CS 466 Final Project Report

Shuijing Liu (sliu105)

Mu Chen (muchen2)

Erfan Mohagheghian (mohaghe2)

Introduction

Computational DNA motif discovery is central to understanding and controlling gene expression. Motifs are typically short nucleotide sequences that are overrepresented statistically. They may appear several times across or within genes and it is conjectured that they possess biological significance, as they often represent the sequence-specific binding sites for 'transcription factors' (TFs) and other classes of regulatory proteins. Motif finding is a computationally daunting task: given a collection of sequences, one must find an unknown but frequent pattern of unknown length, while taking into account possible mutations, deletions or insertions.

In this project, we studied the popular motif finding algorithms, which will be discussed in Literature Survey section, and implemented RankMotif++ and Gibbs sampling algorithms on subsets of DREAM5 DNA - Motif Recognition Challenge[8] dataset.

Literature Survey

De novo algorithm[1]

Enumeration

In this approach, we simply do the exhaustive search through all possible motifs of specific length and then count the number of occurrence to identify the most frequent sequence as the motif. To make the algorithm more flexibility, we can describe motif as a consensus sequence and an allowed number of mismatches and uses an efficient suffix tree representation to find all such motifs in the target sequences. Enumerating approach will guarantee the global optimum, however, the abstractions needed to achieve an enumerable search space may overlook some of the subtle patterns present in real binding sites.

Probabilistic optimization

One of the most well-know algorithm is Gibbs sampling which is based on the stochastic implementation of the expectation maximization. In this approach, we randomly select the one motif sit in each sequence. Then, the PWM will be constructed based on these random sites. The calculated PWM will slide over a randomly chosen sequence to get the best matching site to the constructed PWM. In the next step, the site with the maximum matching value will be replaced with the previous site of that sequence, and then, the PWM will be updated. This iteration will continue until convergence.

Deterministic optimization

In this approach, the weight matrix of the motif is initialized and the probability of generating the sequences given this PWM will be calculated. Then, the expectation maximization algorithm is used to refine the motif model by maximizing the probability of sequences given the motif. One popular implementation of EM algorithm is MEME, performs a single iteration for each n-mer in the target sequences, selects the best motif from this set and then iterates only that one to convergence, avoiding local maxima.

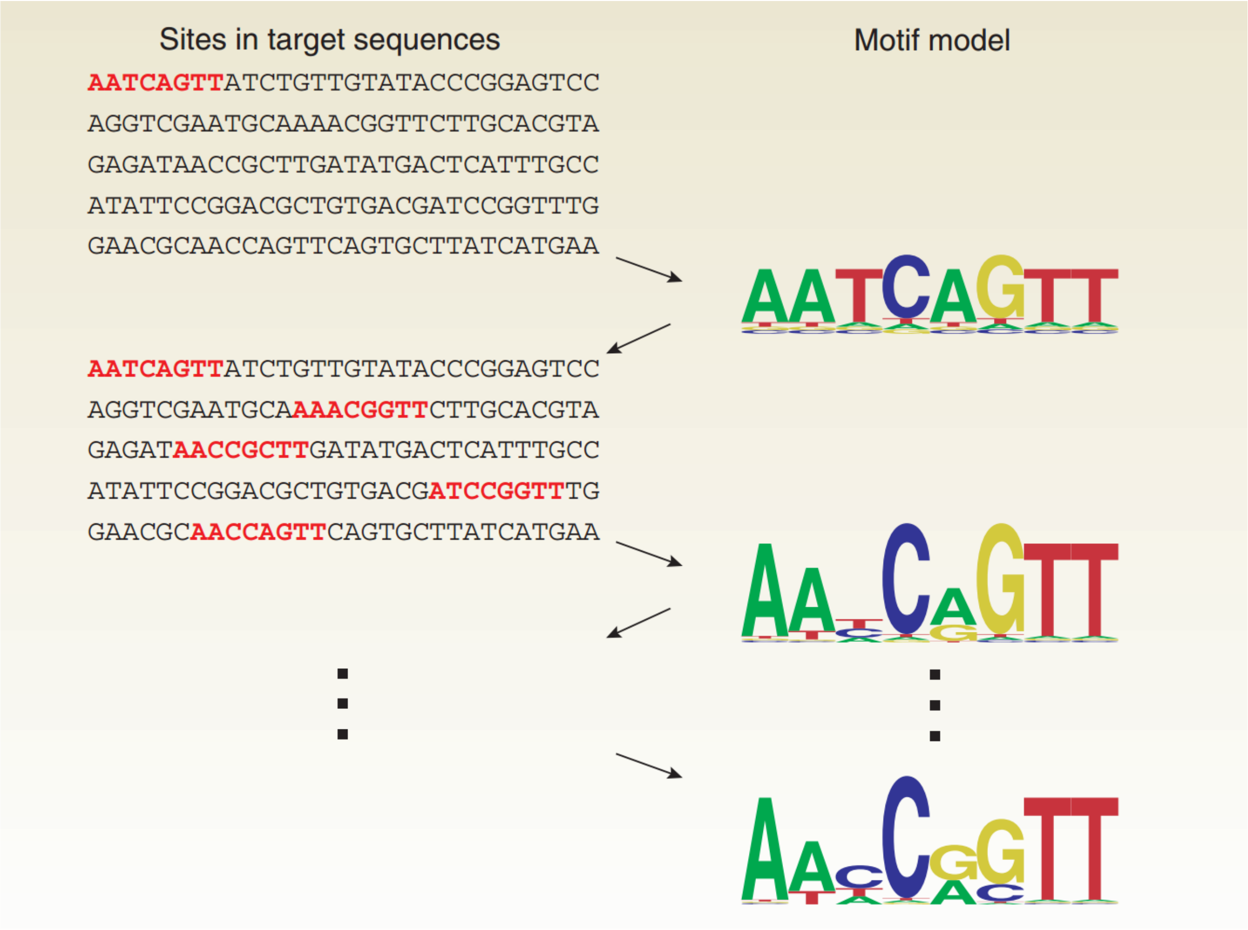

Sites in target sequences
Motif model
AATCAGTTATCTGTTGTATACCCGGAGTCC
AGGTCGAATGCAAAACGGTTCTTGCACGTA
GAGATAACCGCTTGATATGACTCATTTGCC
ATATTCCGGACGCTGTGACGATCCGGTTTG
GAACGCAACCAGTTCAGTGCTTATCATGAA
AATCAGTT
AATCAGTTATCTGTTGTATACCCGGAGTCC
AGGTCGAATGCAAAACGGTTCTTGCACGTA
GAGATAACCGCTTGATATGACTCATTTGCC
ATATTCCGGACGCTGTGACGATCCGGTTTG
GAACGCAACCAGTTCAGTGCTTATCATGAA

