Alignment-Free Sequence Comparison: A Systematic Survey From a Machine Learning Perspective

Katrin Sophie Bohnsack, Marika Kaden, Julia Abel, and Thomas Villmann

Abstract—The encounter of large amounts of biological sequence data generated during the last decades and the algorithmic and hardware improvements have offered the possibility to apply machine learning techniques in bioinformatics. While the machine learning community is aware of the necessity to rigorously distinguish data transformation from data comparison and adopt reasonable combinations thereof, this awareness is often lacking in the field of comparative sequence analysis. With realization of the disadvantages of alignments for sequence comparison, some typical applications use more and more so-called alignment-free approaches. In light of this development, we present a conceptual framework for alignment-free sequence comparison, which highlights the delineation of: 1) the sequence data transformation comprising of adequate mathematical sequence coding and feature generation, from 2) the subsequent (dis-)similarity evaluation of the transformed data by means of problem-specific but mathematically consistent proximity measures. We consider coding to be an information-loss free data transformation in order to get an appropriate representation, whereas feature generation is inevitably information-lossy with the intention to extract just the task-relevant information. This distinction sheds light on the plethora of methods available and assists in identifying suitable methods in machine learning and data analysis to compare the sequences under these premises.

Index Terms—Alignment-free sequence comparison, machine learning, systematization, proximity measures, data transformation, feature generation

1 Introduction

The encounter of large amounts of biological sequence data generated during the last decades and the algorithmic and hardware improvements have offered the possibility to apply machine learning techniques for real world problems in bioinformatics.

The number of accessible sequence data increased steadily [1] due to the constantly improving and newly available sequencing hardware [2]. Likewise, machine learning approaches were enhanced and accompanied by fundamentally new techniques such as deep learning approaches [3], efficient recursive methods for sequence processing [4] as well as generative adversarial networks [5] and reinforcement learning [6]. These approaches became feasible by means of massive parallel hardware developments such as specialized graphic processing units (GPUs). As a result, machine learning gained popularity in many research areas and, particularly, in bioinformatics as an alternative to classical statistical methods [7].

However, these approaches usually require huge databases and efficient implementations of the respective training algorithms supporting the hardware architectures [8]. Notably, interpretable machine learning requires a sophisticated data preprocessing for efficient training and meaningful inference. These algorithms conduct inherent data comparisons for learning.

Then again, the concepts of sequence data preprocessing and comparison are frequently not distinguished in bioinformatics [9]. However, from a machine learning perspective their delineation might be beneficial so that consistent and meaningful combinations can be made supporting the machine learning algorithm in use.

Taking this prospect, this paper aims to analyze and review existing concepts and methods of sequence data[1] preprocessing and comparison and generates a systematic framework as a result. Thereby, we concentrate on alignment-free (AF) methods, whose choice we will explain in more detail later.

In order to design this framework, we rely on a process-oriented perspective. This is due to the fact that suitable comparative measures for sequences no longer only depend on the biological application but, when aiming for self-learning procedures, also on the choice of these algorithms themselves. Their respective requirements may determine the necessary concepts in (sequence) data processing. This

• *The authors are with the Saxon Institute for Computational Intelligence and Machine Learning, University of Applied Sciences Mittweida, 09648 Mittweida, Germany. E-mail: {bohnsac1, kaden1, abel, villmann}@hs-mittweida.de.*

Manuscript received 3 June 2021; revised 29 October 2021; accepted 3 January 2022. Date of publication 6 January 2022; date of current version 3 February 2023.
This work was supported in part by the European Social Fund (ESF) under Grant 100381749 to Katrin Sophie Bohnsack, Marika Kaden, and Julia Abel. (Corresponding author: Katrin Sophie Bohnsack.)
Digital Object Identifier no. 10.1109/TCBB.2022.3140873

1. Sequence data in this article are always assumed to be nucleotide sequences, i.e., we do not consider sequences like time series as a data source.

stands in stark contrast to the widespread problem-oriented categorizations based on domain knowledge.

In particular, we propose a systematic categorization, which emphasizes two key concepts that have to be combined for a successful comparison analysis: 1) the data transformation comprising adequate mathematical sequence coding and feature generation, and 2) the subsequent (dis-) similarity evaluation of the transformed data by means of problem-specific but mathematically consistent proximity measures. This distinction sheds light on the plethora of methods available and assists in identifying suitable methods in machine learning and data analysis to compare the sequences under these premises.

The paper is structured as follows: Section 2 motivates the focus on alignment-free comparison methods, followed by the general need for their systematization from a machine learning perspective. In Section 3, we present our respectively elaborated conceptual framework. However, the description of the individual systematization modules is limited to a verbal illustration in order to not lose the reader in the details of mathematical formalism. Section 4 is dedicated to show the validity and applicability of our systematization in practice by considering common approaches from the relevant literature. Likewise, an in-text description, used to clarify the systematics parts, is omitted. The reader is advised to consult the alphabetically sorted description in the glossary (Section 6) if they are unfamiliar with a particular method or are not satisfied with a verbal description. Section 5 summarizes our key findings.

2 The Need for Systematization

2.1 Motivation From a Biological Perspective

Concepts for comparative sequence analysis are one of the most significant and far-reaching achievements in bioinformatics. Decisive for this development was the introduction of algorithms for global [10] and local [11] sequence alignments and their rapid and continuous development [12]. However, alignments make assumptions about the collinearity and evolution of the sequences to be compared, which are not necessarily valid [9]. Notably, multiple sequence alignments are computationally not viable in that their calculation is NP-hard if approximations are avoided [13], requiring the resort to heuristics [14]. Due to this computational complexity and the resulting time consumption they imply limited usability in many machine learning algorithms, which typically require huge amounts of data. Moreover, sequence alignment methods generally do not provide numerical representations of the data besides the demanded similarity value [15]. However, those representations could be valuable in subsequent data processing and machine learning for deeper data analysis.

Motivated by these deficiencies, the branch of alignment-free (AF) methods has emerged as an alternative to alignment-based approaches for the quantization of similarities between sequences [16]. Current applications of AF procedures cover, among other things, the domains of phylogenetics [17], [18], [19], (meta-)genomics [20], [21], database similarity search [22] or next-generation sequencing data analyses [23], [24], [25]. Due to their inherent design, some approaches can even be applied to unassembled sequencing reads [26] or can handle segmented genomes as they can occur in viruses [27].

As will become apparent in the course of this contribution, AF approaches usually rely on an appropriate data preprocessing and consistently designed mathematical data comparison in terms of well-defined similarity measures and categories. This aspect in particular makes AF methods interesting for users in the fields of machine learning and statistical data analysis, since respective methods usually rely on specific mathematical properties of the data and/or the similarity measure in use. The successful combination of AF approaches with machine learning techniques was demonstrated in [28], [29], [30], [31], [32], [33], [34], [35] for supervised and in [31], [36], [37], [38] for unsupervised scenarios. Selected approaches are reviewed in [39].

It is the user's responsibility to choose the data transformation and dissimilarity measure in a sense that the former is appropriate for the intended machine learning algorithm and the latter is in turn consistent with the data transformation. The data transformation may involve a task specific feature generation, whose outcome is used for quantifying the sequences' degree of compliance.

Nowadays, AF sequence comparison is an interdisciplinary field of research that constantly brings forth novel approaches. Many of these have common underpinnings, though often conveyed with different notations. It is for this reason that a comprehensive review of AF concepts is in demand.

As a matter of fact, there are reviews that classify existing approaches according to certain characteristics [9], [40]. Although, frequently no distinction is made between data preprocessing and dissimilarity evaluation of the sequences, i.e., mapping sequences to and comparing them in a feature space [35]. Further, regarding the data comparison, little attention is paid to the importance and influence of different (dis-)similarity, i.e., proximity measures [41]. Both concepts constitute relevant considerations for the objective of sequence comparison, especially in machine learning applications.

These shortcomings are consistent with the observation that only some aspects of AF methods are considered in the available contributions: For instance, most of them seem to equate the totality of AF procedures with the subgroup of word-based methods by focusing exclusively on the latter [26], [35], [41], [42], [43], [44]. Although the suitability, i.e., performance, of various AF methods was recently evaluated in the context of typical sequence comparison tasks [44], those results are not intended to reveal the method's inherent commonalities. Therefore, our focus is to proclaim the importance of dealing with the merits and pitfalls of all utilized (data transformation) components of a complex AF method in a modular thinking, rather than aiming at a plain listing of publicly available tools/software packages. Likewise, we do not aim at claiming a single method's superiority, as, in general, according to the "no free lunch" theorem [45], there will always be a use case where one method performs better than others. Thus, the choice of the AF method should not solely be based on the performance for the present task, but should also reflect biological knowledge and resulting constraints which can be interpreted in the task's context [46].

In this contribution, we particularly concentrate on research questions where the follow-up application of AF sequence comparison is not the inference of a phylogenetic tree. The latter usually relies on an evolutionary model [47] and the unweighted pair group method with arithmetic mean (UPGMA) [48] or Neighbor-joining [49] clustering algorithm. Excellent AF approaches for this purpose were presented in [50], [51], [52], [53], [54], [55], [56]. Instead, we consider applications beyond this scope such as the detection of quadruplex structures from sequences [34] or the identification of real sequence snippets among artificially generated ones [57].

2.2 Motivation Form a Machine Learning Perspective

Machine learning (ML) approaches for sequence analysis comprise, among others, algorithms for unsupervised and supervised learning [58]. Unsupervised methods are mainly considered for clustering and pattern recognition whereas supervised approaches primarily deal with classification tasks in this context [59].

Generally, at least three basic concepts can be identified as machine learning paradigms: artificial neural networks for model generation (including deep [3] and convolutional networks [60]), evolutionary/genetic algorithms for model optimization [61] and reinforcement learning as minimum information learning [6]. Here we concentrate on the first one – artificial neural networks and related methods like support vector machines (SVMs) [62].

Usually, machine learning methods assume a certain representation of the data objects as mathematical quantities, i.e., as real- or complex-valued numbers, vectors or matrices, etc. These quantities are frequently obtained by an appropriate data transformation and/or feature generation. Afterwards, the analysis of objects takes place, feeding the representation data as input into the ML-method. Multilayer perceptrons (MLPs) for classification and regression, which include deep networks, as well as SVMs, process the sample data implicitly determining decision hyperplanes by means of calculating inner products between the input vectors and weights [3]. It should be noted here that inner products generally are neither similarity nor dissimilarity measures [63]. Convolutional neural networks assume images as input and process data by means of methods which rely on image properties like convolutions, optical flow etc. [64]. Prototype based approaches like supervised and unsupervised vector quantizers compare the data with internal prototype sets in terms of an appropriately chosen proximity measure [65]. Thereby, the choice of the data representation, the proximity measure as well as the used algorithm depend on the given task [66]. Thus, the design of the data processing settings requires a thoughtful selection of these methods. Particularly, one has to keep in mind the internal processing of the data in the selected ML method. For example, proximity based methods like k-nearest neighbor (k-NN) [58] in standard implementations takes the euclidean distance but would also work with more sophisticated proximity measures taking biological domain knowledge into account. Deep networks internally compare vectorial data with network weights by means of the euclidean inner product which usually does not reflect any biological knowledge. Like (k-NN), prototype-based classifiers and cluster algorithms are distance based methods realizing a nearest prototype principle instead of nearest neighbors, which constitute more robust but easy to interpret approaches for data representation and classification on the basis of the reference principle (prototypes together with a (dis-) similarity measure), which are also successfully applied in bioinformatics [63], [67]. In the following, we address some aspects and pitfalls related to these thoughts for sequence data processing:

First, and obviously, biological sequence data are usually given as non-numerical data in the form of symbolic sequences. Hence, such data must either be transformed into a suitable numerical representation or task-specific numerical features have to be generated from them. Thereby, the subsequent ML-algorithm may restrict the choice of data representation and/or feature generation as well as the comparison measure.

