Digital Discovery

ROYAL SOCIETY OF CHEMISTRY

PAPER

View Article Online
View Journal | View Issue

Cite this: *Digital Discovery*, 2023, **2**, 502

Krein support vector machine classification of antimicrobial peptides

Joseph Redshaw,[a] Darren S. J. Ting,[bef] Alex Brown,[c] Jonathan D. Hirst *[a] and Thomas Gärtner[d]

Antimicrobial peptides (AMPs) represent a potential solution to the growing problem of antimicrobial resistance, yet their identification through wet-lab experiments is a costly and time-consuming process. Accurate computational predictions would allow rapid *in silico* screening of candidate AMPs, thereby accelerating the discovery process. Kernel methods are a class of machine learning algorithms that utilise a kernel function to transform input data into a new representation. When appropriately normalised, the kernel function can be regarded as a notion of similarity between instances. However, many expressive notions of similarity are not valid kernel functions, meaning they cannot be used with standard kernel methods such as the support-vector machine (SVM). The Kreĭn-SVM represents generalisation of the standard SVM that admits a much larger class of similarity functions. In this study, we propose and develop Kreĭn-SVM models for AMP classification and prediction by employing the Levenshtein distance and local alignment score as sequence similarity functions. Utilising two datasets from the literature, each containing more than 3000 peptides, we train models to predict general antimicrobial activity. Our best models achieve an AUC of 0.967 and 0.863 on the test sets of each respective dataset, outperforming the in-house and literature baselines in both cases. We also curate a dataset of experimentally validated peptides, measured against *Staphylococcus aureus* and *Pseudomonas aeruginosa*, in order to evaluate the applicability of our methodology in predicting microbe-specific activity. In this case, our best models achieve an AUC of 0.982 and 0.891, respectively. Models to predict both general and microbe-specific activities are made available as web applications.

Received 19th January 2023
Accepted 22nd February 2023

DOI: 10.1039/d3dd00004d

rsc.li/digitaldiscovery

1 Introduction

Kernel methods are a class of machine learning algorithms that incorporate a kernel function in order to model non-linear relationships. Standard kernel methods assume that a given kernel function is positive-definite. Those kernel functions which do not satisfy this assumption are known as indefinite kernels. The assumption of positive-definiteness is restrictive, as it limits the number of applicable functions. Recent developments in the theory of learning with indefinite kernels have now removed this requirement, allowing a much broader class of functions to be incorporated into kernel methods.[1–3] Leveraging these developments, we study the effectiveness of learning with established sequence-similarity functions for the classification of antimicrobial peptides (AMPs) based on their amino acid sequences. We evaluate the ability of the proposed methodology to predict general antimicrobial activity, as well as antimicrobial activity against specific species.

AMPs, also known as host defense peptides, are a class of evolutionary conserved molecules that form an important component in the innate immune system.[4–6] These molecules are usually made of 12 to 50 amino acid residues, and they typically possess certain properties, including cationicity, 30–50% hydrophobicity, and amphiphilicity. They exhibit good antimicrobial activity against a broad range of bacteria, viruses, fungi, and parasites. In addition, they have an inherent low risk of developing antimicrobial resistance (AMR), largely attributed to their underlying rapid membrane permeabilising activity.[4,7,8] Such broad-spectrum and rapid antimicrobial activity has prompted researchers to consider AMPs as a potential remedy to the growing problem of AMR, which is a major global health threat.[9,10] Nonetheless, there has so far been a lack of success in translating AMP-based therapy to clinical use, due to challenges such as complex structure–activity relationship (SAR), drug toxicity, instability in host and infective environment, and low financial incentives.[11,12] Owing to the complex SAR and the

[a]*School of Chemistry, University of Nottingham, University Park, Nottingham, NG7 2RD, UK. E-mail: jonathan.hirst@nottingham.ac.uk*

[b]*Academic Ophthalmology, School of Medicine, University of Nottingham, Nottingham, NG7 2UH, UK*

[c]*Artificial Intelligence and Machine Learning, GSK Medicines Research Centre, Gunnels Wood Road, Stevenage, SG1 2NY, UK*

[d]*Machine Learning Group, TU Wien Informatics, Vienna, Austria*

[e]*Academic Unit of Ophthalmology, Institute of Inflammation and Ageing, University of Birmingham, Birmingham, UK*

[f]*Birmingham and Midland Eye Centre, Birmingham, UK*

costly and time-consuming process of wet-lab experiments associated with AMP investigations, many researchers have proposed computational approaches, including molecular dynamics (MD) simulations and machine learning (ML) algorithms, to accelerate the discovery and development of potential AMPs for clinical use.[13–19]

Several studies have highlighted the promise of ML algorithms in predicting the antimicrobial activity, dissecting the complex SAR, and informing the drug design of AMPs.[13–15] A wide range of ML algorithms have been utilised, including random forests,[20] support vector machines (SVMs)[20–24] and artificial neural networks.[20–22,25,26] Many of these algorithms are used in combination with a carefully selected set of peptide features, which can be divided into two categories: compositional and physicochemical. The amino acid composition is the simplest example of a compositional feature, which is a vector containing counts of each amino acid in a given peptide. There are various extensions, such as the reduced amino acid composition[27] and the pseudo amino acid composition.[28] When computing the reduced amino acid composition, a peptide is represented in a reduced alphabet in which similar amino acids are grouped together. The pseudo amino acid composition accounts for composition as well as sequence-order information, as this is not considered in the standard amino acid composition. The set of physicochemical features include peptide properties such as the charge, hydrophobicity and isoelectric point.[20,24,29] These features are typically average values of the respective properties calculated over the length of the peptide.

Classical sequence alignment algorithms, such as the Smith–Waterman[30] and Needleman–Wunsch[31] algorithms, are computationally intensive and do not scale well to large problems. Many papers have advocated the use of alignment-free methods to determine sequence similarity.[19,32–35] The success of these endeavours notwithstanding, sequence alignment functions are effective notions of biological-sequence similarity that can reflect ancestral, structural or functional similarity and therefore should not be overlooked. Several studies have utilised sequence alignment functions for AMP prediction. For example, Wang *et al.*[36] and Ng *et al.*[37] utilised the BLAST algorithm[38] in a classification model by comparing the BLAST similarity scores of a query peptide to all those in the training set. Whilst these approaches led to accurate models, the BLAST algorithm is a heuristic method that finds only approximate optimal alignments. This approximation leads to generally faster results than what could be obtained by the Smith–Waterman algorithm, and it is one of the main reasons practitioners choose to use it. However, on the relatively small datasets in the aforementioned studies, it is interesting to consider whether the same approaches using the optimal alignment score would improve the models.

The SVM is a well-known ML algorithm for classification and can incorporate a kernel function in order to learn non-linear classification boundaries. The kernel function greatly influences the performance of the resulting classification model. When appropriately normalised, a kernel function can be regarded as a similarity function. A useful kernel function should produce similarities that are relevant to the problem. Many expressive notions of similarity are not valid kernel functions,[39–42] in that they are indefinite, meaning they cannot be used with an SVM. Recent developments have now alleviated this problem, facilitating a much larger class of similarity functions to be used in conjunction with an SVM. Loosli *et al.*[1] present an algorithm for learning an SVM with indefinite kernels. Their approach relies on a method of stabilisation, meaning there is no guarantee of global optimality. On the other hand, the Kreĭn-SVM[3] is an algorithm for learning an SVM with indefinite kernels that is guaranteed to find a globally optimal solution.

