Interpretable Clustering and Classification of an Imbalanced Dataset of DNA Sequences

Shekh Ahammed Adnan Bashir
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology
1018052026@grad.cse.buet.ac.bd

***Abstract*—Clustering or classification of DNA sequences is more difficult than doing those with any other type of sequences due to various reasons. Such tasks become more difficult when they have to be done on genus or family. Further difficulty is added when the class samples vary in number- that is, the dataset of DNA sequences is imbalanced. However, through using proper feature extraction techniques and dataset sampling technique such tasks are becomes feasible. The dataset we work with in this paper has 9 classes and the number of samples vary from 4 to 1269. In this work, we present a feature extraction technique inspired by popular Natural Language Preprocessing algorithm GloVe [1] to make the classification and clustering of such huge and imbalanced dataset possible. The feature extraction routine is static rather than being a learning one. This eased the interpretation of the machine learning tasks easier. Using interpretable shallow learning techniques, we achieved an accuracy score around 99.8% and a V-measure of 0.5363.**

I. Introduction

As the rapid development of next generation genome sequencing techniques, newer species are being classified quickly. It is important to know the class (family or genus) that a newly sequenced DNA belong to besides learning how the new species are related to the other ones. Here comes the necessity of DNA sequence classification and clustering. In both classification and clustering, a given collection of items is separated into a few to several subcollections so that the items in one subcollection are as similar as possible with items in two different subcollections are as different as possible. However, they differ in the way they do it.

In classification, labeled data is provided to the classifier and the classifier learns an implicit measure of similarity upon seeing the labels of the provided data. This implicitly learned similarity measure can take a very compilcated mathematical form. On the other hand, in clustering a measure of similarity is provided explicitly and the clusterer speratаes the items into subcollections according to that similarity score. The clusterer does not need data labels. The better the similarity measure provided to the clusterer, the better the separation is and hence better the V-measure score is. As the similarity measure used in clustering is already understood, we can gain an insight into the data in the light of that. Broadly speaking, classification discovers the spearating boundary in a given collection to divide that into subcollections and clustering is discovering the groups based on a provided similarity measure. Clustering provides us a view of how the items in a collection form homogeneous groups of items and classification draws decision boundaries among the groups.

A sequence is a list of elements. In a sequence, items of the provided collection are arranged one after another. There might or might not be sequential dependencies among the items. That is, value of an item appearing in the latter positions might or might not depend the value of item or items appearing in the previous positions. This dependency relation varies from problem to problem and this can only either be inferred from the data or provided to the algorithm as parameters. Classification and clustering tasks on the sequences hence are very difficult with data analysis and static algorithms only.

DNA sequences are composed of 4 different nucleotides- Adenine, Cytosine, Guanine, and Thiamine. Therefore, it is very likely that DNA sequence of two species share some common region because of the small state space. There are noises in the DNA sequences. There are intra-class difference of the DNA sequences when it comes classifying genus or families of DNA sequences. Through using preprocessing techniques that maintains the class invariant properties yet transforming all the sequences into sequences of same length or extracting features we can successfully classify or cluster the sequences in such cases.

Interpretability is important to better understand the data so that some static algorithm can be employed to do something of importance with a certain guarantee. When a classifier or a clusterer does their repective tasks, it can do so interpretably or not. Using static feature preprocessing and understanding the parameters and attributes of the machine learning models helps result interpretation. Decision tree based classifiers, probabilistic classifiers, manifold learning algorithms provide interpretable results because they do their in a predefined step by step manner rather optimizing the decision boundary based on different hyperparameters only like black box machine learning models.

In this work, our main focus has been on interpreting the results obtained from the clusterer and the classifier. Specifically, our contributions are:

- Designing a feature extraction procedure that maintains the class invariant property of the DNA sequences.

- Using interpretable machine learning techniques to cluster and classify the DNA sequences.
- Interpretation of the results obtained from the clusterer and the classifier.

II. Previous Works

Better the features better the classification accuracy is. Works done so far on the classification of DNA sequences first extracts features. Nearly all the feature extraction procedure are hand engineered features. Also many of them do not employ popular classification algorithms on those hand-engineered extracted features. Many of them incorporates the use of very popular DNA sequence matching algorithm BLAST for thr discrimination of the sequences purpose. Wang et al. proposed two new methods to classify DNA sequences [2]. The first technique relies on the comparison of a given sequence to a group of active motifs discovered from the classes it knows. The second technique generates and matches gapped fingerprints of a given unlabeled sequence with the elements of classes it knows. Stranneheim et al. classified DNA sequences through using Bloom filters to keep track of novelties of the sequence reads [3]. Seo transformed sequence data to real-valued vectors so that Support Vector Machine (SVM) can be applied [4] to the data for the purpose of classification. In MEGAN [5], a sequence is searched against multiple sequence databases for assigning the lowest common ancestor of the best matches in different databases to the sequence. PhymmBL [6], [7] uses interpolated Markov models to characterize variable-length oligonucleotides and combines with BLAST matching score to yield better accuracy. The Naïve Bayes classifier [8] learns a Bayesian rule from the k-mer distribution of genomes in the training data and applies that rule to classify. In Kraken [9], each k-mer in the sequence is mapped to the lowest common ancestor (LCA) of the genomes that contain that k-mer in a database. For classifcation, the taxa associated with the k-mers in the sequence form a pruned subtree of the general taxonomy tree. Another classifier CLARK [10] defines k-spectrum. k-spectrum of a string x is the vector of dimension 4^k that collects the number of occurences of all possible k-mers in x. k-spectrum represents each DNA sequence. The CLARK algorithm first computes the k-spectrum of a unlabeled DNA sequence. Then the algorithm removes the common regions of the sequences to make the remaining k-mers to be discriminative. Finally, based on the maximum similarity in k-mer frequencies a class is assigned to an unlabeled DNA sequence.

As we already discussed, clustering of DNA sequences is done to study the similarities among the DNA sequences through the lens of a predefined similarity measure. Plenty of works, such as DNACLUST [11], d2 cluster [12], CD-HIT [13], UCLUST [14] have already been done to cluster DNA sequences and they are interpretable. However, seldom of them used adhoc clustering algorithms rather than the ones which are widely known and their inner workings are understood in depth. MeShClust [15] addressed the issue of the sensitivity of clusterer output due to predefined algorithm paramaters. It repurposes the popular Mean Shift algorithm [16] for the DNA sequence clustering. It is a nonparametric feature space analysis technique and seeks the maxima of a density function. The space spanned by DNA sequences can be considered as feature space. The distribution has a density function in that space and modes are the cluster in the input space. However, considering the maxima of the density function as in Mean Shift algorithm makes it very difficult to work with highly imbalanced training dataset- that is, if the number of samples from different classes varies largely.

III. Methodology

A. Sequence Embedding

1) Popular techniques: Embedding is the process of mapping some signal from one domain to another. It is done for the convenience of postprocessing. Sequence embedding maps an entire sequence to another domain. Sequence embedding eases the modeling the sequence for further tasks. Sequence modeling is predicting the distribution of the smaller units of the sequence. It also can be used to extract features of the sequences. Feature extraction falls under the umbrella of feature learning. Feature extraction is done through discovering the patterns among the smaller units in the sequence. Feature extraction routines can be learning procedures or static procedures. Therefore, feature extraction can be done using static algorithms those rely on domain specific knowledge or hypotheses regarding the sequence or can be done using neural networks or shallow learning algorithms.

In the field of Natural Language Processing, word embedding, which is a sequence embedding technique, refers to a set of techniques for language modeling and feature learning. In NLP, word embedding is done on words or phrases to map them to real valued vectors. This real-valued vectors are used for further tasks- such as classification or clustering. There are severalp opular word embedding techniques including word2vec [17], GloVe [18], fastText [19] [20], etc. Each of the routines transforms word sequences such as sentences, paragraphs, etc. into real valued vectors for feasible classification, clustering, etc.

word2vec reduces the number of dimensions using neural networks to learn the embedding. It produces the sequence embedding in two exclusive methods. In one method, called continuous bag of words, the feature learning model is trained to predict a word from the context or surrounding words of it. In another method, called skip-gram, the context words are predicted from a given word through assigning some probability score to a one hot vector of the vocabulary under consideration. GloVe uses a static method to produce the embedding. GloVe model is trained on aggregated global word-word co-occurrence statistics from a corpus. Representations obtained from GloVe demonstrate interesting linear substructures of the word vector space. fastText learns a representation for each character n-gram (a number of neighboring characters around

a character in a word) instead of directly learning a vector representation for a word (as with word2vec). Each word is represented as a bag of character n-grams, so the overall word embedding is a sum of these character n-grams.

2) Our approach: We extend two sequence embedding techniques to the domain of biological sequences- word2vec and GloVe. However, rather directly using them, we modify them for using with k-mers. As k-mers we pick 3-mers, that is the codons. We did not pick the value of k experimentally. We picked a value of k that corresponds to a already known and studied biological structure.

We first split the sequence into nonoverlapping subsequence of length k. Then we construct a *weighted* cooccurence vector out of that. To construct the vector we first define a number of neighbors on the right and the left side of a given k-mer and initialize and a matrix of 0s of dimension $4^k \times 4^k$ with 4 being the number of DNA bases and k being the k value of the k-mer we consider. Therefore, the rows and the columns of the matrix corresponds all possible k-mers. That is, an item $C_{i,j}$ is the weight of association between k-mer i and k-mer j in a given sequence. We call this matrix cooccurence matrix.

We take maximum that number of left and right neighbor. For example, for a sequence of length $100 * k$ (k-mer sequence of length 100) and left and right neighbor count of 3, the first k-mer in the sequence we only have the right neighbor (k-mers in the 2^{nd}, 3^{rd}, and 4^{th} positions), the last k-mer has only left neighbors (k-mers in the 97^{th}, 98^{th}, and 99^{th} positions), and the 10^{th} k-mer has k-mers in 7^{th}, 8^{th}, and 9^{th} position as the left neighbors and k-mers in the 11^{th}, 12^{th}, and 13^{th} positions in the sequence as the right neighbors. Now we add $\frac{1}{3}$ to the weight between the k-mer at the 10^{th} position and the k-mer at the 7^{th} position and the k-mer at the 10^{th} position and the k-mer at the 13^{th} position. That is, we do $\mathbf{C}_{10,7} = \mathbf{C}_{10,7} + \frac{1}{3}$ and $\mathbf{C}_{10,13} = \mathbf{C}_{10,13} + \frac{1}{3}$. We also do $\mathbf{C}_{10,8} = \mathbf{C}_{10,8} + \frac{2}{3}$ and $\mathbf{C}_{10,12} = \mathbf{C}_{10,12} + \frac{2}{3}$ and finally, $\mathbf{C}_{10,9} = \mathbf{C}_{10,9} + 1$ and $\mathbf{C}_{10,11} = \mathbf{C}_{10,11} + 1$. Finally, we flatten the matrix to obtain the cooccurence vector, that is, $\vec{C} = \mathbf{C}_{1,1}, \ldots, \mathbf{C}_{1,N}, \mathbf{C}_{2,1}, \ldots, \mathbf{C}_{2,N}, \ldots, \mathbf{C}_{N-1,N}, \mathbf{C}_{N,1}, \ldots, \mathbf{C}_{N,N}$ Later, we divide the vector with the largest element in it in order to normalize it. This is how we get feature vector of constant size. The significance of this vector can be thoght of to express cooccurence intensity between the k-mers.

B. Visualization through dimension reduction algorithms

Many datasets can be represented as a matrix, such that each row represents a set of attributes (or features or dimensions). Attributes describe a particular instance of something. Larger the number of attributes (columns) exponentially large the space of unique possible rows. Large number of data dimension causes difficulties in data sampling and pattern recognition. Reducing data into fewer dimensions often makes the analysis of algorithms more efficient, and can help machine learning algorithms make more accurate predictions.

We want to have our clusters as nonoverlapping as possible. Visualization is important to understand how good our embedding routine is. We first visualize the clusters through dimension reduction algorithms. We pick three algorithms for that:

- Isomap
- Locally-Linear Embedding (LLE)
- T-distributed Stochastic Neighbor Embedding (t-SNE)

In Isomap first neighbors of each data point are found. Then a neighborhood graph is constructed with each point is connected to other if it is a K nearest neighbor and edge length equal to Euclidean distance. Then in that neighborhood graph the shortest path between the nodes are computed. Finally, a lower-dimensional embedding is computed using multidimensional scaling. We notice that in the Isomap plot Figure 1 of our features many classes are largely overlapped.

Locally-Linear Enbedding leverages the sparsity of the data matrix. First, LLE finds a set of the nearest neighbors of each point. Then it computes a set of weights for each point that best describes the point as a linear combination of its neighbors. Finally, an eigenvector-based optimization is done to find the low-dimensional embedding of points, such that each point is still described with the same linear combination of its neighbors. LLE does not have a mechanism to work with data points with nonuniform density. It performs poor in those cases. This is observed in our LLE plot Figure 2.

t-SNE models each high-dimensional data point by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability. First, a probability distribution over pairs of high-dimensional objects is constructed. Second, a similar probability distribution is defined over the points in the low-dimensional map, and then t-SNE minimizes the KullbackLeibler divergence between the two distributions with respect to the locations of the points in the map. Reducing dimensionality with t-SNE we are able to see the clusters better upon plotting Figure 3. However, feeding this reduced dimensioned data to the classifiers does not yield better classification.

This is error is due to the unsupervised nature of the dimension reduction algorithms. The classification boundary in all problems cannot be properly defined by the well-studied similarity measure(s) alone. Therefore, dimension reduction in an unsupervised fashion does not always work. This is the reason we did not reduce the data dimension for supervised training.

C. Sequence Clustering

D. Sequence Classification

IV. Interpretation

References

[1] J. Pennington, R. Socher, and C. Manning *Glove: Global Vectors for Word Representation.* EMNLP, 14, page 1532–1543. (2014)

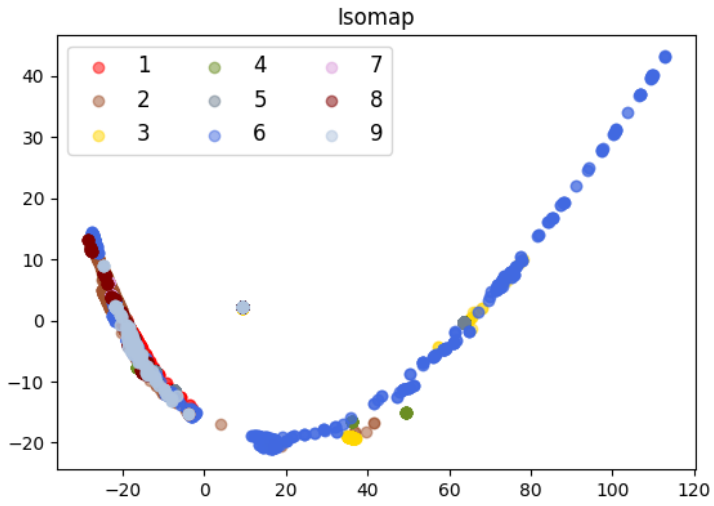

Fig. 1: Isomap on the embeddings obtained from our embedding algorithm.

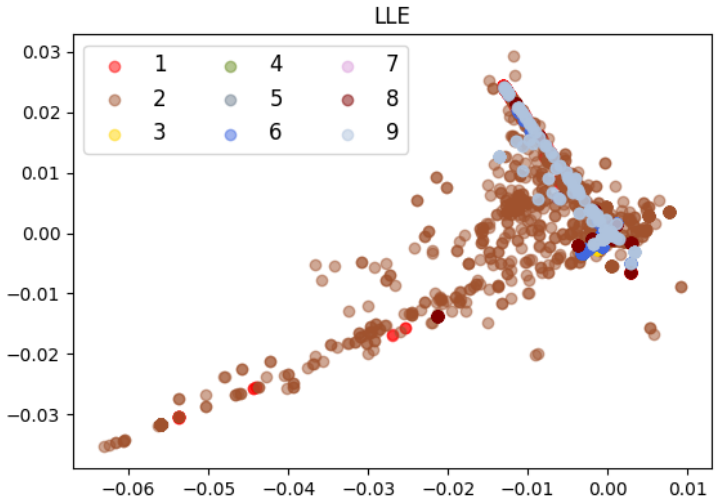

Fig. 2: Locally-Linear Enbedding on the embeddings obtained from our embedding algorithm.

[2] Wang JT, Rozen S, Shapiro BA, Shasha D, Wang Z, Yin M. *New Techniques for DNA sequence classification*. In J Comput Biol. 1999 Summer;6(2):209-18.

[3] Henrik Stranneheim1, Max Käller, Tobias Allander, Björn Andersson, Lars Arvestad and Joakim Lundeberg. *Classification of DNA sequences using Bloom filters*. Vol. 26 no. 13 2010, pages 15951600, doi:10.1093/bioinformatics/btq230.

[4] Seo, TK. *Classification of nucleotide sequences using support vector machines*. J Mol Evol. 2010 Oct;71(4):250-67. doi: 10.1007/s00239-010-9380-9.

[5] Huson D, Auch A, Qi J, Schuster S: *MEGAN analysis of metagenomic data*. Genome Res 2007, 17:377386.

[6] Brady A, Salzberg S. *Phymm and PhymmBL: metagenomic phylogenetic classification with interpolated Markov models*. Nat Methods 2009, 6:673676.

[7] Brady A, Salzberg S. *PhymmBL expanded: confidence scores, custom databases, parallelization and more*. Nat Methods 2011, 8:367.

[8] Rosen G, Garbarine E, Caseiro D, Polikar R, Sokhansanj B. *Metagenome fragment classification using N-mer frequency profiles*. Adv Bioinformatics 2008, 2008:112.

[9] Wood, Derrick E and Salzberg, Steven L. *Kraken: ultrafast metagenomic sequence classification using exact alignments*. Genome Biology 2014, 15:R46.

[10] Rachid Ounit, Steve Wanamaker, Timothy J Close, Stefano Lonardi *CLARK: fast and accurate classification of metagenomic and genomic sequences using discriminative k-mers*. BMC Genomics (2015) 16:236 DOI 10.1186/s12864-015-1419-2.

[11] Ghodsi,M., Liu,B. and Pop,M. (2011) *DNACLUST: accurate and efficient clustering of phylogenetic marker genes*. BMC Bioinformatics, 12, 271.

[12] Burke,J., Davison,D. and Hide,W. (1999) *d2 cluster: a validated method for clustering EST and full-length cDNA sequences*. Genome Res., 9, 11351142.

[13] Li,W. and Godzik,A. (2006) *Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences*. Bioinformatics, 22, 1658.

[14] Edgar,R.C. (2010) *Search and clustering orders of magnitude faster than BLAST*. Bioinformatics, 26, 24602461.

[15] James, Benjamin T., Luczak, Brian B., Girgis, Hani Z. *MeShClust: an intelligent tool for clustering DNA sequences*. Nucleic Acids Research, 2018, Vol. 46, No. 14 doi: 10.1093/nar/gky315

[16] Comaniciu, D. Meer, P. *Mean shift: A robust approach toward feature*

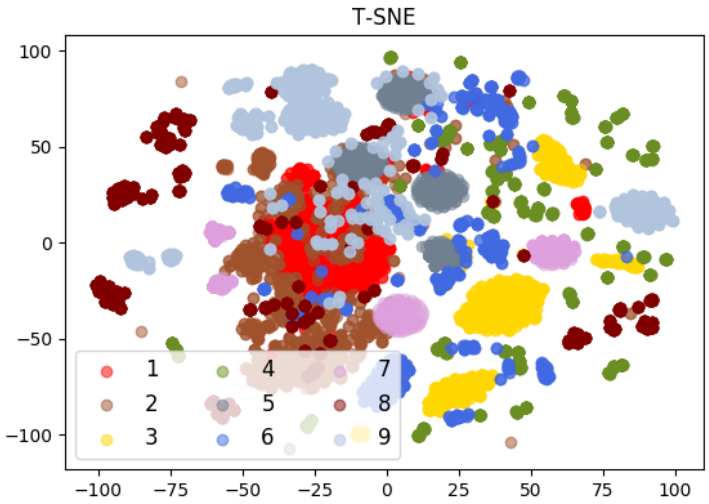

Fig. 3: T-distributed Stochastic Neighbor Embedding on the embeddings obtained from our embedding algorithm.

space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence (2002)

[17] Mikolov, Tomas; Sutskever, Ilya; Chen, Kai; Corrado, Greg S.; Dean, Jeff (2013). *Distributed representations of words and phrases and their compositionality*. Advances in Neural Information Processing Systems. arXiv:1310.4546.

[18] Pennington, J., Socher, R. and Manning, C. D. *Glove: Global Vectors for Word Representation*. EMNLP 2014.

[19] Joulin, A., Grave, E., Bojanowski, P., Mikolov, T. *Bag of Tricks for Efficient Text Classification* (cite arxiv:1607.01759)

[20] Bojanowski, P., Grave, E., Joulin, A., Mikolov, T. (2017). *Enriching Word Vectors with Subword Information*. Transactions of the Association for Computational Linguistics.