Second, as mentioned above, data have to be compared in terms of either an inner product or a certain proximity measure. Both, the inner product as well as the proximity measure have to be consistent with the data representation. Thereby, the chosen proximity has to, on the one hand side, reflect the biological meaning of the task. On the other hand, the selected proximity implies particular mathematical properties, which may restrict the set of available ML-methods. For example, some unsupervised algorithms require similarities such as Affinity propagation [68], while others rely on dissimilarities such as the fuzzy c-means [69] or the Neural Gas [70] algorithm. Although similarities are generally convertible into dissimilarities, this only applies if some basic requirements (see Section 3.3) are fulfilled and the similarity is bounded [66]. Another set of examples is SVMs for classification problems which require a matrix of inner products between the data samples, where the resulting matrix has to be positive definite [71].

Distance based algorithms frequently suppose mathematical distances as dissimilarity measures fulfilling certain properties or inequalities like the triangle inequality. Yet, those properties cannot always be assumed for commonly used "distance measures" like evolutionary distances [72] or the standard correlation measure [73]. A rigorous mathematical categorization of proximity measures can be found in [66]. Moreover, different mathematical proximities may deliver equivalent results in combination with certain ML-methods. For example, a k-NN approach [74] yields the same result for the euclidean distance as the Radial basis function (RBF) kernel [75] would do [66].

Third, the obtained data representation or feature generation together with the chosen proximity may further restrict the ML-methods that can be applied to solve the problem. Several ML-methods assume specific properties for the proximity measure. For example, differentiability is usually supposed in the context of stochastic gradient descent learning variants, which is, in fact, the most applied adaptation method [3], [76]. Summarizing, the choice of the data representation, the proximity measure and the ML-method cannot be made independently and has to be related to the task in question.

Finally, there is a class of clustering and classification algorithms, which only deal with proximity matrices for the data objects as obtained, for example, by alignment methods [77]. These approaches are known as relational methods. They are characterized by the fact that they do not require the data in any explicit form (e.g., as vectors) but only the proximity relations between the data objects.

For classification learning, prominent methods based on proximity data are the relational and median variants of learning vector quantization [78], [79], whereas SVMs require similarity data as input, which can be interpreted as kernel values under certain conditions [62], [66].

For unsupervised learning, affinity propagation and relational neural gas are established for clustering and vector quantization in case of relational data, respectively [68], [80]. Fuzzy clustering based on dissimilarity data is known as relational fuzzy c-means [81].

However, these methods usually suppose specific properties of those proximity matrices like positive definiteness or the euclidean property for mathematical consistence [65]. Unfortunately, these assumptions frequently are violated in practice such that these methods are either not applicable or may yield incorrect results [82]. However, in case of small deviations from the strong setting the relational methods frequently still achieve a promising performance using moderate correction procedures [65], [66], [78].

In summary, for all these aspects of machine learning, the data preprocessing seriously influences the choice of the machine learning method. Inversely, keeping a specific machine learning approach in mind for data processing, consistent preprocessing has to be selected. At this point, we emphasize again that even if proximity data are not explicitly used as input to machine learning models, but instead vectorial data, the data comparison is computed using inherent mathematical methods of the algorithms. This contrasts with the relational and median machine learning methods, where data relations serve as input but which may also suppose certain mathematical properties to be fulfilled.

Hence, a process-oriented view on the whole data preprocessing is necessary, which will be introduced in the following. It is remarked that although the framework is motivated by a machine learner's perspective, its scope is considerably broader: the framework remains valid for any desired data processing technique in sequence analysis other than machine learning procedures.

3 The Process-Oriented Systematization

Taking a process-oriented perspective, we present a conceptual framework for AF sequence comparison. It encompasses three separable modules (sub-tasks) connected in series, which all are interdependent and should relate to the given task:

1) *Coding*: A sequence coding may take place to get an appropriate data representation.
2) *Feature generation*: A feature generation can be applied to extract information relevant for the task.
3) *Proximity measure*: A proximity measure has to be selected for data comparison chosen consistently with the features and/or representation as well as to the required mathematical properties of the ML-method in question.

In the following, we briefly present a characterization of these modules and introduce an uniform notation to overcome the hurdles of communication and understanding between researchers from different research areas participating in the work on AF methods. Of particular importance is the chosen delineation of coding and feature generation, as these concept notations are often not precise enough or used synonymously in the literature.

Coding. In this paper, we define coding as a data transformation or mapping of the source X into a coded data (target) Y provided that all information of the source is preserved. Accordingly, we demand that a coding is information loss-free. This permits the original data to be accurately rebuilt (exactly recovered) from the coded data[2]. Succinctly stated, a coding transforms the data representation but preserves the information. We remark at this point that this definition of coding allows adding of information during the coding process. This implies that the information content of the mapped data might be more than that of the source X. For example, mapping text to a vector representation adds the information that the coded data are now elements of a vector space equipped with the respective mathematical properties, which of course cannot be attributed simply to texts. Another possibility of information addition is to integrate domain knowledge into the coding.

Feature Generation. It is understood as a data transformation with a certain intention regarding the further data processing (task) in mind. Hence and in contrast to coding, it is inevitably accompanied by information loss.

Coding and feature generation together constitute the overall data transformation which sometimes is also considered as preprocessing for the machine learning application.

Proximity Measure. The final but crucial step in this framework constitutes the comparison of the transformed data by quantifying their shared or discriminating features, i.e., object commonalities and differences [84]. This is achieved by applying an appropriately chosen proximity measure, i.e., (dis-)similarity measure [66], which has to be consistent with the transformed data as well as with the subsequent machine learning approach.

It is emphasized that coding as well as feature generation are performed for each of the sequences individually, while, obviously, any proximity measure requires a data pair to calculate their respective (dis-)similarity value. Fig. 1 visualizes this relation.

In the following, we explain these three modules in detail and provide a theoretical characterization, before investigating their applicability for exemplary sequence comparison procedures in Section 4. Thereby, if a concept/tool is *cursively* typeset, it is explained in the glossary (Section 6) for better readability.

2. It should be noted that our definition of coding differs from that common in coding theory [83] where reversibility is introduced as an additional property of a code (uniquely decipherable).

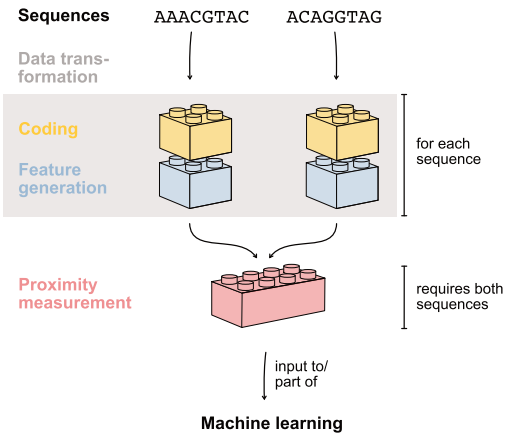

Fig. 1. Schematic overview of the systematization's categories for AF sequence comparison in proximity-based machine learning: coding, feature generation and proximity measurement.

3.1 Sequence Coding

First of all we turn to define more precisely the coding types related to sequence analysis as required for RNA/DNA investigations. Although various categorizations of those coding types would be possible, we distinguish from a process-oriented point of view: context-dependent, element-wise and holistic coding. This distinction allows to specify how biological domain knowledge is incorporated into the data transformation. Subsequent to the definitions of these coding types, we provide respective tangible examples to illustrate their discrepancies. At this point, we only give a literal definition of these sub-categories in order to avoid a mathematical overload.

Roughly speaking, we now assume an alphabet (set) of symbols/elements. A sequence is a concatenation of those elements. Short sequences with particular meaning are also denoted as words over the alphabet. We distinguish source and code alphabets.

We start with the context-dependent coding as it presents the most common form of the mentioned ones:

Definition 1 (Context-dependent coding). *In a context-dependent coding, the mapping of a sequence element from the source alphabet to an element of, or word over the code alphabet depends on its position in the sequence and its neighborhood, i.e., (portions of) the sequence itself.*

Example 1. We consider the *k-mer coding* [85] as a context-dependent coding, where the context of a sequence element is determined by the $k-1$ surrounding sequence elements. This can be considered as encoding a nucleotide sequence into a sequence of words over the nucleotide alphabet.

A special case of context-dependent coding constitutes the recursive coding. For this, the considered neighborhood comprises solely the direct predecessor of the element to be mapped. By means of a recursive coding, a nucleotide is mapped onto a number or a vector, but the assignment is dependent upon that of the previous nucleotide in the sequence:

Definition 2 (Recursive coding). *The recursive coding is an iterative scheme. The mapping of a considered sequence element depends on the result of the mapping of the previous element(s) and the current element itself.*

Example 2. A recursive coding is given by *Universal Sequence Maps* (USMs) [86], whose foundations lie in the *Chaos Game Representation* (CGR) [87]. USMs are claimed to be bijective [88], [89] and, hence, constitute a coding. Another example constitutes the *2D DNA walk* [90].

A more trivial coding is the element-wise coding. Thereby, the reliance on any context given by the sequence is dropped:

Definition 3 (Element-wise coding). *The element-wise coding describes the mapping of each nucleotide of a sequence onto an individual element of or word over the code alphabet. The coding is independent from the context given by the sequence itself.*

Example 3. There are numerous approaches incorporating an element-wise coding. The *Integer-* [91] as well as the *Real number representation* [92] are listed here. The latter also shows that knowledge can be included into the coding process, in this case the classification of bases into purines and pyrimidines. Further examples are the Voss representation [93], i.e., *One-hot coding*, as well as the *Tetrahedron representation* [94] and the *Three sequence method* [95]. For the first two we recognize that the code alphabet is of numerical type, whereas in the latter case it is of symbolic type. Interestingly, we also can consider vectors as words over an alphabet which is the $\mathbb{R}^n$.

In the special case that the context is given by the whole sequence, we introduce an extra category named holistic codings:

Definition 4 (Holistic coding). *For a holistic coding, the elements of the target (encoded sequence) are determined in dependence of the properties of the whole sequence.*

Example 4. *Natural vectors* [96], [97] as well as the *Fourier transform* [98] and the *Wavelet transform* [99] are prominent examples for holistic codings. *Natural vectors* consider statistical properties, whereas both previous domain transforms rely on frequency properties. It is proven in [96] that the correspondence of a sequence and its *Natural vector* is one-to-one, justifying it to be a coding. In fact, the *Fourier transform* belongs to coding procedures due to the Fourier inversion theorem [98].

At this point it becomes apparent that some coding schemes as e.g., the *Fourier transform* may require another appropriate coding beforehand, in order to have a numeric data representation as input. We address this peculiarity and how it fits nicely into our scheme explicitly in Section 4.2.

3.2 Feature Generation

According to our process-oriented systematization, the overall aim of feature generation is to condense information in terms of the machine learning task and to establish data comparability regarding the proximity measure. In this view, the feature generation differs from a coding in that it is inevitably accompanied by information loss, though its

intention is to preserve most of the "relevant information" for the task in mind.

Generally, the term feature refers to any descriptive quantity or property which can be extracted from the data sample, here being an input sequence. Thereby, the underlying process of determination has to be consistent with, or rather motivated by mathematical concepts from statistics, information theory and frequency analysis but may also apply biological domain knowledge. Hence, the extracted features may reflect both, the mathematical and/or the biological properties, of the applied approaches. At this point, it should only be mentioned that features can also be generated or extracted by means of a machine learning method [100], [101].

Because feature generation comprises a huge variety of methods, it is difficult to divide them into subcategories that are all-encompassing. Further, many methods cover several aspects. For example, frequency analysis takes the particular perspective of signal processing and draws on respective methods but also counts as a statistical method. For this reason, here, the term feature generation covers all data transformations that are, on the one hand side not loss-free and therefore, do not constitute a coding, and on the other hand, do not meet the requirements of the definitions of proximity measures in Section 3.3.

Frequently, feature generation is followed by a feature selection or feature weighting. We aggregate all these methods simply as feature generation. Further, one can distinguish reference-free features from reference-dependent features. For the latter one, the features are obtained taking into account a reference set, i.e., one or multiple additional sequences originating from the data set under consideration, such as demonstrated in [102] or in [31]. Another possibility is to take other domain knowledge for that. For the reference-free feature generation, external information is not required, i.e., only the information contained in the considered sample is extracted.