In this work, we utilised the Kreĭn-SVM algorithm to assess the effectiveness of sequence alignment functions for AMP classification. We performed an empirical comparison of both the local alignment score and Levenshtein distance[43,44] on two AMP datasets from the literature. Furthermore, we tested the ability of our approach to detect activity against specific species on a dataset of experimentally validated peptides measured against both *Staphylococcus aureus* ATCC 29213 and *Pseudomonas aeruginosa* ATCC 27853, henceforth referred to as *S. aureus* ATCC 29213 and *P. aeruginosa* ATCC 27853. Our trained models are made available as web applications at **http://comp.chem.nottingham.ac.uk/KreinAMP**, for the prediction of both general and species-specific activities.

2 Methods

AMP prediction models were developed using the Kreĭn-SVM algorithm in conjunction with sequence alignment functions, which we formally define in this section. We initially describe the more familiar SVM, before moving on to the Kreĭn-SVM. We then define the Levenshtein distance[44] and the local alignment score. Finally, we describe our computational and microbiological methodology.

2.1 SVM and Kreĭn-SVM

2.1.1 SVM. The SVM is a ML algorithm used for classification. The result of training an SVM is a hyperplane whose distance to the closest training instance, in either class, is maximal. Furthermore, instances from each class are required to reside on separate sides of the hyperplane. New instances are classified based solely on which side of the hyperplane they are located. The distance from the hyperplane to the closest training instance is known as the margin. The hyperplane that maximises the margin is the maximum-margin hyperplane and this is what is produced when training an SVM.

The decision surface associated with a hyperplane is inherently linear, which can be restrictive when the two classes are not linearly separable. This issue is mitigated through the use of a kernel function, which implicitly maps the instances into a new space. The space in which the instances are mapped is known as a reproducing kernel Hilbert space (RKHS), and every kernel function is uniquely associated to a RKHS. When incorporating a kernel function, the SVM finds the maximum-margin hyperplane in the associated RKHS and this can

correspond to a non-linear decision surface in the original space. This surface allows greater flexibility when separating the two classes but also increases the possibility of overfitting. Perfectly separating a noisy dataset is often indicative of overfitting. The Soft Margin SVM helps alleviate this problem by allowing some instances to reside on the incorrect side of the hyperplane. Eqn (1) presents the optimisation problem that is solved when training a Soft Margin SVM with L2 loss, commonly known as an L2-SVM. We opted for an L2-SVM as greater penalisation is placed on instances residing on the incorrect side of the hyperplane.

$$\begin{aligned} &\underset{f\in\mathscr{H},\xi\in\mathbb{R}^n}{\arg\min} && \sum_{i=1}^{n}\xi_i^{\,2}+\lambda\|f\|_{\mathscr{H}}^{\,2}, \\ &\text{subject to} && y_i f(x_i)\geq 1-\xi_i, \quad i=1,\ldots,n \end{aligned} \tag{1}$$

We denote by x_i the i-th training instance and by y_i its corresponding label. Furthermore, $\mathscr{H}$ denotes the RKHS from which a solution is found. The solution is a function $f\in\mathscr{H}$ and a vector of slack variables ξ which minimise the objective function, subject to the constraints. The constraint $y_i f(x_i)\geq 1-\xi_i$ imposes that the i-th instance lies on the correct side of the hyperplane and that its distance to the hyperplane is greater than or equal to the margin. An instance x_j which does not satisfy this constraint contributes a value of $\xi_j^{\,2}$ to the objective function, where ξ_j is the distance between x_j and the margin. The $\sum_{i=1}^{n}\xi_i^2$ term in the objective function is the total contribution of all instances which do not satisfy the constraints. The $\|f\|_{\mathscr{H}}^{\,2}$ term is the norm of the considered function and acts as a measure of complexity. The hyperparameter λ can be tuned to provide a balance between the number of instances that violate the constraints and the complexity of the considered function.

2.1.2 Kreĭn-SVM. The Kreĭn-SVM is a classification algorithm that is defined to incorporate a much broader class of kernel functions, known as indefinite kernel functions. Similarly to a standard kernel function, an indefinite kernel function implicitly maps instances into a new space. However, the space associated to an indefinite kernel function is known as a reproducing kernel Kreĭn Space (RKKS). Whilst operating in different spaces, the SVM and Kreĭn-SVM are conceptually similar. Both algorithms incorporate a kernel function, find the maximum-margin hyperplane in the associated space and are capable of learning non-linear decision surfaces.

$$\begin{aligned} &\underset{f\in\mathcal{K},\xi\in\mathbb{R}^n}{\arg\min} && \sum_{i=1}^{n}\xi_i^{\,2}+\lambda_+\|f_+\|_{\mathscr{H}_+}^{\,2}+\lambda_-\|f_-\|_{\mathscr{H}_-}^{\,2}. \\ &\text{subject to} && y_i f(x_i)\geq 1-\xi_i, \quad i=1,\ldots,n. \end{aligned} \tag{2}$$

Eqn (2) presents the optimisation problem that is solved when training the Kreĭn-SVM with L2 loss. We denote by $\mathcal{K}$ the associated RKKS. Similarly to the SVM, the solution is a function $f\in\mathcal{K}$ and a vector of slack variables ξ which minimise the objective function. The first term in the objective function, as well as the constraints, have the same interpretation as in eqn (1). The only notable difference to the SVM is the method of regularisation. Any RKKS $\mathcal{K}$ can be expressed as a direct sum of the form

$$\mathcal{K}=\mathscr{H}_+\oplus\mathscr{H}_-,$$

where $\mathscr{H}_\pm$ are RKHSs. This means that any function $f\in\mathcal{K}$ can be decomposed as $f=f_+\oplus f_-$, where $f_\pm\in\mathscr{H}_\pm$. Hence, regularisation in eqn (2) is performed by separately regularising each component of the decomposition. The hyperparameters $\lambda_\pm$ can be tuned to provide a balance between the number of instances that violate the constraints and the regularisation of each decomposition component.

2.2 Sequence similarities and distances

We now proceed to define the sequence similarities used throughout this work. This section closely follows the works of Setubal and Meidanis[45] and Yujian and Bo.[43] First, we clarify our terminology.

2.2.1 Notation and terminology. Let Σ be a finite alphabet, $\Sigma^n\subseteq\Sigma$ be the set of all strings of length n from Σ and Σ^* the set of all strings from that alphabet. A string $s\in\Sigma^*$ of length n is a sequence of characters that can be indexed as $s=s_1\ldots s_n$. Given a string $s\in\Sigma^n$, we say that $u\in\Sigma^m$ is a subsequence of s if there exists a set of indices $I=\{i_1,\ldots,i_m\}$ with $1\leq i_1\leq\ldots\leq i_m\leq n$, such that $u_j=S_{i_j}$ for $j=1,\ldots,m$. We write $u=s[I]$ for short. We say that $v\in\Sigma^l$ is a substring of s if v is a subsequence of s with index set $J=\{j_1,\ldots,j_l\}$ such that $j_{r+1}=j_r+1$ for $r=1,\ldots,l-1$. That is, v is a subsequence consisting of consecutive characters of s.

