K-mer Neural Embedding Performance Analysis Using Amino Acid Codons

Muhammad Nabeel Asim*†, Muhammad Imran Malik‡, Andreas Dengel*†, Sheraz Ahmed*
Email: firstname.lastname@dfki.de
*German Research Center for Artificial Intelligence (DFKI), 67663 Kaiserslautern, Germany
†TU Kaiserslautern, Kaiserslautern, Germany
‡National Center for Artificial Intelligence (NCAI), National University of Sciences and Technology, Islamabad, Pakistan

I. Abstract

Exponential growth of genome-wide assays of gene expressions and their public access open new horizons for machine learning methodologies to effectively perform genetic analysis. In this work, domain specific pre-train k-mer embeddings of DNA sequences are generated by utilising FastText approach. Sequence co-expression pattern information is embedded into 200 dimensional vectors by training Fasttext model on 317,151 samples of DNA sequences (with k-mers representation). We propose a novel idea to utilize the information of various codons present in amino acids for the evaluation of learned sequence vectors. We employ two diverse techniques to compare the performance of generated task-specific k-mer embeddings with state-of-the-art publicly available generic k-mer embeddings of genome. Firstly, we utilize a dimensionality reduction approach namely PCA to alleviate the dimensions of DNA sequences upto 50 features by preserving almost 85% of sequence features information. Afterwards, TSNE algorithms is used to visualize k-mer embeddings and to make sure whether different codons representing the same amino acid are more closer to each other than the ones representing different amino acids. Secondly, to assess the analogy of k-mer embeddings, generated domain specific k-mer embeddings are compared with state-of-the-art k-mer embeddings by estimating the cosine similarity among those codons vectors which represent same amino acid. Overall, we believe that task-specific distributed representation of k-mers would be useful for DNA methylation and Histone occupancy prediction tasks.

***Index Terms*—Histones, Word2vec, FastText, Pretrain k-mer embeddings,**

II. Introduction

Organisms have a lot of biological sequences which communicate throughout their life spans. In genetics, communication between sequences, and cells happens through certain symbols and signs that plays a pivotal role in maintaining multifarious body functions. Nowadays artificial intelligence is being extensively used in various genetic applications such as analysis of protein-DNA interaction [1], prediction and representations of protein coding regions [2], identification of splice sites, nucleosome positioning [3], and histone markers identification [4]. State-of-the-art machine and deep learning methodologies process k-mers of DNA sequences just like words of natural language processing. Considering this sophisticated analogy, we perform an in-depth exploration of existing natural language processing (NLP) based k-mer neural embeddings to acquire a better and explanatory understanding of how exactly these methodologies support DNA sequence analysis.

In natural language processing (NLP), and DNA sequence analysis, for machine learning methodologies, feature selection has significant importance for various tasks such as classification [5], [6] and clustering [7]. While the most sophisticated algorithms perform poorly if inappropriate features are used, simple methods manage to show great performance when they are fed with the appropriate features [5]. Contrarily, deep learning methodologies automate the process of feature engineering, however for these methodologies, feature representation plays a pivotal role [8]. The performance of deep learning methodologies decrease when the features are represented through inappropriate feature representation approach as it creates vanishing and exploding gradients problems during back propagation and badly affect the process of feature extraction [9]. Previously, one hot vector encoding was considered better feature representation technique for deep learning methodologies [10]. However, with the rise and huge success of neural word embeddings, the performance of diverse deep learning methodologies has been revolutionized. Nowadays in NLP, deep learning methodologies along with pre-trained word embeddings are producing state-of-the-art performance for various tasks such as text document classification [11] [12], Text summarization [13], information extraction, and retrieval [14]. Although DNA sequence classification can be considered another natural language processing task, however, deep learning methodologies have been failed to imitate similar promising performance for diverse sequence analysis tasks such as sequence classification [15], codon region prediction [2], nucleosome positioning [16]. This performance gap is due to the lack of pre-trained word embeddings which are in abundance for general natural language processing tasks.

In order to alleviate this performance gap and to raise the performance of deep learning methodologies for DNA Sequence analysis, Asgari et al. [17] provided 100 dimensional pre-trained word vectors (bio-vec) for biological sequences

This work was supported by the SAIL (Sartorius ArtificialIntelligence Lab)