Example 5. A widely applied feature generation in sequence processing is the *Bag of words* [85] approach relying on the frequency statistics of words from a *k-mer encoded* sequence. Further, *normalization* of vector data may lead to an information loss. According to our understanding of feature generation, the approaches *1D-DNA-walk* [103] and the *Paired numeric representation* [104] belong to this category since the mapping is not reversible, i.e., the original sequences cannot be recovered from the encoded samples. The frequently used *Power spectrum* [98] of a *Fourier transform* constitutes a feature generation, because the Fourier transform, which itself is a one-to-one mapping, cannot be reconstructed from the spectrum. If, however, a reduced number of Fourier features of a (complete) *Fourier transform* are used, Fourier analysis constitutes a feature generation instead of being a coding. Analogously, an incomplete calculation of *Natural vectors* (restricted number of considered moments) is no longer a coding but yields a feature generation.

At this point it should be noted that a feature does not necessarily need to be quantitative. For example, a descriptive property can also be of categorical type (present or not) or even symbolic (textual).

Example 6. Consider the mapping of a single sequence element onto the IUPAC degenerate base notation [105]. Here the element (base) is assigned to either the group of purines (R) or pyrimidines (Y). Obviously, this is neither a coding nor a quantitative property.

3.3 Proximity Measurement

When speaking of (dis-)similarities and distances between objects one needs to be strict in notation as well as in interpretation. The latter one is of particular importance because different similarity types show different mathematical properties which may be important for subsequent data processing and machine learning. We distinguish them in line with our process-oriented point of view but follow the exact mathematical definitions presented in [66]:

Definition 5 (Primitive dissimilarity measure). *A primitive dissimilarity measure is a map* $d : X \times X \to \mathbb{R}$ *fulfilling at least the following requirements* $\forall\ \boldsymbol{x}, \boldsymbol{y} \in X$

1) *Minimum principle:* $d(\boldsymbol{x}, \boldsymbol{x}) \leq d(\boldsymbol{x}, \boldsymbol{y})$ *and* $d(\boldsymbol{x}, \boldsymbol{x}) \leq d(\boldsymbol{y}, \boldsymbol{x})$
2) *Non-negativity:* $d(\boldsymbol{x}, \boldsymbol{y}) \geq 0$.

Definition 6 (Primitive similarity measure). *A primitive similarity measure* $s : X \times X \to \mathbb{R}$ *realizes, in contrast, a maximum principle* $s(\boldsymbol{x}, \boldsymbol{x}) \geq s(\boldsymbol{x}, \boldsymbol{y})$ *and* $s(\boldsymbol{x}, \boldsymbol{x}) \geq s(\boldsymbol{y}, \boldsymbol{x})$ *in addition to the non-negativity. For basic similarities, the positiveness property is dropped.*

Hence, both definitions state that an object is most similar to itself as a basic principle in agreement with interpretations in cognitive science [84]. A more strict dissimilarity measure is a distance or, equivalently, a metric.

Definition 7 (Metric). *A metric is a dissimilarity measure fulfilling the following properties* $\forall\ \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in X$

1) *Symmetry:* $d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$
2) *Positive definiteness:* $d(\boldsymbol{x}, \boldsymbol{y}) \geq 0$ *and* $d(\boldsymbol{x}, \boldsymbol{y}) = 0 \Leftrightarrow \boldsymbol{x} = \boldsymbol{y}$
3) *Triangle inequality:* $d(\boldsymbol{x}, \boldsymbol{y}) \leq d(\boldsymbol{x}, \boldsymbol{z}) + d(\boldsymbol{z}, \boldsymbol{y})$.

Example 7. An example for a dissimilarity measure, which is not a distance, is the *Kullback-Leibler divergence* [106]. It is no metric because it violates the symmetry property.

The *RBF kernel* [75] as well as the *Cosine measure* [66] as normalization of an inner product are similarity measures. Inner products themselves are, contrary to popular belief, neither primitive similarity nor basic similarity measures since they may violate both, the maximum principle and the non-negativity [66].

In the following, we denote primitive (dis-)similarities simply as (dis-)similarities or proximities. But for distances and respective metrics we explicitly require the properties given in Definition 7. For detailed considerations of types of proximities we refer to [66].

It should be remarked that in many publications the term distance is not used in its mathematical strictness, i.e., not all of the three properties are necessarily valid/fulfilled. Thus a careful consideration is demanded. In this work, however, measures denoted as distance have to fulfill the metric properties. Otherwise we denote them as dissimilarities.

Several proximities suppose specific data requirements. This concerns data structures like vectors, matrices, symbols (text) or others to be compared [107] as well as numeric conditions. For example, divergences can only be applied to vectorial data if $x_i \in [0,1]$ and $\sum x_i = 1$. If, however, only the first argument holds, generalized divergences may be taken into consideration.

Further, there might be technical or computational requirements by the machine learning method for the proximity measure. For example, symmetry is not always required whereas differentiability is assumed for machine learning approaches based on stochastic gradient learning.

Example 8. The frequently applied Levenshtein distance [108] is not differentiable. Divergences like the *Kullback-Leibler-divergence* [106] or the Rényi-α divergence [109], which are the basis for information theoretic models in machine learning, are non-symmetric dissimilarities.

For further examples of proximities we refer to [110] and with a more biological context to [41].

4 The Systematization in Practice

In this section, we demonstrate the validity and applicability of our systematization by breaking down common alignment-free sequence comparison methods into their components coding, feature generation and proximity measurement. In doing so, we explain how these blocks work together and have to be consistently combined.

According to our best knowledge, the vast majority of available sequence comparison approaches can be fit smoothly into this scheme. Though, we acknowledge that a clear confirmation of this statement is, due to the sheer mass of publications on this subject, beyond our capabilities. Thereby, we neither claim completeness nor do we evaluate the methods appropriateness for a given task.

We distinguish simple combinations of these blocks from nested combinations. For simple combinations, repeated application of blocks is prohibited, for example two codings in a row. In contrast, nested combinations allow the repeated coding as well as repeated feature generation. Examples for simple combinations are investigated in Section 4.1, examples for nested combinations are studied in Section 4.2.

In the following, we only roughly describe the methods such that the categorization into the proposed scheme becomes visible. The methods blocks are precisely described in the glossary (Section 6), so that the reader is not confused by the details here. This is indicated by *cursive* typesetting of the respective data transformations and proximity measures.

4.1 Simple Combinations

Simple combination does not necessarily assume that all three blocks are present. More specifically, the coding and/or the feature generation blocks can be omitted depending on the available data structure and the investigation task. Obviously, the proximity part is mandatory for comparison analysis.

Therefore, we can consider four cases of combinations as depicted in Fig. 2. In the following we give examples for each case and show how these examples fit into the scheme.

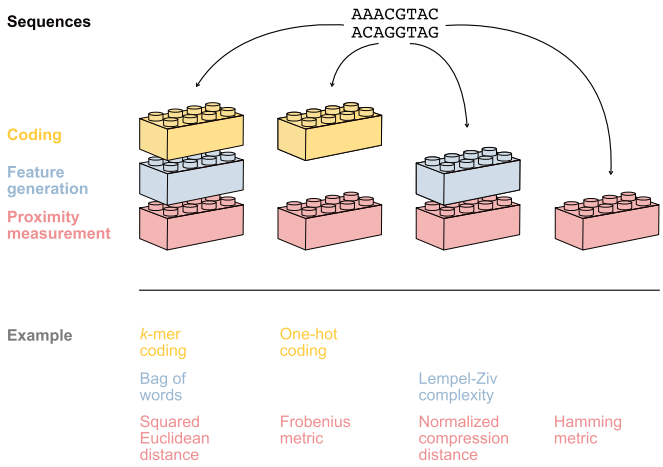

Fig. 2. Demonstration of different ways to combine the modules coding, feature generation and proximity measurement to assess two sequences (dis-)similarity.

4.1.1 Case 1 – All Three Blocks are Present: Coding, Feature Generation and Proximity Measurement

One of the most commonly used alignment-free methods is combining the (context-dependent) *k-mer coding* with the *Bag of words (BoW)* feature generation followed by an adequate proximity measure. In its simplest form this was introduced by [85] taking the squared *euclidean distance* between the word statistics vectors as proximity. The well-known D_2 statistic [41], [111] is based on the same coding and feature generation combination. The subsequently applied *euclidean inner product* is unfortunately not a proximity measure, as emphasized before. In difference to the last approach, in [18] the *BoW* allows wildcards of length n within a word. Further, the used proximity measure is derived from the D_2 statistic.

Yet, the *k-mer* coding offers the possibility for other feature generation than *BoW*. An example is proposed in [112], where truncated *Natural vectors* are generated from the k-mer distribution of the sequence. In fact, this is not a coding because only a restricted number of moments is considered as described in Section 3.2. The dissimilarity measure was chosen to be a re-scaled *cosine measure*.

If k-mer sets are constructed from a k-mer encoded sequence, this constitutes another frequent feature generation since the frequency and position information is lost. Comparison is achieved by set similarities such as the *Jaccard similarity coefficient*, also known as Tanimoto coefficient. It should be remarked that this procedure is only conclusive given large values of k during the coding procedure, as otherwise sequences tend to share all short k-mers.

In [113], the Multiple encoding vector is proposed, where the coding is to generate a symbolic matrix via the *Three Sequence Method* (element-wise), the feature generation is the calculation of *natural vectors* simultaneously on each row of this matrix (the variance is the highest considered moment) and the similarity is chosen to be the *euclidean distance*.

An approach relying on a recursive coding scheme, namely the *Chaos Game Representation* (CGR), is presented in [89], where oligonucleotides are represented as data points in a two-dimensional CGR space (plane) obtained by a recursive mapping procedure. As stated by the authors, this

mapping is an one-to-one mapping and, hence, a coding. For feature generation this plane is partitioned by a rectangular grid into cells. The number of points in each cell is determined, which then is denoted as Frequency CGR (FCGR). The FCGRs of sequences are compared by means of a weighted *Pearson correlation coefficient*.

4.1.2 Case 2 – Only Coding and Proximity Measurement are Present

Generally, if sequences of different lengths have to be compared, a direct application of a proximity measure after coding is impossible, because the coding preserves the complexity according to the one-to-one property.

If though, this assumption is fulfilled, we can directly apply a consistent proximity measure onto the encoded sequences. For instance, sequences in the *one-hot coding* can be compared by means of a matrix dissimilarity, e.g., the *Frobenius distance* induced by the Frobenius norm. Likewise, sequences in the *Integer coding* [91] can be compared by vector proximities like the *euclidean distance* or a *correlation measure*.

Applying a proximity measure just after coding is not limited to numeric codings. For example, the *Hamming metric* can be taken to compare words, i.e., the *k-mer coding*.

Given a recursive coding as in [114], we can interpret the encoded sequences as functional data. A geometrically inspired proximity measure based on L_p*-norms* is proposed in [115] where time series are considered as discretization of continuous functions of time (in our case sequence position). Functional metrics in mathematics are related to Sobolev-norms based on L_p*-norms* [116].

4.1.3 Case 3 – Only Feature Generation and Proximity Measurement are Present

One of the most prominent examples is the *Lempel-Ziv complexity* [117] of a sequence based on the principle of minimum description length of objects. Those complexity values can be consistently compared by the *Normalized compression distance (NCD)* [118]. Several variants of *NCD* are known using different normalizations and/or compressors [119] yielding a distance measure under certain conditions. Particular complexity measures for nucleotide sequences taking biological domain knowledge into account, such as the DSIC complexity introduced by [120] can also be considered for *NCD*.

In [121] a feature generation based on the Averaged Mutual Information (AMI) between nucleotide distributions within a sequence for a given range k was suggested. Then, the AMI-values for different k-ranges build the AMI profile (feature) vector, which can be compared by any vector proximity measure, e.g., *correlations*. The further development of this approach by [122] also falls into this category.

As described in Section 3.2, approximations in the *Natural vector* using a restricted number of moments constitute a feature generation that is frequently combined with the *euclidean distance* [32], [96].