2.2.2 Global alignments. The goal of a sequence alignment is to establish a correspondence between the characters in two sequences. In the context of bioinformatics, a pairwise alignment can indicate ancestral, structural or functional similarities between the pair of sequences. In this section, we provide a formal review of global sequence alignment.

Definition 2.1 (global alignment). Let Σ be an alphabet and let $s\in\Sigma^n$ and $t\in\Sigma^m$ be two strings over Σ. Define $\Sigma_g=\Sigma\cup\{\text{“}-\text{”}\}$ as the extension of Σ with the gap character “$-$”. The tuple $\alpha(s,t)=(s',t')$ is a global alignment of sequences s and t if and only if

(1) $s',t'\in\Sigma_g^*$

(2) $|s'|=|t'|=l$, such that $\max(n,m)\leq l\leq m+n$,

(3) The subsequence of s' obtained by removing all gap characters is equal to s,

(4) The subsequence of t' obtained by removing all gap characters is equal to t,

(5) $\{i\,|\,s_i'=\text{“}-\text{”}\}\cap\{i\,|\,t_i'=\text{“}-\text{”}\}=\varnothing$.

Definition 2.1 provides a formal definition of global alignment. Whilst many possible alignments exist for two strings, the goal of sequence alignment is to find an alignment that optimises some criterion. A scoring function, presented in Definition 2.2, can be used to quantify the “appropriateness” of an alignment. An optimal global alignment is then one which is optimal with respect to a given scoring function, as shown in Definition 2.3.

Definition 2.2 (scoring functions). Let Σ be an alphabet, $\Sigma_g=\Sigma\cup\{\text{“}-\text{”}\}$ be the extension of Σ with the gap character “$-$” and $p:\Sigma_g\times\Sigma_g\to\mathbb{R}$ be a function defined over the elements of Σ_g. We say p is a similarity scoring function over Σ_g if, for all $x,y\in\Sigma$, we have

(1) $p(x,x)>0$,

(2) $p(x, x) > p(x, y)$,
(3) $p(x, y) = p(y, x)$,
(4) $p(x, \text{“}-\text{”}) \leq 0$,
(5) $p(\text{“}-\text{”}, \text{“}-\text{”}) = -\infty$.

Similarly, we say p is a distance scoring function over Σ_g if, for all $x, y, z \in \Sigma$, we have
(1) $p(x, x) = 0$,
(2) $p(x, y) > 0$,
(3) $p(x, y) = p(y, x)$,
(4) $p(x, \text{“}-\text{”}) > 0$,
(5) $p(x, z) \leq p(x, y) + p(y, z)$,
(6) $p(\text{“}-\text{”}, \text{“}-\text{”}) = \infty$.

Definition 2.3 (optimal global alignment). Let Σ be an alphabet, $\Sigma_g = \Sigma \cup \{\text{“}-\text{”}\}$ be the extension of Σ with the gap character “−” and consider two strings $s \in \Sigma^n$, $t \in \Sigma^m$. Let $\alpha(s, t) = (s', t')$ be a valid global alignment of s and t (valid in the sense that it satisfies the conditions of Definition 2.1), $p : \Sigma_g \times \Sigma_g \to \mathbb{R}$ be a scoring function over Σ_g and $\mathcal{A}(s, t)$ be the space of all valid alignments of s and t. The score $\mathcal{S}_p(\alpha(s, t))$ of $\alpha(s, t)$ with respect to the scoring function p is defined as

$$\mathcal{S}_p(\alpha(s,t)) = \sum_{i=1}^{l} p\left(s_i', t_i'\right).$$

If p is a similarity scoring function, then the optimal global alignment $\alpha^*(s, t)$ with respect to p is defined as

$$\alpha^*(s,t) = \underset{\alpha(s,t)\in\mathcal{A}(s,t)}{\arg\max} \mathcal{S}_p(\alpha(s,t)).$$

Similarly, if p is a distance scoring function then the optimal global alignment $\alpha^*(s, t)$ with respect to p is defined as

$$\alpha^*(s,t) = \underset{\alpha(s,t)\in\mathcal{A}(s,t)}{\operatorname{argmin}} \mathcal{S}_p(\alpha(s,t)).$$

Since an alignment is optimal with respect to a given scoring function, it is natural to consider which scoring function to use in order to obtain the most meaningful alignments. In the context of biological sequences, researchers have been considering this problem for many years. A number of families of scoring matrices have been designed to encode useful notions of similarity. In this work, we only considered the BLOSUM62 scoring matrix,[46] as it is a standard choice when performing sequence alignment.

2.2.3 Levenshtein distance. The string edit distance defines a useful notion of distance between a pair of strings. It is informally defined as the minimum number of edit operations required to transform one string into another. The Levenshtein distance is a variant of the string edit distance that allows the operations of substitution, deletion and insertion of characters, and these are defined in Definition 2.4.

Definition 2.4 (elementary edit operations). Let Σ be an alphabet. For two characters $a, b \in \Sigma$, we denote by $a \to b$ the elementary edit operation that substitutes a with b. Denoting by ε the null character (the empty string), we can define the elementary edit operations of insertion and deletion as $\varepsilon \to b$ and $a \to \varepsilon$, respectively.

In order to define more complex transformations, one can consider the consecutive application of a sequence of elementary edit operations. Of special interest to the Levenshtein distance are those sequences of operations that transform one string into another. Such a sequence is known as an edit path and its length is defined as the number of operations in the sequence. The Levenshtein distance between two strings is defined as the length of the minimum length edit path, as seen in Definition 2.5.

Definition 2.5 (Levenshtein distance). Let Σ be an alphabet and consider two strings $s \in \Sigma^n$ and $t \in \Sigma^m$ from Σ. An edit path from s to t is denoted by $P_{s,t}$ and represents a sequence of elementary edit operations that transforms s into t. Denote by $|P_{s,t}|$ the number of operations contained in $P_{s,t}$ and by $\mathcal{P}_{s,t}$ the space of all edit paths from s to t. The Levenshtein distance $d_L(s, t)$ between s and t is defined as

$$d_L(s,t) = \min_{P_{s,t}\in\mathcal{P}_{s,t}} \left|P_{s,t}\right|$$

Definition 2.5 shares some interesting similarities with Definition 2.3. Both problems solve a combinatorial optimisation problem and, indeed, the Levenshtein distance can be realised as a special case of global alignment. More specifically, consider the distance scoring function p: $\Sigma_g \times \Sigma_g \to \{0, 1\}$ defined as

$$p(x,y) = \begin{cases} 0, & \text{if} \quad x = y \\ 1, & \text{otherwise} \end{cases}$$

For two strings $s \in \Sigma^n$ and $t \in \Sigma^m$, let their optimal global alignment with respect to p be equal to $\alpha^*(s, t) = \{s', t'\}$. Define the set U as

$$U = \left\{ i \middle| s_i' \neq t_i' \right\}.$$

Then the score $\mathcal{S}_p(\alpha^*(s, t))$ of $\alpha^*(s, t)$ with respect to p is equal to the cardinality of U. This is exactly equal to the Levenshtein distance between s and t.

2.2.4 Local alignments. A global alignment produces an alignment which spans the whole length of a pair of strings. It is based on the assumption that the strings are related in their entirety. This assumption can be restrictive, since it is often the case that certain substrings exhibit high similarity whilst others do not. A local alignment produces an alignment that finds those high similarity substrings. That is, it finds the highest scoring global alignment from all possible substrings of the pair of strings. We formalise this notion in Definition 2.6.

