# Comparison of Computational Methods for Removing Redundancy in Motif Identification Process

**Esther Yu[1], Hanqing Li[2], Rainney Wan[3], Sicheng Ma[4], and Warren Xu[5]**

[1]Computer Science and Mathematics, 2025, yy465
[2]Biological Science, Chemistry, and Computer Science, 2025, hl698
[3]Computer Science and Mathematics, 2025, rw476
[4]Biological Science and Computer Science, 2025, sm2287
[5]Biological Science and Computer Science, 2026, jx62

## ABSTRACT

DNA motifs are conserved patterns typically functioning as the transcription factor binding sites. Computational biologists developed motif discovery algorithms for motif identification from chIP-seq datasets, however these algorithms exhibit redundancy in their motif output. By removing the redundancy, clustering algorithms fosters the identification of transcription factor binding sites. In this project, we reimplemented and benchmarked three popular clustering algorithms: hierarchical agglomerative clustering, k-medoids clustering, and hidden Markov Model (HMM) based clustering. Specifically, benchmarking motif datasets with different features are fetched from JASPAR. Ungapped alignment-based motif distance calculation methods are implemented to quantify motif similarity and find the optimal offset. The three clustering algorithms are evaluated by matching output clusters with correct clustering results from JASPAR,and clustering results are visualized for direct comparison about clustering choices. Generally, pearson correlation distance as position-wise comparison clusters most accurately. Hierarchical clustering performs more accurate than K-medoid clustering, which might attribute to a biased dataset. The randomness of K-medoid clustering can be eliminated by initializing K-medoid clustering with preliminary clustering result. For the HMM method, the novel pipeline was based on the traditional profile-HMM model but with several modifications made to adapt to the ungapped motif matrices dataset. It has a promising potential in its ability to extract profiles from a cluster as well as generalizing results using a probabilistic approach. All methods provides relatively correct clustering on datasets of various scales, while they are all over-sensitive to highly-conserved sequences in the motifs which leads to wrong clustering. Other motif representations and similarity quantification methods might be explored in the future.

**Keywords (minumum 5): DNA motif clustering, alignment-based motif similarity, hierarchical agglomerative clustering, k-medoids clustering, hidden Markov Model clustering, JASPAR, greedy algorithm, position probability matrix**

**Project type: Reimplementation & Benchmark**

**Project repository: https://github.com/HanqingLiLHQ/cs4775_mc**

## 1 Introduction

Eukaryotic transcription is regulated by transcription factors that bind to DNA sequences directly. These transcription factors contain conserved protein substructures, including zinc fingers[1], helix-turn-helix[2], homeodomain[3], that can recognize different conserved DNA sequences correspondingly. Given the sophisticated level of regulation, it can be expected that the transcription start sites are flanked by many of these conserved DNA sequences, appearing recurrently in different genes to recruit the same type of transcription factors. These recurrent patterns of DNA sequences, typically less than 20 bp, are known as regulatory DNA motifs[4].

DNA sequence motifs exhibit significant insight about gene expression and regulation. To perform a site-specific regulatory function, the transcription factor should bind tightly to the DNA sequence via electrostatic interactions and hydrogen bonds with bases in the major groove[5]. By characterizing these binding specificities, biologists may investigate transcription factors from gene sequence, and further co-activator (or co-repressor) analysis would enhance the mechanistic understanding of the overall transcriptome regulations. DNA motifs as transcription factor binding sites also provide insight to cancer[6]. Mutations in these motifs would significantly change the binding affinity of the transcription factor, therefore possibly turning on an oncogene or

deactivating a tumor suppressor gene to induce cancer[7]. For instance, Lou et al. identified a single-nucleotide polymorphism located at transcription factor binding sites that "confers risk of colorectal cancer in the Chinese population[8]."

Molecular biologists develop numerous techniques to detect interactions between DNA motifs and transcription factors. Compared to older techniques like DNase 1 footprinting[9], chIP-seq[10] (chromatin immunoprecipitation sequencing) is powerful in terms of pulling down the DNA sequences directly bound by a protein. As next-generation sequencing techniques[11] significantly promoted biological research, molecular biologists obtained gigantic chIP-seq datasets. Motif discovery algorithms[12] (i.e Gibbs sampling[13]) are applied to these datasets to identify DNA motifs. However, these algorithms usually output redundant motifs[14], making it difficult to identify the specific transcription factors binding to them.

Therefore, it needs to reduce redundancy in the motif identification process through motif clustering[15]. Hierarchical clustering is among the most popular choices for motif clustering. For instance, KanKainen et al. developed a hierarchical clustering based motif matching tool with dynamic programming approach to evaluate alignment of motifs represented by position frequency matrix (PFM) or consensus sequence (CS) [16]; Mondragon et al, on the other hand, compared the position-specific scoring matrices (PSSM) representation of motifs and applied hierarchical clustering[17]. K-medoids clustering, an unsupervised clustering algorithm, was also adapted for motif clustering with motif-wise similarity quantified in an alignment-free convention[18]. Hidden Markov Models (HMM) based clustering algorithms, which were already adapted for protein motif clustering[19], also exhibits prospects for DNA motif clustering.

In this project, we reimplement and benchmark these motif clustering algorithms to compare their effect in removing motif redundancy. We choose position probability matrix (PPM) as the primary representation of the motifs due to its simplicity for understanding. We apply an alignment-based method to quantify motif similarities: we implement column-wise distance calculation methods and slide the motifs against one another to check possible ungapped alignments[18]. While the motif-wise distance can be used in HMM clustering algorithms, we also calculate a distance matrix for each pair of motifs. The distance matrix could then be fed into our hierarchical agglomerative clustering algorithms and our K medoids algorithms. By comparing the performance on benchmarking datasets from JASPAR[20], we will gain insight on the appropriate quantification of motif similarity and the effective clustering algorithms useful for reducing motif redundancy, thereby facilitating a more efficient motif identification process.

## 2 Methods

### 2.1 Data Set

The datasets used are pulled from JASPAR[20], a regularly maintained open-access database storing manually curated transcription factors (TF) binding profiles. The original dataset used are composed of 178 validated motifs from the fungi kingdom. Motifs are fetched from JASPAR dataset as Biopython.motif, and we mainly access their representation as position probability matrix(see Fig. 1). The dataset used includes both gapped and ungapped motifs with various length.

The correct clusters are based on the matrix clustering result from JASPAR generated by RSAT matrix-clustering, a hierarchical clustering algorithm. Specifically, 178 motifs are clustered to 65 clusters, within which there are great variances between the sizes of the clusters(variance of the sizes of clusters equals 12.7). Since fungi have a higher proportion of adenine (A) and thymine (T) nucleotides compared to guanine (G) and cytosine (C) while the proportion of adenine (A) and thymine (T) normally doesn't exceed 0.6, the background nucleotide frequency used in the project is A=0.27 C=0.23 G=0.23 T=0.27.

Besides the validated fungi dataset, to test the performance of the algorithms on gapped motifs, a dataset composed of clusters riched in gapped motifs is manually created from the vertebrate motif database on JASPAR, which is composed of 30 motifs clustered into 6 clusters. Also, in order to study the performance of the algorithms on larger datasets, two dataset composed of both validated and unvalidated motifs from fungi and vertebrates are used. Although unvalidated motifs may not have a biological meaning, they are still usable for testing algorithms, since they are discovered from real ChIP-seq/-exo sequences. The unvalidated fungi dataset is composed of 229 motifs from fungi genome, which is clustered into 61 clusters. The unvalidated vertebrate dataset is composed of 1196 motifs from vertebrate, which is clustered into 321 clusters.

### 2.2 Motif-wise Distance Calculation

#### *2.2.1 Preliminary Motif Distance Calculation without Alignment Using Jensen Shannon Distance*

Jensen-Shannon Distance is based on information theory, which is well-suited to the analysis of biological data. In the context of gene motifs, it can be interpreted as measuring the information lost when one motif distribution is used to approximate another, providing an insight into how much information is unique to each motif. The basic pipeline as follows:

- **Input**: Two motif matrices P and Q that you want to compare

- **Intermediate Step**: Calculate the average distribution $M$, which is the pointwise mean of $P$ and $Q$

$$M = \frac{P+Q}{2}$$

- **Calculation**: Compute the KL divergence of $P$ from $M$ and $Q$ from $M$, and then find the average of these two divergences. KL divergence of $P$ from $M$, for example, is given by

$$KL\_Divergence(P,M) = \sum_{x \in X} P(x) \log \frac{P(x)}{M(x)}$$

  Taking the average divergence would be

$$D_{ave}(P,Q) = \frac{KL\_Divergence(P,M) + KL\_Divergence(Q,M)}{2}$$

- **Result**: The square root of this average $\sqrt{D_{ave}(P,Q)}$ is the Jensen-Shannon distance

Specifically, using log for KL divergence not only make it easier to work with mathematically, but also ensures that the JS divergence can provide a robust and interpretable measure of similarity that is less sensitive to outliers and works well even when there are significant differences in the distributions being compared.

#### *2.2.2 Alignment-Based Motif Distance Calculation*

The preliminary method for distance calculation directly aligns the motifs head-to-head and compute the Jensen-Shannon divergence based on the overlapping region. However, motif matrices (on JASPAR at least) preconditions do not restrict the first position as the beginning site of a "conserved" region, which is usually flanked by non-conserved positions where the nucleotide frequency is close to the background. This might attribute to the different motif-length parameters in different motif-discovery runs. To reduce redundancy, it would be crucial to consider all the possible alignments between motifs in similarity quantification.

In this section, we adapt an alignment-based motif distance calculation method: we first calculate the position-wise distance, then compare motif-wise distance based on all the ungapped alignments. We do not account gap insertions, since a large insertion (or deletion) requires the mutation in both the transcription factor and all its binding sites, which might be impractical evolutionarily. However, we acknowledge that some transcription factors can bind motifs with different gap lengths (still restricted)[21]: for instance, AP-2 family transcription factors contain a basic helix-span-helix protein motif that recognize dyad symmetric DNA motifs via dimerization[22], and the flexibility of protein structure between the DNA binding dimers confer flexibility to the gap length. We will examine our calculated motif distance with AP-2 transcription factors in the results (also see Fig. 4).

**Position-wise Distance Calculation** As each nucleotide position in a PPM is denoted by a 4 dimensional vector, the position-wise distance calculation functions take in two 4D vectors and output a distance that is symmetric and 0 to itself.

- **Euclidean Distance** , one of the most classic distance measurement, has already been applied for protein motif comparison[23]. We appreciate its property to penalize quadratically for evident dissimilarities between conserved nucleotides, which follows the biological evidence that mutating a conserved nucleotide (in cancer) impacts extensively on transcription factor binding affinity [7].

$$D_{l_2}(P_1,P_2) = \sqrt{\sum_{N \in \{A,C,G,T\}} (P_1(N) - P_2(N))^2}$$

- **Pearson Correlation Distance** is also a classic way for characterizing protein motif similarity[24]. We appreciate its calculation related to the entries' numerical difference with the vector's mean value, which is close to the nucleotide background frequency. The difference from the background frequency give the motif their biological significance. (Side note: we set the average entry value as 0.25000001 in our implementation to avoid calculating denominator as 0). Pearson distance equals 1 - PCC , giving it a range of [-1, 1].

$$D_{PCC}(P_1,P_2) = 1 - \frac{\sum_{N \in \{A,C,G,T\}} (P_1(N) - \bar{P})(P_2(N) - \bar{P})}{\sqrt{\sum_{N \in \{A,C,G,T\}} (P_1(N) - \bar{P})^2} \sqrt{\sum_{N \in \{A,C,G,T\}} (P_2(N) - \bar{P})^2}} \tag{1}$$

$$= 1 - \frac{\sum_{N \in \{A,C,G,T\}} (P_1(N) - 0.25)(P_2(N) - 0.25)}{\sqrt{\sum_{N \in \{A,C,G,T\}} (P_1(N) - 0.25)^2} \sqrt{\sum_{N \in \{A,C,G,T\}} (P_2(N) - 0.25)^2}} \tag{2}$$

$$\tag{3}$$

- **Kullback-Leibler Distance** has already applied to PWMs (position weight matrices) for motif comparison[25]. As the original KL divergence is not symmetric, we calculate the KL divergence for both direction and average them. The specifics of KL divergence is already introduced in section 2.2.1.

$$D_{KL}(P_1,P_2) = \frac{KL\_Divergence(P_1,P_2) + KL\_Divergence(P_2,P_1)}{2}$$

- **Jensen-Shannon Distance** is another distance measurement based on KL divergence, and is already introduced in section 2.2.1.