4.1.4 Case 4 – Only Proximity Measurement is Present

There are a number of proximity measures that can work directly on the non-transformed sequences, i.e., these proximities work also for non-numeric inputs.

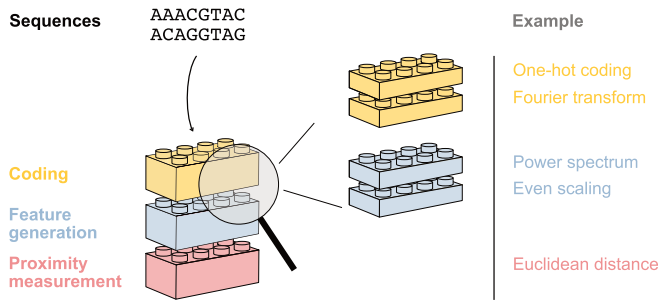

Fig. 3. Illustrating the possibility for nested codings and feature generations. The right-side example refers to the approach presented in [128].

A prominent example is the *Hamming metric* [123], which simply counts the number of nucleotide disagreements. If the measure includes a normalization step to transform absolute to relative disagreements, this is known as p-distance in biology [107], and should be carefully distinguished from L_p*-distances* which are sometimes also abbreviated as p-distances.

Another non-numeric dissimilarity measure for sequences is the Levenshtein metric [108], though this is sidelined here as it represents the basis for all alignment procedures. It is a variant of the more general earth-mover's-distance [124] and constitutes an optimization problem.

The *Eigen-McCaskill-Schuster distance* denotes the number of transversions (purine ↔ pyrimidine) [107] and, thus is working directly with sequences. However, one could decompose the comparison process into a feature generation, which maps the nucleotides to a one-letter representation of either purines or pyrimidines [105] followed by the *Hamming metric* measurement.

Other frequently applied proximities based on text are the *Average common substring distance* [125] or the related concepts of k-mismatch Averange common substring [126] and Shortest unique substring [127].

4.2 Nested Combinations

As mentioned above, nested combinations allow repeated codings as well as repeated feature generations. These repetitions are frequently convenient in terms of better interpretabilty of the data pre-processing before comparison and/or to obtain suitable structures for the comparison. For the following examples we restrict ourselves to only present the nested coding approach or the nested feature generation process independently, not considering combinations thereof, which might be available. Further, we omit the subsequent proximity measurement step to avoid a technical overload. An illustrative example of nested coding and feature generation is depicted in Fig. 3.

4.2.1 Nesting Codings

For nested codings, first we present approaches with an initial *k-mer coding*. For the subsequent coding we consider the following: In [33] the individual k-mers are subsequently *one-hot encoded*. In [31], the k-mers are uniquely mapped onto complex numbers before applying even a further coding, the *Wavelet transform* onto it. In the PN-curve approach from [129], a k-mer coding is followed by an element-wise coding based on the *integer representation* peppered with some

additional information, namely the k-mer's cumulative occurrence (counting) and a continuous number (sequence position). The recursive encoding of the k-mer sequence constitutes another promising idea falling into this category.

Second, we consider selected coding approaches that follow an initial numeric coding: Numerical coding yields a vector or matrix representation of a sequence. These representations frequently are further coded by the *Fourier* or *Wavelet transform*. We emphasize that both transformations are reversible and, hence, really are codings. Depending on the numeric encoding (vector/matrix), 1D- or 2D transformations can be applied. An *Integer coding* combined with the *Fourier transform* was proposed by [91] or combined with the *Wavelet-transform* in [36] using the Haar wavelet. In [90] the combination of a *1D DNA walk* with a *Wavelet transform* was investigated. We remark that *Wavelet* and *Fourier transform*, if applied to representations of different lengths, are to be compared carefully. Truncation of the complete Fourier or wavelet coefficients is a kind of feature generation (here feature selection). Yet, avoiding this difficulty applying *zero-padding* [91] still gives a (nested) coding for these transformations supposing that the nucleotides are coded neither as zeros nor as zero vectors for 1D- and 2D transformations, respectively.

As a last example for nested codings, we consider initial recursive coding in combination with visualization techniques as subsequent coding: A popular approach, though often solely used for visualization purposes, considers the graph (2 or 3D plot) of a recursively encoded sequence. This plotting may, under certain premises, constitute a reversible coding. This applies e.g., to the plotting of *CGR* coordinates proposed in [89]. If we take this view, we emphasize that recursive codings first yield vectorial representations [130] which afterwards are uniquely embedded into the 2D-plane. Yet, the authors presented their approach as a single step coding into a visualization space denoted as the CGR-space. A similar approach was presented in [114], where the visualization mapping yields a curve instead of a point distribution in a 2D-plane. We remark that a plotting of encoded sequences into a visualization space is not a coding if the mapping is not one-to-one, see for example [131]. These visualizations can be compared by appropriate distance measures used in respective machine learning models like vector quantizers [132]. Convolutional neural networks [60] directly take images as inputs but internally rely on inner products and, hence, are not proximity-based.

4.2.2 Nesting Feature Generations

Usually, before conducting an analysis, a data preprocessing takes place to extract or generate the relevant information for the task in mind. Frequently, not all of these information can be obtained by a single processing step. More realistically, several features are generated by different methods in line which constitutes a nested approach. In the following we illustrate a selection of respective approaches based on *k-mer encoded* sequences:

A convenient nesting of feature generations is to first apply a *Bag of words* histogram analysis as an initial feature generation step, followed by a *normalization* to obtain relative word frequencies as the second step. The finally resulting relative histogram vector constitutes the vector of features, which allows an interpretation as discrete representations of probability densities. This motivates their comparison using divergences [57]. Exactly this scheme is realized in [133] applying the *Jensen-Shannon divergence*. Without *normalization* the described procedure is no longer nested but rather a simple combination (see Section 4.1) of the blocks well-known from [85] taking their the squared *euclidean distance* as proximity measure.

Yet, the extracted *BoW* features can also be normalized in other ways: The Composition vector (CV) considered in [134] as well as the D_2^S approach [41], [111] apply a background adjustment, i.e., the subtraction of random background and expected frequencies, respectively.

In [135], a bunch of feature generations is applied after the *k-mer coding*: the *BoW*-histograms of two sequences are paired, statistical descriptors of concordant/discordant pairs are generated and, finally compared by the Kendall correlation statistic as proximity measure. This method is denoted simply as K_2 by the authors.

Of course, other features than *BoW* can be extracted. For instance the so-called Distance Measure based on k-tuples (DMk) approach [136] as well as the Return Time Distribution (RTD) method [137] utilize multiple feature generation steps in a row relying on determining the interspaces between the words in the the k-mer sequence and the statistics and information theoretic features thereof.

Obviously, both, nested coding as well as nested feature generation can be combined within one complex alignment-free sequence comparison approach. Examples can be found in [37] or [128]. However, this might be crucial: the individual parts must always be consistently chosen and carefully deliberated. In particular, one has to take care of a precise determination of information preserving operations for coding and non-preserving feature generation. Only keeping protocol of these processes allows a meaningful interpretation of the results.

5 Conclusion

Applying machine learning methods in the context of molecular biology has been gaining increasing interest during the last years and provides alternatives to classical statistical approaches. As stated in [7] "machine learning approaches are best suited for areas where there is a large amount of data but little theory", which in fact applies to many bioinformatics data analysis tasks. However, these tools usually have explicit but also implicit data requirements which crucially influence the results and their interpretation. The users of the methods are frequently unaware of these constraints. Otherwise, the user has to select one of the many available data transformation approaches which may be accompanied by information loss and/or incorporate domain knowledge. This selection has to be mathematically consistent with the machine learning method. Hence, a clear distinction of 1) data transformations to obtain suitable representations from 2) data comparison by means of proximity measures or inner products is beneficial. Therefore, we presented a respective framework decomposing the full data preparation before machine learning application into sequence data coding, feature generation and subsequent comparison. This modularization

allows to compare different strategies according to their mathematical and information properties in order to achieve a machine learning consistent data representation. We explicitly categorized a selection of already proposed approaches into this scheme to show its validity. Further, it provides inspiration to combine modules from several approaches to improve the capability and the interpretability: the decomposition into the integral components helps to show similarities and differences of the methods and allows conclusions to be drawn about the information used. Additionally, this systematization helps the biology experts to recognize data representation problems depending on the task and select appropriate plausible machine learning approaches or to communicate these challenges to machine learning experts. Likewise, machine learners may profit by including biological domain knowledge into their model design for better interpretability. Now that a scheme has been established to assess or rather classify the methods information preservation or loss, future analyses should also reflect upon the approaches time complexity, i.e., runtime and, hence compatibility with and applicability in the big data era.

6 Glossary of the Described Methods and Concepts

In the following, methods and concepts for data transformations and proximity measurement, which have been mentioned in the paper, are briefly described in the upcoming two subsections, respectively. For the best usability, we ordered the methods alphabetically.

For each method, we give a short description accompanied by the respective key characteristics. For data transformation approaches, this includes 1) the method's classification as either coding or feature generation as well as 2) common follow ups of the respective methods. For proximity measures and related inner products, we conversely provide a) the method's classification as similarity, dissimilarity or distance as well as b) their input requirements.

6.1 Data Transformation Methods

Let $S = [s_i]_{i=1,\dots,n}$ denote a sequence with length n. The elements s_i belong to an alphabet $\mathcal{A} = \{\alpha_j\}_{j=1,\dots,n_{\mathcal{A}}}$. Thus, the cardinality of $\mathcal{A}$ is $n_{\mathcal{A}}$. Further, $n_\alpha(S)$ denotes the number of occurrences of an $\alpha \in \mathcal{A}$ within a sequence S. We simply write n_α if it is clear from the context which sequence S is considered.

Bag of Words (BoW)

Generation of feature vectors of length $(n_{\mathcal{A}})^k$ containing the counts of all possible k-mers regarding $\mathcal{A} = \{A, C, G, T\}$ [85]. The term BoW is only rarely found in the bioinformatics community, though the notation is used here to be consistent with its origin in natural language processing and machine learning [138].

Classification: feature generation *Follow ups:* normalization (feature generation), vector proximity measure

Chaos Game Representation (CGR)

The sequence S is transformed into a matrix $\boldsymbol{X} \in \mathbb{R}^{2\times n}$. For this, first, each symbol of the alphabet $\mathcal{A} = \{A, C, G, T\}$ is assigned to a corner of the unit square in $\mathbb{R}^2$ located at the origin [87]. The coding starts selecting randomly a start point x_0 inside the square. The sequence elements s_i are iteratively coded into vectors $\boldsymbol{x}_i \in \mathbb{R}^2$ to generate $\boldsymbol{X}$ by

$$\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \frac{1}{2}(\boldsymbol{y}_i - \boldsymbol{x}_i), \tag{1}$$

whereby

$$\boldsymbol{y}_i^T = \begin{cases} (0,0) & \text{if } s_i = A \\ (0,1) & \text{if } s_i = C \\ (1,0) & \text{if } s_i = G \\ (1,1) & \text{if } s_i = T \end{cases}, \tag{2}$$

and $i = 1, \dots, n$ [139].

Classification: coding (recursive) *Follow ups:* Frequency CGR (feature generation), matrix proximity measure

DNA-Walk (1D)

This method returns a vector $\boldsymbol{x} \in \mathbb{R}^n$ according to

$$x_i = \begin{cases} x_{i-1} + 1 & \text{if } s_i \in \{C, T\} \\ x_{i-1} - 1 & \text{otherwise} \end{cases}, \tag{3}$$

for $i = 1, \dots, n$ with the initialization $x_0 = 0$. Thus, it includes the biological domain knowledge of nucleotides belonging to the group of either purines or pyrimidines [103].