Definition 2.6 (optimal local alignment). Let Σ be an alphabet, $\Sigma_g = \Sigma \cup \{\text{“}-\text{”}\}$ be the extension of Σ with the gap character “−” and $p : \Sigma_g \times \Sigma_g \to \mathbb{R}$ be a similarity scoring function. For a string $s \in \Sigma^n$, let $\mathcal{I}_s$ be the space of all index sets such that for any $I_s \in \mathcal{I}_s$, $s[I_s]$ is a valid substring of s. Similarly, for a string $t \in \Sigma^m$, let $\mathcal{I}_t$ be the space of all index sets such that for any $I_t \in \mathcal{I}_t$, $t[I_t]$ is a valid substring of t. For any $I_s \in \mathcal{I}_s$ and $I_t \in \mathcal{I}_t$, denote by $\alpha^*(s[I_s], s[I_t])$ the optimal global alignment of $s[I_s]$ and $t[I_t]$ with respect to

p. The optimal local alignment $\alpha_L^*(s,t)$ of s and t with respect to p is defined as

$$\alpha_L^*(s,t) = \underset{I_s \in \mathcal{I}_s, I_t \in \mathcal{I}_t}{\arg\max} \mathcal{S}_p\left(\alpha^*(s[I_s], t[I_t])\right).$$

2.3 Computational methodology

This section discusses the setup of our computational evaluation, as well as the datasets used.

2.3.1 Computational setup. To assess the usage of learning with sequence alignment functions, we performed a series of computational experiments on a number of AMP datasets. In each of our evaluations, we tested both the local alignment score (LA) and the Levenshtein distance (LEV) in conjunction with the Kreĭn-SVM algorithm. We compare against two baselines: an SVM with amino acid composition kernel and an SVM using the gapped k-mer kernel. The former is a positive-definite kernel function; peptides are represented *via* their amino acid composition and the kernel is defined as the inner product under this representation. The latter is also a positive-definite kernel function. It has produced accurate models in a number of biological-sequence classification tasks[47–49] and hence makes for a useful baseline. When applicable, we also compared our models with AMP identification tools from the literature. The parasail package[50] was used to compute local alignment scores. We only considered normalised variants of the local alignment score and Levenshtein distance, with the normalisation performed according to Schölkopf *et al.*[51] and Yujian and Bo,[43] respectively. We report the accuracy, the area under the receiver operating characteristic curve (AUC) and Matthews correlation coefficient (MCC) to compare models. The accuracy and MCC are defined in eqn (3), where TP, TN, FP and FN are the number of true-positives, true-negatives, false-positives and false-negatives, respectively.

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}} \tag{3}$$

$$\mathrm{MCC} = \frac{(\mathrm{TP} \times \mathrm{TN}) - (\mathrm{FN} \times \mathrm{FP})}{(\mathrm{TP} + \mathrm{FN}) \times (\mathrm{TN} + \mathrm{FP}) \times (\mathrm{TP} + \mathrm{FP}) \times (\mathrm{TN} + \mathrm{FN})} \tag{4}$$

The AUC is defined as the probability that a classifier will score a randomly chosen positive instance higher than a randomly chosen negative instance.[52] In order to allow for a fair comparison, all models used the same training and test splits. The optimal hyperparameters were selected by performing an exhaustive grid search over the training set, using 10-fold cross validation. The λ hyperparameter of the SVM algorithm, as well as the λ_+ and λ_- hyperparameters of the Kreĭn-SVM algorithm, were selected from $\{0.01, 0.1, 1, 10, 100\}$. The Levenshtein distance and amino acid composition kernel have no hyperparameters to control; we used the default values for the hyperparameters of the local alignment score. The gapped k-mer kernel has two hyperparameters g and m and is quite susceptible to their values. The optimal value of g was selected from $\{1, 2, 3, 4, 5\}$ and the optimal value of m was selected from $\{1, 2, 3, \ldots, 10\}$. It is required that $g > m$, so only valid combinations of the two were considered. In our nested cross-validation experiments, we used 10 inner and 10 outer folds and the reported results are averaged over the outer fold test sets.

2.3.2 General antimicrobial datasets. We selected two AMP classification datasets from the literature, which we refer to as AMPScan[26] and DeepAMP,[25] in order to test the ability of approach to predict general antimicrobial activity. Detailed discussions on the creation of these datasets can be found in the original studies. The AMPScan and DeepAMP datasets contain 3556 and 3246 instances, respectively. Each dataset also contains a 50 : 50 ratio of AMPs to non-AMPs, allowing us to avoid issues that result from class imbalance. Associated to each dataset is a specific test set, and reporting results on this set allows comparison with the authors' proposed models. Despite being of similar size, one major differentiating factor between the two datasets is the length of peptides. Fig. 1 displays the empirical distribution of peptide lengths for both datasets, partitioned by peptide classification. In both cases, the distributions corresponding to AMPs and non-AMPs are very similar. However, the distributions across datasets are clearly very different. The Kolmogorov–Smirnov two-sample test[53] provides evidence to reject the null hypothesis that the peptide length distributions of the AMPScan and DeepAMP datasets are identical. DeepAMP contains generally shorter peptides than AMPScan. Indeed, the DeepAMP dataset was curated since short-length AMPs have been shown to exhibit enhanced activity, lower toxicity and higher stability as opposed to their longer counterparts.[54,55] More importantly, synthesis is cheaper for the short AMPs than the full-length AMPs, which increases the potential for clinical translation and commercialisation.[12] Fig. 2 displays the empirical amino acid distributions for both datasets, indicating their similarity.

2.3.3 Species-specific datasets. To test the ability of our methodology to identify activity against specific species, we have utilised an external dataset of 16 peptides with minimum inhibitory concentration (MIC) measured against *S. aureus* ATCC 29213 and *P. aeruginosa* ATCC 27853. To make this dataset suitable for classification, we label a peptide as active if its MIC <100 $\mu g\ mL^{-1}$ and inactive otherwise. Further details of the microbiological experimentation used to construct this dataset can be found in Section 2.4.

Vishnepolsky *et al.*[56] have shown that, given an appropriate training dataset, predictive models of peptide activity against specific species can be constructed. This involves training a separate model for each species of interest, which, in this case, was the DBSCAN algorithm.[57] Their model predicting activity against *E. coli* ATCC 25922 achieved a balanced accuracy of 0.79, which was greater than a number of common AMP prediction tools.[56] Furthermore, models to predict activity against *S. aureus* ATCC 25923 and *P. aeruginosa* ATCC 27853 were made publicly available as web-tools.