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

by adopting a deep learning methodology namely word2vec. To extract the relationship between the k-mers in a specific context, a Skip-gram based word2vec model was trained on 546,790 sequences (RNA, DNA, and proteins) collected from Swiss-Prot database. T-Stochastic Neighbor Embedding (TSNE) approach was used to validate the integrity of developed embeddings. A similar method was also used by Patrick Ng. [18] to provide pre-trained word embedding of 100 dimensions. For HG38 dataset, K-mers (k=3,4,...8) DNA fragments were used to train the model and extract the relationships among different k mers. Needleman-Wunsch algorithm was used for measuring the alignment and cosine similarity of word vectors. Moreover, Jingcheng Du et al. [19] prepared skip-gram embeddings in various dimensions (50, 100, 200 and 300) on 984 datasets taken from Gene Expression Omnibus (GEO) database. Gene-to-gene interaction was performed as a down stream task. Jianliang et al. [20] proposed another feature representation approach for protein sequences from five datasets H.sapiens, M.musculus,D.melanogaster , C.elegans, and S.cerevisiae taken from IntAct database. They utilized Protein to protein interaction (PPI) network using graphs where each node was represented by a vector of 100 dimensions. In all three discussed research papers, training corpora were consisted of biological sequences taken from different domains. In natural language processing recently, wang et al. [21] performed extensive experimentation to compare the integrity of various pretrained word embeddings. Their experimental results prove that word embeddings trained on domain specific corpus perform better as compared to the training them on a large corpus which contain both domain specific and non domain specific data. Inspired from their research findings, we use Fasttext model to generate word embeddings on a corpus of 317,151 DNA sequences which are only positively charged. To evaluate the integrity of generated word vectors, we performed quantitative and qualitative analysis. Qualitative analysis is performed by the combination of two dimensionality reduction approaches principal component analysis (PCA) and TSNE. Finally, two dimensional vectors of each k-mer are plotted using TSNE algorithm. From plotted k-mers, we analysed weather different k-mers that belongs to the same amino acid are nearest each other as compared to k-mers that belongs to different amino acids. For quantitative analysis, by using cosine similarity measure, we assess the level of similarity between codon vectors of particular amino acid. To sum up, by performing deep analysis of generated domain specific vectors and general k-mer embeddings, we believe, domain specific embeddings will largely assist deep learning models to perform computational analysis of histone related problems.

III. Distributed Representation

Machine learning methodologies require numeric feature space in order to perform multifarious tasks such as clustering [22], classification [23], and summarization [24]. The process of creating numeric feature space is called feature representation or vectorization. Researchers have proved that even best machine learning algorithms perform worse with bad feature representation, and less effective machine learning algorithms produce good performance with better feature representation. Feature representation has always been an attractive area of research for numerous researchers and practitioners. With each passing day, still several feature representation approaches are being proposed and experimented. Initially, researchers tend to use bag-of-words (BOW) based vectorization approaches such as Count Vectorizer or term frequency (TF), term frequency inverse document frequency (TF-IDF), Latent Dirichlet Allocation (LDA), and Latent Semantic Analysis (LSA), to estimate continuous representations of corpus words. In bag-of-words (BOW) based approach, every document is represented with a fixed sized vector whose length is exactly equivalent to the vocabulary of underlay corpus. Later on, researchers utilized boolean vector encoding scheme known as One Hot encoding. One hot encoding represents categorical attributes in the form of binary vectors where 1 represents the presence and 0 represents the absence of particular feature. These approaches do not consider contextual information and also produce high dimensional and sparse feature vectors which are less memory efficient and hard to construct for large scale datasets. Although Co-Occurrence matrix captures the contextual information when constructed with a reasonable context window, however it requires humongous memory and computational time which makes it almost infeasible for large datasets.

Contrarily, continuous distributed feature representation or prediction based word embeddings are dense low dimensional feature vectors which capture both syntactic and semantic properties [25]. Furthermore, these embeddings are trained in an unsupervised manner and can improve the performance of diverse downsteam NLP tasks. Firstly, Bengio et al [26]. introduced the term “Word Embeddings” and trained them with a deep language model in 2003. The proposed deep language model was consist of feed forward neural network having one hidden layer, which used to predict the subsequent word of a sequence. Primary building blocks of Bengio model embedding layer, intermediate layers, and softmax layer are still an integral part of several word embedding and language models.

Later on in 2008, Collobert et al. [27] showed that word embeddings contain significant amount of syntactic and semantic information when trained on a substantially large data. They also presented the utilization of pre-trained neural word embeddings. Their most significant publication “A unified architecture for natural language processing” did not only reveal a deep neural architecture (CW model) which is a part of several current approaches, but it also prove that word embeddings are extremly effective for downstream natural language processing (NLP) tasks [27]. Nonetheless, the ultimate popularization of neural word embeddings is mostly accredited to Mikolov et al. [28], who is the pioneer of robust word embedding toolkit known as Word2vec. They presented two architectures namely Continuous bag-of-words (CBOW), and Skip-gram in quest of learning neural word embeddings and

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

also improved these models by utilizing additional approaches to raise training speed along with accuracy [29]. Both proposed neural architectures vary on the basis of training objective. While Continuous bag-of-words (CBOW) preditcs the target word by considering the context of n surrounding words, Skip-gram utilizes the target word to predict the neightbouring words. Former one does not capture rare words as its primary focus is to predict the word of highest probability, however latter one capables to capture rare words and considered more optimal option.

Undoubtedly, Word2vec exploded the research in word embeddings as Word2vec toolkit provides the simplest way to utilize pre-trained word embeddings or seamless training for downstream NLP tasks. Inspiring from Word2vec, in 2014, Pennington et al. [30] came up with a competitive but similar set of pre-trained neural word embeddings and named it as "Glove". They utilized word co-occurrence to construct this model as they considered that co-occurring words have semantic or syntactic similarities. Word2vec, and Glove associate a distinct vector to every single word and ignores the word morphology, thus these models find it hard to build word embeddings for those languages which have huge vocabularies and substantial amount of rare words. To overcome this limitation, Bojanowski et al. [31] came up with a naive approach based on Skip-gram model in which every word is expressed as a bag of n-grams characters. Each n-gram character gets a vector representation, words being expressed as the addition of these representations. For instance, consider the word "Abnormal" with n-gram equals to 3. FastText representations of n-grams characters are <ab, bno, nor, orm, rma, mal, al>, where angle brackets are embedded as boundary symbols to effectively disparate n-grams from the actual word. FastText representations have not only been proved fast but it also enabled the representations of those words which did not appear in the corpus as it utilizes character level embeddings.