Classification: feature generation *Follow ups:* Fourier transform (coding), Wavelet transform (coding), vector proximity measure

DNA-Walk (2D)

This method returns a complex valued vector $\boldsymbol{x} \in \mathbb{C}^n$ according to

$$x_i = \begin{cases} x_{i-1} + 1 & \text{if } s_i = A \\ x_{i-1} - 1 & \text{if } s_i = G \\ x_{i-1} + \mathrm{i} & \text{if } s_i = T \\ x_{i-1} - \mathrm{i} & \text{if } s_i = C \end{cases}, \tag{4}$$

for $i = 1, \dots, n, \mathrm{i}^2 = -1$ and initial value $x_0 = 0$ [90]. This procedure extends the *DNA walk (1D)* to the complex plane which now turns out to be a coding.

Classification: coding (recursive) *Follow ups:* Fourier transform (coding), Wavelet transform (coding), vector proximity measure

Fourier Transform

A mathematical method to describe a sequence signal in the frequency domain. The Discrete Fourier transform (DFT) of a numerically encoded sequence $\boldsymbol{x}$ can be written as

$$DFT(k) = \sum_{m=0}^{n-1} \boldsymbol{x}_m \; e^{-\mathrm{i}\frac{2\pi}{n}km}, \tag{5}$$

where $\mathrm{i}^2 = -1$. The vector consisting of the values $DFT(k)$ is denoted as spectrum.

Classification: coding (holistic) *Follow ups:* Power spectrum (feature generation), frequency selection (feature generation), denoising/filtering, matrix proximity measure

Integer Representation

This methods returns a vector $\boldsymbol{x} \in \mathbb{R}^n$. Each of the four nucleotides in the original sequence is replaced by a unique fixed integer value using the mapping rule: $A \mapsto 0$, $C \mapsto 1$, $G \mapsto 2$ and $T \mapsto 3$ [91].

Classification: coding (element-wise) *Follow ups:* Fourier transform (coding), Wavelet transform (coding), vector proximity measure

k-mer Coding

The k-mer coding returns a sequence W of words w_i (k-mers) over the alphabet $\mathcal{A}$ with length k. A k-mer refers to a segment of the sequence with k symbols, where $k \leq n$. Frequently, the words are obtained using a sliding window of length k to go through the sequence S from position 1 to position $n-k+1$ step by step [40]. Consequently, the number of possible words is $n_w = (n_{\mathcal{A}})^k$.

Classification: coding (context-dependent) *Follow ups:* Bag of words (feature generation), one-hot-coding (element-wise coding), matrix proximity measure

Lempel-Ziv complexity

This complexity measure known from theoretical computer science returns the quantity $C_{LZ}(S)$, which is defined as the least number of steps necessary for the synthesis of S. At each step of the synthesis only one of the following procedures is permitted: 1) the copying of the longest fragment which was already synthesized and 2) the generation of an additional symbol to ensure uniqueness of each component. It is an approximation of the Kolmogorov complexity as an universal complexity measure [117].

Classification: feature generation *Follow ups:* Normalized compression distance

Natural Vectors

This method returns a vector $\boldsymbol{x}$ containing statistical descriptors of the sequence. The statistical descriptors are:

1) The frequencies n_α of the four nucleotides $\alpha \in \mathcal{A} = \{A, C, G, T\}$.
2) The mean values μ_α of the total distances T_α for the four nucleotides according to: $T_\alpha = \sum_{i=1}^{n_\alpha} t_{\alpha i}$ with $t_{\alpha i}$ being the position of the ith occurrence of α minus one. μ_α is obtained as $\mu_\alpha = \frac{T_\alpha}{n_\alpha}$.
3) The normalized central moments

$$M_\alpha^j = \sum_{i=1}^{n_\alpha} \frac{(t_{\alpha i} - \mu_\alpha)^j}{n_\alpha^{j-1} n^{j-1}}, \tag{6}$$

for $j = 2, \ldots, \max_{\alpha \in \mathcal{A}}(n_\alpha)$ and $M_\alpha^j = 0$ if $j > n_\alpha$ [96]. Frequently, it is stated that, in dependence of the data set, even central moments of second order are sufficient to maintain this relationship [96], [112], though this cannot be taken as the rule.

Classification: coding (holistic), only the first $k < \max_{\alpha \in \mathcal{A}}(n_\alpha)$ moments are calculated (feature generation) *Follow ups:* normalization (feature generation), vector proximity measure

Normalization

Normalization of a vector, is understood here as a general re-scaling of the vector entries whereas in mathematics normalization is related to a vector norm and re-scales the vector in a way that after normalization it has the norm equal to one. Generally, after normalization we cannot reconstruct the original vector from the normalized vector itself.

Classification: feature generation *Follow ups:* vector or matrix proximity measure

One-Hot Coding

It returns a matrix $\boldsymbol{X} \in \mathbb{R}^{n_{\mathcal{A}} \times n}$ with components

$$x_{\alpha i} = \begin{cases} 1 & \text{if } s_i = \alpha \in \mathcal{A} \\ 0 & \text{otherwise} \end{cases}, \tag{7}$$

for $i = 1, \ldots, n$ [128]. The ith column vector $\boldsymbol{x}_i$ of $\boldsymbol{X}$ corresponds to a corner of the $n_{\mathcal{A}}$-dimensional hypercube. This coding is known as Voss representation if $\mathcal{A} = \{A, C, G, T\}$ [93].

Classification: coding (element-wise) *Follow ups:* Fourier transform separately for each of the rows (coding), matrix proximity measure

Power Spectrum

The power spectrum is the absolute square of the *Fourier transform*. For the discrete Fourier transform of a vector $\boldsymbol{x} \in \mathbb{C}^n$ the kth component reads as

$$PS(k) = |DFT(k)|^2, \tag{8}$$

with $k = 0, 1, \ldots$.

Classification: feature generation *Follow ups:* vector proximity measure, spectral interpretation

Real Number Representation

This method returns a vector $\boldsymbol{x} \in \mathbb{R}^n$. It works equivalently to the *integer representation* but takes the mappings: $A \mapsto -1.5$, $C \mapsto 0.5$, $G \mapsto -0.5$ and $T \mapsto 1.5$. Thus, purines (A and G) are replaced by negative and pyrimidines (C and T) by positive real values [92], thereby incorporating biological domain knowledge.

Classification: coding (element-wise) *Follow ups:* Fourier transform (coding), Wavelet transform (coding)

Tetrahedron Representation

It returns a matrix $\boldsymbol{X} \in \mathbb{R}^{3 \times n}$ and was introduced in [94]. It is only applicable to alphabets of cardinality 4. If $\mathcal{A} = \{A, C, G, T\}$, each nucleotide corresponds to the vertex of a tetrahedron by mapping $A \mapsto (0, 0, 1)^T$, $C \mapsto (-\frac{\sqrt{2}}{3}, \frac{\sqrt{6}}{3}, -\frac{1}{3})^T$, $G \mapsto (-\frac{\sqrt{2}}{3}, -\frac{\sqrt{6}}{3}, -\frac{1}{3})^T$ and $T \mapsto (\frac{2\sqrt{2}}{3}, 0, -\frac{1}{3})^T$ such that all corners are equidistant in the euclidean metric [91]. It is equivalent to *one-hot coding* but has lower dimensionality.

Classification: coding (element-wise) *Follow ups:* matrix proximity measure

Three Sequence Method (TSM)

The TSM returns a symbolic matrix $\boldsymbol{X}$ of size $3 \times n$. A nucleotide is coded according to its chemical properties: 1) purin (R) or pyrimidine (Y) group, 2) amino (M) or keto (K) group and 3) weak (W) or strong (S) hydrogen bond group [95]. The nucleotides s_i of the sequence are transformed according to

$$\begin{aligned} c_{RY}(s_i) &= \begin{cases} R & \text{if } s_i \in \{A, G\} \\ Y & \text{otherwise} \end{cases} \\ c_{MK}(s_i) &= \begin{cases} M & \text{if } s_i \in \{A, C\} \\ K & \text{otherwise} \end{cases} \\ c_{WS}(s_i) &= \begin{cases} W & \text{if } s_i \in \{A, T\} \\ S & \text{otherwise} \end{cases} \end{aligned} \tag{9}$$

for $i = 1, \ldots, n$, resulting in

$$\boldsymbol{X} = \begin{pmatrix} c_{RY}(s_1) \ldots c_{RY}(s_n) \\ c_{MK}(s_1) \ldots c_{MK}(s_n) \\ c_{WS}(s_1) \ldots c_{WS}(s_n) \end{pmatrix}. \tag{10}$$

For example, $A \mapsto (R, M, W)^T$, etc.

Classification: coding (element-wise) *Follow ups:* Natural vectors separately for each of the three sequences (feature generation)

Universal Sequence Maps (USMs)

Extension of the *Chaos Game Representation* for alphabets with cardinality n [86]. This is realized by assigning each symbol of the alphabet to a corner of a hypercube according to a *one-hot coding* scheme [139].

Classification: coding (recursive) *Follow ups:* matrix proximity measure

Wavelet Transform

The wavelet transform is a unique linear time-frequency transformation. It is similar to the *Fourier transform* but pays attention to the spatial resolution. As for the Fourier transform discrete variants exist. For further details, see [99].

Classification: coding (holistic) *Follow ups:* denoising/filtering, explicit feature selection, vector proximity measure

Zero Padding

Zero padding refers to increasing the length of vector $\boldsymbol{x} \in \mathbb{R}^n$ with $x_n \neq 0$ by adding zero elements. For matrices $\boldsymbol{X} \in \mathbb{R}^{m \times n}$, zero column vectors are added provided that the nth column vector of $\boldsymbol{X}$ is a non-zero vector.

Classification: coding (holistic) *Follow ups:* Fourier and Wavelet transform (coding), vector/matrix proximity

6.2 Proximity Measures and Related Concepts

Average Common Substring (ACS) Distance

The ACS distance is defined for two sequences S_1 and S_2 with length n and m respectively. Starting from a position $i \in \{1, \ldots, n\}$ in sequence S_1, the length l_i of the longest substring that can be found also in sequence S_2 is searched. The obtained lengths are averaged giving $L(S_1, S_2) = \frac{\sum_{i=1}^{n} l_i}{n}$. Information theoretic rescaling gives the non-symmetric quantity

$$d(S_1, S_2) = \frac{\log(m)}{L(S_1, S_2)} - \frac{2\log(n)}{n}, \tag{11}$$

symmetrization of which delivers

$$d_{ACS}(S_1, S_2) = \frac{d(S_1, S_2) + d(S_2, S_1)}{2}, \tag{12}$$

as ACS distance [125].

Classification: distance *Input:* non-numerical and numerical sequences

Correlation Measures

Correlation measures determine the statistical relations of two variables. There exist several correlation measurements like the *Pearson correlation* or the rank correlation coefficients such as those from Spearman or Kendall. Usually, the range is between -1 and 1, whereby 1 means highly correlated, 0 no correlation and -1 highly negative correlated. Sometimes, divergences are denoted as generalized correlation measures including higher moments.

Classification: no (primitive) similarity *Input:* vectors, matrices

Cosine Measure

Normalized euclidean inner product representing the cosine of the angle between $\boldsymbol{x}$ and $\boldsymbol{y}$

$$s_{cos}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\langle \boldsymbol{x}, \boldsymbol{y} \rangle_E}{\|\boldsymbol{x}\|_E \cdot \|\boldsymbol{y}\|_E}. \tag{13}$$

Classification: similarity *Input:* vectors

Eigen-McCaskill-Schuster Distance

Describes the number of transversions (purine $\leftrightarrow$ pyrimidine) between two nucleotide sequences S and S^* of length n [107]

$$d(S, S^*) = \sum_{i=1}^{n} \delta_i, \tag{14}$$

with

$$\delta_i = \begin{cases} 1 & \text{if } s_i \in \{A, G\} \wedge s_i^* \in \{T, C\} \\ 1 & \text{if } s_i \in \{T, C\} \wedge s_i^* \in \{A, G\} \\ 0 & \text{otherwise} \end{cases}. \tag{15}$$

Classification: distance *Input:* nucleotide sequences

euclidean Distance/euclidean Norm.

$$d_E(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_E, \tag{16}$$

with $\|\cdot\|_E$ being the euclidean norm $\|\boldsymbol{x}\|_E = \sqrt{\sum_{i=1}^{n} |x_i|^2}$.
Classification: distance/norm *Input:* vectors

euclidean Inner Product. Given vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^n$, the inner product is

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_E = \sum_{k=1}^{n} \bar{x}_k \cdot y_k, \tag{17}$$

where $\bar{x}_k$ is the conjugate complex of x_k.