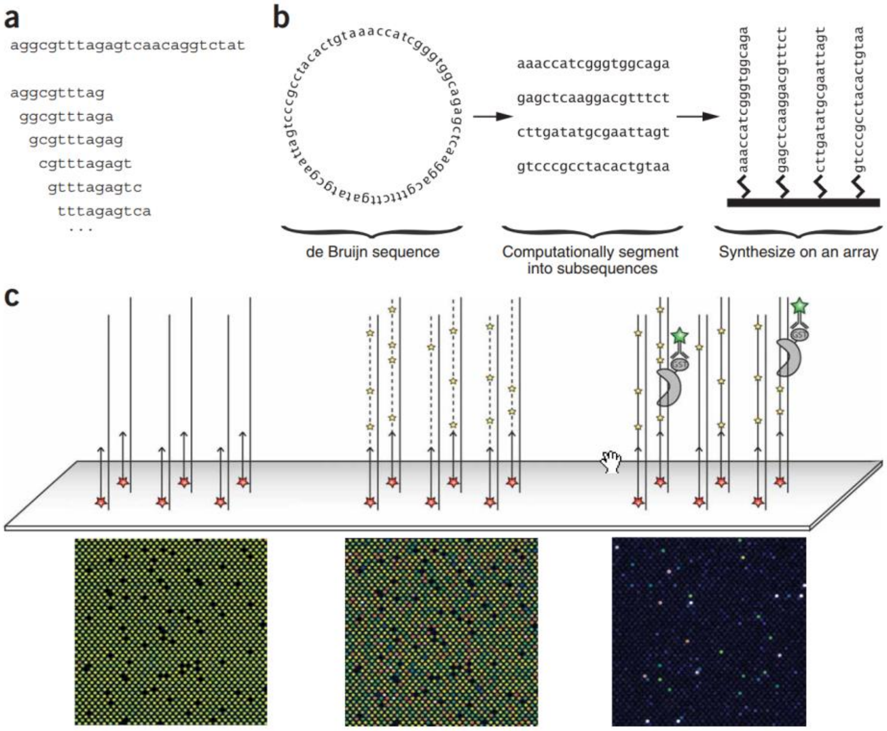
Figure 1 Starting from a single site, expectation maximization algorithms such as MEME[4] alternate between assigning sites to a motif (left) and updating the motif model (right). Note that only the best hit per sequence is shown here, although lesser hits in the same sequence can have an effect as well.

Chip-Seq

In the Chip-seq experiment, first the cells with the defined treatment will be fixed to preserve the binding of specific proteins (transcription factor or polymerase) and the DNA. Then after cell lysis and sonication to make the DNA fragments, the targeted proteins will be precipitated using the microbead coated with antibody specific to the protein. The short end of precipitated DNA fragments will be sequenced using high throughput sequencing and aligned to the reference genome. Finally, the high enriched segments of DNA indicating the TF binding sites will be calculated using peak calling. Predicting transcription factor binding *in vivo* is more difficult because it is affected by other proteins, the chromatin state and the physical accessibility of the binding site, and the sequence specificity of a TF is only one of several factors that determine where it binds *in vivo* (others include cofactors and DNA accessibility)

PBM[2]

In this approach, sequences containing all 4^k overlapping k-mers exactly once are named de Bruijn sequences of order k. The de Brujin sequence of order 10 is partitioned into sub-sequence of L with overlap of k-1 so that every 8-mer will occur at least 16 times. These subsequences will be attached to the DNA microarray which later will be incubated with target fluorescent-labelled antibody specific to the transcription factor.



Rankmotif [3]

In this algorithm, the purpose is maximizing the probability of observing a preference in the microarray data. The result of PBM microarray will first be converted to a subset of the binary random variables in X. where Xij=1 if probe sequence i is observed to be preferred over sequence j (i>j). As a result, given $S = \{s_1, s_2, .., s_n\}$ and the parameters of the motif mode (Θ, w). The probability of binding preference has been modeled as the log of binding affinity ratio which is modeled to be the.

$$L(\Theta, w) = \sum_{i>j} \log P(X_{ij} = 1 | \Theta, w, S)$$

Θ represents the parameters of the motif model. The model also includes a scaling factor w that governs the effect that changes in predicted binding affinity have on the probability of observing a preference in the microarray data. The parameters Θ and w were calculated using gradient descent optimization with backpropagation with chain rules.

Seed and Wobble[2]

Berger et al. used their seed-wobble motif finding algorithm. In their PBM data, each 8-mer occurs at least at 32 different probes and 8-mers with up to three gaps are represented 16 times on each de Bruin sequence. In this approach, the 8 mers with up to 3 gapped positions were scored with their enrichment score defined as $\frac{1}{B+F}\left[\frac{\rho_B}{B}-\frac{\rho_F}{F}\right]$ where B and F are the sample sizes of the background and foreground, respectively, and ρ_B and ρ_A are the sums of their respective ranks. Here, the 'foreground' features were defined as those containing a match to the 8-mer, and the 'background' features defined as all others. The next step, 8-mer (continuous or gapped) with highest E-score was identified as the seed.

Then, $F_{i,j,p}$ defined as set of all features on array $i \in \{1,2\}$ that contain a match to the variant of the seed that has letter $j \in \{A,C,G,T\}$ at position $p \in \{1,2,\ldots,8\}$ and $\overline{F_{i,j,p}}$ corresponding to features with no match to variant of the seed with letter j at position p .Using the above notation, the relative preferences of the TF for each of different variants of the seed at specific location was calculated as:

$$\psi_{i,j,p} = \frac{1}{N_{i,j,p} + \overline{N_{i,j,p}}}\left[\frac{\rho_{i,j,p}}{N_{i,j,p}} - \frac{\overline{\rho_{i,j,p}}}{\overline{N_{i,j,p}}}\right]$$

Here is N is the number of feature in that set and rho is the rank-sum of features in $F_{i,j,p.}$

Using the normalized probabilities of $\Psi_{j,p}$ ($P_{j,p}$), The least informative position of 8-mer seed was identified using the relative entropy of each position and then removed to get the final 7-mer. Finally, all extension of 7-mer to 8-mer with no more than e gaps were tested and the probability of each 4 variants to extend 7-mer to the 8-mer with maximum three gaps were calculate in the same process based on $\Psi_{j,p}$ and $P_{j,p}$.

Deep learning[4]

They adapted deep learning methods to the task of predicting sequence specificities. Their approach, called DeepBind, is based on deep convolutional neural networks and can discover new patterns even when the locations of patterns within sequences are unknown—a task for which traditional neural networks require an exorbitant amount of training data. In their approach, their network has four computation layers including convolution, rectification, pooling and neural network each with trainable parameters including motif detectors M, threshold b and weights W.