**Motif-wise Distance Calculation** Basically, we slide one motif through the other to evaluate all ungapped alignments. For each alignment, there could be positions from one motif that is not aligned to anything. We treat these alignments in two ways: 1, (overlap strategy) to only consider the overlapping region (Biopython also applies this strategy in dist_pearson comparison of PWMs[26]); 2, (expand strategy) to supply all the unaligned positions with background frequency. We take 0.27, 0.23, 0.23, 0.27 for A,C,G,T respectively as background frequency, regarding that AT is slightly more abundant than GC in many organisms. Either considering overlap or supplying with background frequency fits the two motifs into same dimensions, where we can simply average the position-wise distance to get the motif distance for this particular alignment. We take the alignment with minimal motif distance to return.

After comparing the two strategies (elaborated in results), we choose to refine the expanding strategy with a threshold calculated from alignment distance with 0 overlap (in that case, all the positions from the two motifs are compared to background frequency). If the minimum distance from expanding strategy is smaller than the threshold, this might indicate a meaningful alignment and the algorithm returns the minimum. If the minimum is larger than threshold, we just take the average of all possible alignment distance and return because the transcription factor might not have preference for any of these alignments. (See Algorithms in the appendix for more details)

In short, we choose the motif-expansion strategy with threshold and different column-wise comparison methods for later motif distance (matrix) calculations.

## 2.3 Clustering

### *2.3.1 Hierarchical Agglomerative Clustering*

Hierarchical agglomerative clustering is a tree-construction based clustering algorithm[27, 28]. It builds a tree and then cuts it to give clusters from some similarity measurements, and is commonly applied for motif clustering tasks[16, 17]. Tree-building algorithms including single-linkage, complete-linkage, UPGMA (unweighted pair group method with arithmetic mean), and WPGMA (weighted pair group method with arithmetic mean) are reimplemented and applied in our study because they output trees with same branch length (unlike neighbor-joining [29], another tree-building algorithm popular for evolutionary modeling). Each tree innernode could then be assigned with a height convenient for cutting.

Specifically, based on our calculated distance matrix, in every iteration the algorithm joins the two closest element into one new cluster, then it updates the distance to the new cluster accordingly. Based on the two original elements in the new cluster, A, B, for every other element C, different tree-building algorithms return different distances (the only difference between the four algorithms):

$$D_{single-linkage}(C,(A,B)) = min(D(C,A),D(C,B))$$
$$D_{complete-linkage}(C,(A,B)) = max(D(C,A),D(C,B))$$
$$D_{UPGMA}(C,(A,B)) = \frac{|A|}{|A|+|B|}D(C,A) + \frac{|B|}{|A|+|B|}D(C,B)$$
$$D_{WPGMA}(C,(A,B)) = \frac{1}{2}D(C,A) + \frac{1}{2}D(C,B)$$

With these distance update rules, at the beginning for each iteration, the elements (X,Y) to be merged has the minimal distance D expressed as:

$$D_{single-linkage}(X,Y) = min_{x\in X, y\in Y}D(x,y)$$
$$D_{complete-linkage}(X,Y) = max_{x\in X, y\in Y}D(x,y)$$
$$D_{UPGMA}(X,Y) = \frac{1}{|X|\cdot|Y|}\sum_{x\in X, y\in Y} D(x,y)$$

$$D_{WPGMA}((x_1,x_2),Y) = \frac{1}{2}(D_{WPGMA}(x_1,Y), D_{WPGMA}(x_2,Y))$$

The height of the innernodes are defined as the half distance between its two child nodes. Each iteration reduces the number of elements by 1, and finally we generate a rooted tree with same branch-length. Given the number of clusters we want to produce from this tree, we traverse and split the innernodes starting from the highest to the lowest ones.

Normally, complete-linkage algorithm is preferred for clustering because it typically generates more compact clusters than single-linkage algorithm. UPGMA assumes a constant rate of evolution while WPGMA does not[27]. Evolutionary changes in transcription factor binding motifs might need the co-evolution of the transcription factor itself, therefore might not necessarily possess a constant evolution rate (might be punctuated equilibrium instead of gradualism). Actually, it is beyond doubt to assume evolutionary process in our task because many motifs in the same cluster originates simply from redundancy of motif discovery algorithms. Still, all of these four tree-construction algorithms exhibit prospect in motif clustering.

#### *2.3.2 K-Medoid Clustering*

Given the pre-computed pairwise distance matrix measuring the similarity between motifs in the dataset, naturally K-medoid clustering can be useful as an unsupervised learning method based on pairwise distances. K-medoid is also suitable for large datasets in terms of runtime efficiency; in fact, its performance in runtime is much better than alternative clustering methods when applying to relative large dataset in our evaluation. There're three specific reasoning to use it instead of more traditional K-means clustering:

- K-medoids chooses actual data points as the center of clusters, unlike K-means, which uses the mean. This makes K-medoids more robust to noise and outliers, as medoids are less influenced by extreme values.
- Since the medoids are actual data points (i.e., actual gene motifs), they can be directly interpreted biologically. In this case, we can extract a "representative" motif for each of the final cluster outcome, which may be some animal species.
- Empirically, we also lack the proper methodology to compute mean value of several motifs with good biological rationale, given that they can have different lengths in terms of DNA sequence.

Overall, K-medoid is an useful tool in bioinformatics for clustering gene motifs to discover functionally related groups and to analyze the regulatory mechanisms in genetic sequences. The workflow of K-Medoid Clustering can be described as follows: (find pseudocode for this in section **Algorithms**)

- Select initial cluster centers randomly from all motifs
- Assign each motif M to closest cluster center based on the distance (similarity score) between M and the cluster center

  $$\text{Assign } x_i \text{ to cluster } C_j \text{ if } \forall t, d(x_i,m_j) \leq d(x_i,m_t)$$

  Here, $d(x_i,m_j)$ is the distance from data point $x_i$ to the medoid $m_j$ of the cluster $C_j$, and $m_t$ are medoid of other clusters.
- Re-compute cluster centers by looping through all motifs in the cluster, for each motif M, compute the average distance between M and other motifs in the cluster; set the motif with smallest average distance as new cluster center

  $$m'_j = argmin_{x_i \in C_j} \sum_{x_p \in C_j} d(x_i,x_p)$$

  Here, $m'_j$ is the new medoid for cluster $C_j$, $x_i$ and $x_p$ are points in cluster $C_j$.

The objective function that K-medoids tries to minimize is the total dissimilarity between points and their corresponding medoid, which can be represented as:

$$J = \sum_{j=1}^{k} \sum_{x_i \in C_j} d(x_i,m_j)$$

Here, $J$ is the cost function, $k$ is number of clusters, $C_j$ is set of data points assigned to cluster $j$, and $d(x_i,m_j)$ is the distance from data point $x_i$ to the medoid $m_j$ of cluster $C_j$.
Due to the random nature of initialization of cluster centroids which produced instabilities across different trials, we have developed an improved version of K-Medoid by using result of hierarchical clustering as initialization of clusters, which was shown to produce more accurate and stable result (See **Result** for more detail). The trade-off is increasing runtime which was originally an advantage of using K-Medoid.

#### 2.3.3 HMM Clustering

Given the intrinsic challenges of clustering motifs and the need to efficiently handle large datasets, the Profile Hidden Markov Model (HMM) algorithm emerges as a potent tool for motif identification and clustering. Similar to the K-medoid clustering approach, the HMM algorithm leverages pre-computed pairwise probabilities to capture the relationships between motifs within a dataset during the initial clustering, while the HMM algorithm is utilized in the consequent merging of naïve clusters[19]. The nature of HMM allows for a nuanced representation of motifs, accommodating variations in sequence patterns more effectively than traditional pairwise distance-based methods[19]. Specific reasons to use HMM include:

- HMM utilizes a probabilistic model. By considering the transition probabilities between hidden states, HMM captures subtle correlations within motifs, allowing for a richer understanding of the underlying biological processes driving motif evolution[30].
- HMM excels in handling sequences of varying lengths, a common challenge in biological datasets. Unlike methods relying on fixed-length motifs, the alignment process in the initial greedy clustering of HMM provides a more realistic representation of the diversity present in genetic sequences.
- HMM's probabilistic framework facilitates the incorporation of prior knowledge or additional information, enhancing the interpretability of clustering results.

In summary, the HMM algorithm offers a unique approach to motif clustering based on its probabilistic framework to capture intricate relationships within biological sequences. The workflow of HMM Clustering can be described as follows (find pseudocode for this in section **Algorithms**):

- Start initial cluster from a random motif
- For each current cluster, update the representative matrix. The representative matrix is calculated by:
  Let $M_1, M_2, \ldots, M_k$ be $n \times 4$ matrices after alignment.
  - For each matrix $M_i$ where $1 \leq i \leq k$:
    1. Calculate the sum of each column:

$$\text{column\_sum}(M_i) = \left( \sum_{j=1}^{n} M_{ij1}, \sum_{j=1}^{n} M_{ij2}, \sum_{j=1}^{n} M_{ij3}, \sum_{j=1}^{n} M_{ij4} \right)$$

    2. Calculate the mean of each index across columns:

$$\text{mean\_matrix}(M_i) = \left( \frac{1}{n}\sum_{j=1}^{n} M_{ij1}, \frac{1}{n}\sum_{j=1}^{n} M_{ij2}, \frac{1}{n}\sum_{j=1}^{n} M_{ij3}, \frac{1}{n}\sum_{j=1}^{n} M_{ij4} \right)$$

  - The representative matrix is obtained by repeating these steps for each matrix and then taking the mean matrix across all $M_i$:

$$\text{representative\_matrix} = \frac{1}{k}\sum_{i=1}^{k} \text{mean\_matrix}(M_i)$$

- For each motif, calculate the distance to each representative matrix of clusters. The distance is calculated by the Euclidean method described before.
- If the distances are all greater than a predefined threshold, then the motif starts a new cluster by itself. Otherwise, it joins the cluster with the smallest distance.
- For each cluster, build a Profile-HMM model according to the representative matrix:
  - Let $R$ be a $4 \times n$ representative matrix, and consider HMM with two states: Ontarget and Offtarget. The transition probabilities between states are fixed at 0.5.
  - For the Ontarget state, the emission probability for each column of $R$ is used:

$$\text{emission\_probability}(\text{Ontarget}, j) = \left( \frac{R_{1j}}{\sum_{i=1}^{4} R_{ij}}, \frac{R_{2j}}{\sum_{i=1}^{4} R_{ij}}, \frac{R_{3j}}{\sum_{i=1}^{4} R_{ij}}, \frac{R_{4j}}{\sum_{i=1}^{4} R_{ij}} \right)$$

  - For the Offtarget state, the emission probability is always $[0.25, 0.25, 0.25, 0.25]$.
  - The complete HMM model is defined by the set of states $\{\text{Ontarget}, \text{Offtarget}\}$, transition probabilities, and emission probabilities as specified.
- Use Viterbi Algorithm, compute the Sum-of-Distances score between each of the possible alignments between the two clusters.

- Merge the two clusters with lowest Sum-of-Distances score according to the best alignment.

The algorithm was designed to assess profile-HMM algorithms, yet it was tailored for a different context than the original algorithms. While profile-HMM algorithms typically operate between sequences and clusters, our algorithm was intended for clusters and clusters. Additionally, our assumption of ungapped motifs rendered the insertion/deletion states in profile HMM models unnecessary[31]. In response, we introduced experimental modifications to the classical profile-HMM model. These changes involved replacing the insertion/deletion state with an offtarget state and utilizing predefined column distances algorithms to calculate the odds score between observation and hidden states.