Classification: no similarity *Input:* vectors

Frobenius Metric/Frobenius Norm.

$$d_{Frob}(\boldsymbol{X}_1, \boldsymbol{X}_2) = \|\boldsymbol{X}_1 - \boldsymbol{X}_2\|_{Fr}, \tag{18}$$

with

$$\|\boldsymbol{X}\|_{Fr} = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |x_{ij}|^2}, \tag{19}$$

being the Frobenius norm of the $m \times n$ matrix $\boldsymbol{X} = \big((x_{ij})\big)$ [107].

Classification: distance/norm *Input:* matrices

Hamming Metric

Counts the number of nucleotide disagreements between two sequences of length n [123]

$$d_{Ham}(S, S^*) = \sum_{i=1}^{n} \delta_i, \tag{20}$$

with

$$\delta_i = \begin{cases} 1 & \text{if } s_i \neq s_i^* \\ 0 & \text{otherwise} \end{cases}. \tag{21}$$

Classification: distance *Input:* non-numeric or numeric sequences

Jaccard Similarity Coefficient

Measures set similarity by considering their intersection and union size. It was independently introduced in [140] and [141] and is also known as Tanimoto coefficient

$$s_{Jac}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}. \tag{22}$$

If runtime is a crucial concern, MinHash (min-wise independent permutations locality sensitive hashing scheme), introduced by [142], should be considered as an approximation. Speed up is achieved by avoiding the explicit calculation of both set operations, but instead considering the intersection of the resulting MinHash signatures. *Classification:* similarity *Input:* sets

Jensen-Shannon Divergence

The Jensen-Shannon divergence [143] is based on the *Kullback-Leibler divergence*, though symmetric and delivers always a finite value

$$D_{JS}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2}\Big(D_{KL}(\boldsymbol{x}, z) + D_{KL}(\boldsymbol{y}, z)\Big), \tag{23}$$

with $z = \frac{\boldsymbol{x}+\boldsymbol{y}}{2}$. The square root of this divergence is a metric, i.e., the Jensen-Shannon distance [144].

Classification: dissimilarity, distance (square root version) *Input:* probabilistic/density vectors $\boldsymbol{x}$, i.e., $x_i \in [0, 1]$ and $\sum x_i = 1$

Kullback-Leibler Divergence.

$$d_{KL}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{j=1}^{n} x_j \cdot \log \frac{x_j}{y_j}, \tag{24}$$

[106] where the $\log$ is to an arbitrary basis.

TABLE 1
Categorization of Surveyed Data Transformation Methods

Category	Method	Reference
Coding	Chaos game representation	[87]
	DNA-walk (2D)	[90]
	Fourier transform	
	Integer representation	[91]
	k-mer coding	
	Natural vectors	[96]
	One-hot coding	
	Real number representation	[92]
	Tetrahedron representation	[94]
	Three sequence method	[95]
	Universal sequence maps	[86]
	Wavelet transform	[99]
	Zero padding	
Feature generation	Bag of words	[85], [138]
	DNA-walk (1D)	[103]
	Lempel-Ziv complexity	[117]
	Natural vectors	[96]
	Normalization	
	Power spectrum	

Classification: dissimilarity *Input:* probabilistic/density vectors $\boldsymbol{x}$, i.e., $x_i \in [0, 1]$ and $\sum x_i = 1$

L_p-Distance/Norm.

$\|\cdot\|_p$ is defined as

$$\|\boldsymbol{x}\|_p = \sqrt[p]{\sum_{i=1}^{n} |x_i|^p}, \tag{25}$$

for $p \geq 1$ and the distance is given as

$$d_p(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_p. \tag{26}$$

Classification: distance/norm *Input:* vectors

Normalized Compression Distance

Uses the sizes $C(S)$ of compressed sequences as input, derived from an appropriate compression algorithm following Kolmogorov's minimal description length principle, see also *Lempel-Ziv complexity*

$$d_{NCD}(S_1, S_2) = \frac{C(S_{12}) - \min\big\{C(S_1), C(S_2)\big\}}{\max\big\{C(S_1), C(S_2)\big\}}, \tag{27}$$

where $S_{12} = S_1 S_2$ is the concatenation of S_1 and S_2 [118].

Classification: distance or dissimilarity in dependence upon the compressor *Input:* complexity values (scalar) for both sequences and their concatenation

Pearson Correlation Coefficient

For vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ with mean values $\bar{x} = 0$ and $\bar{y} = 0$, respectively, the correlation is

$$r(\boldsymbol{x}, \boldsymbol{y}) = \langle \boldsymbol{x}, \boldsymbol{y} \rangle_E. \tag{28}$$

See also *correlation measures*.

Classification: no similarity *Input:* vectors

TABLE 2
Categorization of Surveyed Proximity Measures and Related Concepts

Category	Method	Reference
Distance	Average common substring distance	[125]
	Eigen-McCaskill-Schuster distance	[107]
	euclidean distance	
	Frobenius metric	
	Hamming metric	[123]
	Jensen-Shannon distance	[144]
	L_p-distance	
	Normalized compression distance	[118]
Dissimilarity	Jensen-Shannon divergence	[143]
	Kullback-Leibler divergence	[106]
Similarity	Cosine measure	
	Jaccard similarity coefficient	[140], [141]
	Radial basis function kernel	
Neither	euclidean inner product	
	Pearson correlation	

Radial Basis Function (RBF) Kernel

This kernel is defined as

$$s_{RBF}(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{y}\|_E^2}{2\sigma^2}\right). \tag{29}$$

Classification: similarity *Input:* vectors

6.3 Tabular Overview of the Surveyed Methods

Tables 1 and 2 summarize the surveyed methods of the systematization categories coding, feature generation and proximity measures, respectively.

ACKNOWLEDGMENTS

The authors would like to thank the members of the Computational Intelligence research group at the University of Applied Sciences Mittweida for useful discussions and impulses.

REFERENCES

[1] Z. D. Stephens *et al.*, "Big data: astronomical or genomical?" *PLOS Biol.*, vol. 13, no. 7, 2015, Art. no. e1002195.
[2] J. M. Heather and B. Chain, "The sequence of sequencers: The history of sequencing DNA," *Genomics*, vol. 107, no. 1, pp. 1–8, Jan. 2016.
[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[4] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, 2015.
[5] I. Goodfellow *et al.*, "Generative adversarial networks," in *Proc. 27th Int. Conf. Advances Neural Inf. Process. Syst*, 2014, pp. 2672–2680.
[6] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT-Press, 2018.
[7] P. Baldi and S. Brunak, *Bioinformatics: The Machine Learning Approach*. Cambridge, MA, USA: MIT Press, 1998.
[8] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "A survey of machine learning for big data processing," *EURASIP J. Advances Signal Process.*, vol. 2016, no. 1, Dec. 2016, Art. no. 67.
[9] A. Zielezinski, S. Vinga, J. Almeida, and W. M. Karlowski, "Alignment-free sequence comparison: Benefits, applications, and tools," *Genome Biol.*, vol. 18, no. 1, Dec. 2017, Art. no. 186.
[10] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *J. Mol. Biol.*, vol. 48, no. 3, pp. 443–453, Mar. 1970.
[11] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *J. Mol. Biol.*, vol. 147, no. 1, pp. 195–197, Mar. 1981.
[12] C. Kemena and C. Notredame, "Upcoming challenges for multiple sequence alignment methods in the high-throughput era," *Bioinformatics*, vol. 25, no. 19, pp. 2455–2465, Oct. 2009.
[13] W. Just, "Computational complexity of multiple sequence alignment with SP-score," *J. Comput. Biol.*, vol. 8, no. 6, pp. 615–623, Nov. 2001.
[14] D.-F. Feng and R. F. Doolittle, "Progressive sequence alignment as a prerequisitetto correct phylogenetic trees," *J. Mol. Evol.*, vol. 25, no. 4, pp. 351–360, Aug. 1987.
[15] Y. Wang, H. Wu, and Y. Cai, "A benchmark study of sequence alignment methods for protein clustering," *BMC Bioinf.*, vol. 19, no. S19, Dec. 2018, Art. no. 529.
[16] G. Kucherov, "Evolution of biosequence search algorithms: A brief survey," *Bioinformatics*, vol. 35, no. 19, pp. 3547–3552, Oct. 2019.
[17] B. Haubold, "Alignment-free phylogenetics and population genetics," *Briefings Bioinf.*, vol. 15, no. 3, pp. 407–418, May 2014.
[18] C. X. Chan, G. Bernard, O. Poirion, J. M. Hogan, and M. A. Ragan, "Inferring phylogenies of evolving sequences without multiple sequence alignment," *Sci. Rep.*, vol. 4, no. 1, May 2015, Art. no. 6504.
[19] K. Hatje and M. Kollmar, "A phylogenetic analysis of the brassicales clade based on an alignment-free sequence comparison method," *Front. Plant Sci.*, vol. 3, 2012, Art. no. 192.
[20] Y.-W. Wu and Y. Ye, "A novel abundance-based algorithm for binning metagenomic sequences using l-tuples," *J. Comput. Biol.: A J. Comput. Mol. Cell Biol.*, vol. 18, no. 3, pp. 523–534, Mar. 2011.
[21] G. Leung and M. B. Eisen, "Identifying Cis-regulatory sequences by word profile similarity," *PLoS One*, vol. 4, no. 9, Sep. 2009, Art. no. e6901.
[22] B. T. *et al.*, "RAFTS3G: An efficient and versatile clustering software to analyses in large protein datasets," *BMC Bioinf.*, vol. 20, no. 1, Jul. 2019, Art. no. 392.
[23] N. L. Bray, H. Pimentel, P. Melsted, and L. Pachter, "Near-optimal probabilistic RNA-seq quantification," *Nature Biotechnol.*, vol. 34, no. 5, pp. 525–527, May 2016.
[24] D. R. Zerbino and E. Birney, "Velvet: Algorithms for de novo short read assembly using de Bruijn graphs," *Genome Res.*, vol. 18, no. 5, pp. 821–829, May 2008.
[25] F.-D. Pajuste, L. Kaplinski, M. Möls, T. Puurand, M. Lepamets, and M. Remm, "FastGT: An alignment-free method for calling common SNVs directly from raw sequencing reads," *Sci. Rep.*, vol. 7, no. 1, May 2017, Art. no. 2537.
[26] K. Song, J. Ren, G. Reinert, M. Deng, M. S. Waterman, and F. Sun, "New developments of alignment-free sequence comparison: Measures, statistics and next-generation sequencing," *Briefings Bioinf.*, vol. 15, no. 3, pp. 343–353, May 2014.
[27] Y. Li, K. Tian, C. Yin, R. L. He, and S. S.-T. Yau, "Virus classification in 60-dimensional protein space," *Mol. Phylogenetics Evol.*, vol. 99, pp. 53–62, Jun. 2016.
[28] M. Kaden *et al.*, "Learning vector quantization as an interpretable classifier for the detection of SARS-CoV-2 types based on their RNA sequences," *Neural Comput. Appl.*, pp. 1–12, Apr. 2021.
[29] T. Villmann *et al.*, "Searching for the origins of life – Detecting RNA life signatures using learning vector quantization," in *Proc. Advances Self-Organizing Maps, Learn. Vector Quantization, Clustering Data Visualization – Proc. 13th Int. Workshop Self-Organizing Maps and Learning Vector Quantization, Clustering and Data Visualization*, 2019, pp. 324–333.
[30] S. Spänig and D. Heider, "Encodings and models for antimicrobial peptide classification for multi-resistant pathogens," *BioData Mining*, vol. 12, no. 1, Dec. 2019, Art. no. 7.
[31] J. Lin, J. Wei, D. Adjeroh, B.-H. Jiang, and Y. Jiang, "SSAW: A new sequence similarity analysis method based on the stationary discrete wavelet transform," *BMC Bioinf.*, vol. 19, no. 1, Dec. 2018, Art. no. 165.
[32] C. Yu *et al.*, "Real time classification of viruses in 12 dimensions," *PLoS One*, vol. 8, no. 5, May 2013, Art. no. e64328.