With the rise of prediction based embeddings, it has become a pivotal part of feature representation as they are readily available, and do not need expensive annotation. Moreover, they have literally improved the performance of diverse downsteam NLP tasks.

IV. Generating DNA Sequnece Embeddings using FastText

FastText word embeddings have proved extremely effective and outperformed word2vec, and Glove for diverse natual language processing (NLP) tasks. FastText splits each word into a set of n-gram characters (sub-words) which are eventually added together in order to construct the word as an ultimate feature. It utilizes the Skip-gram core objective along with negative sampling, where sub-words are positive instances, and the random samples from the corpus dictionary are considered as negative instances. In this way, FastText word embeddings embed sub-word information and effectively tackle the representation of those words which have not appeared in the training data.

This paper utilizes 3-mers of DNA sequences, and FastText distributed representation learning approach for 317151 DNA sequences of 18 public corpora to prepare a task-specific word embeddings. In order to generate 3-mers of DNA sequences we slide a window of size 3 on the sequence with stride size 1. Using FastText approach we generated 200 dimensional vectors for each k-mer. Moreover, in embedding corpora we discarded all k-mers which occurs less then 5 times in a sequence. K-mer representation is learned by rotating a window of size 5.

V. Embeddings Datasets

This section briefly describes basic concepts of genetics along with detailed description of DNA sequence data which is used to train fasttext model for the generation of task-specific embeddings.

Genome is a genetic material and most significant part of all living organisms, which contains hereditary information e.g. running, building, maintenance of an organism and reproduction. Human genome is made up of 3 billion nucleotides (A, C, G and T) organized in a proper sequence and form a DNA of an organism. Histones are lightweight proteins (14-18 kDa) also known as basic building blocks of genomes. They are essential part of all eukaryotic organisms (like human, animals) and are made up of amino-acids in which 20-24 % sequences contain lysine and arginine amino acid. There are five core histone proteins H1, H2A, H2B, H3, H4 and each one has its variants and post-translational modifications (PTMs). Humans have 55 histone variants which differ from each other on the basis of amino acid sequences especially at the tail N-terminal, while PTMs are the modification of acetylene, phosphorylation, methylene or ubiquitination etc. on tails of sequences by writers or erasers. Histone PTMs have a significant role in cellular processes like delimiting the boundary regions for euchromatin and heterochromatin, stemness maintenance and process of controlling the cell cycle. Combination of two tetramer H3-H4 and two dimers H2A-H2B form histone octamer, which is the core part of nucleosome and present at the centre of it. In nucleosome, it is superhelically wrapped by 146 base pairs of DNA in eukaryotes and plays a significant role in the process of transcription (DNA to RNA) which is further used for translation (RNA to Protein) process.

We have prepared the 3-mer DNA sequence representation using H3, H4 histone proteins and some of its PTMs which are explained in table I with 500 base sequence length. Where H3 is 15-16 kDa histone having 135 amino acid. Its total variants are 216 in different species while the human body has 6 variants. K and its leading number denote the K^{th} modified amino acid, for example, K4 represents that modification of 4th amino acid. Each amino acid modification affects differently like H3K4 is considerably behind the activation of both methyl or acetyl group while H3K36 is considered a fine wine; intriguing, complex and attracts several researchers to this. In methylation process, PTMs are performed by three types of modifications mono, di and tri methylation as

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

Dataset Name	Description	Positive Samples	Nagative Samples	Length of Sequence
H3	H3 occupancy	7667	7298	500
H4	H4 occupancy	6480	8121	500
H3K4me1	H3K4 mono-methylation relative	17266	14411	500
H3K4me2	H3K4me2 H3K4 di-methylation relative to H3	18143	12540	500
H3K4me3	H3K4me3 H3K4 tri-methylation relative to H3	19604	17195	500
H3K36me3	H3K36me3 H3K36 tri-methylation relative to H3	18892	15988	500
H3K79me3	H3K79me3 H3K79 tri-methylation relative to H3	15337	13500	500
H3K9ac	H3K9 acetylation relative to H3	15415	12367	500
H3K14ac	H3K14 acetylation relative to H3	18771	14277	500
H4ac	H4 acetylation relative to H4	18410	15686	500
Promoters	E. coli promoter gene sequences with partial domain theory	53	53	57
Genomes	Nucleosome Positioning Dataset for nucleusomal and linker DNA sequences	1880	1740	150
Homo-Sapiens	Nucleosome Positioning Dataset for formation and inhibiting nuclesome DNA segment	2,273	2,300	147
Drosophila Melanogaster	Nucleosome Positioning Dataset for formation and inhibiting nuclesome DNA segment	2,900	2,850	147
Caenorhabditis Elegans	Nucleosome Positioning Dataset for formation and inhibiting nuclesome DNA segment	2,567	2,608	147
DSB genome	Double strand DNA break sites in genomes	3600		1001
SNP genome	Single Nucleotide Polymorphism in genomes	6637		1001
ORI genome	Origin of replication sites in genomes	322		1001