## 2.4 Evaluation

Upon the completion of our motif clustering process, we conducted a comprehensive evaluation to ascertain the biological relevance of our method. This was achieved by contrasting our clustering results with established biological clusters obtained from JASPAR[20]. This comparative analysis was instrumental in validating the effectiveness of our clustering approach against a recognized biological benchmark, providing insights into the functional roles of the identified motifs.

To critically evaluate the effectiveness of our clustering approach, we adopted three recognized scoring metrics, each meticulously applied via the 'sklearn.metrics' module[32] from the scikit-learn library. These metrics are:

- Adjusted Rand Score (ARS)
- Fowlkes-Mallows Score (FMS)
- Normalized Mutual Information Score (NMIS)

The application of these metrics facilitated a consistent and reproducible evaluation of our clustering outcomes. The following exposition details each of these metrics comprehensively.

### *2.4.1 Adjusted Rand Score*

The Adjusted Rand Score (ARS) is a metric for quantifying the similarity between two data clusterings. It evaluates the agreement of two assignments, disregarding permutations. ARS is defined as:

$$\text{ARS} = \frac{\text{RI} - \text{Expected\_RI}}{\max(\text{RI}) - \text{Expected\_RI}} = \frac{\sum_{ij}\binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}]/\binom{n}{2}}$$

In this equation, *RI* stands for the Rand Index, and *Expected_RI* is its expected value under random clustering. Also, $n_{ij}$ represents the count of common elements between clusters $X_i$ and $Y_j$, $a_i$ and $b_j$ denote the sizes of clusters $X_i$ and $Y_j$, respectively, and $n$ is the total number of elements. The ARS is adjusted for chance, allowing for an unbiased comparison of clustering, even with varying numbers and sizes of clusters. It ranges from -0.5 to 1, where 1 indicates perfect agreement and values close to 0 or negative indicate random or discordant clustering.

### *2.4.2 Fowlkes-Mallows Score*

The Fowlkes-Mallows Score is another index used to compare the similarity of two clusterings. For two sets of clusterings $X$ and $Y$, it is defined as:

$$\text{FMS} = \frac{TP}{\sqrt{(TP+FP)(TP+FN)}}$$

Here, $TP$ denotes the number of True Positives, which is the count of pair points that belong to the same clusters in both *true_labels* and *pred_labels*. $FP$ is the count of False Positives, indicating pairs in the same cluster in *true_labels* but not in *pred_labels*, while $FN$ represents the count of False Negatives, referring to pairs in the same cluster in *pred_labels* but not in *true_labels*. The FMS ranges from 0 to 1, where a score of 1 signifies an exact match between two clusterings.

### *2.4.3 Normalized Mutual Information Score*

The Normalized Mutual Information Score (NMIS) is an adjusted measure of the Mutual Information (MI) for comparing the similarity of two clusterings. It is normalized by the average of the entropies of each clustering. It is defined as:

$$\text{NMIS} = \frac{2I(X;Y)}{H(X)+H(Y)}$$

where $I(X;Y)$ is the mutual information between $X$ and $Y$, and $H(X)$ and $H(Y)$ are the entropies of $X$ and $Y$ respectively. A score of 1 indicates a perfect correlation, and 0 implies no mutual information.

### 2.5 Visualization

In our project, we utilized Biopython's WebLogo[26] tool to generate graphical representations of motifs. This approach enabled us to create clear and informative sequence logos for each motif, which visually depict the frequency and conservation of nucleotides within the motifs. These sequence logos were instrumental in understanding the characteristics of each motif, including their variability and conservation patterns.

To enhance our analysis, we arranged the sequence logos of motifs from the same cluster together, facilitating a cohesive and comparative view within each identified cluster. Furthermore, we aligned our motif clusters with those from the JASPAR database. This alignment was primarily based on the similarity of motifs within the clusters: clusters from our analysis were placed in correspondence with JASPAR clusters on the basis that they shared a greater number of motifs in common than with other clusters. This method of alignment allowed for a direct and meaningful comparison between our findings and established data, providing a deeper insight into the similarities and differences in motif distribution and conservation.

## 3 Results

### 3.1 Motif Distance Calculation

As we compare the two alignment strategies(two examples are shown in Fig. 2), we realized that only comparing the overlapping region is erroneous because many times the returned distance is from an 1-position alignment instead of from the correct one. Alignments with too "little" overlap might be meaningless because transcription factors tend to bind to multiple nucleotides. The expanding strategy also assigns shorter distance for alignments with shorter overlap, which is usually returned as motif distance if the two motifs cannot align well themselves. Based on these results, we decide to use the expansion method preferentially, and set a threshold (described in methods) to distinguish meaningful alignments.

Different position-wise distance calculations exhibit different characteristics (Fig. 3). Generally, Euclidean distance has similar results as Jensen-Shannon distance, and Kullback-Leibler distance fluctuates more since it does not include square-root in its calculation as both Euclidean and Jensen-Shannon do. Pearson correlation distance does not necessarily calculate shorter distance as the other three do for alignments with shorter overlap, and it might because Pearson distance takes 0.25 (mean entry) as baseline, which will not privilege background frequencies comparisons as the three other distances do. Despite their difference, they are all capable to identify the optimal alignment, and generally they have the same patterns for distance calculations. Meanwhile, they can also appropriately quantify the similarity between motifs of the same family with variable gap length (Fig. 4). They identify the two possible alignments of the AP-2 dimer binding motif, indicated by their significantly lower distances from these two alignments than from other less meaningful alignments. We therefore apply all of the four position-wise distance calculations in further experiments to explore their clustering preferences.

### 3.2 Scores of Clustering Result

The scores of either the clustering results of K-medoid or hierarchical clustering as well as the HMM clustering on the validated fungi dataset are satisfying. The scores of K-medoid clustering and four subtypes of hierarchical clustering(pairwise single-linkage clustering, pairwise complete-linkage clustering, UPGMA (unweighted pair group method with arithmetic mean), and WPGMA (weighted pair group method with arithmetic mean)) are listed in table 1. ARI and FMI are lower and NMI since the first two are biased by the variance in the sizes of the clusters while NMI is not. The performance of the UPGMA is the highest among all four subtypes of hierarchical clustering and the K-medoid clustering. The performance of the pairwise complete-linkage clustering is the lowest among all five methods. The performances of the hierarchical clustering algorithms except the pairwise complete-linkage clustering are higher than the performance of the K-medoid clustering.

There are four distance calculation methods used in distance-based clustering, which are Euclidean Distance, Pearson Correlation Coefficient Distance, Kullback Leibler Distance, and Jensen Shannon Distance. The performances of hierarchical clustering(UPGMA) and K-medoid clustering using different kinds of distance methods are listed in table 2, which are scored by NMI due to their unbiased characteristics. The NMI score of Pearson Correlation Coefficient Distance is the highest among all four distance calculation methods.

The performances of UPGMA hierarchical clustering using Pearson Correlation Coefficient Distance and K-medoid clustering initialized by the clustering result of the hierarchical clustering on the vertebrate dataset riched in gapped motifs are shown in table 3. The performance of the UPGMA hierarchical clustering using Pearson correlation Distance is the highest, while all other distance methods have lower yet consistent performance(including the K-medoid Clustering) on the gapped dataset.

The results of UPGMA hierarchical clustering using Euclidean and K-medoid clustering initialized by the clustering result of the hierarchical clustering on larger datasets are listed in table 4.

### 3.3 Clustering Result

The clustering result of UPGMA hierarchical clustering using Pearson Correlation Coefficient Distance, the clustering result of K-medoid clustering based on UPGMA hierarchical clustering using Pearson Correlation Coefficient Distance as well as the clustering result of HMM clustering using Euclidean Distance is shown in Figure 10, Figure 11, Figure 12, Figure 13, Figure 14, Figure 15, and Figure 16. We format them in several figures to present all 65 clusters.

## 4 Discussion

Among the four position-wise distance calculation methods we implemented, Pearson Correlation Coefficient distance performs the best (Table 2). This might owe to that PCC calculation is based on 0.25 while other distances do not. 0.25, which is quite close to background frequency, might not exhibit motif-specific features, and it is reasonable to compare the probabilities based on their difference to 0.25. The advantage of PCC is also displayed in Fig. 3, where it is found to be less biased for alignments with little overlap as other calculation methods do. Euclidean distance and Jensen-Shannon distance performs similarly in clustering, as they are found to output similar distance in Fig.3. In our task, Kullback-Leibler distance did not vary too much from its logarithmic alternative, but still not performing as well as PCC distance, which was found as an effective way to model motif similarity [24].

### 4.1 Performance of Hierarchical Clustering Methods on the fungi dataset

The performance of UPGMA is the highest among all four types of hierarchical clustering methods, probably suggesting a constant evolutionary rate of motifs in the fungi kingdom (Table 1). The performance of pairwise single-linkage clustering is the lowest since this method intends to produce stringy long clusters, while most correct clusters are short (many of which only have one motif). Except for the single-linkage method and HMM, all other methods exhibit an NMI score above 0.8, suggesting the validity of our computational pipeline of hierarchical clustering.

### 4.2 Performance of different distance methods on the fungi dataset

On the original fungi dataset, the distance matrix calculated by the Pearson Correlation Coefficient Distance has the best performance over the other three methods when using UPGMA hierarchical clustering and UPGMA-based K-medoid clustering, while the running time of Pearson correlation Distance is much longer than other distance methods.

However, on other datasets, combinations of other distance methods using hierarchical clustering other than UPGMA clustering can have the same or even more correct clustering result than UPGMA hierarchical clustering using Pearson CC Distance. Analyzing the detailed differences between different combinations of clustering methods and distance methods on different datasets are one of the major future points for research.

The performance of our clustering methods on the manually selected gapped dataset is very high. However, the clusters in the gapped dataset all have a highly conserved sequence, which can be biased by the distance methods. New gapped datasets composed of more various clusters with motifs sharing less conserved sequences can be designed or simulated to test the performance of distance methods on gapped locis.

### 4.3 Clustering result of Hierarchical Clustering Methods on the fungi dataset

The clustering results of the UPGMA hierarchical Clustering using Pearson Correlation Coefficient Distance show that the algorithm is biased to highly conserved sequences between motifs. If there are several highly conserved sequences between some otherwise very different motifs, the algorithm will be biased by the highly conserved sequences and neglect other differences between these motifs.[see Figure 5 and 6]Since in correct clusters, motifs within the same cluster might not share a very conserved sequence, such mistakes steal motifs from their correct clusters, leading to more errors. Possible improvement might include to "penalize" more for variance at conserved positions in calculating motif distance. The problem of the highly conserved sequences becomes more serious when it comes to long motifs that don't have nucleotide preferences on the 5' and 3' end of the motif. The distance calculation methods don't give enough weight to the differences in length, since many locals on the 3' and 5' end don't have much information to align, which leads to errors when there are higher conserved sequences between motifs with quite different lengths. (see figure 7)

Since the number of expected clustering is a required parameter in the hierarchical clustering, the algorithm will cluster some wrong clusters, since all motifs of some clusters have been clustered to other clusters due to highly conserved sequences while there are many clusters with only one motif according to the correct clustering. To solve the problem of requiring the expected

number of clusters, the silhouette score can be applied to different clustering results based on different expected cluster numbers, and the optimal expected cluster number with the highest average silhouette score can then be determined.

### 4.4 Clustering result of K-medoid clustering on the fungi dataset

While stability and relatively fast computation are major advantages of K-Medoid clustering, the performance of the K-medoid methods is generally lower than the best combination in the hierarchical clustering of all datasets in terms of scores, and this might be caused by the bias in the database we used since the JASPAR database uses hierarchical clustering which might cause some bias.

The problem of the highly conserved sequence also exists in K-medoid clustering, and it becomes even more severe in some clusters, clustering more motifs with conserved sequences to the wrong clusters. (see figure 8) The algorithm clusters a lot of motifs to wrong clusters due to this problem and creates a lot of incorrect new clusters to fulfill the expected cluster counts. Besides the problem of highly conserved sequences, the method also makes mistakes in clusters with motifs that are gapped at different loci.(see figure 9).

One possible future improvement is to run K-Medoid specifying different cluster numbers when this information is not given. The idea behind this method is to identify a point in the graph of a chosen metric (such as the sum of squared errors or inertia within clusters) against the number of clusters, where the rate of decrease sharply changes. This point resembles an "elbow" in the graph. In other words, after a certain point, adding more clusters doesn't significantly decrease inaccuracies. We could then use the $k$ as pre-defined number of clusters. It's also possible to identify best possible cluster number based on observations of the dataset or results from other clustering models.

### 4.5 Performance of HMM clustering on the fungi dataset

Although the results of the innovative Profile-HMM based pipeline were a little below our expectations as indicated by lower scores in comparison to traditional clustering methods, clear clustering patterns were still seen consistently across the identified clusters. The adaptability of HMM to ungapped motifs might be a factor contributing to its lower performance, as the assumption may not fully align with the characteristics of the original algorithm. Further exploration into fine-tuning HMM parameters, considering alternative probabilistic models, or even integrating additional features to better accommodate the dataset's nuances could be ways for improving the algorithm's performance in motif clustering. Additionally, seeking insights from domain experts and conducting a thorough analysis of the dataset's specific characteristics could provide valuable guidance for refining the HMM-based approach.

### 4.6 Clustering result of HMM clustering on the fungi dataset

The clustering results from the HMM clustering show that it is capable of capturing the general clustering patterns, but falls behind in identifying the nuances between individual motifs, especially when the motif displays similar patterns to more than one clusters. This might be attributed to the assigned off-target penalty score of 0.5 fails to distinguish the differences between the mismatch from distinctly patterned emission states and complexly patterned emission states. To improve this issue, a dynamic penalty score could be used that is assigned based on the deviation of the current emission states and the general off-target emission state of [0.25, 0.25, 0.25, 0.25].

It is noteworthy that the ari-scores and nmi-scores significantly increased after the final number of clusters were fixed at 65 instead of 155, but the fmi-scores decreased after sequential merging. This indicates that our HMM pipeline displays stronger capability in the initial merging between the closely-linked motifs, but is less accurate in the merging process between pre-identified clusters later on. This could possibly be improved by re-implementing a way to calculate the representative ppms of each cluster before merging so that more distinct patterns could come up with a larger weight in the overall ppm rather than an equal weight regardless of the strength of the pattern.

### 4.7 Conclusion

The performance of the hierarchical clustering is generally better than the K-medoid clustering on all datasets, which might due to the biased database. There are great differences between the performances of different combinations of distance methods and clustering methods, and the UPGMA hierarchical clustering using the Pearson CC Distance method provides the most accurate clustering result on our main dataset.

For future tasks of this project, the HMM method needs to be further improved, and a silhouette score analysis can be added to the algorithm to eliminate the dependence on an expected cluster number. To reduce the sensitivity of highly conserved

sequences, penalization on different kind of loci in motifs can be improved in the distance methods. Other motif representations, including PWM (position weight matrix), might also be applied and compared to PPM for performance.

## Author Contributions

This section is required to all project groups with more than a single member. Place group member's name under the appropriate contribution.

- Study design: Hanqing Li
- Coding:
  - Dataset fetching: Warren Xu, Hanqing Li
  - Preliminary Jensen-Shannon motif distance: Rainney Wan
  - Alignment-based position/motif-wise distance: Hanqing Li
  - Hierarchical Clustering: Hanqing Li
  - K-medoids Clustering: Rainney Wan
  - HMM clustering (greedy + profile-HMM based merge): Sicheng Ma
  - Clustering evaluation and visualization: Esther Yu
- Experiments: Warren Xu, Esther Yu, Hanqing Li
- Analyses: Warren Xu, Hanqing Li
- Writing:
  - *Introduction: Hanqing Li*
  - *Methods: All*
  - *Results: Warren Xu, Hanqing Li, Sicheng Ma*
  - *Discussion: Warren Xu, Sicheng Ma, Hanqing Li, Rainney Wan*
  - *Latex debugging: Esther Yu*

## References


**1.** Cassandri M, Smirnov A, Novelli F, Pitolli C, Agostini M, Malewicz M, Melino G, Raschellà G, 2017. Zinc-finger proteins in health and disease. *Cell Death Discovery*, 3(1):17071.

**2.** Aravind L, Anantharaman V, Balaji S, Babu M, Iyer L, 2005. The many faces of the helix-turn-helix domain: Transcription regulation and beyond. *FEMS Microbiology Reviews*, 29(2):231–262.

**3.** Bürglin TR, Affolter M, 2016. Homeodomain proteins: an update. *Chromosoma*, 125(3):497–521.

**4.** D'haeseleer P, 2006. What are DNA sequence motifs? *Nature Biotechnology*, 24(4):423–425.

**5.** Jen-Jacobson L, Engler LE, Jacobson LA, 2000. Structural and Thermodynamic Strategies for Site-Specific DNA Binding Proteins. *Structure*, 8(10):1015–1023.

**6.** Yiu Chan CW, Gu Z, Bieg M, Eils R, Herrmann C, 2019. Impact of cancer mutational signatures on transcription factor motifs in the human genome. *BMC Medical Genomics*, 12(1):64.

**7.** Liu M, Boot A, Ng AWT, Gordân R, Rozen SG, 2021. Mutational processes in cancer preferentially affect binding of particular transcription factors. *Scientific Reports*, 11(1):3339.

**8.** Lou J, Gong J, Ke J, Tian J, Zhang Y, Li J, Yang Y, Zhu Y, Gong Y, Li L, *et al.*, 2016. A functional polymorphism located at transcription factor binding sites, rs6695837 near *LAMC1* gene, confers risk of colorectal cancer in Chinese populations. *Carcinogenesis*, :bgw204.

**9.** Brenowitz M, Senear DF, Shea MA, Ackers GK, 1986. [9] Quantitative DNase footprint titration: A method for studying protein-DNA interactions. In *Methods in Enzymology*, volume 130, pages 132–181. Elsevier.

**10.** Park PJ, 2009. ChIP–seq: advantages and challenges of a maturing technology. *Nature Reviews Genetics*, 10(10):669–680.

**11.** Slatko BE, Gardner AF, Ausubel FM, 2018. Overview of Next-Generation Sequencing Technologies. *Current Protocols in Molecular Biology*, 122(1):e59.

**12.** Liu XS, Brutlag DL, Liu JS, 2002. An algorithm for finding protein–DNA binding sites with applications to chromatin-immunoprecipitation microarray experiments. *Nature Biotechnology*, 20(8):835–839.

**13.** Thompson W, 2003. Gibbs Recursive Sampler: finding transcription factor binding sites. *Nucleic Acids Research*, 31(13):3580–3585.

**14.** Habib N, Kaplan T, Margalit H, Friedman N, 2008. A Novel Bayesian DNA Motif Comparison Method for Clustering and Retrieval. *PLoS Computational Biology*, 4(2):e1000010.

**15.** MacIsaac KD, Fraenkel E, 2006. Practical Strategies for Discovering Regulatory DNA Sequence Motifs. *PLOS Computational Biology*, 2(4):1–10. Publisher: Public Library of Science.

**16.** Kankainen M, Löytynoja A, 2007. MATLIGN: a motif clustering, comparison and matching tool. *BMC Bioinformatics*, 8(1):189.

**17.** Castro-Mondragon JA, Jaeger S, Thieffry D, Thomas-Chollier M, van Helden J, 2017. RSAT matrix-clustering: dynamic exploration and redundancy reduction of transcription factor binding motif collections. *Nucleic Acids Research*, 45(13):e119–e119.

**18.** Broin P, Smith TJ, Golden AA, 2015. Alignment-free clustering of transcription factor binding motifs using a genetic-k-medoids approach. *BMC Bioinformatics*, 16(1):22.

**19.** Krejci A, Hupp TR, Lexa M, Vojtesek B, Muller P, 2016. Hammock: a hidden Markov model-based peptide clustering algorithm to identify protein-interaction consensus motifs in large datasets. *Bioinformatics*, 32(1):9–16.

**20.** Sandelin A, 2004. JASPAR: an open-access database for eukaryotic transcription factor binding profiles. *Nucleic Acids Research*, 32(90001):91D–94.

**21.** Giaquinta E, Fredriksson K, Grabowski S, Tomescu AI, Ukkonen E, 2014. Motif matching using gapped patterns. *Theoretical Computer Science*, 548:1–13.

**22.** Williams T, Tjian R, 1991. Analysis of the dna-binding and activation properties of the human transcription factor ap-2. *Genes Dev*, 5(4):670–682.

**23.** Choi IG, Kwon J, Kim SH, 2004. Local feature frequency profile: A method to measure structural similarity in proteins. *Proceedings of the National Academy of Sciences*, 101(11):3797–3802.

**24.** Pietrokovski S, 1996. Searching Databases of Conserved Sequence Regions by Aligning Protein Multiple-Alignments. *Nucleic Acids Research*, 24(19):3836–3845.

**25.** Roepcke S, Grossmann S, Rahmann S, Vingron M, 2005. T-Reg Comparator: an analysis tool for the comparison of position weight matrices. *Nucleic Acids Research*, 33(suppl$_2$):W438–W441.

**26.** Cock PJA, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, Friedberg I, Hamelryck T, Kauff F, Wilczynski B, *et al.*, 2009. Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423.

**27.** Nielsen F, 2016. *Hierarchical Clustering*, pages 195–211.

**28.** Sokal RR, Michener CD, 1958. A statistical method for evaluating systematic relationships. *University of Kansas Scientific Bulletin*, 28:1409–1438.

**29.** Saitou N, Nei M, 1987. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425.

**30.** Söding J, 2004. Protein homology detection by HMM–HMM comparison. *Bioinformatics*, 21:951–960.

**31.** Yoon BJ, 2004. Hidden Markov Models and their Applications in Biological Sequence Analysis. *Curr Genomics*, 10(6):402–415.

**32.** Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, *et al.*, 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

## Figures

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A: | .2 | .2 | 0 | 0 | 0 | 0 | .9 | .1 | .1 | .1 | .3 | 0 |
| C: | .1 | .6 | 0 | 0 | 0 | 0 | 0 | .4 | .1 | .2 | .4 | .6 |
| G: | 0 | 0 | 1 | 1 | .9 | .9 | .1 | 0 | 0 | 0 | 0 | 0 |
| T: | .7 | .2 | 0 | 0 | .1 | .1 | 0 | .5 | .8 | .7 | .3 | .4 |


**Figure 1. An example of a motif's weblogo and its PPM**. Each column sums to 1, with the four entries denoting the probability of observing the given nucleotide at a specific position.

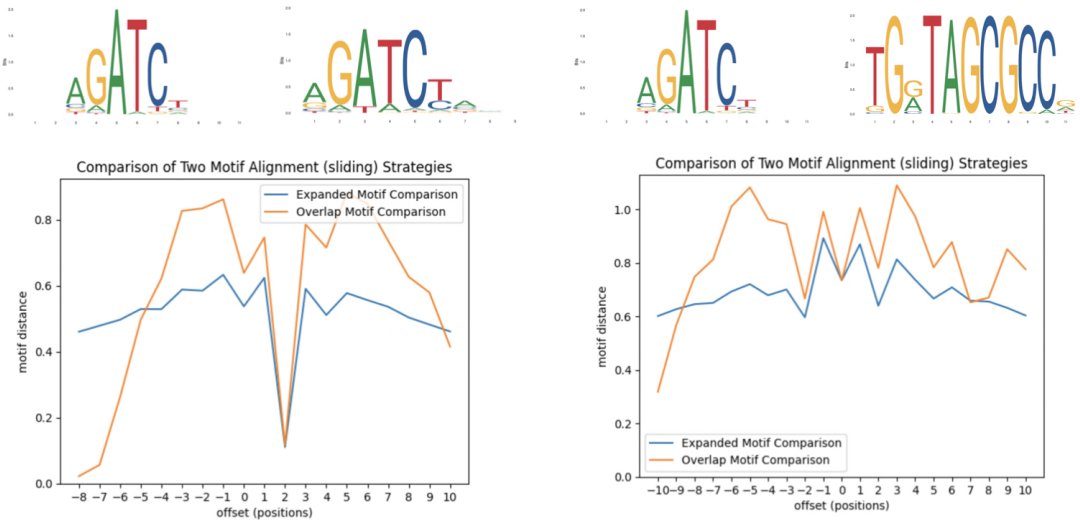


**Figure 2. Comparison of Overlap Alignment and Expansion Alignment illustrated with similar motifs**. Euclidean distance was used for position-wise distance calculation in both alignment examples. **Left**: Two C4 zinc-finger binding motifs (MA0293.1 and MA0301.1, top-left and top-right respectively) which exhibit an optimal alignment at an offset of 2 are compared based on the two alignment strategies. While the minimum distance is at offset 2 for expanded alignment, offset -8 confers the minimum distance to overlap alignment when there are only one position aligned and calculated for position-wise distance. Meanwhile, both alignment strategies tend to have shorter distances for alignments with less overlap except for the optimal alignment offset. **Right**: A C4 zinc-finger binding motif (MA0291.1, top-left) and a Myb/SANT domain factors motif (MA0384.1, top-right) which do not exhibit optimal alignment are compared. While the expansion strategy tend to be more flattened with lower distance at the borders, the overlap strategy has a minimal distance with 1 nucleotide overlap again. Motif matrices are from jaspar[20].

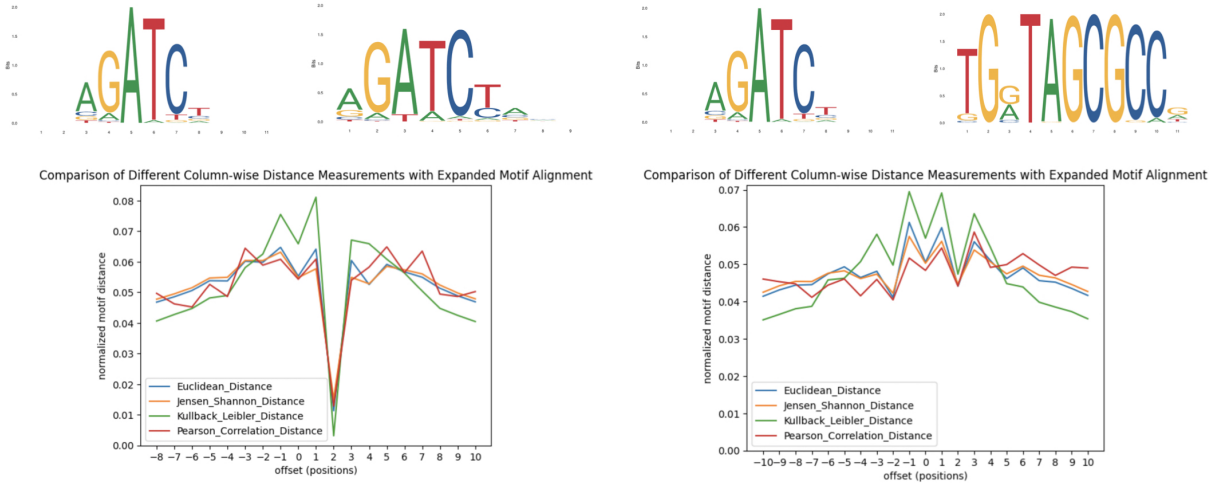


**Figure 3. Comparison of Four Position-wise Distance Calculation for Motif Comparison**. The alignment distances are calculated after motif expansion. The distances are normalized to sum as 1 for all alignments for more straightforward comparison. The background frequencies are 0.27, 0.23, 0.23, 0.27 for A,C,G,T, respectively. **Left**: Two similar motifs (same as Fig. 2) which exhibit an optimal alignment at an offset of 2 are compared based on four position-wise distance calculation methods. All the distance methods successfully find the optimal alignment. **Right**: Two dissimilar motifs (same as Fig. 2) are compared. To compare the distance methods through these two figures, we notice that Euclidean distance tend to have similar results with Jensen-Shannon distance, and Kullback-Leibler distance tend to fluctuate more, possibly because it does not have a square-root operator during the calculation as both Euclidean and Jensen-Shannon do. While these three distance measurements tend to calculate shorter distance for alignment with shorter overlap, the Pearson distance generally exhibits a more flattened line, possibly because it sets the frequency baseline as 0.25 (the mean entry) in the calculation, thus not privileging the alignments expanded with background frequencies.

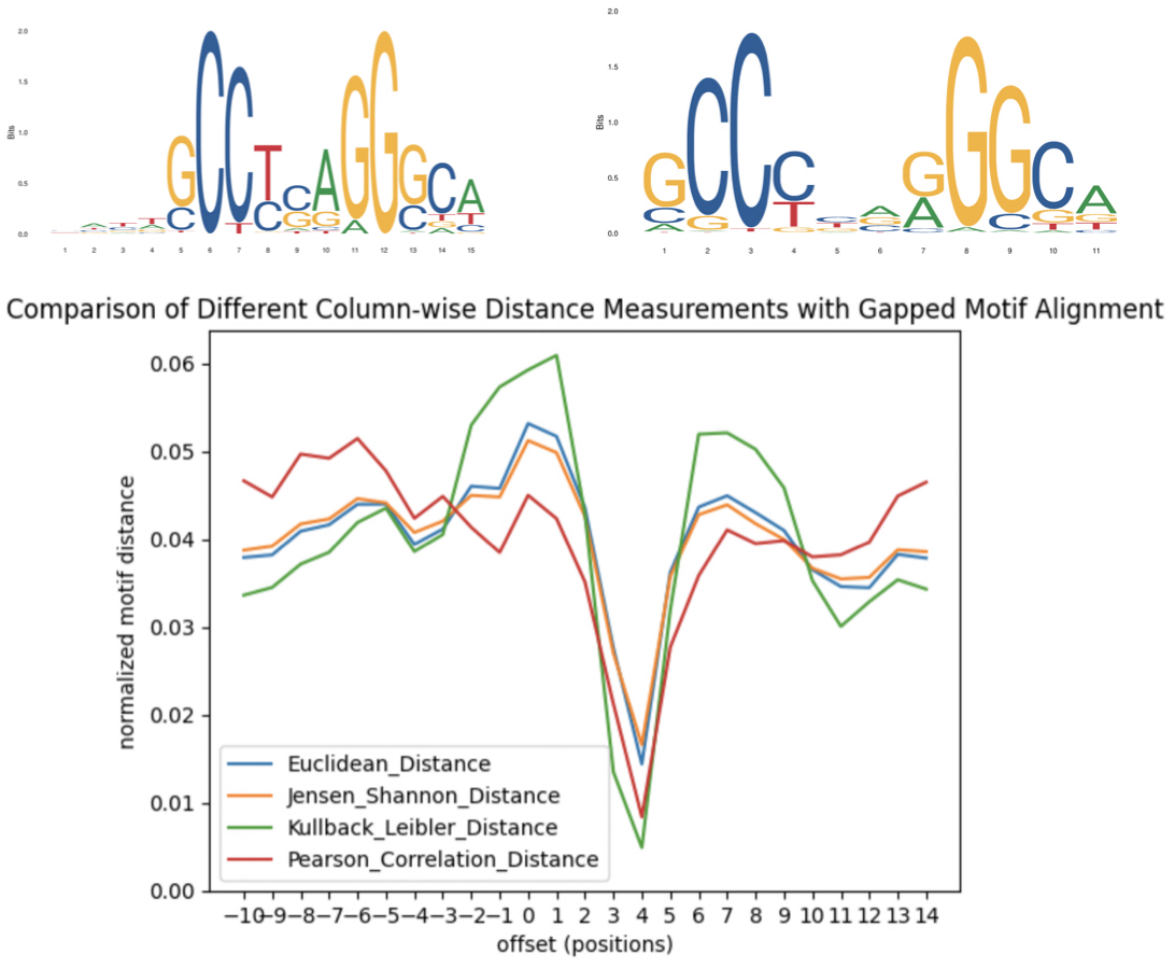


**Figure 4. Performance of Four Position-wise Distance Calculation with Expanded Alignments on Gapped Motif with Variable Gap Length**. Two motifs from AP-2 family of transcription factors with variable gap length (MA0003.2, MA0524.3, at top-left and top-right respectively) are studied for their alignments. As AP-2 family transcription factors recognize palindromic repeats[22], these two motifs both have conserved CC and GG nucleotides separated by 3 or 4 nucleotides that presumably not bound by the transcription factor. Offset 3 would align the two G nucleotides, while offset 4 would align the two C nucleotides. All our four distance calculation methods choose offset 4 as the best alignment, while they also detect another possible alignment at offset 3. The distance calculated at these offsets are much less than the average, indicating that our motif distance calculation method might still recognize the similarity between gapped motifs with different gap length. Motifs matrices and logos are from JASPAR[20].

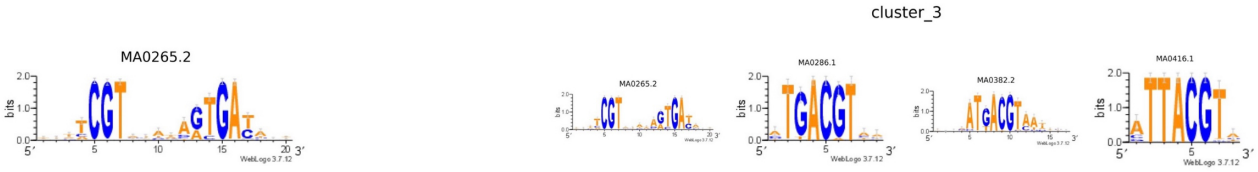


**Figure 5.** This is part of the cluster result of the Fungi Dataset. The cluster on the left is the correct cluster, while the cluster on the right is the predicted cluster. The hierarchical algorithm is biased by the highly conserved sequence "CGT", leading to clustering wrong motifs together

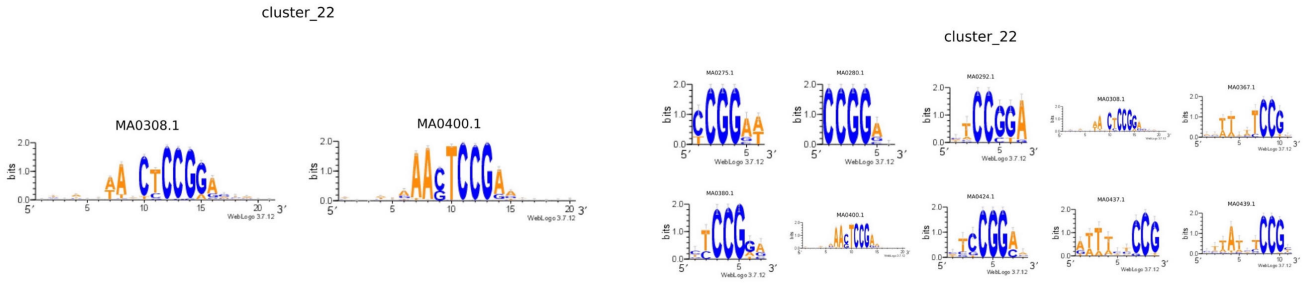


**Figure 6.** This is part of the cluster result of the Fungi Dataset. The cluster on the left is the correct cluster, while the cluster on the right is the predicted cluster. The hierarchical algorithm is biased by the highly conserved sequence "CCGG", leading to clustering wrong motifs together

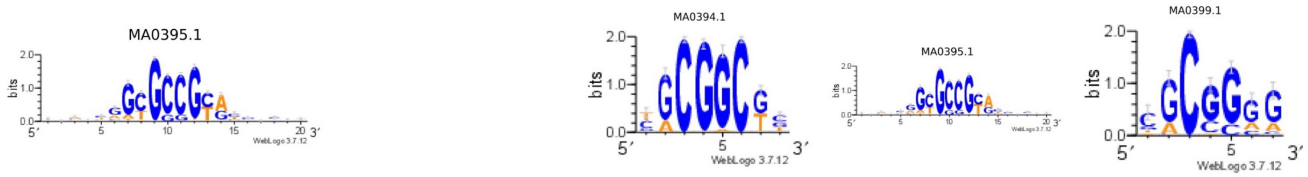


**Figure 7.** This is part of the cluster result of the Fungi Dataset. The cluster on the left is the correct cluster, while the cluster on the right is the predicted cluster. The hierarchical algorithm is biased by highly conserved CGs sequences, and under-weigh the differences between the lengths of the motifs