[33] N. G. Nguyen *et al.*, "DNA sequence classification by convolutional neural network," *J. Biomed. Sci. Eng.*, vol. 09, no. 05, pp. 280–286, 2016.

[34] J.-M. Garant, J.-P. Perreault, and M. S. Scott, "Motif independent identification of potential RNA G-quadruplexes by G4RNA screener," *Bioinformatics*, vol. 33, no. 22, pp. 3532–3537, Nov. 2017.

[35] L. Pinello, G. Lo Bosco , and G.-C. Yuan, "Applications of alignment-free methods in epigenomics," *Briefings Bioinf.*, vol. 15, no. 3, pp. 419–430, May 2014.

[36] J. Bao and R. Yuan, "A wavelet-based feature vector model for DNA clustering," *Genetics Mol. Res.*, vol. 14, no. 4, pp. 19 163–19 172, 2015.

[37] J. Bao, R. Yuan, and Z. Bao, "An improved alignment-free model for dna sequence similarity metric," *BMC Bioinf.*, vol. 15, no. 1, Dec. 2014, Art. no. 321.

[38] K. D. Murray, C. Webers, C. S. Ong, J. Borevitz, and N. Warthmann, "kWIP: The k-mer weighted inner product, a de novo estimator of genetic similarity," *PLoS One Comput. Biol.*, vol. 13, no. 9, Sep. 2017, Art. no. e1005727.

[39] A. Yang, W. Zhang, J. Wang, K. Yang, Y. Han, and L. Zhang, "Review on the application of machine learning algorithms in the sequence data mining of DNA," *Front. Bioeng. Biotechnol.*, vol. 8, Sep. 2020, Art. no. 1032.

[40] S. Vinga and J. Almeida, "Alignment-free sequence comparison–a review," *Bioinformatics*, vol. 19, no. 4, pp. 513–523, Mar. 2003.

[41] B. B. Luczak, B. T. James, and H. Z. Girgis, "A survey and evaluations of histogram-based statistics in alignment-free sequence comparison," *Briefings Bioinf.*, vol. 20, no. 4, pp. 1222–1237, Jul. 2019.

[42] O. Bonham-Carter , J. Steele, and D. Bastola, "Alignment-free genetic sequence comparisons: A review of recent approaches by word analysis," *Briefings Bioinf.*, vol. 15, no. 6, pp. 890–905, Nov. 2014.

[43] J. Ren *et al.*, "Alignment-free sequence analysis and applications," *Annu. Rev. Biomed. Data Sci.*, vol. 1, pp. 93–114, 2018, Art. no. 22.

[44] A. Zielezinski *et al.*, "Benchmarking of alignment-free sequence comparison methods," *Genome Biol.*, vol. 20, no. 1, Dec. 2019, Art. no. 144.

[45] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[46] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Mach. Intell.*, vol. 1, no. 5, pp. 206–215, 2019.

[47] P. Lemey, M. Salemi, and A.-M. Vandamme, Eds., *The Phylogenetic Handbook: A Practical Approach to Phylogenetic Analysis and Hypothesis Testing*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[48] R. R. Sokal and C. D. Michener, "A statistical method for evaluating systematic relationships," *Univ. Kansas Sci. Bull.*, vol. 38, no. 22, pp. 1409–1438, 1958.

[49] N. Saitou and M. Nei, "The neighbor-joining method: A new method for reconstructing phylogenetic trees," *Mol. Biol. Evol.*, vol. 4, no. 4, pp. 406–425, Jul. 1987.

[50] B. D. Ondov *et al.*, "Mash: Fast genome and metagenome distance estimation using MinHash," *Genome Biol.*, vol. 17, no. 1, Dec. 2016, Art. no. 132.

[51] G. Bernard *et al.*, "Alignment-free inference of hierarchical and reticulate phylogenomic relationships," *Briefings Bioinf.*, vol. 20, no. 2, pp. 426–435, Mar. 2019.

[52] T. Dencker, C.-A. Leimeister, M. Gerth, C. Bleidorn, S. Snir, and B. Morgenstern, "Multi-SpaM': A maximum-likelihood approach to phylogeny reconstruction using multiple spaced-word matches and quartet trees," *NAR Genomics Bioinf.*, vol. 2, no. 1, Mar. 2020, Art. no. lqz013.

[53] B. Haubold, F. Klötzl, and P. Pfaffelhuber, "Andi: Fast and accurate estimation of evolutionary distances between closely related genomes," *Bioinformatics*, vol. 31, no. 8, pp. 1169–1175, Apr. 2015.

[54] F. Klötzl and B. Haubold, "Phylonium: Fast estimation of evolutionary distances from large samples of similar genomes," *Bioinformatics*, vol. 36, no. 7, pp. 2040–2046, Apr. 2020.

[55] C.-A. Leimeister, M. Boden, S. Horwege, S. Lindner, and B. Morgenstern, "Fast alignment-free sequence comparison using spaced-word frequencies," *Bioinformatics*, vol. 30, no. 14, pp. 1991–1999, Jul. 2014.

[56] H. Z. Girgis, B. T. James, and B. B. Luczak, "*Identity* : Rapid alignment-free prediction of sequence alignment identity scores using self-supervised general linear models," *NAR Genomics Bioinf.*, vol. 3, no. 1, Feb. 2021, Art. no. lqab001.

[57] T. Villmann *et al.*, "Searching for the origins of life – detecting RNA life signatures using learning vector quantization," in *Proc. 13th Int. Workshop*, 2019, pp. 324–333.

[58] S. Haykin, *Neural Networks - A Comprehensive Foundation*. Piscataway, NJ, USA: IEEE Press, 1994.

[59] C. Bishop, *Pattern Recognition and Machine Learning*. Berlin, Germany: Springer, 2006.

[60] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. 25*, 2012, pp. 1097–1105.

[61] T. Bäck, D. B. Fogel, and Z. Michalewicz, Eds., *Handbook of Evolutionary Computation*. London, U.K.: Institute of Physics Publishing and Oxford Univ. Press, 1997.

[62] B. Schölkopf and A. Smola, *Learning with Kernels*. Cambridge, MA, USA: MIT Press, 2002.

[63] M. Biehl, B. Hammer, and T. Villmann, "Prototype-based models in machine learning," *Wiley Interdisciplinary Rev.: Cogn. Sci.*, no. 2, pp. 92–111, 2016.

[64] Y. LeCun and Y. Bengio, *The handbook of brain theory and neural networks*. Cambridge, MA, USA: MIT Press, 1995, pp. 255–258.

[65] E. Pekalska and R. Duin, *The Dissimilarity Representation for Pattern Recognition: Foundations and Applications*. Singapore: World Scientific, 2006.

[66] D. Nebel, M. Kaden, A. Villmann, and T. Villmann, "Types of (dis-)similarities and adaptive mixtures thereof for improved classification learning," *Neurocomputing*, vol. 268, pp. 42–54, Dec. 2017.

[67] S. Bittrich, M. Kaden, C. Leberecht, F. Kaiser, T. Villmann, and D. Labudde, "Application of an interpretable classification model on early folding residues during protein folding," *BioData Mining*, vol. 12, no. 1, pp. 1:1–1:16, 2019.

[68] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007.

[69] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters," *J. Cybern.*, vol. 3, no. 3, pp. 32–57, Jan. 1973.

[70] T. M. Martinetz, S. G. Berkovich, and K. J. Schulten, "'Neural-gas' network for vector quantization and its application to time-series prediction," *IEEE Trans. Neural Netw. Learn. Syst*, vol. 4, no. 4, pp. 558–569, Jul. 1993.

[71] I. Steinwart and A. Christmann, *Support Vector Machines*. Berlin, Germany: Springer Verlag, 2008.

[72] A. Zharkikh, "Estimation of evolutionary distances between nucleotide sequences," *J. Mol. Evol.*, vol. 39, no. 3, pp. 315–329, Sep. 1994.

[73] L. Sachs, *Angewandte Statistik*, 7th ed. Berlin, Germany: Springer Verlag, 1992.

[74] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[75] B. Schölkopf, K. Tsuda, and J.-P. Vert, *Kernel Methods in Computational Biology*. Cambridge, MA, USA: MIT Press, 2004.

[76] B. Rumelhart, G. Hinton, and R. Williams, "Learning internal representations by backpropagating errors," *Nature*, vol. 323, no. 99, pp. 533–536, 1986.

[77] B. Mokbel, B. Paassen, F.-M. Schleif, and B. Hammer, "Metric learning for sequences in relational LVQ," *Neurocomputing*, vol. 169, pp. 306–322, Dec. 2015.

[78] B. Hammer, D. Hofmann, F.-M. Schleif, and X. Zhu, "Learning vector quantization for (dis-)similarities," *Neurocomputing*, vol. 131, pp. 43–51, 2014.

[79] D. Nebel, B. Hammer, K. Frohberg, and T. Villmann, "Median variants of learning vector quantization for learning of dissimilarity data," *Neurocomputing*, vol. 169, pp. 295–305, Dec. 2015.

[80] B. Hammer and A. Hasenfuss, "Relational neural gas," in *Proc. 30th Annu. German Conf. Adv. Artif. Intell.*, 2007, pp. 190–204.

[81] R. Hathaway and J. Bezdek, "NERF c-means: Non-euclidean relational fuzzy clustering," *Pattern Recognit.*, vol. 27, no. 3, pp. 429–437, 1994.

[82] F.-M. Schleif and P. Tiňo, "Indefinite proximity learning: A review," *Neural Comput.*, vol. 27, no. 10, 2015, Art. no. 2.