TABLE I: Characteristics of Corpus used for embedding generation

Amino Acid	DNA codons
Isoleucine	ATT, ATC, ATA
Leucine	CTT, CTC, CTA, CTG, TTA, TTG
Valine	GTT, GTC, GTA, GTG
Phenylalanine	TTT, TTC
Methionine	ATG
Cysteine	TGT, TGC
Alanine	GCT, GCC, GCA, GCG
Glycine	GGT, GGC, GGA, GGG
Proline	CCT, CCC, CCA, CCG
Threonine	ACT, ACC, ACA, ACG
Serine	TCT, TCC, TCA, TCG, AGT, AGC
Tyrosine	TAT, TAC
Tryptophan	TGG
Glutamine	CAA, CAG
Asparagine	AAT, AAC
Histidine	CAT, CAC
Glutamic acid	GAA, GAG
Aspartic acid	GAT, GAC
Lysine	AAA, AAG
Arginine	CGT, CGC, CGA, CGG, AGA, AGG
Stop codons	TAA, TAG, TGA

TABLE II: 64 DNA codons which represents to twenty amino acids and stop codons [32]. Different amino acids are represented by different number of codons.

H3k4me1, H3k4me2 and H3k4me3, H3K36me3, H3K79me3 respectively. Enzymes category lysine methyltransferases performed these modifications. Methylation modification does not affect the interaction of proteins with DNA, as the charge remain same on histones and it protects the transcription process. Acetyl PTMs are catalysed by specific enzyme lysine acetyltransferases. The interaction between histone and DNA is hampered by this modification. Here we have used 2 acetyls PTM of H3 H3K9ac and H3k14ac with the modification of 9^{th} and 14^{th} amino acid. These two acetyl histones with H3K4me3 have large significance as they are the benchmark for the activation of gene promoters.

H4 is 11.3 kDa size protein with 102 amino acids. It has 116 variants in all organisms while humans have no variant except H4 but it has many modifiers by writing or erasing the acetyl, phosphate or methyl group. The importance of these H4 PTMs have been well depicted in diverse biological processes such as transcriptional activation, DNA damage, heterochromatinisation, and cancer hallmarks. H4 is used as docking site for all other histones. We use its only one modifier H4ac. These all DNA sequence datasets are classified into 2 classes based on methyl and acetyl occupancy. If methyl and acetyl occupancy in the middle position is greater than 1.2 than it is categorized as a positive class, and if less than 0.8 than it is classified a negative class, otherwise nothing. Furthermore, we include the benchmark nucleosome positioning datasets in genome, and specifically in Homo sapiens (humans), Drosiphila melanogaster (species of fly), and Caenorhabditis elegans (transparent nematode, e.g. worms). These DNA sequence datasets also belong to two classes classified as positive and negative. In these nucleosome positioning datasets, positive DNA segments represent the formation of nuclosome, while negative represents the inhibiting nucleosome with 147-bp length. We also include broken off the double strand of DNA named as DBS DNA, SNP for nuclutide positioning dataset and replication initiation in genome ORI dataset. Details of all used datasets is summarised in Table I.

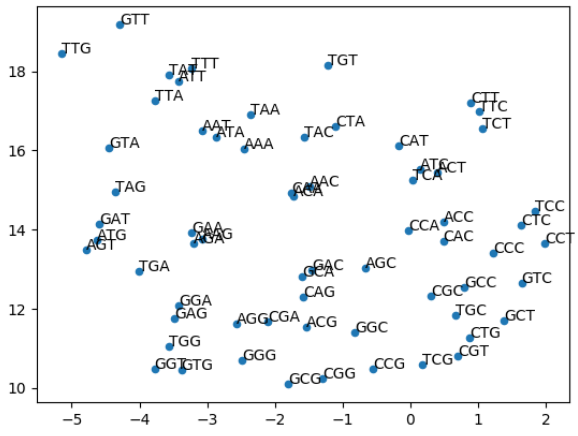
VI. K-mer Embedding Evaluation

K-mer embeddings, real-valued representations of words or k-mers produced by distributional semantic models (DSMs) have been an active area of research. However, their limitations about semantic knowledge extraction are still not well understood. One of the most important questions in the studies of distributional semantics is how to evaluate the quality of generated embeddings. There is still no consensus in the scientific community about which evaluation method should be used: NLP engineers are more interested in performing downstream tasks for the evaluation of pre-trained neural word embeddings, while computational linguistics are used to explore the quality of pre-train neural word embeddings through visualisation or by measuring similarity across different vectors using cosine similarity measure. In the process of evaluation through visualization or cosine similarity measure, pre-train neural word embeddings are considered better if similar words have more similarity among their word vectors as compared to their similarity with other dissimilar word vectors. For example, as

