davis, R. J., Waltho, J. P., & Sharrocks, A. D. (1996). Determinants of DNA-binding specificity of ETS-domain transcription factors. *Molecular and Cellular Biology*, *16*(7), 3338–3349.

Szymczyna, B. R., & Arrowsmith, C. H. (2000). DNA-binding specificity studies of 4 ETS proteins supports an "indirect read-out" mechanism of protein-DNA recognition. *Journal of Biological Chemistry*, *275*(37), 28363–28370.

Wei, G.-H., Badis, G., Berger, M. F., Kivioja, T., Palin, K., Enge, M., ... Taipale J. (2010). Genome-wide analysis of ETS-family DNA-binding in vitro and in vivo. *The EMBO Journal*, *29*(13), 2147–2160.

Wilking, D., & Röfer, T. (2004). Realtime object recognition using decision tree learning. In *Robot Soccer World Cup* (pp. 556–563).

Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European Conference on Computer Vision* (pp. 818–833).

Appendix A

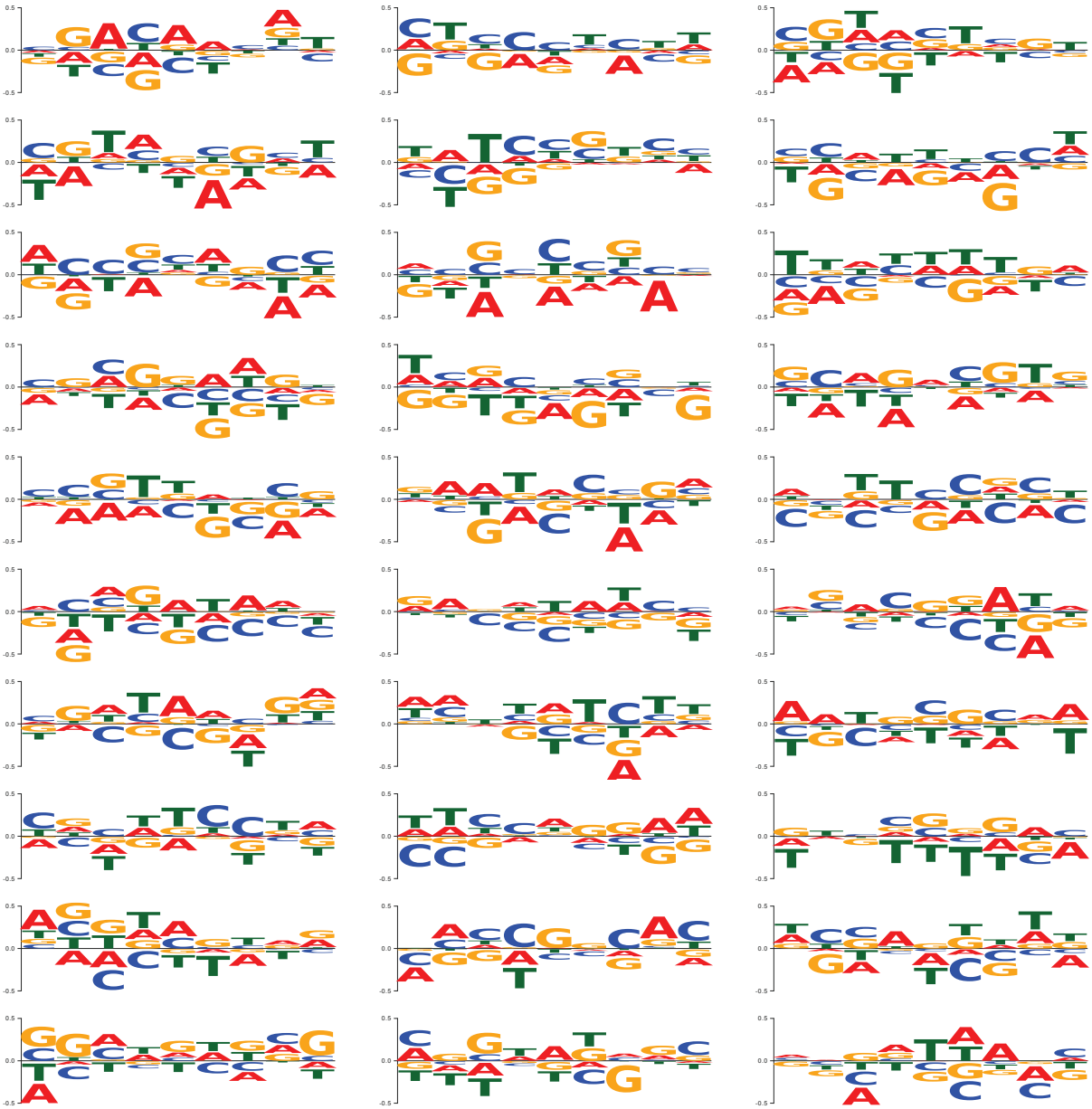

Figure S1. CNN convolutional filters using ”Regular” training: Sequence logo representation of all 30 convolutional filters learned by the CNN model trained to classify ELF1- and GABP-bound DNA sequences using the Regular training strategy described in the Results