[83] A. Lempel, "On multiset decipherable codes (Corresp.)," *IEEE Trans. Inf. Theory*, vol. 32, no. 5, pp. 714–716, Sep. 1986.
[84] A. Tversky and I. Gati, "Studies of similarity," in *Cognition and Categorization*, E. Rosch and B. Lloyd, Eds. Hillsdale, NJ, USA: Erlbaum, 1978, pp. 79–98.
[85] B. E. Blaisdell, "Average values of a dissimilarity measure not requiring sequence alignment are twice the averages of conventional mismatch counts requiring sequence alignment for a variety of computer-generated model systems," *J. Mol. Evol.*, vol. 29, pp. 538–547, Jun. 1989.
[86] J. S. Almeida and S. Vinga, "Universal sequence map (USM) of arbitrary discrete sequences," *BMC Bioinf.*, vol. 3, no. 6, 2002, Art. no. 11.
[87] H. Jeffrey, "Chaos game representation of gene structure," *Nucleic Acids Res.*, vol. 18, no. 8, pp. 2163–2170, 1990.
[88] J. S. Almeida and S. Vinga, "Computing distribution of scale independent motifs in biological sequences," *Algorithms Mol. Biol.*, vol. 1, no. 1, Dec. 2006, Art. no. 18.
[89] J. S. Almeida, J. A. Carrico, A. Maretzek, P. A. Noble, and M. Fletcher, "Analysis of genomic sequences by Chaos Game Representation," *Bioinformatics*, vol. 17, no. 5, pp. 429–437, May 2001.
[90] J. A. Berger, S. K. Mitra, M. Carli, and A. Neri, "New approaches to genome sequence analysis base don digital signal processing," in *Proc. IEEE Workshop Genomic Signal Process. Statist.*, 2002, p. 4.
[91] G. Mendizabal-Ruiz , I. Román-Godínez, S. Torres-Ramos , R. A. Salido-Ruiz , and J. A. Morales, "On DNA numerical representations for genomic similarity computation," *PLoS One*, vol. 12, no. 3, Mar. 2017, Art. no. e0173288.
[92] N. Chakravarthy, A. Spanias, L. D. Iasemidis, and K. Tsakalis, "Autoregressive modeling and feature analysis of DNA sequences," *EURASIP J. Adv. Signal Process.*, vol. 2004, no. 1, Dec. 2004, Art. no. 952689.
[93] N. Voss, "Evolution of long-range fractal correlations and 1/f noise in DNA base sequences," *Physical Rev. Lett.*, vol. 68, no. 25, pp. 3805–3808, Jun. 1992.
[94] B. D. Silverman and R. Linsker, "A measure of DNA periodicity," *J. Theoretical Biol.*, vol. 118, no. 3, pp. 295–300, Feb. 1986.
[95] L. Shi and H. Huang, "DNA sequences analysis based on classifications of nucleotide bases," *Adv. Intell. Soft Comput.*, vol. 137, pp. 379–384, 2012.
[96] M. Deng, C. Yu, Q. Liang, R. L. He, and S. S.-T. Yau, "A novel method of characterizing genetic sequences: Genome space with biological distance and applications," *PLoS One*, vol. 6, no. 3, Mar. 2011, Art. no. e17293.
[97] L. Liu, Y.-K. Ho, and S. Yau, "Clustering DNA sequences by feature vectors," *Mol. Phylogenetics Evol.*, vol. 41, no. 1, pp. 64–69, Oct. 2006.
[98] R. J. Marks, *Handbook of Fourier Analysis & Its Applications*. New York, NY, USA: Oxford Univ. Press, 2009.
[99] D. F. Walnut, *An Introduction to Wavelet Analysis*, J. J. Benedetto, Ed., Boston, MA, USA: Birkhäuser, 2004.
[100] H. Liang, X. Sun, Y. Sun, and Y. Gao, "Text feature extraction based on deep learning: A review," *EURASIP J. Wireless Commun. Netw.*, vol. 2017, no. 1, Dec. 2017, Art. no. 211.
[101] J. Schmidt, M. R. G. Marques, S. Botti, and M. A. L. Marques, "Recent advances and applications of machine learning in solid-state materials science," *npj Comput. Mater.*, vol. 5, no. 1, pp. 1–36, Aug. 2019.
[102] Y. Kobori and S. Mizuta, "Similarity estimation between DNA sequences based on local pattern histograms of binary images," *Genomics, Proteomics Bioinf.*, vol. 14, no. 2, pp. 103–112, Apr. 2016.
[103] C.-K. Peng, *et al.*, "Long-range correlations in nucleotide sequences," *Nature*, vol. 356, no. 6365, pp. 168–170, Mar. 1992.
[104] M. Akhtar, J. Epps, and E. Ambikairajah, "On DNA numerical representations for period-3 based exon prediction," in *Proc. IEEE Int. Workshop Genomic Signal Process. Statist.*, 2007, pp. 1–4.
[105] A. Cornish-Bowden , "Nomenclature for incompletely specified bases in nucleic acid sequences: Recommendations 1984," *Nucleic Acids Res.*, vol. 13, no. 9, pp. 3021–3030, May 1985.
[106] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, Mar. 1951.
[107] M. M. Deza and E. Deza, *Encyclopedia of Distances*, 4th ed. Berlin, Germany: Springer-Verlag, 2016.
[108] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Sov. Phys.-Doklady*, vol. 10, no. 8, pp. 707–710, 1966.
[109] A. Rényi, "On measures of entropy and information," in *Proc. 4th Berkeley Symp. Math. Statist. Probability*, 1961, pp. 547–561.
[110] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *Int. J. Math. Models Methods Appl. Sci.*, vol. 1, no. 4, pp. 300–307, 2007.
[111] G. Reinert, D. Chew, F. Sun, and M. S. Waterman, "Alignment-free sequence comparison (I): Statistics and power," *J. Comput. Biol.*, vol. 16, no. 12, pp. 1615–1634, Dec. 2009.
[112] J. Wen, R. H. Chan, S.-C. Yau, R. L. He, and S. S. T. Yau, "K-mer natural vector and its application to the phylogenetic analysis of genetic sequences," *Gene*, vol. 546, no. 1, pp. 25–34, Aug. 2014.
[113] Y. Li, L. He, R. Lucy He , and S. S.-T. Yau, "A novel fast vector method for genetic sequence comparison," *Sci. Rep.*, vol. 7, no. 1, Dec. 2017, Art. no. 12226.
[114] S. S. T. Yau, "DNA sequence representation without degeneracy," *Nucleic Acids Res.*, vol. 31, no. 12, pp. 3078–3080, Jun. 2003.
[115] J. A. Lee and M. Verleysen, "Generalization of the Lp norm for time series and its application to Self-Organizing Maps," in *Proc. Self-Organizing Maps*, 2005, pp. 733–740.
[116] M. Lange, D. Zühlke, O. Holz, and T. Villmann, "Applications of -norms and their smooth approximations for gradient based learning vector quantization," in *Proc. Eur. Symp. Artif. Neural Netw. Comput. Intell. Mach. Learn.*, 2014, pp. 271–276.
[117] A. Lempel and J. Ziv, "On the complexity of finite sequences," *IEEE Trans. Inf. Theory*, vol. 22, no. 1, pp. 75–81, Jan. 1976.
[118] M. Li, X. Chen, X. Li, B. Ma, and P. Vitanyi, "The similarity metric," *IEEE Trans. Inf. Theory*, vol. 50, no. 12, pp. 3250–3264, Dec. 2004.
[119] D. Sculley and C. Brodley, "Compression and machine learning: A new perspective on feature space vectors," in *Proc. Data Compression Conf.*, 2006, pp. 332–332.
[120] V. N. Babenko, P. S. Kosarev, O. V. Vishnevsky, V. G. Levitsky, V. V. Basin, and A. S. Frolov, "Investigating extended regulatory regions of genomic DNA sequences," *Bioinformatics*, vol. 15, no. 7, pp. 644–653, Jul. 1999.
[121] M. Bauer, S. M. Schuster, and K. Sayood, "The average mutual information profile as a genomic signature," *BMC Bioinf.*, vol. 9, no. 1, 2008, Art. no. 48.
[122] K. S. Bohnsack, M. Kaden, J. Abel, S. Saralajew, and T. Villmann, "The resolved mutual information function as a structural fingerprint of biomolecular sequences for interpretable machine learning classifiers," *Entropy*, vol. 23, no. 10, Oct. 2021, Art. no. 1357.
[123] R. W. Hamming, "Error detecting and error correcting codes," *Bell Syst. Tech. J.*, vol. 29, no. 2, pp. 147–160, Apr. 1950.
[124] Y. Rubner, C. Tomasi, and L. J. Guibas, "A metric for distributions with applications to image databases," in *Proc. 6th Int. Conf. Comput. Vis.*, 1998, pp. 59–66.
[125] I. Ulitsky, D. Burstein, T. Tuller, and B. Chor, "The average common substring approach to phylogenomic reconstruction," *J. Comput. Biol.*, vol. 13, no. 2, pp. 336–350, Mar. 2006.
[126] C.-A. Leimeister and B. Morgenstern, "Kmacs: The k -mismatch average common substring approach to alignment-free sequence comparison," *Bioinformatics*, vol. 30, no. 14, pp. 2000–2008, Jul. 2014.
[127] B. Haubold, N. Pierstorff, F. Möller, and T. Wiehe, "Genome comparison without alignment using shortest unique substrings," *BMC Bioinf.*, vol. 6, no. 1, 2005, Art. no. 123.
[128] C. Yin, Y. Chen, and S. S.-T. Yau, "A measure of DNA sequence similarity by Fourier Transform with applications on hierarchical clustering," *J. Theor. Biol.*, vol. 359, pp. 18–28, Oct. 2014.
[129] Z.-H. Qi and T.-R. Fan, "PN-curve: A 3D graphical representation of DNA sequences and their numerical characterization," *Chem. Phys. Lett.*, vol. 442, no. 4–6, pp. 434–440, Jul. 2007.
[130] J. Joseph and R. Sasikumar, "Chaos game representation for comparison of whole genomes," *BMC Bioinf.*, vol. 7, no. 1, Dec. 2006, Art. no. 243.
[131] M. A. Gates, "Simpler DNA sequence representations," *Nature*, vol. 316, no. 6025, pp. 219–219, Jul. 1985.
[132] K. Domaschke, M. Kaden, M. Lange, and T. Villmann, "Learning matrix quantization and variants of relevance learning," in *Proc. Eur. Symp. Artif. Neural Netw. Comput. Intell. Mach. Learn.*, 2015, pp. 13–18.
[133] G. E. Sims, S.-R. Jun, G. A. Wu, and S.-H. Kim, "Alignment-free genome comparison with feature frequency profiles (FFP) and optimal resolutions," in *Proc. Nat. Acad. Sci. USA*, vol. 106, no. 8, pp. 2677–2682, Feb. 2009.
[134] J. Qi, B. Wang, and B.-I. Hao, "Whole proteome prokaryote phylogeny without sequence alignment: A K -String Composition Approach," *J. Mol. Evol.*, vol. 58, no. 1, pp. 1–11, Jan. 2004.

[135] J. Lin, D. A. Adjeroh, B.-H. Jiang, and Y. Jiang, "K 2 and K2*: Efficient alignment-free sequence similarity measurement based on Kendall statistics," *Bioinformatics*, vol. 34, no. 10, pp. 1682–1689, May 2018.
[136] D. Wei, Q. Jiang, Y. Wei, and S. Wang, "A novel hierarchical clustering algorithm for gene sequences," *BMC Bioinf.*, vol. 13, no. 1, Dec. 2012, Art. no. 174.
[137] P. Kolekar, M. Kale, and U. Kulkarni-Kale , "Alignment-free distance measure based on return time distribution for sequence analysis: Applications to clustering, molecular phylogeny and subtyping," *Mol. Phylogenetics Evol.*, vol. 65, no. 2, pp. 510–522, Nov. 2012.
[138] Y. Goldberg, "Neural network methods for natural language processing," *Synthesis Lectures Human Lang. Technol.*, vol. 10, no. 1, pp. 1–309, Apr. 2017.
[139] S. Vinga and J. S. Almeida, "Rényi continuous entropy of DNA sequences," *J. Theor. Biol.*, vol. 231, no. 3, pp. 377–388, Dec. 2004.
[140] P. Jaccard, "The distribution of the flora in the alpine zone.1," *New Phytologist*, vol. 11, no. 2, pp. 37–50, 1912.
[141] T. T. Tanimoto, "An elementary mathematical theory of classification and prediction," *Internal IBM Tech. Report*, Armonk, NY, USA: Int. Bus. Mach. Corporation, 1957.
[142] A. Broder, "On the resemblance and containment of documents," in *Proc. Compression Complexity SEQUENCES*, 1998, pp. 21–29.
[143] J. Lin, "Divergence measures based on the shannon entropy," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 145–151, Jan. 1991.
[144] T. Yamano, "Some bounds for skewed -jensen-shannon divergence," *Results Appl. Math.*, vol. 3, Oct. 2019, Art. no. 100064.

Katrin Sophie Bohnsack received the MSc degree in molecular biology/bioinformatics and is currently working toward the PhD degree at the Computational Intelligence Research Group, University of Applied Sciences Mittweida, Germany. Her research is focused on developing machine learning tools in the context of biological sequence data analysis.

Marika Kaden (formerly Kästner) received the PhD degree in computer science from University Leipzig, Germany, in 2016, and since 2010 she is with the Computational Intelligence Group at the University of Applied Sciences Mittweida, Germany, currently as a postdoctoral researcher. Her research interests include interpretable machine learning models, optimized prototype-based classification learning, integration of expert knowledge, and properties of dissimilarity structures in context of machine learning.

Julia Abel received the MSc degree in molecular biology/bioinformatics and currently is working toward the PhD degree at the Computational Intelligence Research Group, University of Applied Sciences Mittweida, Germany. The focus of her research is on the application of machine learning tools and their interpretation in biological domains.

Thomas Villmann is currently full professor for mathematics and computational intelligence at the University of Applied Sciences Mittweida, Germany. He is head of the Saxon Institute for Computational Intelligence and Machine Learning. His research focus is the theory of machine learning, interpretable models and information theoretic methods as well as applications in bioinformatics, medicine, remote sensing and autonomous driving.