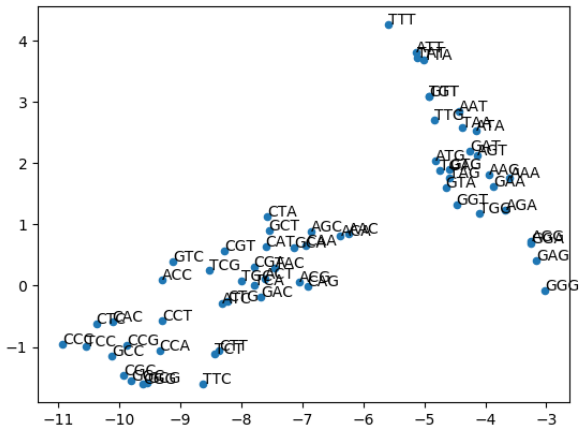
978-1-7281-6926-2/20/$31.00 ©2020 IEEE

2*Amino Acid	2***DNA codons**	**Top 6 Most Similar word vectors**		
		DNA2vec	**Protien2vec**	**Fast_text**
Isoleucine	ATT, ATC, ATA	TTT, TTA, **ATC**, TAT, AAT, TTG	TTT, TAT, AAT, TTA, TAA, **ATA**	TTT, **ATC**, **ATA**, ATG, GTT, CTT
Leucine	CTT, CTC, CTA, CTG, TTA, TTG	TTT, CCC, CCT, **CTC**, CGC, TCT	CCT, **CTC**, TCC, CCC, TTC, ATC,	**CTC**, **CTG**, CTA, TTT, ATT, GTT
Valine	GTT, GTC, GTA, GTG	TTT, GTC, GTG, GGT, TGT, **GTA**	TAT, GAT, TTT, TTG, ATG, ATT	GTC, GTA, **GTG**, TTT, ATT, CTT
Phenylalanine	TTT, TTC	ATT, CTT, GTT, TTG, **TTC**, TAT	TAT, TTA, ATT, TGT, **TTC**, AAT	CTT, **TTC**, ATT, GTT, TTA, TTG
Methionine	ATG	GTG, ATC, TGG, GCG, GGG, GGT	AAG, GTA, TAG, TAA, AAT, AGA	TTG, GTG, CTG, ATT, ATA, ATC
Cysteine	TGT, TGC	TGG, GTG, CGT, GGT, GGG, CCG	GTG, TAT, TTT, TGA, TCT, GAA	AGT, GGT, CGT, TGG, TGA, **TGC**
Alanine	GCT, GCC, GCA, GCG	GCC, CCT, GGG, GGC, **GCG**, CCC	**GCA**, CCA, TGT, CTT, TGC, TCG	**GCA**, **GCC**, GCG, TCT, CCT, ACT
Glycine	GGT, GGC, GGA, GGG	**GGG**, AGG, GTG, GCG, TGG, **GGC**	**GGA**, AGG, AGA, TAA, TTG, GCA	**GGA**, **GGC**, TGT, **GGG**, AGT, CGT
Proline	CCT, CCC, CCA, CCG	**CCC**, GCC, TCC, CTC, ACC, CCG	CTC, **CCC**, TCC, **CCG**, CTT, CAC	**CCA**, **CCG**, TCT, **CCC**, ACT, GCT,
Threonine	ACT, ACC, ACA, ACG	ACC, CTC, CCC, **ACG**, CCT, CGC	CCG, CCC, CGA, CCT, CAG, TCC	**ACA**, **ACC**, TCT, CCT, GCT, **ACG**
Serine	TCT, TCC, TCA, TCG, AGT, AGC	CTC, CCT, CCC, GCT, **TCC**, GTC	CTC, CCC, CGC, CCT, **TCC**, CCG	CCT, **TCA**, **TCC**, ACT, GCT, TTT
Tyrosine	TAT, TAC	TTT, TTA, ATA, TGT, CAT, ATT	TTT, GTT, TTA, ATT, TGT, TAA	TAA, **TAC**, TAG, AAT, CAT, GAT
Tryptophan	TGG	GGG, GCG, GGC, GGA, CGG, CCG	AGG, TAG, GAG, GTA, AGT, AGA	AGG, CGG, GGG, TGT, TGA, TGC
Glutamine	CAA, CAG	AAA, **CAG**, ACA, AAG, CCA, CGG	CTA, **CAG**, ACG, TAC, CGA, ATC	**CAG**, CAT, CAC, AAA, TAA, GAA
Asparagine	AAT, AAC	AAA, TAA, ATA, **AAC**, CAA, AGC	AAA, TAA, AAG, TTA, AGA, CAA	AAA, GAT, TAT, CAT, AAG, **AAC**
Histidine	CAT, CAC	**CAC**, CGC, CCC, CAG, GGC, CCT	CCC, CGA, CCG, CTC, TAC, ATC	CAA, **CAC**, CAG, AAT, TAT, GAT
Glutamic acid	GAA, GAG	AAA, AGC, GGA, GAG, AGA,AAG	AAG, AAA, GGA, AGG, AAT, AGA	**GAG**, GAT, AAA, GAC, TAA, CAA
Aspartic acid	GAT, GAC	GGG, CGA, **GAC**, GGC, GGA, GCG	GTT, AAG, AGT, AAT, CGA, GAA	GAA, **GAC**, GAG, AAT, TAT, CAT
Lysine	AAA, AAG	CAA, TAA, GAA, ATA, **AAG**, ACA	**AAG**, TAA, AAT, AGA, AGG, GAA	GAA, **AAG**, AAT, TAA, CAA, AAC
Arginine	CGT, CGC, CGA, CGG, AGA, AGG	CCG, GCG, ACG, GGT, GTG, **CGG**	**CGA**, TCG, CCG, CTC, TCC, **CGG**	**CGA**, **CGG**, **CGC**, TGT, AGT, GGT
Stop codons	TAA, TAG, TGA	AAA, AAT, TAC, TAG, ATA, CCA	AAA, AAT, AAG, ATG, TTA, TAT	TAT, **TAG**, TAC, AAA, AGG, CAA