We follow the methodology of Vishnepolsky *et al.* to construct useful training datasets for our problem. We utilised the Database of Antimicrobial Activity and Structure of Peptides

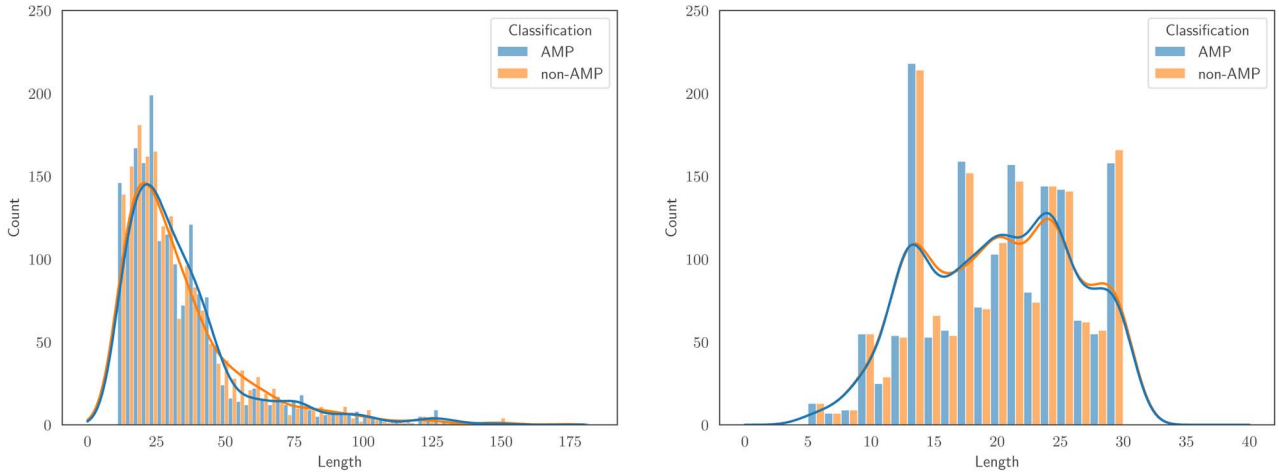

Fig. 1 The distribution of peptide lengths for (a) the AMPScan and (b) DeepAMP datasets.

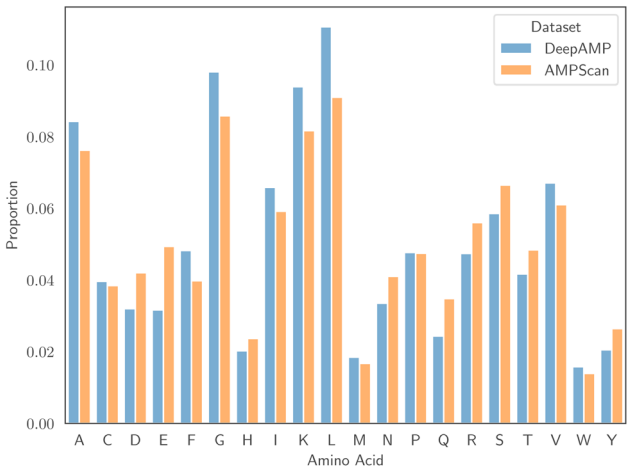

Fig. 2 The distribution of amino acids for the AMPScan and DeepAMP datasets.

(DBAASP) as a source of data. DBAASP contains peptide activity measurements against a wide-range of species,[58] including those of interest to us. We extracted from DBAASP all peptides with activity measured against *S. aureus* ATCC 29213, *S. aureus* ATCC 25923 or *P. aeruginosa* ATCC 27853 subject to the following conditions: (i) peptide length in the range [6, 18], (ii) without intrachain bonds, (iii) without non-standard amino acids and (iv) MIC measured in μg mL^{-1} or μM. Condition (i) was imposed as that is the range of peptide lengths in our external test set. Conditions (ii) and (iii) were imposed following the recommendation of the Vishnepolsky *et al.* and condition (iv) was imposed as conversion from μM to μg mL^{-1} is possible by estimating the molecular weight of a given sequence. Since no web-tool to predict activity against *S. aureus* ATCC 29213 was available, we couldn't directly compare our results. Instead, we collected data for peptides active against *S. aureus* ATCC 25923. This allowed us to compare our models with the state of the art provided by Vishnepolsky *et al.*

Three separate datasets of peptides with activity measured against *S. aureus* ATCC 29213, *S. aureus* ATCC 25923 and *P. aeruginosa* ATCC 27853 were created using the data collected from DBAASP. We refer to these datasets as SA_{29213}, SA_{25923}, and PA_{27853}, respectively. The peptides in the respective datasets were labelled according to their activity against the specific strain. Each dataset is constructed from highly active peptides (MIC ≤25 μg mL^{-1}) and inactive peptides (MIC ≥100 μg mL^{-1}). A peptide with 25 μg mL^{-1} < MIC <100 μg mL^{-1} would not be included in our training dataset. This large interval allows us to account for experimental errors, which in turn increases the confidence in our class labels. In the case that a peptide was associated to multiple activity measurements, the median value was taken to represent its activity. As shown in Table 1, the three training datasets are all relatively small and contain slightly more active peptides than inactive peptides.

2.4 Microbiological experiments

A previously established dataset of 16 short-length peptides (18 amino acids or shorter in length) was used to test the ability of the developed ML algorithms in predicting antimicrobial activity against *S. aureus* and *P. aeruginosa*.[7] These peptides were commercially synthesised by Mimotopes (Mulgrave Victoria, Australia) *via* solid phase Fmoc synthesis and were purified by reverse phase high performance liquid chromatography (RP-HPLC) to >95% purity. The efficacy of these peptides was already experimentally validated using established MIC assay with broth microdilution method approved by the Clinical and

Table 1 Descriptive statistics of the SA_{29213}, SA_{25923} and PA_{27853} datasets

Dataset	Size	Class ratio
SA_{29213}	463	0.644
SA_{25923}	808	0.646
PA_{27853}	686	0.547

Laboratory Standards Institute. Full detail of the previously conducted microbiological experiment can be found in the previous study.[7]

3 Results

In Section 3.1, we discuss the ability of our models to identify general antimicrobial activity. We observe that our proposed models consistently outperform the baselines and, in some cases, the models proposed in the literature. One shortcoming of any computational model that identifies general antimicrobial activity is that it cannot be used to identify activity against specific species. We address this shortcoming in Section 3.2, by training our models to identify activity against *S. aureus* ATCC 29213 and *P. aeruginosa* ATCC 27853 on an experimentally validated dataset of 16 peptides, for which the proposed approach produces accurate models.

3.1 Identifying general antimicrobial activity

3.1.1 Nested cross-validation. The performance of the AMP classifiers on the considered datasets is reported in Table 2. The results are averaged over the multiple test sets generated by nested-cross validation. On both datasets, the proposed models achieve a greater average accuracy, AUC and MCC than the baselines, with the local alignment score achieving the best values in all cases. The Welch *t*-test,[59] with $p = 0.05$, is used to compare the test set AUC of our proposed models against the baseline SVM with Gapped *k*-mer kernel across the outer folds of nested cross-validation. Adjusting for the testing of multiple hypotheses with the Bonferroni correction, we observe a significant difference between the mean AUC of the local alignment score and that of the baseline SVM with Gapped *k*-mer kernel on the AMPScan dataset. All other comparisons, including those on the DeepAMP dataset, are not significant. The performance of all models is greater on the AMPScan dataset than on the DeepAMP dataset. This may, in part, be due to the fact that DeepAMP contains generally shorter peptides. Shorter peptides provide less information to the models, which can limit their discriminative ability.