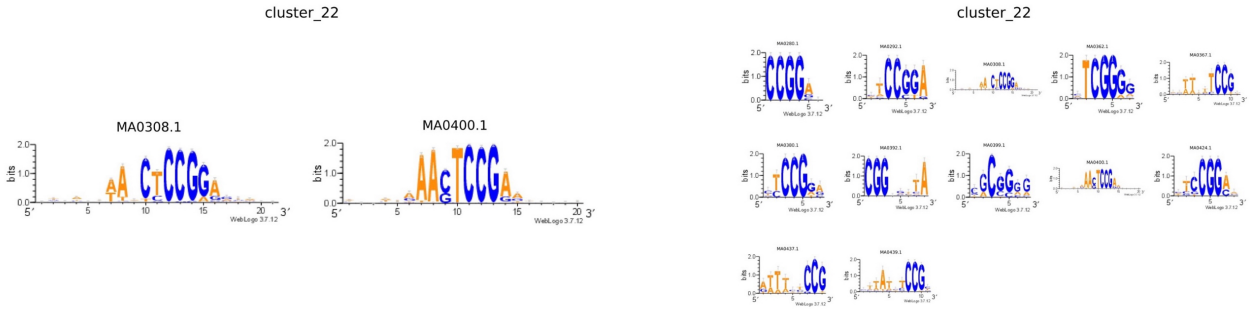


**Figure 8.** This is part of the cluster result of the Fungi Dataset. The cluster on the left is the correct cluster, while the cluster on the right is the predicted cluster. The algorithm is biased by highly conserved CGG and CCG sequences. The K-medoid algorithm clusters two extra motifs with the conserved sequences to the wrong cluster

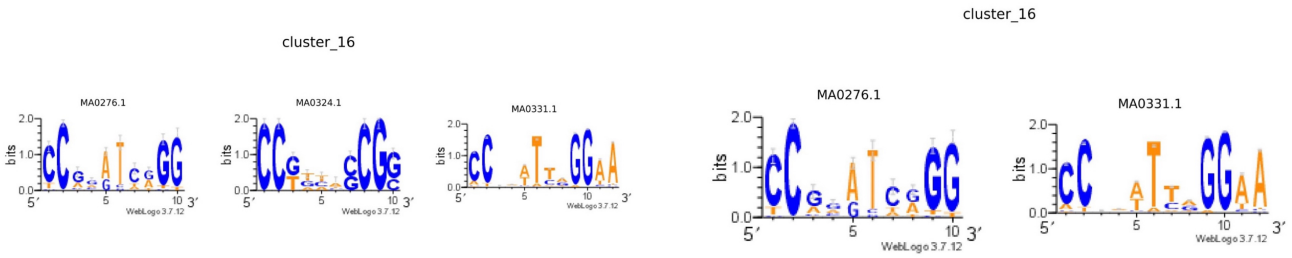


**Figure 9.** This is part of the cluster result of the Fungi Dataset. The cluster on the left is the correct cluster, while the cluster on the right is the predicted cluster. MA0276.1 and MA0331.1 have gapped pattern on the third and forth loci, while MA0324.1 has gapped pattern on the fifth and sixth loci. K-medoid algorithm fail to cluster MA324.1 with MA0276.1 and MA0331.1

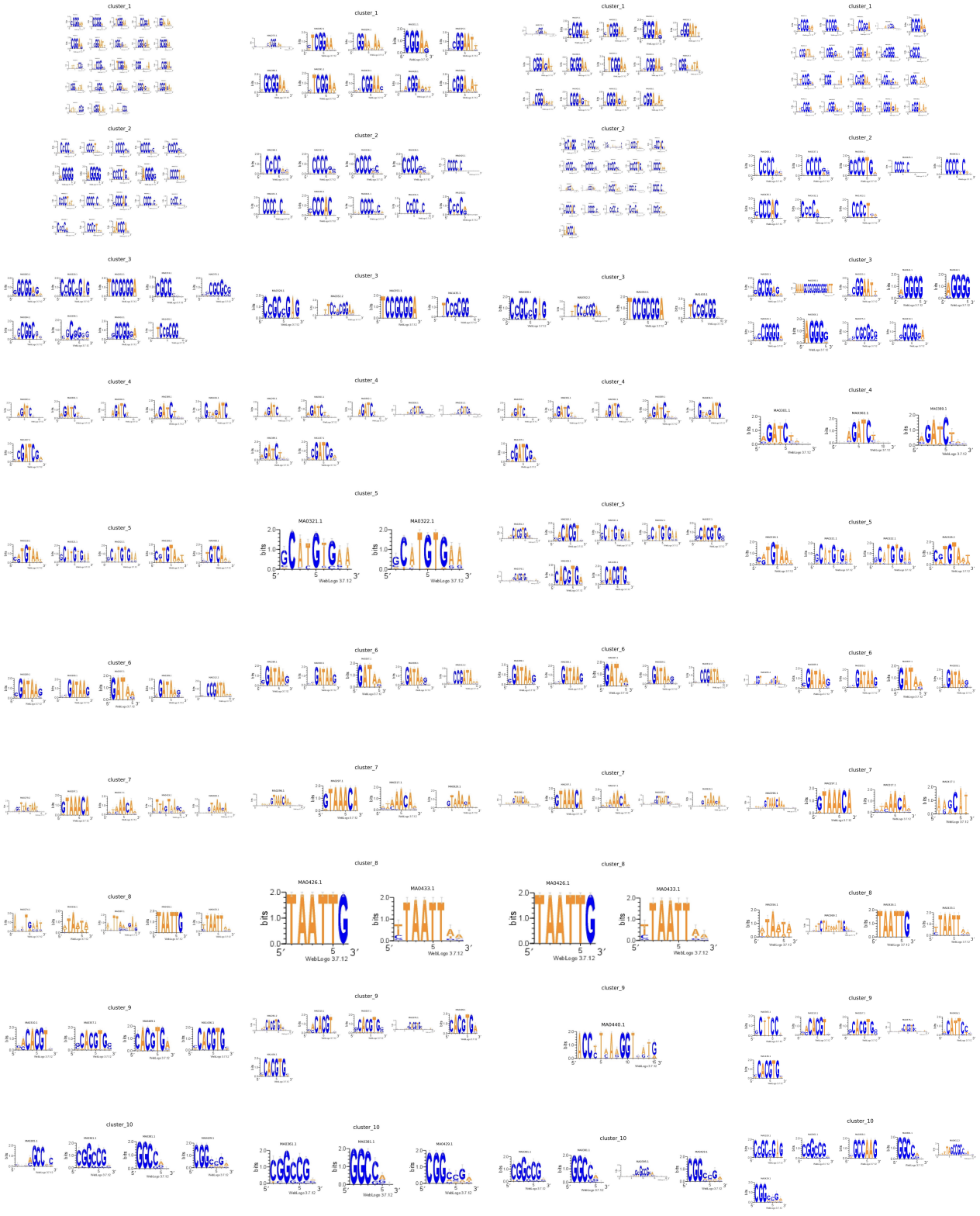

**Figure 10.** Clusters 1 - 10. Left 1: JASPAR Clustering. Left 2: Hierarchical Clustering. Right 2: K-Medoids Clustering. Right 1: HMM Clustering.

**Figure 11.** Clusters 11 - 20. Left 1: JASPAR Clustering. Left 2: Hierarchical Clustering. Right 2: K-Medoids Clustering. Right 1: HMM Clustering.

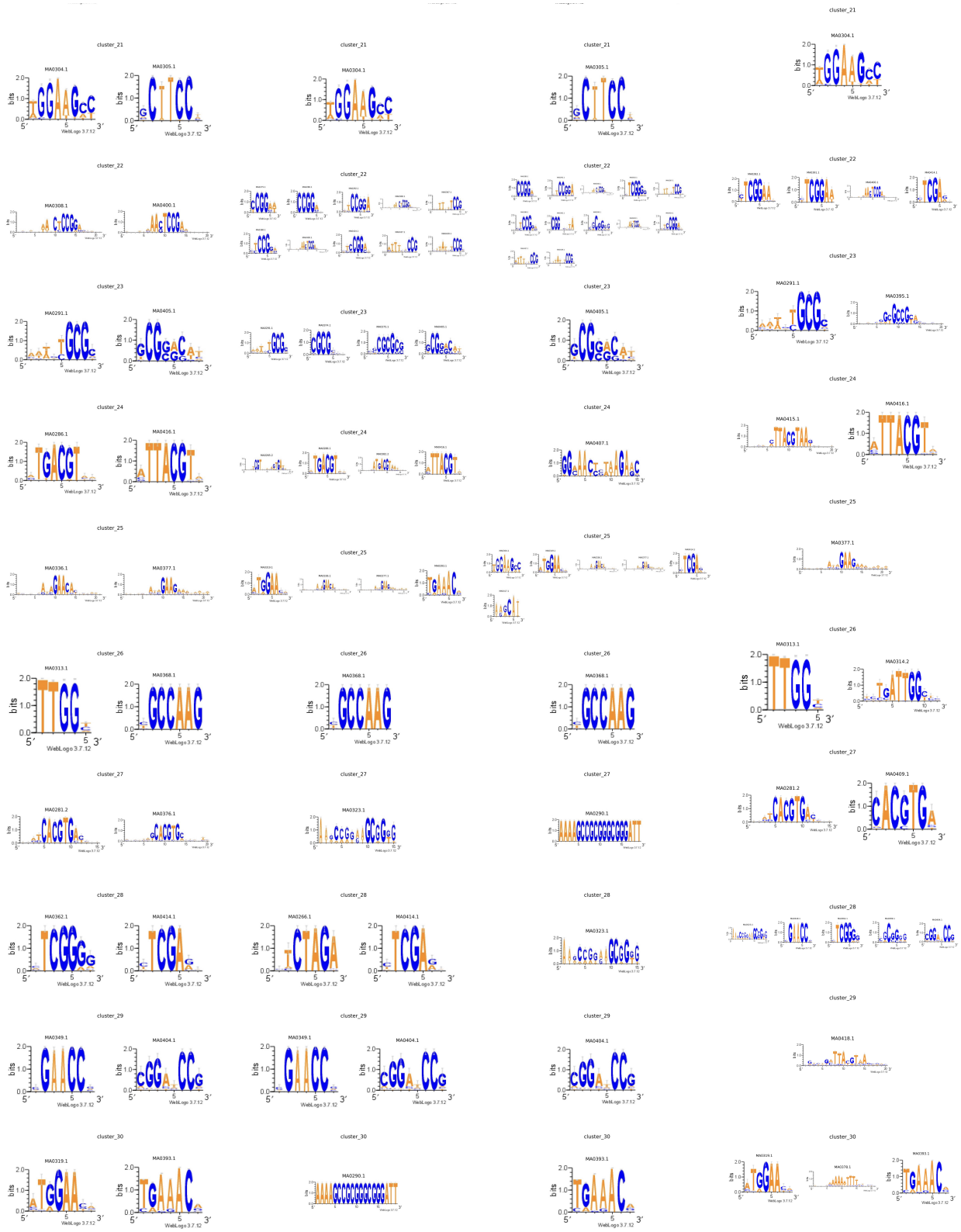

**Figure 12.** Clusters 21 - 30. Left 1: JASPAR Clustering. Left 2: Hierarchical Clustering. Right 2: K-Medoids Clustering. Right 1: HMM Clustering.

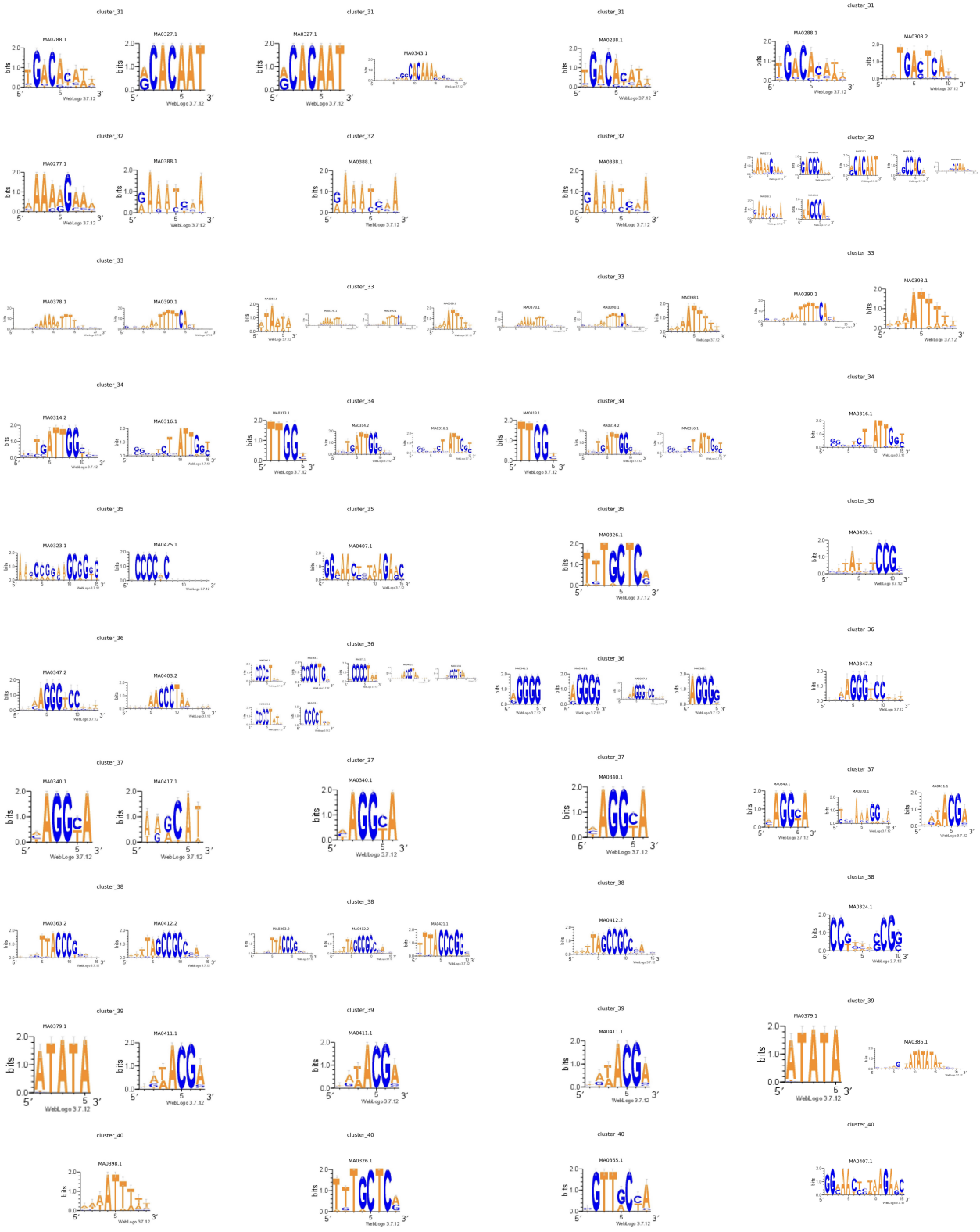

**Figure 13.** Clusters 31 - 40. Left 1: JASPAR Clustering. Left 2: Hierarchical Clustering. Right 2: K-Medoids Clustering. Right 1: HMM Clustering.

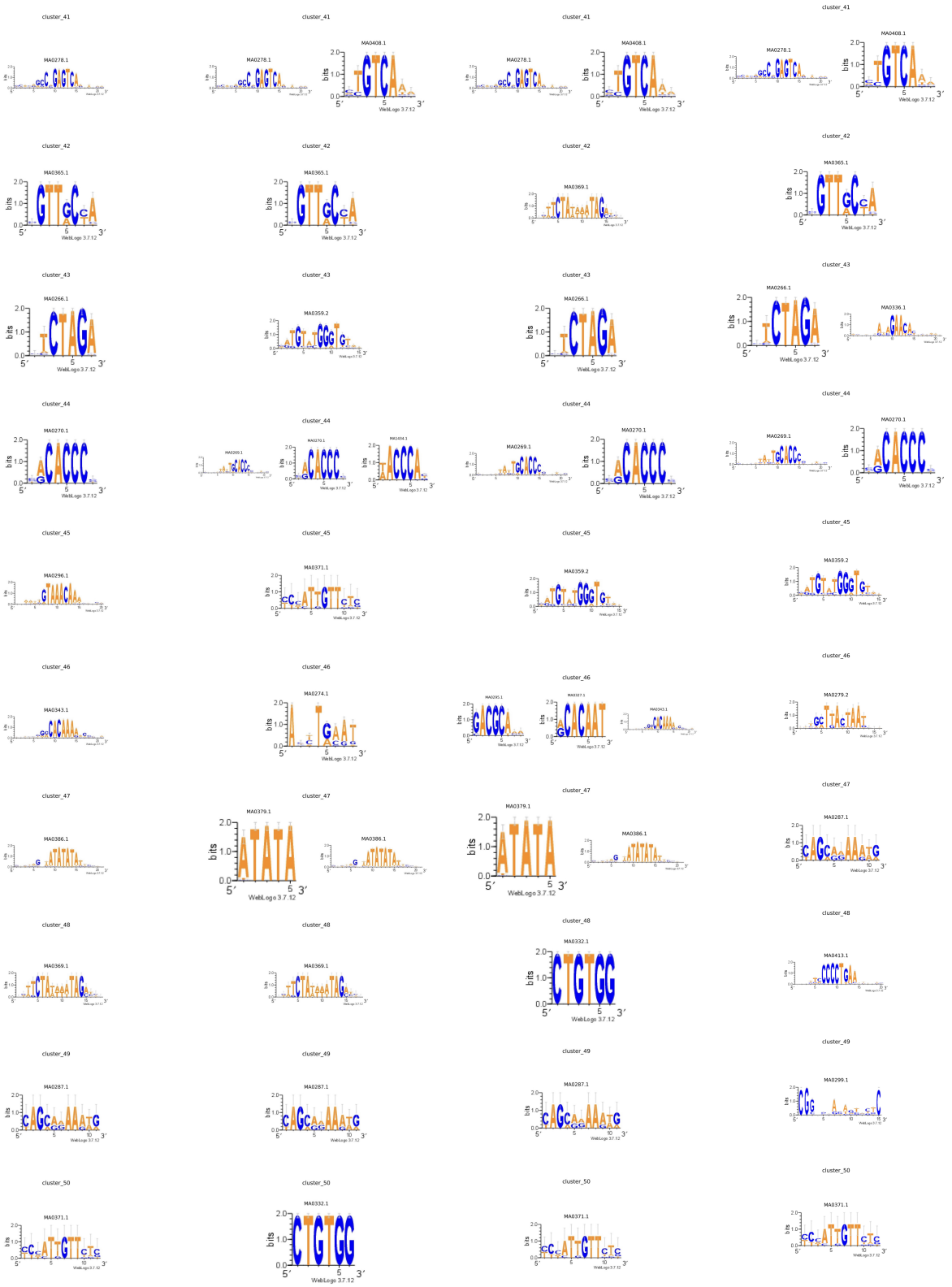

**Figure 14.** Clusters 41 - 50. Left 1: JASPAR Clustering. Left 2: Hierarchical Clustering. Right 2: K-Medoids Clustering. Right 1: HMM Clustering.

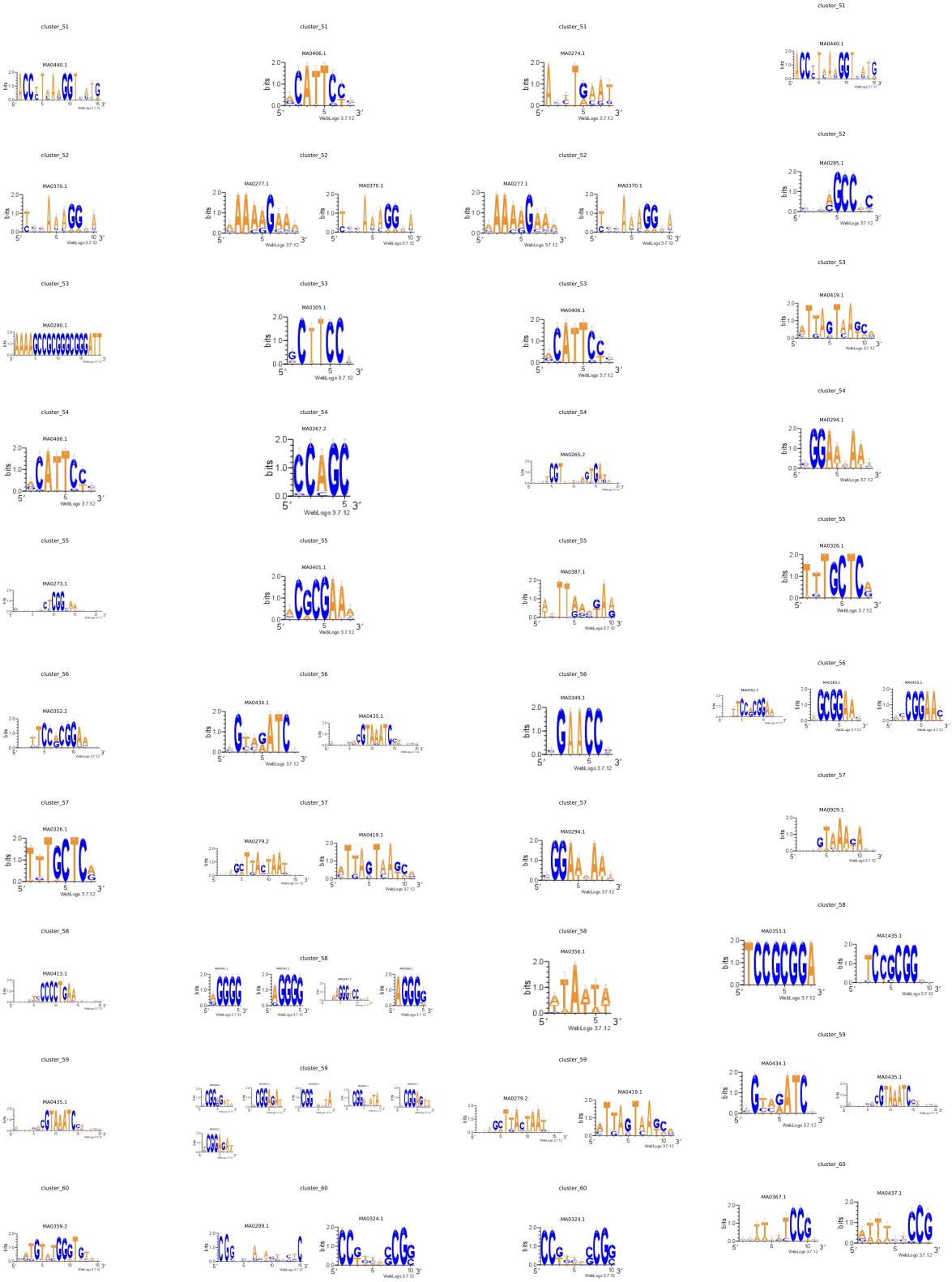

**Figure 15.** Clusters 51 - 60. Left 1: JASPAR Clustering. Left 2: Hierarchical Clustering. Right 2: K-Medoids Clustering. Right 1: HMM Clustering.

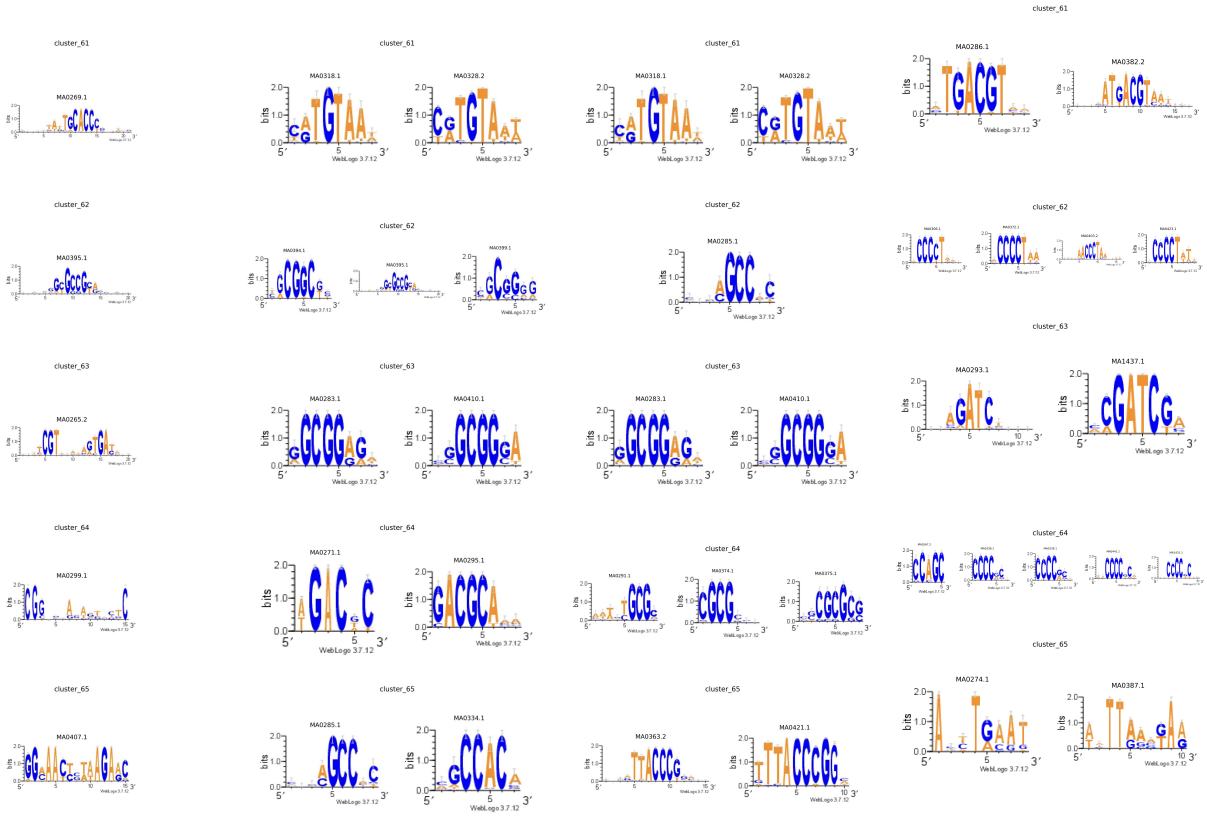

**Figure 16.** Clusters 61 - 65. Left 1: JASPAR Clustering. Left 2: Hierarchical Clustering. Right 2: K-Medoids Clustering. Right 1: HMM Clustering.

## Algorithms

### 1 Alignment-based Motif-wise Distance Calculation

**Algorithm 1** MotifDistance(pos_dist, ppm1, ppm2, align_method, bg = [0.27, 0.23, 0.23, 0.27], average = np.mean)

Initialize $short\_dist = \infty$
Initialize $average\_dist = 0$
**for** all possible alignments of $ppm1$,$ppm2$ with overlap > 0 **do**
    **if** $align\_method =$ "overlap" **then**
        Cut ppm1, ppm2 to their overlapping region
    **end if**
    **if** $align\_method =$ "expand" **then**
        Expand ppm1, ppm2 to contain each other's unoverlapping region with bg
    **end if**
    distance = average($pos\_dist$(pos1, pos2) for pos1, pos2 in ppm1, ppm2)
    Update $short\_dist$, $average\_dist$
**end for**
**if** $align\_method =$ "overlap" **then**
    **return** $short\_dist$
**end if**
**if** $align\_method =$ "expand" **then**
    $threshold =$ average($pos\_dist(pos1, pos2)$ for $pos1, pos2$ in $[ppm1, \text{len}(ppm2) \times bg], [\text{len}(ppm1) \times bg, ppm2]$)
    **if** $short\_dist \leq threshold$ **then**
        **return** $short\_dist$
    **end if**
    **return** $average\_dist$
**end if**

## 2 Hierarchical Agglomerative Clustering

**Algorithm 2** Hierarchical Clustering(distance_matrix,num_clusters,distance_update)

**for** iter in (#motifs - 1) **do**
    Search for *min_distance* between *ele*1 and *ele*2 in *distance_matrix*
    Set new inner node *ele*3 at height *min_distance*/2 with *ele*1, *ele*2 as children
    Update *distance_matrix* to remove *ele*1 and *ele*2 and add *ele*3 according to *distance_update*
**end for**
Initialize active_nodes as [*root*]
**for** iter in (*num_clusters* - 1) **do**
    Remove *node* highest in *active_nodes*
    Add *node*'s two child nodes to *active_nodes*
**end for**
**for** *inner_node* in *active_nodes* **do**
    Degrade *inner_node* and get motifs into a cluster
**end for**

## 3 K-medoids Clustering

**Algorithm 3** KMedoids(dist_mat, num_clusters, max_iter)

Initialize $centroids[num\_clusters]$ randomly
Initialize $clusters = []$
Initialize $prev\_centroids = []$
**for** iteration from 1 to $max\_iter$ **do**
    Set $clusters = []$ for each cluster
    **for** each data point $x_i$ in $dist\_mat$ **do**
        Find the closest centroid $x_j$ from centroids to $x_i$
        Append $x_i$ to the corresponding cluster of $x_j$ in $clusters$
    **end for**
    $prev\_centroids = centroids$
    $centroids = []$
    **for** each $C_j$ in $clusters$ **do**
        **if** $C_j \neq []$ **then**
            $min\_avg\_dist = \infty$
            **for** each motif $x_i$ in $C_j$ **do**
                Calculate $avg\_dist$ as the average distance from $x_i$ to other motifs in the $C_j$
                **if** $avg\_dist < min\_avg\_dist$ **then**
                    $min\_avg\_dist = avg\_dist$
                    $centroid$ of $C_j = x_i$
                **end if**
            **end for**
            Append $centroid$ of $C_j$ to $centroids$
        **end if**
    **end for**
    **if** $centroids = prev\_centroids$ **then**
        Result converged, break from loop
    **end if**
**end for**
Return $clusters$

## 4 HMM Clustering - Part 1

**Algorithm 4** Greedy Naïve Clustering(motifs)

Initialize *threshold* = Predefined threshold
Initialize *clusters* = []
*clusters*[0] = first motif from list
**while** iteration through unsorted motif **do**
    *clus* = id of the best cluster
    *min_dist* = ∞
    Test distance on the new matrix candidate to each existing cluster, Attempt to find minimum distance smaller than threshold value
    **if** no distance under threshold was found: **then**
        Create new cluster
        *clusters*[*curr_clus_id*] =motif being iterated
        *curr_clus_id*+ = 1
    **else**
        Adds motif being iterated to the identified cluster and recalculate columns
        Align motif according to the best offset
        *modified_mtfs* = motifs after alignment
    **end if**
    *curr_mtx_id*+ = 1
**end while**
**Return** clusters

## 5 HMM Clustering - Part 2

**Algorithm 5** Profile-HMM Scoring Aligning Merge(clusters)

**for** key of first cluster to compare in range(# number of clusters): **do**
  **for** key of second cluster in range(# number of uncompared clusters): **do**
    Initialize two clusters for comparison, offset for alignment, best hidden states and minscore.
    **for** $i$ in range(offset + 1): **do**
      Initialize Hidden States, Transition Probabilities, Viterbi Matrix and Backtracing Matrix
      $vb\_m[0][0] =$ Euclidean Distance between Ontarget Hidden State of first column in cluster1 and first emission column in cluster2
      $vb\_m[1][0] =$ Euclidean Distance between Offtarget State [0.25,0.25,0.25,0.25] and first emission column in cluster2
      **for** $t$ in range(number of observations): **do**
        **for** curr_state in range(# number of hidden states = 2): **do**
          $min\_score = \infty$
          $best\_prev\_state =$ placeholder
          **for** prev_state in range( number of hidden states = 2): **do**
            Find minimum Sum-of-Distances score from previous states.
          **end for**
          **if** curr_state == Ontarget **then**
            $vb\_m[curr\_state][t] =$ min_score + Euclidean Distance between Ontarget Hidden State of current column in cluster1 and current emission column in cluster2
          **else**
            $vb\_m[curr\_state][t] =$ min_score + Euclidean Distance between Offtarget State [0.25,0.25,0.25,0.25] and current emission column in cluster2
          **end if**
          Records best previous final state
        **end for**
      **end for**
      Update best hidden states and best final state.
      **for** $t$ in range(n_obs, 0, -1): **do**
        Backtrace to the best previous hidden state with best min score
      **end for**
      $mycore =$ Calculated score plus penalty for offmatching (0.5 per offmatch)
      **if** $myscore < minscore$ : **then**
        Record information for the minscore, offset and best hidden states
      **end if**
    **end for**
    **if** $minscore < best\_minscore$ : **then**
      record information for the minscore, offset and best hidden states
    **end if**
  **end for**
**end for**
return clusters with cluster[id1] and cluster[id2] merged

## Tables

| Score Type | pairwise complete-linkage clustering | pairwise single-linkage clustering | UPGMA | WPGMA | K-medoid | HMM |
|---|---|---|---|---|---|---|
| ARI | 0.5003 | 0.1373 | 0.6248 | 0.5880 | 0.5509 | 0.3325 |
| FMI | 0.5417 | 0.3002 | 0.6381 | 0.6026 | 0.5720 | 0.3570 |
| NMI | 0.8858 | 0.7342 | 0.8912 | 0.8867 | 0.8712 | 0.8050 |

**Table 1.** Scores of the performance of hierarchical clustering and k-medoid clustering on the validated fungi dataset. The dataset used are composed of 178 validated motifs from fungi, which is clustered to 65 clusters. ARI scales from -1 to 1. FMI scales from 0 to 1. NMI scales from 0 to 1.The performance of UPGMA is the highest, while the performance of the pairwise single-linkage clustering is the lowest.

| Clustering Method | Euclidean Distance | Pearson Correlation Coefficient Distance | Kullback Leibler Distance | Jensen Shannon Distance |
|---|---|---|---|---|
| Hierarchical Clustering (UPGMA) | 0.8750 | 0.8912 | 0.8604 | 0.8745 |
| K-medoid Clustering | 0.8477 | 0.8712 | 0.8504 | 0.8498 |

**Table 2.** NMI scores of UPGMA Hierarchical Clustering and K-medoid clustering using Pearson Correlation Coefficient Distance, Euclidean Distance, Kullback Leibler Distance, and Jensen Shannon Distance. The NMI scores of the Pearson Correlation's performance with either hierarchical clustering or K-medoid clustering are the highest among all four distance methods.

| Score Type | Euclidean Distance | Pearson Correlation Coefficient Distance | Kullback Leibler Distance | Jensen Shannon Distance | K-medoid |
|---|---|---|---|---|---|
| ARI | 0.8787 | 0.7874 | 0.7874 | 0.7874 | 0.7874 |
| FMI | 0.9023 | 0.8317 | 0.8317 | 0.8317 | 0.8317 |
| NMI | 0.9266 | 0.8688 | 0.8688 | 0.8688 | 0.8688 |

**Table 3.** The performance of the UPGMA hierarchical clustering using four kinds of distance methods and the K-medoid clustering on the vertebrate dataset enriched in gapped motifs. The performance of the UPGMA hierarchical clustering using Pearson CC Distance is the highest, while all other distance methods have lower yet consistent performance(including the K-medoid Clustering) on the gapped dataset.

| | UPGMA Hierarchical Clustering | K-medoid Clustering |
|---|---|---|
| Large Fungi | 0.8706 | 0.8446 |
| Large Vertebrate | 0.8931 | 0.8891 |

**Table 4.** The datasets used are a dataset composed of all potential motifs in fungi and a dataset composed of all potential motifs in vertebrate. The large fungi dataset is composed of 229 motifs from fungi genome, which is clustered into 61 clusters. The large vertebrate dataset is composed of 1196 motifs from vertebrate, which is clustered into 321 clusters.The NMI scores of UPGMA Hierarchical Clustering and K-medoid clustering on the large fungi dataset and the large vertebrate dataset. The NMI scores are close to 1, suggesting that there is a lot of mutual information between the clustering results and the correct clustering. The performance of the hierarchical clustering is better than the K-medoid clustering.