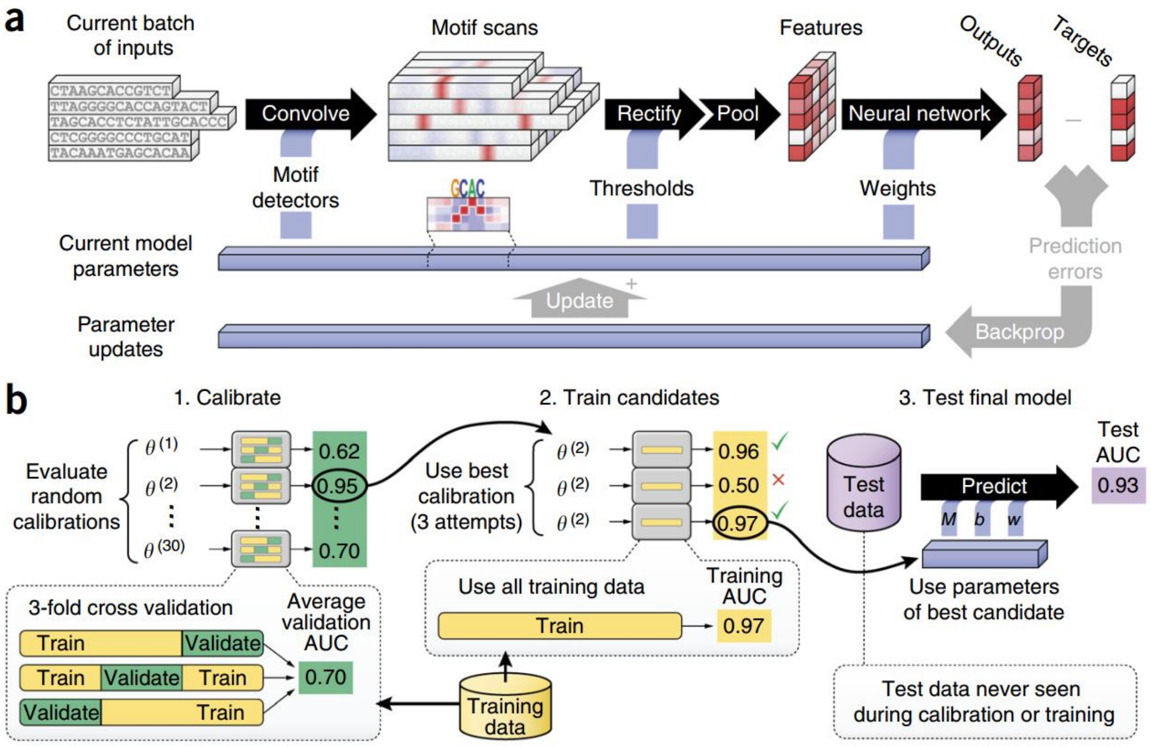


Figure 2 Details of inner workings of DeepBind and its training procedure. (**a**) Five independent sequences being processed in parallel by a single DeepBind model. The convolve, rectify, pool and neural network stages predict a separate score for each sequence using the current model parameters (**Supplementary Notes**, sec. 1). During the training phase, the backprop and update stages simultaneously update all motifs, thresholds and network weights of the model to improve prediction accuracy. (**b**) The calibration, training and testing procedure used throughout (**Supplementary Notes**, sec. 2).

Convolution layer plays a role of motif scan by introducing the motif detector M_k (4 x m matrix like PWM of length m). Before using the convolution layer, the sequence was converted to numerical matrix with pads of "one-of-four" as follows:

$$S_{i,j} = \begin{cases} .25 & \text{if } s_{i-m+1} = \text{N or } i < m \text{ or } i > n-m \\ 1 & \text{if } s_{i-m+1} = j^{\text{th}} \text{ base in (A, C, G, T)} \\ 0 & \text{otherwise} \end{cases}$$

And then passed through the convolution layer:

$$X_{i,k} = \sum_{j=1}^{m} \sum_{l=1}^{4} S_{i+j,l} M_{k,j,l}$$

The next step, the rectification stage will apply the thresholding to the output of convolution layer using the trainable threshold of b and function of max. Next, the pooling will output the maximum or average of rectification outputs. Finally, a fully connected layer will apply to output the final score which will be compared to actual scores

Rectification	Pooling	Fully connected layer
$Y_{i,k} = \max(0, X_{i,k} - b_k).$	$z_k = \max(Y_{1,k}, \dots, Y_{n,k}).$	$p = w_{d+1} + \sum_{k=1}^{d} w_k z_k$

To train the network, the loss was defined with regularization parameters. However, for the PBM data the loss of mean square error was considered. For Chip data, since the label is either 1 indicating the preferable sequence or 0 indicating background, negative log likelihood was considered.

$$\frac{1}{N}\sum_{i=1}^{N} \text{LOSS}\left(f^{(i)}, t^{(i)}\right) + \beta_1 \|M\|_1 + \beta_2 \|W\|_1 + \beta_3 \|w\|_1$$

Loss for PBM data	Loss for Chip-seq data
$\text{MSE}(f,t) = \frac{1}{2}(f-t)^2$	$\text{NLL}(f,t) = -t\log\big(\sigma(f)\big) - (1-t)\log\big(1-\sigma(f)\big)$

The most challenging part of this method is calibration step due to the sensitivity of deep learning to parameters such as the learning rate, the degree of momentum, the mini-batch size, the strength of parameter regularization, and the dropout probability. The calibration step is based on the random sampling of the calibration parameters and doing the 3-fold cross validation on the dataset to find the best parameters. Once the best calibration parameters, the best trained model was used to test dataset.

Energy based methods [5]

Energy-based methods try to reflect the biophysics that underlies TF binding which enables the probability of binding to be calculated for any protein concentration. In addition, contrary to the PWM methods which assumes the independency of binding affinity for each site, the energy-based method can incorporate the dependency assumption. The physical intuition behind this model is that transcription factor–DNA recognition is primarily based on complementarity between the sequence dependent positioning of hydrogen bond donors and acceptors in the grooves of the double helix and those of the amino acids on the surface of the transcription factor. In this model, the probability of a sequence to be bound by the TF ($P(S_i)$ was defined based on association and dissociation constant of TF and the sequence:

$$TF + S_i \overset{k_{on}}{\underset{k_{off}}{\leftrightarrow}} TF{\bullet}S_i \qquad P(S_i) = \frac{[TF{\bullet}S_i]}{[S_i]+[TF{\bullet}S_i]} = \frac{[TF]}{[TF]+K_d} = \frac{1}{1+e^{E_i-\mu}} \qquad P(j) = P(S_i) + (1-P(S_i))P(\bar{S}_i)$$

Where E_i is the difference in free energy between binding to sequence S_i and the reference sequence and μ is the log ratio of free TF concentration and K_d of the reference sequence. $P(j)$ is binding probability to a position j of the probe, with sequence S_i

$$E(S_i) = \sum_{b=A}^{T} \sum_{m=1}^{L} \varepsilon(b,m)S_i(b,m) + \sum_{m=1}^{L-1} \sum_{n=m+1}^{L} \sum_{b=A}^{T} \sum_{c=A}^{T} \varepsilon(b,m,c,n)S_i(b,m,c,n),$$

$e(b,m)$ are the energy contributions of base b at position m, and $\mathrm{S}_i(\mathrm{b,m})$ is an indicator variable with $S_i(b,m) = 1$ if base b occurs at position m of sequence S_i and $S_i(b,m) = 0$ otherwise. In the above equation the dependency of the nucleotide position (dinucleotide contributions) was implemented as the second term. By considering the position of the binding site within a probe, the final binding score was calculated as:

$$F(i) = \sum_{j=1}^{L} P(j)F_{pos}(j)$$

Here, $F_{pos}(j)$ is a position correction factor considering the fact that the position of binding site within a probe significantly influence the signal intensity. Then, using the minimization objective function the parameters of the PWM and μ was calculated by

$$O(\varepsilon,\mu) = \sum_{i} W_i (Y_i - a - cF(i))^2 + \lambda \sum_{b=A}^{T} \sum_{k=1}^{l} \varepsilon(b,k)^2$$

HMM [6]

In this approach, a probabilistic model has been implanted to generate the sequence from two sets: one as background w_b and the other as groups of motifs. The sequence will be defined using a parse or combination of background and motif which are selected based on the transition probability of P_i. Then the probability of generating the sequence given model parameters θ including the P_i and motif set W can be calculated by summing over all the parses of states:

$$Pr(S|\theta) = \sum_{T} Pr(S, T|\theta)$$

And the sequence will be scored based on the log ratio of probability of generating the sequence from the model to the background:

$$F(S,\theta) = \log \frac{Pr(S|\theta)}{Pr(S|\theta_b)}$$

In addition, the generating probability of the sequence $S = \{s_1, s_2, .., s_n\}$ from a PWM of a motif with w_{ij} as probability of sampling j at position i can be calculated as:

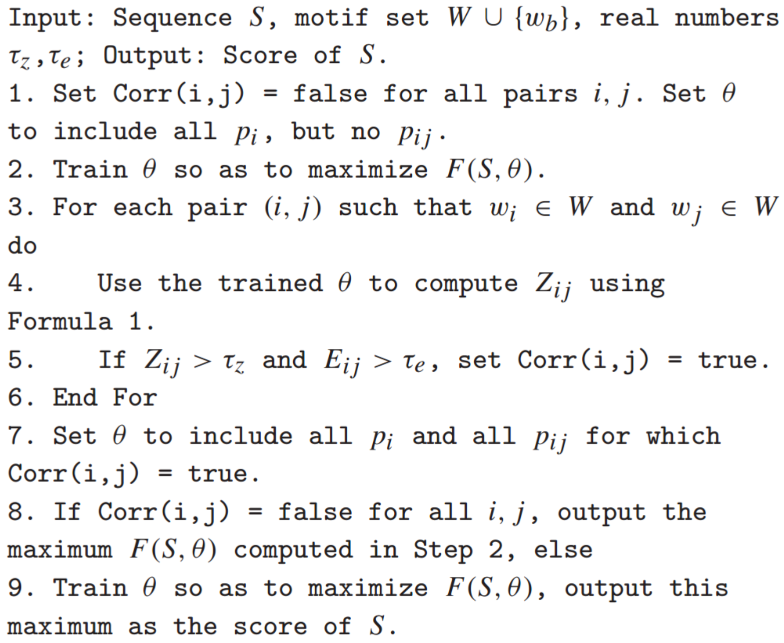
$$Pr(s|w) = \prod_{i=1}^{l} w_{s_i i}$$

The parameters in θ can be calculated using Expectation-Maximization of the F(S, θ).

The more advanced HMM model of motif finding has mitigated the assumption of independency placement of motifs and backgrounds by introducing the correlated transition probability P_{ij} indicating the choice of w_j with probability of P_{ij} if the previous non-background motif was w_i. In this approach, to prevent overfitting, P_{ij} will be added to parameters only if a correlation between w_j and w_i is detected. For correlation detection, sample sequences of X for parameters of the basic HMM were generated and the number of w_j followed by w_i was calculated ($A_{ij}(S)$). The expectation and deviation of the $A_{ij}(X)$ was calculated and Z_{ij} was defined as follows:

$$Z_{ij} = \frac{A_{ij}(S) - E_{ij}}{\sigma_{ij}}$$

Above the specific threshold for E_{ij} and Z_{ij}, the P_{ij} was considered. The refined algorithm is as below:



```
Input: Sequence $S$, motif set $W \cup \{w_b\}$, real numbers
$\tau_z$, $\tau_e$; Output: Score of $S$.
1. Set Corr(i,j) = false for all pairs $i, j$. Set $\theta$
to include all $p_i$, but no $p_{ij}$.
2. Train $\theta$ so as to maximize $F(S, \theta)$.
3. For each pair $(i, j)$ such that $w_i \in W$ and $w_j \in W$
do
4.     Use the trained $\theta$ to compute $Z_{ij}$ using
Formula 1.
5.     If $Z_{ij} > \tau_z$ and $E_{ij} > \tau_e$, set Corr(i,j) = true.
6. End For
7. Set $\theta$ to include all $p_i$ and all $p_{ij}$ for which
Corr(i,j) = true.
8. If Corr(i,j) = false for all $i, j$, output the
maximum $F(S, \theta)$ computed in Step 2, else
9. Train $\theta$ so as to maximize $F(S, \theta)$, output this
maximum as the score of $S$.
```

Algorithm comparison [7]

Weirauch et. al scored the 26 motif finding algorithms on PBM data of 66 mouse TFs. They provided a challenge "DREAM5" by generating the two arrays of PBM data from independent de Brujin sequences of 86 mouse TFs. In DREAM5 challenge, participants were given only one of the PBM data, and they were required to predict the second array which were held back from them. For calibration and testing their algorithm, the participants were provided with PBM data of both array for 20 randomly selected TFs. For 33 TFs, the intensity of array1 and for the remaining 33 TFs the only intensity of array2 were given. The algorithms that were evaluated are listed as below:

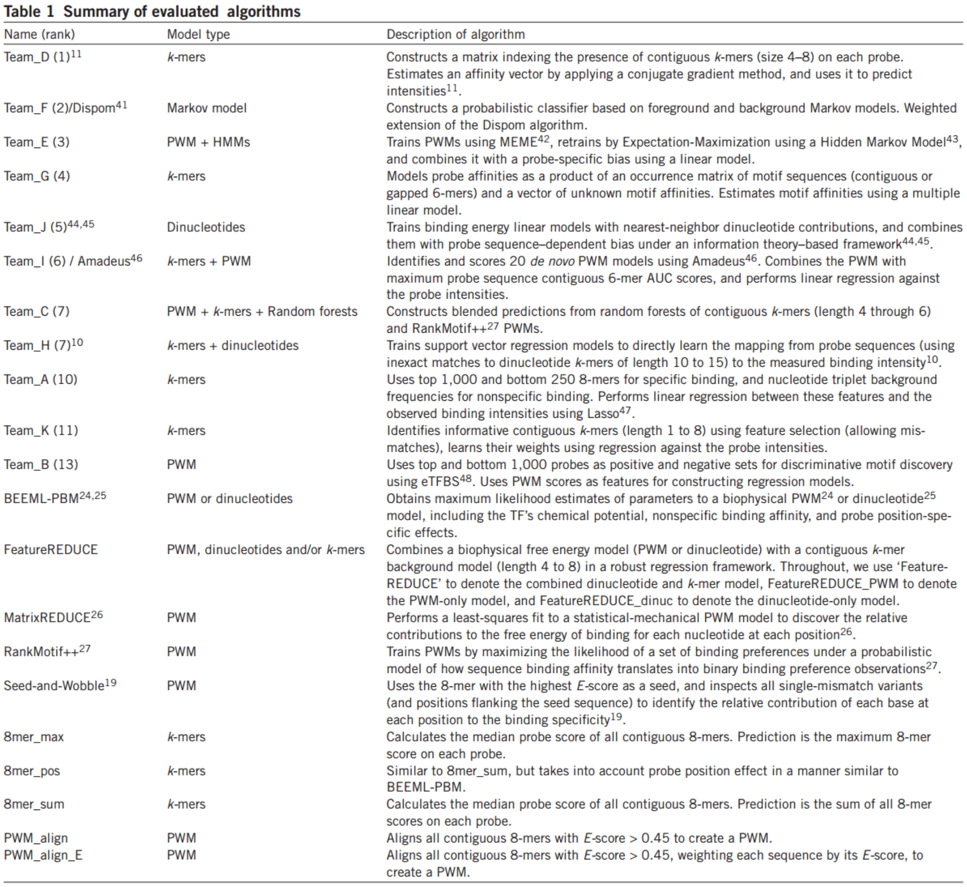


Table 1 Summary of evaluated algorithms

Name (rank)	Model type	Description of algorithm
Team_D (1)[11]	*k*-mers	Constructs a matrix indexing the presence of contiguous *k*-mers (size 4–8) on each probe. Estimates an affinity vector by applying a conjugate gradient method, and uses it to predict intensities[11].
Team_F (2)/Dispom[41]	Markov model	Constructs a probabilistic classifier based on foreground and background Markov models. Weighted extension of the Dispom algorithm.
Team_E (3)	PWM + HMMs	Trains PWMs using MEME[42], retrains by Expectation-Maximization using a Hidden Markov Model[43], and combines it with a probe-specific bias using a linear model.
Team_G (4)	*k*-mers	Models probe affinities as a product of an occurrence matrix of motif sequences (contiguous or gapped 6-mers) and a vector of unknown motif affinities. Estimates motif affinities using a multiple linear model.
Team_J (5)[44,45]	Dinucleotides	Trains binding energy linear models with nearest-neighbor dinucleotide contributions, and combines them with probe sequence–dependent bias under an information theory–based framework[44,45].
Team_I (6) / Amadeus[46]	*k*-mers + PWM	Identifies and scores 20 *de novo* PWM models using Amadeus[46]. Combines the PWM with maximum probe sequence contiguous 6-mer AUC scores, and performs linear regression against the probe intensities.
Team_C (7)	PWM + *k*-mers + Random forests	Constructs blended predictions from random forests of contiguous *k*-mers (length 4 through 6) and RankMotif++[27] PWMs.
Team_H (7)[10]	*k*-mers + dinucleotides	Trains support vector regression models to directly learn the mapping from probe sequences (using inexact matches to dinucleotide *k*-mers of length 10 to 15) to the measured binding intensity[10].
Team_A (10)	*k*-mers	Uses top 1,000 and bottom 250 8-mers for specific binding, and nucleotide triplet background frequencies for nonspecific binding. Performs linear regression between these features and the observed binding intensities using Lasso[47].
Team_K (11)	*k*-mers	Identifies informative contiguous *k*-mers (length 1 to 8) using feature selection (allowing mismatches), learns their weights using regression against the probe intensities.
Team_B (13)	PWM	Uses top and bottom 1,000 probes as positive and negative sets for discriminative motif discovery using eTFBS[48]. Uses PWM scores as features for constructing regression models.
BEEML-PBM[24,25]	PWM or dinucleotides	Obtains maximum likelihood estimates of parameters to a biophysical PWM[24] or dinucleotide[25] model, including the TF's chemical potential, nonspecific binding affinity, and probe position-specific effects.
FeatureREDUCE	PWM, dinucleotides and/or *k*-mers	Combines a biophysical free energy model (PWM or dinucleotide) with a contiguous *k*-mer background model (length 4 to 8) in a robust regression framework. Throughout, we use 'FeatureREDUCE' to denote the combined dinucleotide and *k*-mer model, FeatureREDUCE_PWM to denote the PWM-only model, and FeatureREDUCE_dinuc to denote the dinucleotide-only model.
MatrixREDUCE[26]	PWM	Performs a least-squares fit to a statistical-mechanical PWM model to discover the relative contributions to the free energy of binding for each nucleotide at each position[26].
RankMotif++[27]	PWM	Trains PWMs by maximizing the likelihood of a set of binding preferences under a probabilistic model of how sequence binding affinity translates into binary binding preference observations[27].
Seed-and-Wobble[19]	PWM	Uses the 8-mer with the highest *E*-score as a seed, and inspects all single-mismatch variants (and positions flanking the seed sequence) to identify the relative contribution of each base at each position to the binding specificity[19].
8mer_max	*k*-mers	Calculates the median probe score of all contiguous 8-mers. Prediction is the maximum 8-mer score on each probe.
8mer_pos	*k*-mers	Similar to 8mer_sum, but takes into account probe position effect in a manner similar to BEEML-PBM.
8mer_sum	*k*-mers	Calculates the median probe score of all contiguous 8-mers. Prediction is the sum of all 8-mer scores on each probe.
PWM_align	PWM	Aligns all contiguous 8-mers with *E*-score > 0.45 to create a PWM.
PWM_align_E	PWM	Aligns all contiguous 8-mers with *E*-score > 0.45, weighting each sequence by its *E*-score, to create a PWM.

The evaluation and ranking of the algorithms was based on Pearson correlation between predicted and actual probe intensities (in the linear domain) for those algorithms that reflect the relative preference to a given sequence. The second criterion was the area under the receiver operating characteristic (AUROC) for the algorithms just discriminating the bonded from unbounded sequences. Then, the top -performing algorithm receives 1, and all other were scaled accordingly.

The best performing algorithm was FeatureREDUCE which combines a dinucleotide model and biophysical framework with a k-mer approach. Overall, k-mer and dinucleotide methods outperformed, however, highest PWM methods scored competetavely. The best PWM-based algorithm performs as well as the best *k*-mer–based algorithm for the majority of TFs (**Fig. 2c**), with a median difference of only 0.014. PWM algorithms, in fact, did slightly better than *k*-mer–based algorithms for 18 TFs.

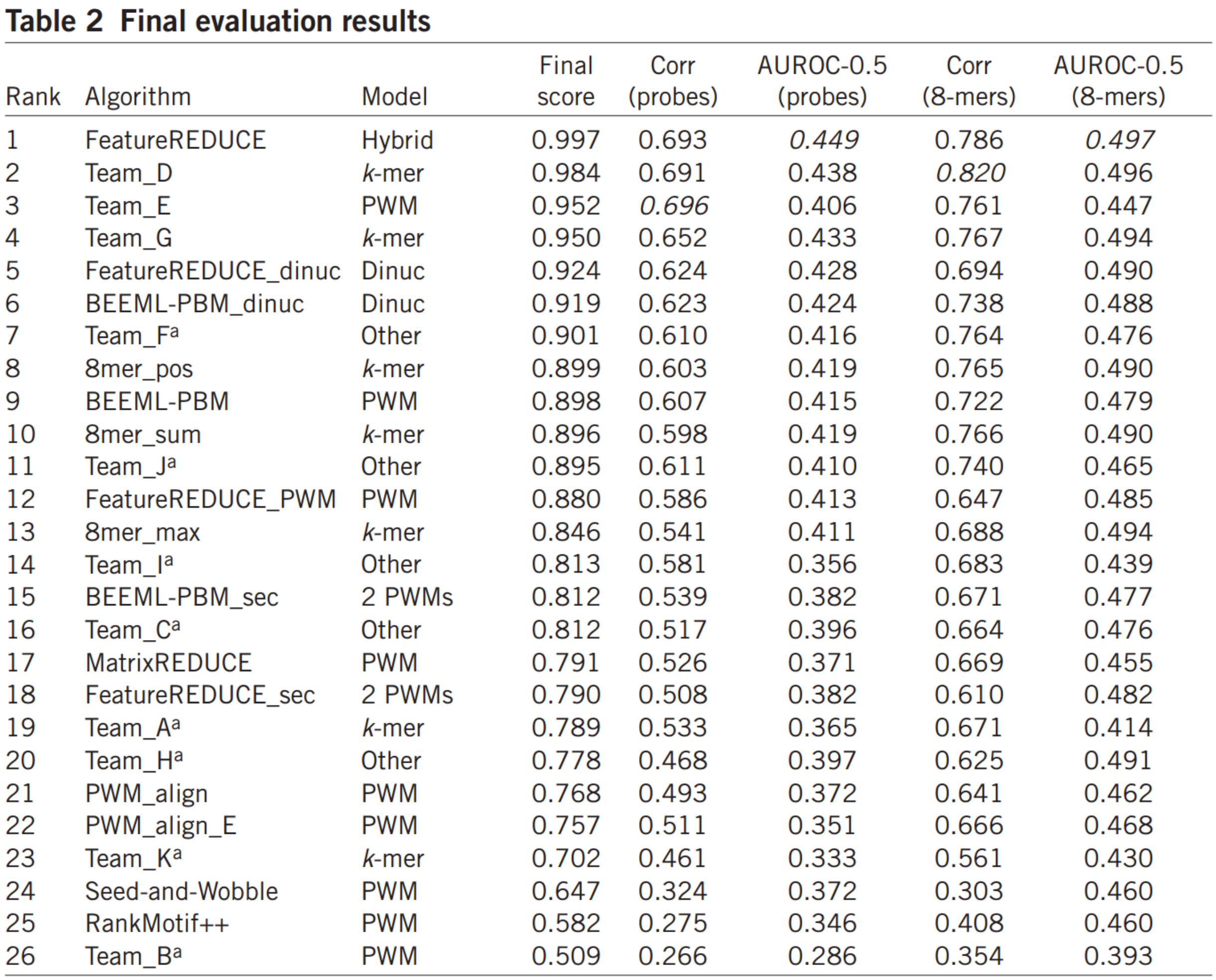
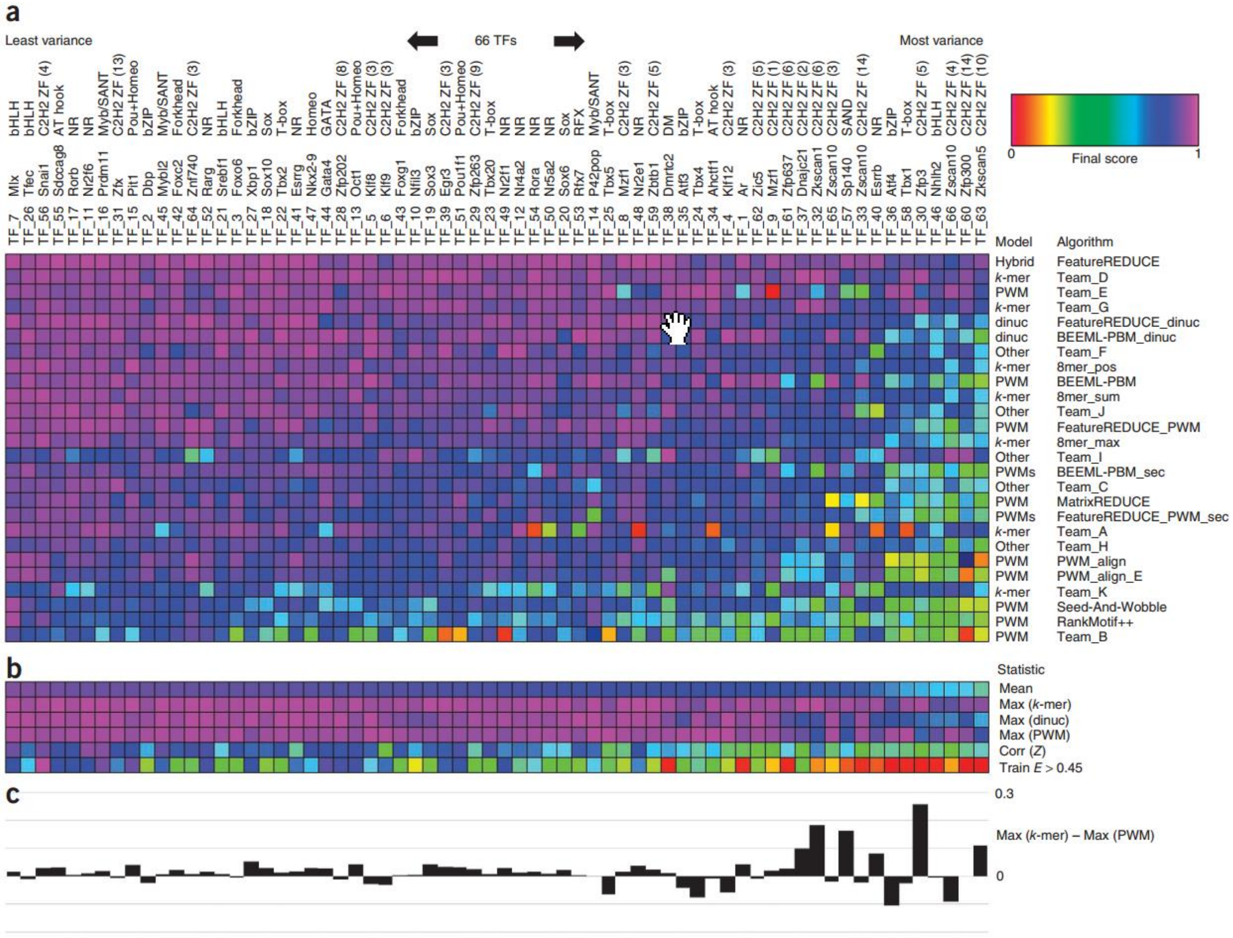


Table 2 Final evaluation results

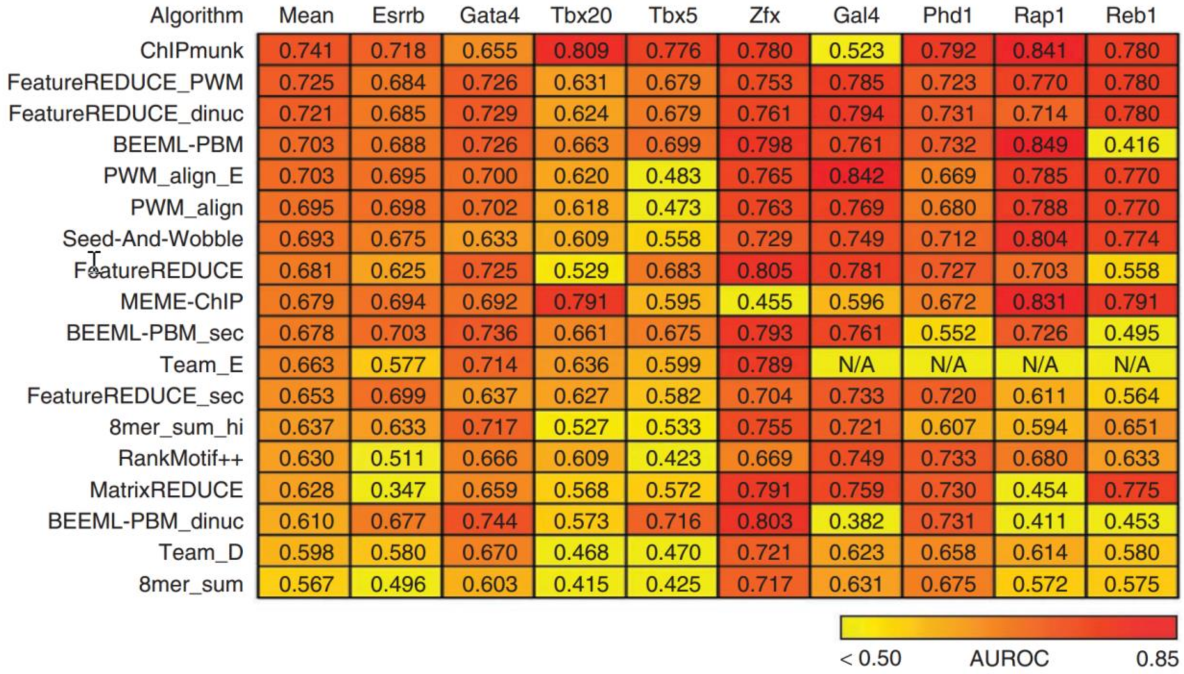
Rank	Algorithm	Model	Final score	Corr (probes)	AUROC-0.5 (probes)	Corr (8-mers)	AUROC-0.5 (8-mers)
1	FeatureREDUCE	Hybrid	0.997	0.693	*0.449*	0.786	*0.497*
2	Team_D	*k*-mer	0.984	0.691	0.438	*0.820*	0.496
3	Team_E	PWM	0.952	*0.696*	0.406	0.761	0.447
4	Team_G	*k*-mer	0.950	0.652	0.433	0.767	0.494
5	FeatureREDUCE_dinuc	Dinuc	0.924	0.624	0.428	0.694	0.490
6	BEEML-PBM_dinuc	Dinuc	0.919	0.623	0.424	0.738	0.488
7	Team_F[a]	Other	0.901	0.610	0.416	0.764	0.476
8	8mer_pos	*k*-mer	0.899	0.603	0.419	0.765	0.490
9	BEEML-PBM	PWM	0.898	0.607	0.415	0.722	0.479
10	8mer_sum	*k*-mer	0.896	0.598	0.419	0.766	0.490
11	Team_J[a]	Other	0.895	0.611	0.410	0.740	0.465
12	FeatureREDUCE_PWM	PWM	0.880	0.586	0.413	0.647	0.485
13	8mer_max	*k*-mer	0.846	0.541	0.411	0.688	0.494
14	Team_I[a]	Other	0.813	0.581	0.356	0.683	0.439
15	BEEML-PBM_sec	2 PWMs	0.812	0.539	0.382	0.671	0.477
16	Team_C[a]	Other	0.812	0.517	0.396	0.664	0.476
17	MatrixREDUCE	PWM	0.791	0.526	0.371	0.669	0.455
18	FeatureREDUCE_sec	2 PWMs	0.790	0.508	0.382	0.610	0.482
19	Team_A[a]	*k*-mer	0.789	0.533	0.365	0.671	0.414
20	Team_H[a]	Other	0.778	0.468	0.397	0.625	0.491
21	PWM_align	PWM	0.768	0.493	0.372	0.641	0.462
22	PWM_align_E	PWM	0.757	0.511	0.351	0.666	0.468
23	Team_K[a]	*k*-mer	0.702	0.461	0.333	0.561	0.430
24	Seed-and-Wobble	PWM	0.647	0.324	0.372	0.303	0.460
25	RankMotif++	PWM	0.582	0.275	0.346	0.408	0.460
26	Team_B[a]	PWM	0.509	0.266	0.286	0.354	0.393

The effect of assumption of the dinucleotide dependencies over the independent inherency of the PWM models were studied by comparing their performance in top two models FeatureREDUCE and BEEML-PBM .These two models were run using both assumptions, and the result showed that there are relatively few cases in which there are bona fide dinucleotide interactions that have a major impact on model performance



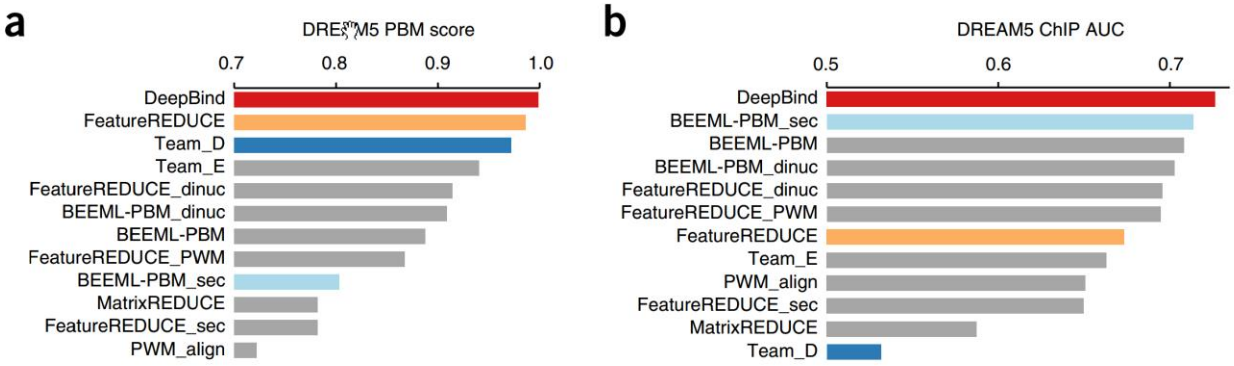
Next, they evaluate those methods on the in-vivo data by first measuring the binding preferences of each TF using training of the PBM data. Then, the trained model was used to to accurately distinguish sequences bound by ChIP-seq and ChIP-exo peaks from control sequence. The trained PWMs were also compared with the PWM trained by running ChIPMunk32 and MEME-Chip33, methods that have been specifically tailored for motif discovery from ChIP-seq data, in a cross-validation setting. The results showed that all PWM-based models can distinquish ChIP-seq and ChIPexo peaks from control sequences to some degree, as evidenced by the fact that the average AUROC scores of all algorithms exceeded the random expectation of

0.5. The algorithms that performed best on the in-vitro data (FeatureREDUCE and Team_D, which both incorporate *k*-mer sequence specificity models) performed the worst in all cases.



Algorithm	Mean	Esrrb	Gata4	Tbx20	Tbx5	Zfx	Gal4	Phd1	Rap1	Reb1
ChIPmunk	0.741	0.718	0.655	0.809	0.776	0.780	0.523	0.792	0.841	0.780
FeatureREDUCE_PWM	0.725	0.684	0.726	0.631	0.679	0.753	0.785	0.723	0.770	0.780
FeatureREDUCE_dinuc	0.721	0.685	0.729	0.624	0.679	0.761	0.794	0.731	0.714	0.780
BEEML-PBM	0.703	0.688	0.726	0.663	0.699	0.798	0.761	0.732	0.849	0.416
PWM_align_E	0.703	0.695	0.700	0.620	0.483	0.765	0.842	0.669	0.785	0.770
PWM_align	0.695	0.698	0.702	0.618	0.473	0.763	0.769	0.680	0.788	0.770
Seed-And-Wobble	0.693	0.675	0.633	0.609	0.558	0.729	0.749	0.712	0.804	0.774
FeatureREDUCE	0.681	0.625	0.725	0.529	0.683	0.805	0.781	0.727	0.703	0.558
MEME-ChIP	0.679	0.694	0.692	0.791	0.595	0.455	0.596	0.672	0.831	0.791
BEEML-PBM_sec	0.678	0.703	0.736	0.661	0.675	0.793	0.761	0.552	0.726	0.495
Team_E	0.663	0.577	0.714	0.636	0.599	0.789	N/A	N/A	N/A	N/A
FeatureREDUCE_sec	0.653	0.699	0.637	0.627	0.582	0.704	0.733	0.720	0.611	0.564
8mer_sum_hi	0.637	0.633	0.717	0.527	0.533	0.755	0.721	0.607	0.594	0.651
RankMotif++	0.630	0.511	0.666	0.609	0.423	0.669	0.749	0.733	0.680	0.633
MatrixREDUCE	0.628	0.347	0.659	0.568	0.572	0.791	0.759	0.730	0.454	0.775
BEEML-PBM_dinuc	0.610	0.677	0.744	0.573	0.716	0.803	0.382	0.731	0.411	0.453
Team_D	0.598	0.580	0.670	0.468	0.470	0.721	0.623	0.658	0.614	0.580
8mer_sum	0.567	0.496	0.603	0.415	0.425	0.717	0.631	0.675	0.572	0.575

The performance of the DeepBind algorithm was also compared by their authors, and ther result showed the outperformance of this motif-finding approach to its counterparts at DREAM5 challenge.