3.1.2 Predefined test set. Table 3 reports the results of all models on the predefined test set associated with each dataset. For the sake of completeness, we also include the performance of the neural network-based classifiers proposed by the authors of each dataset. As for the nested cross-validation results, we observe that all models perform better on the AMPScan dataset than the DeepAMP dataset. Considering the former, the local alignment score achieves the largest accuracy, AUC and MCC, and is followed closely by the literature model. On the DeepAMP dataset, the performance is similar among all methods. The local alignment score achieves the best AUC but the Levenshtein distance achieves the best accuracy and MCC. However, the Levenshtein distance outperforms the literature model against all metrics. On both datasets, the baselines are the least predictive models. It is encouraging to observe that the sequence alignment functions can produce classifiers that match, and also outperform, the neural network-based classifiers.

Table 3 Quality of the predictions on the AMPScan and DeepAMP datasets. The accuracy, AUC and MCC are reported, computed on the predefined test sets. Results are presented for the Kreĭn-SVM with local alignment score (LA-KSVM), the Kreĭn-SVM with Levenshtein distance (LEV-KSVM), the SVM with Gapped *k*-mer kernel (GKM-SVM) and the SVM with amino acid composition kernel (AAC-SVM). Results from the respective neural network-based classifiers[25,26] proposed by the authors of each dataset are also presented, denoted by Literature

	AMPScan			DeepAMP		
Model	Accuracy	AUC	MCC	Accuracy	AUC	MCC
LA-KSVM	0.911	0.967	0.823	0.761	0.863	0.523
LEV-KSVM	0.904	0.960	0.809	0.798	0.860	0.596
GKM-SVM	0.900	0.954	0.801	0.782	0.838	0.564
AAC-SVM	0.870	0.929	0.742	0.771	0.853	0.543
Literature	0.910	0.965	0.820	0.771	0.853	0.543

3.2 Identifying species-specific activity

In this section, we highlight the ability of our models to identify AMPs that are active against specific species, particularly *S. aureus* ATCC 29213 and *P. aeruginosa* ATCC 27853. Table 4 displays the accuracy on the external test set for models trained on the AMPScan and DeepAMP datasets. Once again, we also include the performance of the neural-network-based classifiers proposed by the authors of the AMPScan and DeepAMP datasets. We observe that the performance of all models is very poor. We noticed in our investigations that the majority of models predicted active for a large proportion of the peptides. This

Table 2 Quality of the predictions on the AMPScan and DeepAMP datasets. The average accuracy AUC and MCC are reported (standard deviation in parentheses), computed over the outer fold test sets of the nested cross-validation procedure. Results are presented for the Kreĭn-SVM with local alignment score (LA-KSVM), the Kreĭn-SVM with Levenshtein distance (LEV-KSVM), the SVM with Gapped *k*-mer kernel (GKM-SVM) and the SVM with amino acid composition kernel (AAC-SVM)

	AMPScan			DeepAMP		
Model	Accuracy	AUC	MCC	Accuracy	AUC	MCC
LA-KSVM	0.920 (0.017)	0.969 (0.006)	0.842 (0.033)	0.760 (0.025)	0.821 (0.028)	0.522 (0.051)
LEV-KSVM	0.910 (0.021)	0.966 (0.010)	0.821 (0.042)	0.756 (0.032)	0.819 (0.029)	0.513 (0.063)
GKM-SVM	0.899 (0.015)	0.957 (0.007)	0.799 (0.030)	0.751 (0.032)	0.817 (0.029)	0.506 (0.066)
AAC-SVM	0.865 (0.023)	0.930 (0.009)	0.732 (0.044)	0.723 (0.031)	0.784 (0.035)	0.447 (0.061)

Table 4 Predictive accuracy of the Kreĭn-SVM with local alignment score (LA-KSVM), the Kreĭn-SVM with Levenshtein distance (LEV-KSVM) and the SVM with Gapped *k*-mer kernel (GKM-SVM) on the species-specific test sets of 16 peptides. Results from the respective neural network-based classifiers[25,26] proposed by the authors of each dataset are also presented, denoted by literature. The dataset column indicates which dataset a model was trained on. The heading *S. aureus* indicates the model was predicting activity against *S. aureus* ATCC 29213 and the heading *P. aeruginosa* indicates the model was predicting activity against *P. aeruginosa* ATCC 27853

Model	Dataset	*S. aureus*	*P. aeruginosa*
LA-KSVM	AMPScan	0.312	0.312
	DeepAMP	0.250	0.250
LEV-KSVM	AMPScan	0.312	0.312
	DeepAMP	0.312	0.312
GKM-SVM	AMPScan	0.312	0.312
	DeepAMP	0.375	0.375
AAC-SVM	AMPScan	0.312	0.312
	DeepAMP	0.312	0.312
Literature	AMPScan	0.250	0.250
	DeepAMP	0.438	0.438

general poor performance is to be expected. We attribute it to the fact that these models have been trained to recognise if a peptide exhibits antimicrobial activity against any type of species. It is therefore unreasonable to assume that they are able to discriminate activity against a specific species.

Table 5 displays the accuracy on the external test set for models trained on the species-specific datasets. We also present the performance of the web-tools provided by Vishnepolsky *et al.*[56] As mentioned in Section 2.3.3, at the time of publication, there is no web-tool to predict activity against *S. aureus* ATCC 29213. Hence, we also provide results for models trained on SA_{25923} and tasked with predicting activity against *S. aureus* ATCC 29213. There is clearly a general improvement over the models trained on the AMPScan and DeepAMP datasets, indicating that the models have much greater discriminative power. On the SA_{29213} dataset, the baseline SVM with amino acid composition kernel is the most predictive with respect to all the metrics. All remaining models achieve the same accuracy and MCC. Considering the SA_{25923} dataset, the Levenshtein distance produces a model with the same accuracy and MCC as the web-tool but with a larger AUC. It also achieves the same accuracy and AUC as the baseline SVM with amino acid composition kernel, but with a larger MCC. It is interesting to note that the models trained on SA_{25923} can still make accurate predictions on *S. aureus* ATCC 29213. Whilst these are two different strains, the findings suggest that the antimicrobial susceptibility to the AMPs is similar for both strains, implying similar mechanisms work in the same species. Considering the PA_{27853} dataset, the baseline SVM with amino acid composition kernel performs the best against all metrics. We find that the local alignment score, Levenshtein distance and baseline SVM with Gapped *k*-mer kernel produce equally accurate models, all of which are more accurate than the web-tool. However, the AUC of the local alignment score and Levenshtein distance are considerably higher than that of the baseline SVM with Gapped *k*-mer kernel. Whilst it is difficult to make any strong conclusions on such a small dataset, it is still encouraging to observe that our models achieve similar accuracy to both the baseline models and web-tools.