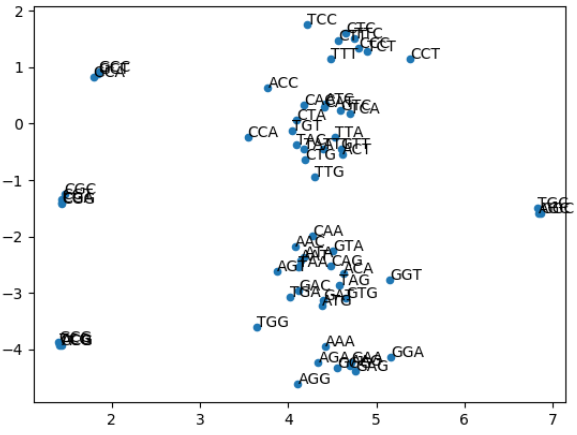
TABLE III: Cosine similarity based top 6 ranked codons for 3 different pre-trained neural k-mer embeddings



(a) DNA2vec [18]



(b) Prot2vec [20]



(c) FastText

Fig. 1: Embedding space representation of 3 different pre-trained Neural k-mer Embeddings

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

words like good and better are semantically quite close to each other, hence their word vectors shall be more similar to each other and less similar to opposite words such as bad.

In order to improve the performance of DNA and RNA sequence analysis tasks, researchers have developed various kinds of pre-trained k-mer embeddings. They have evaluated the performance of pre-trained k-mer embeddings by performing a downstream classification task or by visualising sequence vector space of different classes using classification data sets. Up to date, no one has evaluated the performance of pre-trained k-mer embeddings at k-mer level. The paper in hand, for the very first time performs performance evaluation of pre-trained neural embeddings at k-mer level. As in natural language processing, we have greater understanding of similar and dissimilar words, hence at word level, we can easily evaluate the performance of pre-trained neural word embeddings. Contrarily, in k-mer neural embeddings, we do not know which k-mers could be consider similar or dissimilar.

In this paper, K-mers similarity and dissimilarity information is borrowed from biological domain where different amino acids are represented by various codons which are made up of three different neucelotides. Actually, in human body there are twenty two amino acids where each amino acid is a combination of three nucleotides known as "Codon". There are 64 possible combinations of nucleotides each one referring to a particular amino acid.

As we have generated pre-trained neural embeddings with 3-mers. so, finally we get 64 different pre-trained vectors of 200 length where each vector represents a unique combination of three neucelotides known as 3-mer. Furthermore, we consider 64 possibly generated 3-mers are equal to 64 codons which represents 22 different amino acids.

Table II adopted from [32], illustrates different amino acids along with 64 possible codons which represent particular amino acids. We consider different codons which represent to same amino acid must have same physical and chemical properties. So pre-trained neural 3-mer vectors of different codons or 3-mers which has same physical and chemical properties must be more similar as compare to 3-mer vectors of codons which do not have same properties.

Rather than measuring and comparing similarity across all 3-mers of generated word vectors, we take only 5 different 3-mers vectors whose cosine similarity score with particular selected codon is higher.

To achieve this, we randomly pick one codon from each amino acid and find cosine similarity of this particular codon pre-trained vector with other 63 pre-trained codon vectors. Based on cosine similarity scores, we rank all 64 codons. As we want to measure the similarity across all codons of a particular amino acid and maximum number of codons in any amino acid is only 6, so from ranked codons we only pick top 6 codons for each case. We repeat same process for all three kinds of pre-trained k-mer neural embeddings. For all three pre-trained embeddings, top 6 codons for each case are shown in Table III.

From Table III it can be summarised that domain specific FastText pre-trained 3-mer embedding vectors are more closely related to the concept of degeneracy of codons as compared to two other 3-mer embedding vectors. For instance, four codons GGT, GGC, GGA, and GGG represent Glycine amino acid and we are assuming that based on cosine similarity scores all four codons must have higher similarities and should be present at 9^{th} row of Table III. DNA2vec pre-trained neural embeddings managed to have only two codons GGG and GGC in top 6 most ranked codons and Protien2vec managed to have only one codon GGT in top 6 ranked codons. Where as FastText based pre-trained neural embeddings managed to have 3 codons GGA, GGC, GGG in top 6 ones.

Heading towrds Arginine which is represented by six codons CGT, CGC, CGA, CGG, AGA, and AGG. Among the six most similar vectors computed for codon CGT, DNA2vec pre-trained neural embeddings managed to have only one codon CGG in top 6 most ranked codons and Protien2vec managed to have only two codons CGA and CGG in top 6 ranked codons. Where as FastText based pre-trained neural embeddings managed to have 3 codons CGA, CGG, CGC in top 6 codons. For amino acid Proline, the behaviour of all three pre-trained neural embeddings is exactly similar to their behaviour for Arginine amino acid.