Our Implementations

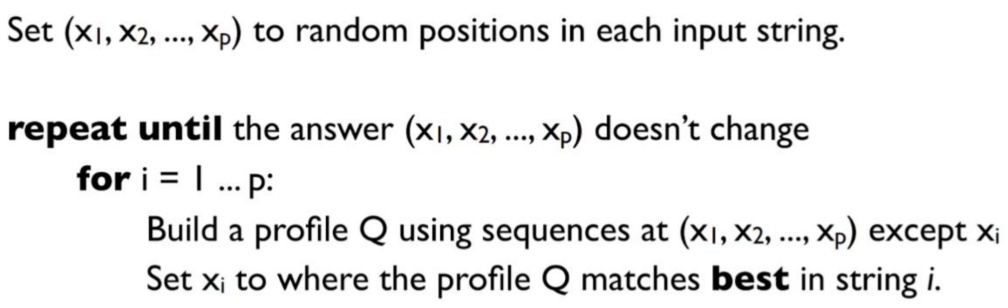
RankMotif++

We implemented the full RankMotif++ algorithm proposed by Chen et al.[3], and tested the model on the PBM dataset from DREAM5 Challenge. To accelerate the development process, we implemented the model with Tensorflow backend and the stochastic gradient descent (SGD) method for optimization.

Gibbs Sampling

Given p strings and a motif length k, our goal is to find p substrings of length k that are most mutually similar. We implemented two versions of the algorithm, according to [9]:

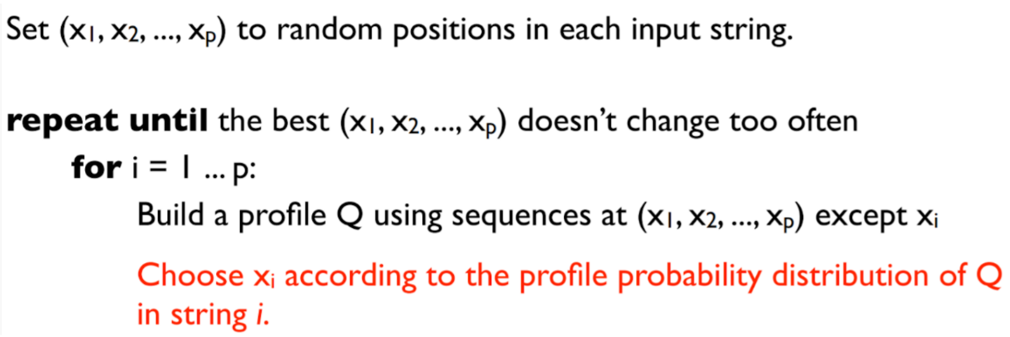
Version 1:



```
Set $(x_1, x_2, ..., x_p)$ to random positions in each input string.

repeat until the answer $(x_1, x_2, ..., x_p)$ doesn't change
    for i = 1 ... p:
        Build a profile Q using sequences at $(x_1, x_2, ..., x_p)$ except $x_i$
        Set $x_i$ to where the profile Q matches best in string i.
```

Version 2:



```
Set $(x_1, x_2, ..., x_p)$ to random positions in each input string.

repeat until the best $(x_1, x_2, ..., x_p)$ doesn't change too often
    for i = 1 ... p:
        Build a profile Q using sequences at $(x_1, x_2, ..., x_p)$ except $x_i$
        Choose $x_i$ according to the profile probability distribution of Q
        in string i.
```

where xi is the starting index of the motif in i-th string. The two versions differs in the way to choose the position of i-th candidate motif xi : version 1 chooses xi that matches the profile matrix Q with highest score, while version 2 adds randomness to the choice of xi , each of which has probability proportional to the matching score. The randomness in version 2 makes it less likely to get stuck at local optimum, and thus more likely to find better motifs.

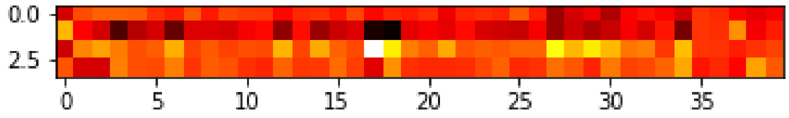
In addition, since the nature of Gibbs sampling is random search, both algorithms need to run multiple times to increase the probability of finding the global optimum.

Experiments and Results

RankMotif++

After a million epochs of training, the model managed to get 64% testing accuracies in predicting TF binding preferences, which is comparable to the accuracies published in the DREAM5 challenge.

The model maintains an underlying Position Weight Matrix (PWM) that is learned through training. After the model is fully trained, the matrix contains distinctive patterns that reflect the structure of the binding site sequences.



Gibbs Sampling

Due to the computational speed constraint, we only used first 30 strings of Dream5 dataset to find motif. We found that smaller k introduces more variance in the motif indices, but the motifs are more uniform. And two versions do not have an obvious difference in results. The results are stored in “CS466_project/gibbs_sampling/results” folder and are named in the format “bestscore/random_k value_number of strings.csv”.

Member Contributions

- **Mu Chen**: Implementing, training, and testing RankMotif++ algorithm;
- **Shuijing Liu**: Implementing and testing Gibbs Sampling algorithm, writing the implementation parts of the report;
- **Erfan Mohagheghian**: Defining the project, reviewing the literature and writing the literature survey part of the report.

GitHub Link

https://github.com/muchen2/CS466_project

References

1. D'Haeseleer, P. How does DNA sequence motif discovery work? *Nature Biotechnology* **24**, 959-961 (2006).

2. Berger, M. F., Philippakis, A., Qureshi, A. M., He, F. S., Estep, P. W., & Bulyk, M. L. (2006). Compact, universal DNA microarrays to comprehensively determine transcription-factor binding site specificities. Nature Biotechnology, 24(11), 1429-1435.

3. Chen, X., Hughes, T. R., Morris, Q. (2007). RankMotif++: a motif-search algorithm that accounts for relative ranks of K-mers in binding transcription factors. Bioinformatics, 23(13), i72-i79.

4. 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.

5. Zhao, Y., Ruan, S., Pandey, M., & Stormo, G. D. (2012). Improved models for transcription factor binding site identification using nonindependent interactions. Genetics, 191(3), 781-790.

6. Sinha, S., van Nimwegen, E., Siggia, E. D. (2003). A probabilistic method to detect regulatory modules. Bioinformatics, 19(Suppl 1), i292-i301.

7. Weirauch, M. T., Cote, A., Norel, R., Annala, M., Zhao, Y., Riley, T. R., ... Hughes, T. R. (2013). Evaluation of methods for modeling transcription factor sequence specificity. Nature Biotechnology, 31(2), 126-134.

8. Dream Challenges. (2014, Apr.). DREAM5 - Transcription-Factor, DNA-Motif Recognition Challenge. [Online]. Available: https://www.synapse.org/#!Synapse:syn2887863/wiki/72185

9. C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald, and J. C. Wootton. *Science*. Bethesda, MD: American Association for the Advancement of Science, 1993.