Table 5 Quality of the predictions of the Kreĭn-SVM with local alignment score (LA-KSVM), the Kreĭn-SVM with Levenshtein distance (LEV-KSVM), the SVM with Gapped *k*-mer kernel (GKM-SVM) and the SVM with amino acid composition kernel (AAC-SVM) on the test sets of 16 peptides. Results from the DBAASP Web-tools are also presented.[56,58] The headings in the third to fifth columns indicate which dataset the models were trained on. The models trained on both SA_{29213} and SA_{25923} were tasked with predicting activity against *S. aureus* ATCC 29213. The models trained on PA_{27853} were tasked with predicting activity against *P. aeruginosa* ATCC 27853. Numbers highlighted in bold indicate the largest AUC achieved on the respective dataset

Model	Evaluation metric	Training dataset		
		SA_{29213}	SA_{25923}	PA_{27853}
LA-KSVM	Accuracy	0.688	0.688	0.750
	AUC	0.873	0.909	0.891
	MCC	0.522	0.405	0.595
LEV-KSVM	Accuracy	0.688	0.875	0.750
	AUC	0.927	**0.982**	0.855
	MCC	0.522	0.764	0.595
GKM-SVM	Accuracy	0.688	0.688	0.750
	AUC	0.945	0.891	0.655
	MCC	0.522	0.405	0.389
AAC-SVM	Accuracy	0.875	0.875	0.875
	AUC	**0.964**	**0.982**	**0.964**
	MCC	0.709	0.709	0.764
DBAASP Web-tool	Accuracy	—	0.875	0.688
	AUC	—	0.945	0.718
	MCC	—	0.764	0.522

4 Conclusions

We have assessed the capabilities of sequence alignment functions coupled with the Kreĭn-SVM as AMP classification models. Our investigations indicate that the proposed methodology produces accurate classifiers of both general and species-specific antimicrobial activity of AMPs. The utility of our methodology is twofold. Firstly, since sequence alignment algorithms operate directly on amino acid sequences, these methods do not explicitly require the use of peptide features. This removes the need for the practitioner to decide which features to use, which is often a detailed and time-consuming process. Secondly, in all of our experiments, we used the local alignment score with its default hyperparameters. Having achieved such promising results, it prompts the question of whether more accurate models could be attained by also tuning the various hyperparameters of the local alignment score, such as the choice of scoring function, which we will explore in future work.

As the chemical space of natural peptides is extremely large, the development of highly accurate classifiers will help accelerate the discovery and development of novel *de novo* AMPs. In addition, the promising results generated from this study open a number of possible avenues for further work. Our identification of activity against specific species could be improved using

a larger external test set, allowing us to draw stronger conclusions. Furthermore, the methodology we have presented is not specific to the classification of AMPs. We suspect that the Kreĭn-SVM coupled with sequence alignment functions could be applied to other biological-sequence classification tasks. Our computational findings demonstrate not only the feasibility of the proposed approach but more generally the utility of the Kreĭn-SVM as a classification algorithm. Its use of indefinite kernel functions provides a means for practitioners to learn from domain-specific similarity functions without the concern of verifying the positive-definite assumption. This is beneficial since it is often well beyond the expertise of the practitioner to verify this assumption. Whilst we have explored its use when combined with sequence alignment functions, there exist many more indefinite kernel functions with which the Kreĭn-SVM could be combined. Furthermore, the theoretical insight of separately regularising the decomposition components of a function in a Kreĭn space could be applied to develop other indefinite kernel-based learning algorithms. A notable example that relates to the current study would be to extend the One-Class SVM[60] to incorporate indefinite kernel functions. The One-Class SVM is a kernel-based learning algorithm that performs anomaly detection.[61] It has previously been applied to identify the domain of applicability of virtual screening models.[62] An indefinite kernel extension of the One-Class SVM could be directly applied to estimate the domain of applicability of our models.

Data availability

The source code and data used to reproduce the computational experiments is available at **https://github.com/Mrjoeybux/KreinAMP**.

Author contributions

Joseph Redshaw: conceptualisation, data curation, formal analysis, investigation, software, methodology, visualisation, writing – original draft. Darren S. J. Ting: conceptualisation, methodology, resources, writing – original draft. Alex Brown: validation, software, writing – review & editing. Jonathan D. Hirst: conceptualisation, methodology, project administration, resources, supervision, writing – review & editing. Thomas Gärtner: conceptualisation, methodology, supervision, writing – review & editing.

Conflicts of interest

A. B. is an employee and shareholder in GSK.

Acknowledgements

J. R. is supported by an NPIF PhD studentship through the MRC IMPACT Doctoral Training Programme [MR/S502431/1]. D. S. J. T. is supported by the Medical Research Council/Fight for Sight (FFS) Clinical Research Fellowship [MR/T001674/1] and the FFS/John Lee, Royal College of Ophthalmologists Primer Fellowship (24CO4). J. D. H. is supported by the Royal Academy of Engineering under the Chairs in Emerging Technologies scheme. We are grateful for access to the University of Nottingham High Performance Computer and to Alexe L. Haywood for useful discussions.

Notes and references

1 G. Loosli, S. Canu and C. S. Ong, *IEEE Trans. Pattern Anal. Mach. Intell.*, 2016, **38**, 1204–1216.
2 D. Oglic and T. Gärtner, *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 3859–3867.
3 D. Oglic and T. Gärtner, *International Conference on Machine Learning*, 2019, pp. 4912–4921.
4 N. Mookherjee, M. A. Anderson, H. P. Haagsman and D. J. Davidson, *Nat. Rev. Drug Discovery*, 2020, **19**, 311–332.
5 R. E. Hancock, E. F. Haney and E. E. Gill, *Nat. Rev. Immunol.*, 2016, **16**, 321–334.
6 D. S. J. Ting, I. Mohammed, R. Lakshminarayanan, R. W. Beuerman and H. S. Dua, *Front. Med.*, 2022, **9**, 835843.
7 D. S. J. Ting, E. T. L. Goh, V. Mayandi, J. M. Busoy, T. T. Aung, M. H. Periayah, M. Nubile, L. Mastropasqua, D. G. Said, H. M. Htoon, *et al.*, *Sci. Rep.*, 2021, **11**, 1–14.
8 V. Mayandi, Q. Xi, E. T. Leng Goh, S. K. Koh, T. Y. Jie Toh, V. A. Barathi, M. H. Urf Turabe Fazil, M. L. Somaraju Chalasani, J. Varadarajan, D. S. J. Ting, *et al.*, *J. Med. Chem.*, 2020, **63**, 3522–3537.
9 C. J. Murray, K. S. Ikuta, F. Sharara, L. Swetschinski, G. R. Aguilar, A. Gray, C. Han, C. Bisignano, P. Rao, E. Wool, *et al.*, *Lancet*, 2022, **399**, 629–655.
10 W. Ali, A. Elsahn, D. S. Ting, H. S. Dua and I. Mohammed, *Antibiotics*, 2022, **11**, 475.
11 C. D. Fjell, J. A. Hiss, R. E. Hancock and G. Schneider, *Nat. Rev. Drug Discovery*, 2012, **11**, 37–51.
12 D. S. J. Ting, R. W. Beuerman, H. S. Dua, R. Lakshminarayanan and I. Mohammed, *Front. Immunol.*, 2020, **11**, 983.
13 P. Das, T. Sercu, K. Wadhawan, I. Padhi, S. Gehrmann, F. Cipcigan, V. Chenthamarakshan, H. Strobelt, C. Dos Santos, P.-Y. Chen, *et al.*, *Nat. Biomed. Eng.*, 2021, **5**, 613–623.
14 N. Y. Yount, D. C. Weaver, E. Y. Lee, M. W. Lee, H. Wang, L. C. Chan, G. C. Wong and M. R. Yeaman, *Proc. Natl. Acad. Sci. U. S. A.*, 2019, **116**, 6944–6953.
15 D. S. J. Ting, J. Li, C. S. Verma, E. T. Goh, M. Nubile, L. Mastropasqua, D. G. Said, R. W. Beuerman, R. Lakshminarayanan, I. Mohammed, *et al.*, *Front. Pharmacol.*, 2021, 2793.
16 A. Capecchi, X. Cai, H. Personne, T. Köhler, C. van Delden and J.-L. Reymond, *Chem. Sci.*, 2021, **12**, 9221–9232.
17 C. Li, D. Sutherland, S. A. Hammond, C. Yang, F. Taho, L. Bergman, S. Houston, R. L. Warren, T. Wong, L. Hoang, *et al.*, *BMC Genomics*, 2022, **23**, 1–15.
18 P. G. Aronica, L. M. Reid, N. Desai, J. Li, S. J. Fox, S. Yadahalli, J. W. Essex and C. S. Verma, *J. Chem. Inf. Model.*, 2021, **61**, 3172–3196.
19 S. A. Pinacho-Castellanos, C. R. García-Jacas, M. K. Gilson and C. A. Brizuela, *J. Chem. Inf. Model.*, 2021, **61**, 3141–3157.