Overall, it can be concluded from all three pre-trained k-mer neural embeddings, domain specific FastText pre-trained k-mer neural embeddings have higher similarities of 3-mer vectors across different codons which have same physical and chemical properties.

Visualization also provides an ease to understand the hidden patterns and relations inside the data. PCA and t-SNE are two most widely used approaches for the task of pre-trained embeddings visualization.

In order to reap the benefits of both PCA and t-SNE approaches, We first pass the embedding vectors to PCA which reduced their dimensions from 200 to 50 dimensions. Then, these low dimensional embedding vectors are passed to t-SNE for further reduction and visualization in two dimensional space. To evaluate the integrity of pre-trained k-mer embeddings, we determine whether different codons or 3-mers representing same amino acid are more closer to each other as compared to other codons which represent different amino acids.

Figure 1 shows the embedding space of 3 different pre-trained k-mer embeddings. Overall from figure 1, it can be inferred that FastText based domain specific pre-trained embedding vectors are more accurate in terms of revealing the concept of codon degeneracy. For example, from Table II we can see two codons TTT and TTC represent the same amino acid Phenylalanine, so both these codons or 3-mers must lies close to each other in the embedding space. Figure 1a shows pretrain 3-mer embedding space of DNA2vec approach, where both 3-mers TTT and TTC are very far from each other. similarly Figure 1b which shows pretrain 3-mer embedding space of Protien2vec, both 3-mers TTT and TTC are close to each other as compaerd to there distance in DNA2vec

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

embeddings space. Figure 1c shows embedding space of domain specific FastText k-mer embeddings, where both 3-mers TTT and TTC are more close as compared to their distance between DNA2vec and Protien2vec embedding spaces. Behaviour of different codons or 3-mers in 4 different amino acids (Phenylalanine Asparagine, Glutamic acid, Aspartic acid, Leucine) is similar to their behaviour in Phenylalanine.

In case of Glutamic acid the distance between there codons GAA GAG for FastText and protien2vec embedding space is nearly same but there is a large gap for DNA2vec embedding space. similarly 3 k-mers TAA, TAG, and TGA represent to k-mers in stop codons and these 3 k-mers are more close for the embedding space of FastText as compare to their distance between embedding spaces of other two approaches. Among other two approaches, in prot2vec although TAG and TGA are very close but TAA is far away while in DNA2vec not a single vector is close to each other.

Turning towrads Leucine amino acid which could be represent by six different codons CTT, CTC, CTA, CTG, TTA, and TTG. Figure shows that six different codons which represents to Leucine amino acid, in FastText embedding space, CTT and CTC are present in one cluster and CTA, CTG, TTA, and TTG are present in another cluster. On the other hand for Protien2vec embedding space, only two codons CTT and CTC are close to each other and remaining 4 codons are far from each other. similarly in DNA2vec embedding space all six codons are far from each other as compare to their distance between embedding space of other two approaches. In a nutshell, it can be concluded that by precisely analysing the embedding spaces of three different k-mer embedding approaches, we can say that the performance of domain specific FastText embeddings is better as compared to the performance of Dna2vec and Protien2vec.

VII. Conclusion

This paper extensively investigates the behaviour of three different pre-train k-mer embeddings by utilising the information of 22 amino acids having different subset of codons. We consider different codons which represent to same amino acid must have their embedding vectors close to each other as compared to embedding vectors of codons which represent to different amino acids. We train k-mer embeddings for DNA sequences by utilising FastText approach. To evaluate the performance impact of domain specific pre-train k-mer embeddings, we compare FastText k-mer embeddings with two other publicly available k-mer embedding approaches DNA2vec and Protien2vec. Performance of all three approaches is evaluated by measuring the similarity of different codon vectors using cosine similarity measure and t-SNE based visualisation. Analysis based on both similarity measures, we conclude that for FatText k-mer embeddings, different codons which represent to same amino acid have more similar k-mer embeddings as compared to their similarity among other two publicly available k-mer embeddings. In future, we will utilise these k-mer embeddings to perform Histon maker identification and classification related tasks.

Acknowledgement

This work was supported by the SAIL (Sartorius Artificial Intelligence Lab) . We thank all members of the Deep Learning Competence Center at the DFKI for their comments and support.

References

[1] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of dna-and rna-binding proteins by deep learning," *Nature biotechnology*, vol. 33, no. 8, p. 831, 2015.

[2] A. L. Swan, A. Mobasheri, D. Allaway, S. Liddell, and J. Bacardit, "Application of machine learning to proteomics data: classification and biomarker identification in postgenomics biology," *Omics: a journal of integrative biology*, vol. 17, no. 12, pp. 595–610, 2013.

[3] M. Tahir, M. Hayat, and S. A. Khan, "inuc-ext-psetnc: an efficient ensemble model for identification of nucleosome positioning by extending the concept of chou's pseaac to pseudo-tri-nucleotide composition," *Molecular Genetics and Genomics*, vol. 294, no. 1, pp. 199–210, 2019.