20 S. Thomas, S. Karnik, R. S. Barai, V. K. Jayaraman and S. Idicula-Thomas, *Nucleic Acids Res.*, 2010, **38**, D774–D780.
21 S. Lata, N. K. Mishra and G. P. Raghava, *BMC Bioinf.*, 2010, **11**, 1–7.
22 M. Torrent, D. Andreu, V. M. Nogués and E. Boix, *PLoS One*, 2011, **6**, e16968.
23 E. Y. Lee, B. M. Fulan, G. C. Wong and A. L. Ferguson, *Proc. Natl. Acad. Sci. U. S. A.*, 2016, **113**, 13588–13593.
24 P. K. Meher, T. K. Sahu, V. Saini and A. R. Rao, *Sci. Rep.*, 2017, **7**, 1–12.
25 J. Yan, P. Bhadra, A. Li, P. Sethiya, L. Qin, H. K. Tai, K. H. Wong and S. W. Siu, *Mol. Ther.-Nucleic Acids*, 2020, **20**, 882–894.
26 D. Veltri, U. Kamath and A. Shehu, *Bioinformatics*, 2018, **34**, 2740–2747.
27 P.-M. Feng, W. Chen, H. Lin and K.-C. Chou, *Anal. Biochem.*, 2013, **442**, 118–125.
28 K.-C. Chou, *Proteins: Struct., Funct., Genet.*, 2001, **43**, 246–255.
29 G. D. Rose, A. R. Geselowitz, G. J. Lesser, R. H. Lee and M. H. Zehfus, *Science*, 1985, **229**, 834–838.
30 T. F. Smith and M. S. Waterman, *J. Mol. Biol.*, 1981, **147**, 195–197.
31 S. B. Needleman and C. D. Wunsch, *J. Mol. Biol.*, 1970, **48**, 443–453.
32 A. Zielezinski, H. Z. Girgis, G. Bernard, C.-A. Leimeister, K. Tang, T. Dencker, A. K. Lau, S. Röhling, J. J. Choi, M. S. Waterman, *et al.*, *Genome Biol.*, 2019, **20**, 1–18.
33 M. R. Kantorovitz, G. E. Robinson and S. Sinha, *Bioinformatics*, 2007, **23**, i249–i255.
34 A. Zielezinski, S. Vinga, J. Almeida and W. M. Karlowski, *Genome Biol.*, 2017, **18**, 1–17.
35 P. Kuksa and V. Pavlovic, *BMC Bioinf.*, 2009, **10**, 1–18.
36 P. Wang, L. Hu, G. Liu, N. Jiang, X. Chen, J. Xu, W. Zheng, L. Li, M. Tan, Z. Chen, *et al.*, *PLoS One*, 2011, **6**, e18476.
37 X. Y. Ng, B. A. Rosdi and S. Shahrudin, *BioMed Res. Int.*, 2015, 212715.
38 S. F. Altschul, W. Gish, W. Miller, E. W. Myers and D. J. Lipman, *J. Mol. Biol.*, 1990, **215**, 403–410.
39 X. Gao, B. Xiao, D. Tao and X. Li, *Pattern Anal Appl.*, 2010, **13**, 113–129.
40 S. Axelsson, *Digit Investig*, 2010, **7**, S24–S31.
41 M. Müller, *Information Retrieval for Music and Motion*, 2007, pp. 69–84.
42 A. Feragen, F. Lauze and S. Hauberg, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3032–3042.
43 L. Yujian and L. Bo, *IEEE Trans. Pattern Anal. Mach. Intell.*, 2007, **29**, 1091–1095.
44 V. I. Levenshtein, *Sov. Phys. Dok.*, 1966, 707–710.
45 J. C. Setubal and J. Meidanis, *Introduction to Computational Molecular Biology*, PWS Pub. Boston, 1997.
46 S. Henikoff and J. G. Henikoff, *Proc. Natl. Acad. Sci. U. S. A.*, 1992, **89**, 10915–10919.
47 R. Wang, Y. Xu and B. Liu, *Sci. Rep.*, 2016, **6**, 1–10.
48 M. Ghandi, D. Lee, M. Mohammad-Noori and M. A. Beer, *PLoS Comput. Biol.*, 2014, **10**, e1003711.
49 D. Blakely, E. Collins, R. Singh, A. Norton, J. Lanchantin and Y. Qi, *Bioinformatics*, 2020, **36**, i857–i865.
50 J. Daily, *BMC Bioinf.*, 2016, **17**, 1–11.
51 B. Schölkopf, K. Tsuda and J.-P. Vert, *Kernel Methods in Computational Biology*, MIT press, 2004.
52 T. Fawcett, *Pattern Recognit. Lett.*, 2006, **27**, 861–874.
53 J. Hodges Jr, *Ark. Mat.*, 1958, **3**, 469–486.
54 H. Kim, J. H. Jang, S. C. Kim and J. H. Cho, *J. Antimicrob. Chemother.*, 2014, **69**, 121–132.
55 S. Ramesh, T. Govender, H. G. Kruger, B. G. de la Torre and F. Albericio, *J. Pept. Sci.*, 2016, **22**, 438–451.
56 B. Vishnepolsky, A. Gabrielian, A. Rosenthal, D. E. Hurt, M. Tartakovsky, G. Managadze, M. Grigolava, G. I. Makhatadze and M. Pirtskhalava, *J. Chem. Inf. Model.*, 2018, **58**, 1141–1151.
57 M. Ester, H.-P. Kriegel, J. Sander and X. Xu, *KDD'96: Proc. Second Intl. Conf. Knowledge Discovery and Data Mining*, 1996, pp. 226–231.
58 M. Pirtskhalava, A. A. Amstrong, M. Grigolava, M. Chubinidze, E. Alimbarashvili, B. Vishnepolsky, A. Gabrielian, A. Rosenthal, D. E. Hurt and M. Tartakovsky, *Nucleic Acids Res.*, 2021, **49**, D288–D297.
59 B. L. Welch, *Biometrika*, 1947, **34**, 28–35.
60 B. Schölkopf, R. C. Williamson, A. Smola, J. Shawe-Taylor and J. Platt, *NIPS'99: Proc. 12th Intl. Conf. Neural Information Processing Systems*, 1999, pp. 582–588.
61 V. Chandola, A. Banerjee and V. Kumar, *ACM Comput. Surv.*, 2009, **41**, 1–58.
62 N. Fechner, A. Jahn, G. Hinselmann and A. Zell, *J. Cheminf.*, 2010, **2**, 1–20.