[4] J. Zhou and O. G. Troyanskaya, "Predicting effects of noncoding variants with deep learning–based sequence model," *Nature methods*, vol. 12, no. 10, p. 931, 2015.

[5] M. N. Asim, M. Wasim, M. S. Ali, and A. Rehman, "Comparison of feature selection methods in text classification on highly skewed datasets," in *2017 First International Conference on Latest trends in Electrical Engineering and Computing Technologies (INTELLECT)*. IEEE, 2017, pp. 1–8.

[6] A. Rehman, K. Javed, H. A. Babri, and N. Asim, "Selection of the most relevant terms based on a max-min ratio metric for text classification," *Expert Systems with Applications*, vol. 114, pp. 78–96, 2018.

[7] S. Alelyani, J. Tang, and H. Liu, "Feature selection for clustering: A review," in *Data Clustering*. Chapman and Hall/CRC, 2018, pp. 29–60.

[8] S. Martinčić-Ipšić, T. Miličić, and L. Todorovski, "The influence of feature representation of text on the performance of document classification," 07 2017.

[9] F. Almeida and G. Xexéo, "Word embeddings: A survey," *arXiv preprint arXiv:1901.09069*, 2019.

[10] K. Potdar, T. S. Pardawala, and C. D. Pai, "A comparative study of categorical variable encoding techniques for neural network classifiers," *International journal of computer applications*, vol. 175, no. 4, pp. 7–9, 2017.

[11] X. Deng, Y. Li, J. Weng, and J. Zhang, "Feature selection for text classification: A review," *Multimedia Tools and Applications*, vol. 78, no. 3, pp. 3797–3816, 2019.

[12] J. Camacho-Collados and M. T. Pilehvar, "On the role of text preprocessing in neural network architectures: An evaluation study on text categorization and sentiment analysis," *arXiv preprint arXiv:1707.01780*, 2017.

[13] A. Kanapala, S. Pal, and R. Pamula, "Text summarization from legal documents: a survey," *Artificial Intelligence Review*, vol. 51, no. 3, pp. 371–402, 2019.

[14] S. Strassel, M. Palmer, K. Knight, K. Griffitt, J. Getman, J. Tracey, A. Bies, and Z. Song, "Reorient: Resources for operationally relevant information extraction from non-explicit text," University of Pennsylvania Philadelphia United States, Tech. Rep., 2019.

[15] R. Villebro, S. Shaw, K. Blin, and T. Weber, "Sequence-based classification of type ii polyketide synthase biosynthetic gene clusters for antismash," *Journal of industrial microbiology & biotechnology*, vol. 46, no. 3-4, pp. 469–475, 2019.

[16] V. B. Teif and C. T. Clarkson, "Nucleosome positioning," *Encyclopedia of Bioinformatics and Computational Biology (ed. S. Ranganathan, M. Gribskov, K. Nakai & C. Schönbach)*, pp. 308–317, 2019.

[17] E. Asgari and M. R. Mofrad, "Continuous distributed representation of biological sequences for deep proteomics and genomics," *PloS one*, vol. 10, no. 11, p. e0141287, 2015.

[18] P. Ng, "dna2vec: Consistent vector representations of variable-length k-mers," *arXiv preprint arXiv:1701.06279*, 2017.

[19] J. Du, P. Jia, Y. Dai, C. Tao, Z. Zhao, and D. Zhi, "Gene2vec: distributed representation of genes based on co-expression," *BMC genomics*, vol. 20, no. 1, p. 82, 2019.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

[20] J. Gao, L. Tian, T. Lv, J. Wang, B. Song, and X. Hu, "Protein2vec: Aligning multiple ppi networks with representation learning," *IEEE/ACM transactions on computational biology and bioinformatics*, 2019.
[21] Y. Wang, S. Liu, N. Afzal, M. Rastegar-Mojarad, L. Wang, F. Shen, P. Kingsbury, and H. Liu, "A comparison of word embeddings for the biomedical natural language processing," *Journal of biomedical informatics*, vol. 87, pp. 12–20, 2018.
[22] J. Wang, S.-T. Wang, Z.-H. Deng *et al.*, "Survey on challenges in clustering analysis research," *Control and Decision*, vol. 27, no. 3, pp. 321–328, 2012.
[23] M. N. Asim, A. Rehman, and U. Shoaib, "Accuracy based feature ranking metric for multi-label text classification," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 10, 2017.
[24] H. Christian, M. P. Agus, and D. Suhartono, "Single document automatic text summarization using term frequency-inverse document frequency (tf-idf)," *ComTech: Computer, Mathematics and Engineering Applications*, vol. 7, no. 4, pp. 285–294, 2016.
[25] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.
[26] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.
[27] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 160–167.
[28] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.
[29] ——, "Efficient estimation of word representations in vector space. cornell university library. 2013," *arXiv preprint arXiv:1301.3781*, 2016.
[30] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.
[31] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.
[32] Y. Wei, P. Thumfort, C. Wurth, and M. Hecht, "Protein design by binary patterning of polar and nonpolar amino acids," *Methods in molecular biology (Clifton, N.J.)*, vol. 352, pp. 155–66, 02 2007.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE