OXFORD

Briefings in Bioinformatics, 2022, **23(2)**, 1–18
https://doi.org/10.1093/bib/bbac008
Problem Solving Protocol

Accelerating the discovery of antifungal peptides using deep temporal convolutional networks

Vishakha Singh, Sameer Shrivastava, Sanjay Kumar Singh, Abhinav Kumar and Sonal Saxena

Corresponding authors: Sameer Shrivastava, Division of Veterinary Biotechnology, ICAR-Indian Veterinary Research Institute, Izatnagar, 243122 Uttar Pradesh, India. E-mail: sameer.shrivastava@icar.gov.in; sameer_vet@rediffmail.com; Sanjay Kumar Singh, Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi 221005, Uttar Pradesh, India. E-mail:sks.cse@iitbhu.ac.in
Vishakha Singh and Sameer Shrivastava both contributed equally to this work.

Abstract

The application of machine intelligence in biological sciences has led to the development of several automated tools, thus enabling rapid drug discovery. Adding to this development is the ongoing COVID-19 pandemic, due to which researchers working in the field of artificial intelligence have acquired an active interest in finding machine learning-guided solutions for diseases like mucormycosis, which has emerged as an important post-COVID-19 fungal complication, especially in immunocompromised patients. On these lines, we have proposed a temporal convolutional network-based binary classification approach to discover new antifungal molecules in the proteome of plants and animals to accelerate the development of antifungal medications. Although these biomolecules, known as antifungal peptides (AFPs), are part of an organism's intrinsic host defense mechanism, their identification and discovery by traditional biochemical procedures is arduous. Also, the absence of a large dataset on AFPs is also a considerable impediment in building a robust automated classifier. To this end, we have employed the transfer learning technique to pre-train our model on antibacterial peptides. Subsequently, we have built a classifier that predicts AFPs with accuracy and precision of 94%. Our classifier outperforms several state-of-the-art models by a considerable margin. The results of its performance were proven as statistically significant using the Kruskal–Wallis H test, followed by a post hoc analysis performed using the Tukey honestly significant difference (HSD) test. Furthermore, we identified potent AFPs in representative animal (Histatin) and plant (Snakin) proteins using our model. We also built and deployed a web app that is freely available at https://tcn-afppred.anvil.app/ for the identification of AFPs in protein sequences.

Keywords: antifungal peptides, artificial intelligence, Snakin, Histatin, mucormycosis, temporal convolutional networks (TCNs), COVID-19.

Introduction

Our battle against disease-causing fungal pathogens has been long-standing, with no apparent winner. According to an estimate, more than one billion people are affected by fungal diseases, out of which 150 million have serious infections [1]. These infections range from cutaneous and subcutaneous infections that affect the skin, keratinous tissue, etc., to life-threatening systemic infections that affect organs like brain, kidneys and liver. Severe systemic infections are very rampant in the immunocompromised individuals [2, 3]. Nowadays, diseases like mucormycosis have created an alarming health crisis due to high morbidity and mortality rate in immunocompromised patients suffering from coronavirus disease 2019 (COVID-19). As of now, there are four conventional classes of antifungal or antimycotic agents that help fight fungal infections: the azoles, which block certain fungal enzymes, thereby impeding the synthesis of ergosterol (the main component of fungal cell membrane); the echinocandins, which inhibit the synthesis of glucan (an essential structural polysaccharide found in the cell wall of fungi); the polyenes, which bind to ergosterol and

Vishakha Singh. She is presently a research scholar in the Department of Computer Science and Engineering at IIT (BHU), Varanasi, India. She completed her M.Tech and B.Tech from IIT (ISM) Dhanbad and IIIT Jabalpur, respectively. She has authored and co-authored several articles in high-impact factor journals. Her research interests include deep learning, machine learning, sequence modeling, therapeutic peptide discovery and multi-objective optimization problems.
Sameer Shrivastava. He is presently a Senior Scientist at Division of Veterinary Biotechnology, IVRI, Izatnagar, India. He has >15 years of teaching and research experience in the area of Animal Biotechnology. His research interests include synthetic peptide biology, cancer biology antimicrobial resistance and biosensors. He has published >70 research papers and has 2 granted patents to his credit.
Sanjay Kumar Singh. He is presently a professor with the Department of Computer Science and Engineering, IIT (BHU), Varanasi, India. He has published over 150 peer-reviewed journal publications, book chapters and conference papers. He has also four patents filed to his credit. His research interests include machine learning, deep learning, computer vision, medical image analysis and pattern recognition.
Abhinav Kumar. He is presently a PhD research scholar in the Department of Computer Science and Engineering at IIT (BHU), Varanasi, India. He has authored and co-authored several top journal articles. He is also a teaching assistant in IIT (BHU), Varanasi, India. His research interests include ML, DL, medical imaging, computer vision and data mining.
Sonal Saxena. She is presently a senior scientist at Division of Veterinary Biotechnology, ICAR-IVRI, Izatnagar and has >12 years of teaching and research experience in the area of Animal Biotechnology. Her current area of research include cancer biology, new generation vaccines and computational biology. She has published >50 research papers in peer-reviewed journals. She is also a recipient of prestigious Lal Bahadur Shastri young scientist award.
Received: November 1, 2021. **Revised:** December 27, 2021. **Accepted:** January 6, 2022

depolarize the fungal cell membrane thereby increasing the membrane permeability and causing cell death, and the fluorinated pyrimidines, which inhibit deoxyribonucleic acid (DNA) and ribonucleic acid (RNA) biosynthesis, by interfering with pyrimidine synthesis in the cell membrane [4]. The constrained modes of action of these antimycotic agents have led to a situation where fungal pathogens have started showing resistance to most of them. This condition is known as multi-drug resistance. Moreover, some antifungals are highly cytotoxic to healthy somatic cells and may cause serious health issues.

In case of humans and animals, diseases caused by diverse species of fungi are very common, and the aforementioned antifungal drugs are routinely used to treat such conditions. But the fungi do not just affect humans and animals; they also have a debilitating impact on plants, resulting in low productivity or even destruction of the entire crop if not handled appropriately [5]. The dead and decaying plants, seeds, soil, weeds, etc., are a few sources of fungal infections in healthy plants. As of now, fungicides are used to prevent and cure plant fungal infections, but due to their toxicity to humankind [6], governments of various countries have started imposing stringent rules and norms on the use of such chemicals. Also, a large number of plant fungi have started exhibiting resistance against existing antifungals [7]. As eukaryotic pathogens, the fungi share many similarities with their host cells, making the development of antifungal compounds all the more difficult [4]. Further, the fungal tropism is highly variable, as these pathogens infect a wide range of cell types; thus, developing safe and non-toxic antifungal compounds is a real challenge.

Nature has provided an in-built defense mechanism in multicellular organisms, which protects them from invading microbes. Antifungal proteins are one such category of molecules that helps in preventing the attack of invading fungi [4]. These proteins are present in all living organisms. In the case of animals, fungi belonging to the genera Rhizopus, Mucor, Rhizomucor, Lichtheimia, Apophysomyces, Cunninghamella, Saksenaea, Candida, Aspergillus, Cryptococcus, etc., are the most common causes of fungal infections [8]. On the other hand, infections in plants are mostly caused by fungi of genera Albugo, Plasmodiophora, Pythium, Sclerotinia, Fusarium, Botrytis, Colletotrichum, Microdochium, etc. [9]. The antifungal proteins either cause lysis of the fungal cell membrane via different mechanisms or interfere with the synthesis of the fungal cell wall or biosynthesis of glucans and chitins, which are essential for maintaining the integrity of the cell wall [10]. These proteins have also shown potent activity in preventing the formation of fungal biofilms and eradication of pre-existing biofilms [11]. The antifungal proteins present in diverse species are supposed to contain some peptidic region or core domain that might perform the antifungal action. This core domain or antifungal peptides (AFPs) have been identified in different plant and animals, and several databases (e.g., [12, 13]) have been constructed for such sequences. Due to their broad-spectrum activity, low cytotoxicity and multiple modes of action, the discovery of novel AFPs has become vital. This is possible by active collaboration between the experts working in the field of bioinformatics, artificial intelligence (AI), and the biological sciences.

We are witnessing a revolution in information technology (IT), which is transforming the entire world. The widespread use of AI techniques like machine learning (ML) and deep learning (DL) is one of the most important reasons for bringing about this revolution. It is hard to think of a field where ML or DL is not being applied presently. Such techniques have several applications in biological sciences such as the prediction of T cell receptor–antigen binding specificity [14], screening of diseases like cervical cancer [15], breast cancer [16–19], dermatological diseases [20], pneumonitis [21], etc. Yet another area of application is the classification and discovery of antimicrobial peptides (AMPs) [22–27], in general, and AFPs, in particular, for which the researchers have built a number of classifiers. Ahmad *et al*. [28] used deep neural networks for predicting AFPs. Authors in [29–31] proposed models based on deep learning approaches like convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to classify AMPs. In [32], authors have developed two classifiers using support vector machine (SVM) and random forest (RF) for predicting antibacterial, antifungal and antiviral peptides. Agrawal *et al*. [33] used a number of machine learning classifiers including SVM, RF, etc. for predicting AFPs. In [34], authors used a variety of ML algorithms and found that SVM and Bagged-C4.5 achieved the highest performance in classifying AFPs. Similarly, in [35], authors used SVM for classification and prediction of plant-based AFPs. In yet another research, Meher *et al*. [36] used SVM to predict antibacterial, antiviral, antifungal and antiparasitic peptides. On the same lines, authors in [37] proposed an SVM-based model for predicting scores for 14 functional activities (antibacterial, antifungal, etc.) in a given peptide. Lin *et al*. [38] addressed the problem of imbalanced classes by constructing a synthetic dataset and building a multi-label (antibacterial, antiviral, antifungal, etc.) classification model. In [39], authors proposed three models based on algorithms such as random forest and RNN to perform classification of antibacterial, antifungal, antiviral peptides, etc. The AMPFUN model [40] consists of two stages, where in the first stage, the peptides are classified as AMPs/non-AMPs and in the second stage they are further classified into various functional types (antiviral, antifungal, targeting gram positive and gram negative bacteria and so on). Xiao *et al*. [41] proposed a fuzzy k-nearest neighbour based two-level classifier for predicting several types of AMPs. In [42], authors used CNN and bidirectional long short term memory (biLSTM) to classify, and identify AFPs. Moreover, in [43], the authors have proposed a multi-class

classification model for predicting various AMPs using bi-LSTM, CNN and SVM.

In general, the researchers working towards discovery of AFPs have restricted themselves to classical machine learning approaches like SVM, which use hand-engineered features like physicochemical properties (molecular weight, net charge, isoelectric point, etc.), structural properties (alpha helix propensity, beta sheet propensity, beta turn propensity), and compositional properties (amino acid (AA) composition, pseudo amino acid composition, etc.). Direct exploration and analysis of the primary structure of an AFP using deep learning sequence models is yet to be explored. Although some researchers have proposed a few deep learning models in this area, there is still a major scope for improvement. This shortcoming has been the main motivation behind the development and deployment of a deep learning model based on **T**emporal **C**onvolutional **N**etworks (TCNs) for **A**nti**f**ungal **p**eptide **pred**iction (TCN-AFPpred). The TCN [44, 45] architecture is an improvement over LSTM which is very frequently used to model sequence data. This model takes the sequence of AA residues of a given peptide as input and predicts whether it is antifungal or not. A free web application has been developed and deployed using this model at https://tcn-afppred.anvil.app/. We identified AFPs present in plant proteins (Snakin) and animal proteins (Histatin) using this app. These proteins are well-known for their antifungal properties. The Snakin proteins are antimicrobial in nature [46], showing broad-spectrum activity against a variety of fungi such as *Fusarium solani*, *Botrytis cinerea*, *Bipolaris maydis*, etc. Similarly, the Histatin proteins show antimycotic activity against fungi like *Candida albicans* [47]. The intent behind finding out AFPs in these proteins was to identify the core antifungal domain(s) that can be tested for antimycotic activity. These proteins were selectively picked from both the animal and plant kingdoms to show that our model can predict AFPs from the genome of organisms belonging to both these kingdoms. We have demonstrated the superior performance of our model with respect to other state-of-the-art models such as MLAMP [38], iAMPpred [36], AMPFUN [40], AMAP [37], Deep-AFPpred [42], iAMP-CA2L [43], AntifpMain_binary_model3 (Antifp) [33], AFPDiscover (AFPD), RNN-AFPDiscover (RNN-AFPD) and Hierarchical AFPDiscover (HAFPD) [39], using various performance metrics like precision, recall, accuracy, etc. We also performed non-parametric statistical analysis called Kruskal–Wallis H test (KWH) [48] to prove the better performance of TCN-AFPpred over other models.

The basic structure of this paper has been explained in Figure 1 and the significant contributions are listed as follows.

- For the first time, a TCN architecture-based model has been proposed for predicting therapeutic peptides.
- To the best of our knowledge, this model is the first to use the concept of transfer learning (TL) (using a pre-trained model) in this field. We initially train the model on a dataset of antibacterial peptides (ABPs), which is followed by fine-tuning of the model weights on the AFP dataset.
- We showed that our model exhibits better performance than other state-of-the-art classifiers and gave a statistical proof to our claim using the KWH [48], followed by a post hoc analysis done using Tukey HSD test [49].
- A web app based on this model has been developed and deployed for free use at https://tcn-afppred.anvil.app/ to predict novel AFPs in a given protein.
- Using the web app, we predicted novel AFPs in the animal (Histatin) and plant (Snakin) proteins and identified the core antifungal domain that can be chemically synthesized and tested for activity against different fungi.

The rest of the paper is organized as follows. Section 2.2 contains a brief overview of the data, tools and techniques used in this paper. In Section 3, the proposed model has been explained and elaborated. Section 4 comprises the experimental evaluation of our proposed model and its comparison with other models using various performance metrics. In this section, we also performed the KWH test and Tukey HSD test to ascertain that our model's performance is statistically different from others. Moreover, this section also presents the identification of core antifungal regions in some antifungal proteins. Lastly, in Section 5, we conclude our work and present future directions.

Materials, tools and techniques

In Section 2.1 the details about the data points (peptides) as well as their pre-processing has been elucidated. The TCN-AFPpred model built for identifying AFPs involves the repurposing of a pre-trained ABP prediction model. This process is known as transfer learning (TL), and it has been thoroughly explained in Section 2.3. In this work, we used three datasets, which have been detailed in Section 2.2.

Data pre-processing

The datasets described in Section 2.2 consist of peptides of different lengths. A peptide is represented by a chain of amino acid (AA) residues, and its length is nothing but the number of AAs contained in it. The AAs are represented by single letters of the English alphabet. To build our model, we considered those peptides whose lengths lie in the interval [4, 30]. Note that we chose only those peptides that contained standard AAs. So, a single data point in our dataset is a string of letters. E.g. *ACDDDEEEAA* is a peptide represented by a string consisting of A, C, D and E. Since machine learning algorithms take input in numerical format, the letters denoting standard AAs

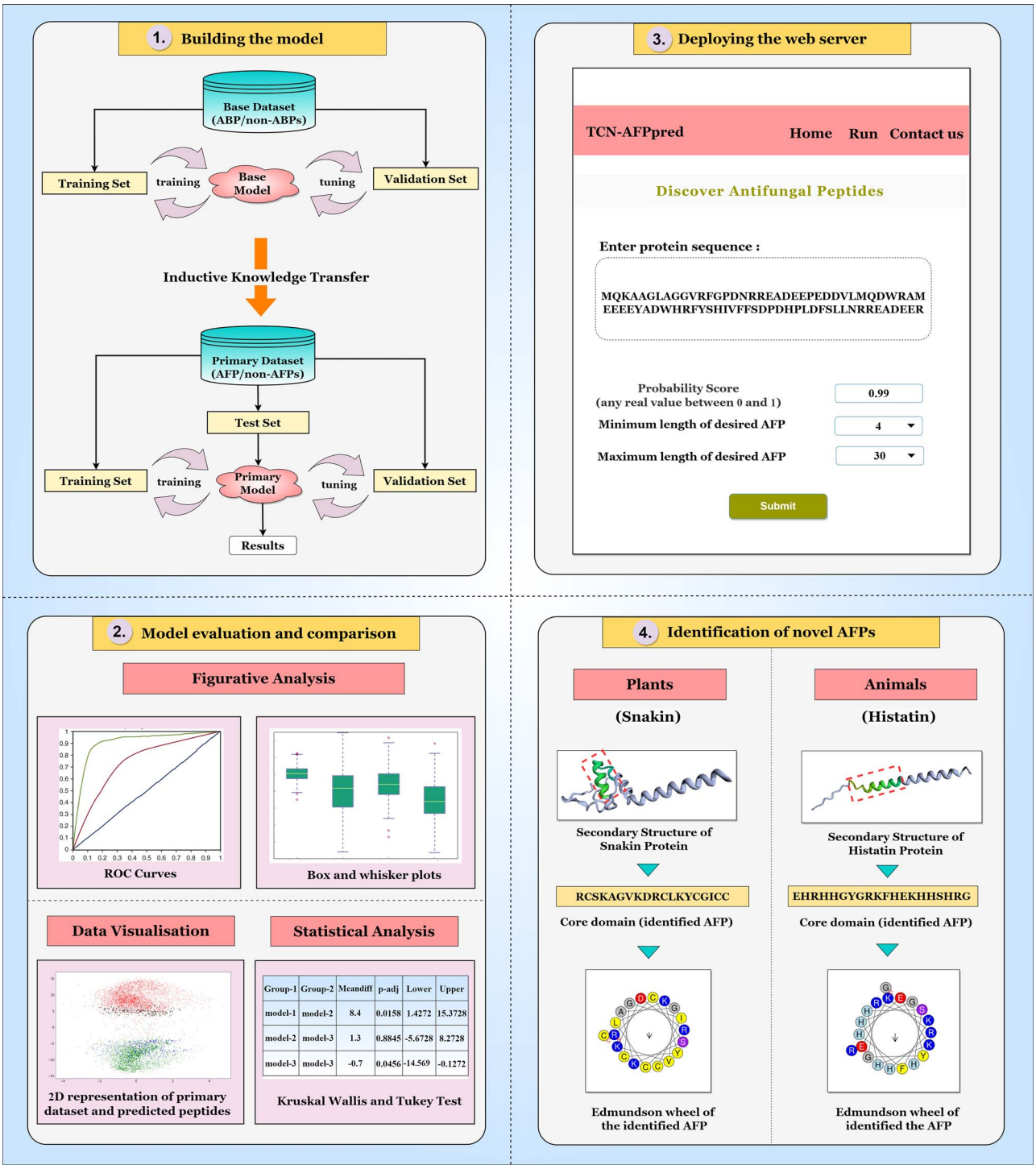

Figure 1. The overall workflow showing the processes involved in building, testing and deploying the TCN-AFPpred model.

were uniquely mapped to a set of integers. Then, each peptide's alphabetical string was tokenized and converted into a string of numbers using this mapping. The representative numerical strings of peptides with lengths less than 30 were padded with zeroes to bring uniformity. These strings were used as input to our model.

Datasets

The AFPs and ABPs present in the datasets were collected from various sources such as, the collection of antimicrobial peptides (CAMP) [50], the structure database of bioactive peptides (StraPep) [12], the bioactive peptide database (BioPD) [51], ANTIMIC database [52], the data repository of antimicrobial peptides (DRAMP) [53], the

antimicrobial peptide database (APD) [54], the milk antimicrobial peptides (MilkAMP) [55] database and the starPep database [13, 56, 57]. The non-ABPs and non-AFPs were obtained from the universal protein resource (UniProt) database [58].

Base dataset

The base dataset consists of 9507 peptides, out of which 3647 were ABPs and 5860 were non-ABPs. It was divided into training and validation sets in the ratio 4:1. Note that we have not used a test set because we just need an initial set of weights obtained after training the model on the base dataset. This pre-trained or base model will then be used for building our primary model using the primary dataset. However, for tuning the base model's hyper-parameters, we have used a validation set.

Primary dataset

The primary dataset consists of 3577 AFPs and 3474 non-AFPs (7051 peptides). It was subgrouped into three sets in which the training set comprised 60% data points, and validation and test sets contained 20% data points, each. The training set was used to retrain the base model (that has learned to classify ABPs using the base dataset). The validation set was used to fine-tune the model, and the test set was used to check its performance on unforeseen data points. Note that there were no common data points between the base and the primary dataset. This ensures that the data points used to build the base model are not used again while training or testing the primary model. Retraining on the same points is not harmful. However, if those points appear in the test dataset while testing the primary model, the model may give exaggerated results since it has already seen these data points while being trained on the base dataset.

Secondary dataset

We also built a secondary dataset comprising 5329 AFPs and non-AFPs in total. This dataset was created using the CD-HIT tool [59–61] that removed similar sequences from the primary dataset. The peptides in the secondary dataset were then divided into training (70%), test (15%) and validation (15%) sets, and the TCN-AFPpred was trained, tuned and tested on these sequences and compared with similar baseline models to assess the generalizability of our proposed model.

Transfer learning

TL is a concept where the knowledge gained after training a model on one task is transferred to retrain the same or another model on a similar task. This entails repurposing of a pre-trained model according to the new task. The process of TL used in this work is also known as an inductive transfer [62]. This concept is used because of the following reasons.

1) Since the tasks (predicting ABPs and AFPs) are somewhat similar, using a pre-trained set of weights is better than random initialization. This would lead to better training because we will be tuning a model that has already acquired some skills and knowledge on the base dataset.
2) When the number of data points for training our model on the primary task are less, it is always better to pre-train the model on a similar task for which we have sufficient data points. This gives an initial jump-start or boost while building our primary model.

The retraining of the base model may be done in two ways: either retrain the entire model (all the constituent layers of the model) or retrain just a few layers of the model. We tried both the alternatives but found that retraining all the layers was a better option.

Proposed Model

The model proposed in this paper is based on a relatively novel deep learning-based sequence modeling architecture called temporal convolutional network (TCN) [44], which has been outperforming various RNN-based models. This section discusses the concept of TCN-AFPpred's constituent layers, and how they are stacked together to form this model.

Layers of the TCN-AFPpred model

The backbone of the TCN-AFPpred model are the blocks of TCN that are used to analyse the input sequence. Many other layers have also been used before and after these blocks. These have been discussed elaborately in this section.

TCN blocks

As mentioned earlier, TCN is a novel deep learning architecture, which has been built by modifying CNNs for modeling sequence data [45]. A TCN block is a stack of multiple one-dimensional convolutional layers with various dilations (d). By dilations, we mean a mechanism that is used to increase the receptive field of a constituent convolutional layer by capturing dependencies between the steps (here the units of hidden layers are referred to as steps), which are separated by multiple other steps, as shown in Figure 2. Note that the receptive field is the scope or extent of the input space to which a particular step (unit) of a layer is exposed. Large dilation values allow for capturing very long-distance dependencies among steps, which may be essential for modeling some sequence-based tasks. The internal architecture of a TCN block having two hidden layers has been illustrated in Figure 2. Due to the introduction of dilations, the conventional convolution operation (i.e. performed by the convolutional layers in CNNs) gets modified in the case of TCNs. The operation so performed depends on whether we are employing causal or acausal TCNs in our model. In causal TCNs, only the past steps are used while calculating the current step t of the next layer. Whereas in the case of acausal TCNs, both past and

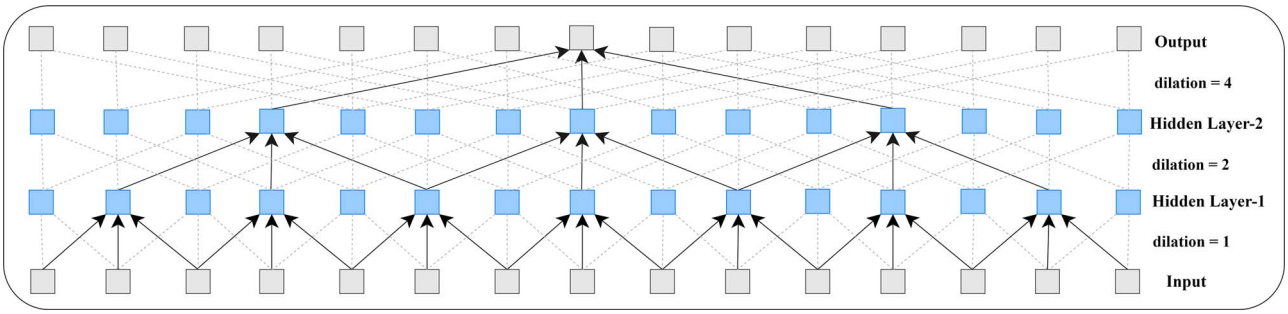

Figure 2. An acausal TCN block having one input, two hidden and one output layer of 15 steps each. Dilations of size 1, 2 and 4 were used.

future steps are taken into cognizance. The TCN-AFPpred model is built using acausal TCNs. Mathematically, the convolution operation ($C(t)$) performed for the t^{th} step of a dilated causal TCN layer is given by the following equation [45].

$$C(t) = (x *_d f)(t) = \sum_{i=1}^{k} f(i).x_{t-d.(i-1)} \quad (1)$$

Here, x represents the input, $*_d$ represents the convolution operation, f denotes a filter, i.e., $f = f(0), f(1), \cdots, f(k)$, with k filter taps (also called as kernel size) for performing the convolution operation, and d is the dilation. The filters are used to extract features, and dilations are used to introduce a fixed number of steps between two consecutive filter taps. Note that in this case, we convolve from $x_{t-d.(k-1)}$ to x_t. The convolution operation performed by a dilated acausal TCN layer, at step t, represented by $A(t)$ is given by Equation 2 [63]. Here, we convolve from $x_{t-d.\lceil (k-1)/2 \rceil}$ to $x_{t+d.\lfloor (k-1)/2 \rfloor}$.

$$A(t) = (x *_d f)(t) = \sum_{i=1}^{\lfloor k/2 \rfloor} f(\lceil k/2 \rceil + i) \cdot x_{t-(d \cdot i)} + \sum_{i=\lfloor k/2 \rfloor + 1}^{k} f(k - i + 1) \cdot x_{t+d.(i-\lfloor k/2 \rfloor + 1)} \quad (2)$$

TCNs are known to possess longer memory than RNNs and its variants (LSTMs, bidirectional-LSTMs, etc.). Also, TCNs allow for parallel processing because of the convolution operation. In other words, unlike LSTMs, to execute its function, there is no need for a step to wait for other steps to finish their calculations. Thus, long sequences are processed in comparatively much lesser duration than LSTMs. Moreover, LSTMs require a considerable amount of memory to store intermediate data like the values of input, output and forget gates, which is not the case with TCNs. Thus, TCNs are a better alternative to LSTMs, biLSTMs and other variants of RNN.

Embedding layer

This layer comes into play when we feed the input to our model. Our input is a string of numbers (which represent a particular peptide), and it has to be converted into a matrix of feature vectors for further processing. This feature matrix is the actual input on which a sequence modeling algorithm works. Following are the options for finding out the feature matrix.

1) **One-hot encoding**- In this method, we convert each integer contained in a given numerical string (that represents a peptide) into a binary feature vector that contains zeroes at all positions except at that position, which is equal to the value of that integer (this position is marked as 1). In our case, the length of each feature vector will be 20 (equal to the number of standard AAs). E.g., suppose we want to construct one-hot encoding of 20 standard AAs, {A, C, D, $\cdots$, W, Y}, which are represented by {1,2,3,$\cdots$,19,20}. Then, one-hot encoded feature vector for AA residue A will be {1, 0, 0, 0, 0,$\cdots$, 0}. Similarly, C will be encoded as {0, 1, 0, 0,$\cdots$, 0}, and so on. So, a peptide of length n shall become a feature matrix of size (n,20). The drawback of using this representation is that the feature matrix is sparse and its size increases dramatically with increase in the length of the peptide. Moreover, this type of encoding considers all AA residues as equally similar/dissimilar to one another, which is not the case in reality (each AA residue is more similar/dissimilar to some AAs than others).
2) **Word2vec embeddings**- This method generates feature matrix using an embedding matrix (in which each row represents the feature vector corresponding to a single AA residue). We usually compute such vectors using either of the two Word2Vec techniques, namely, the continuous-bag-of-words and skip-gram algorithm. The advantage of this technique is that the feature vectors are generated by finding the correlations between the AA residues according to the peptides contained in the dataset under consideration. This means that the algorithms involved in constructing such representations find similarities and dissimilarities between the AA residues. Thus, the AAs that are similar to each other have similar feature vectors. After constructing this embedding matrix, each peptide is transformed into a two-dimensional feature matrix using it.

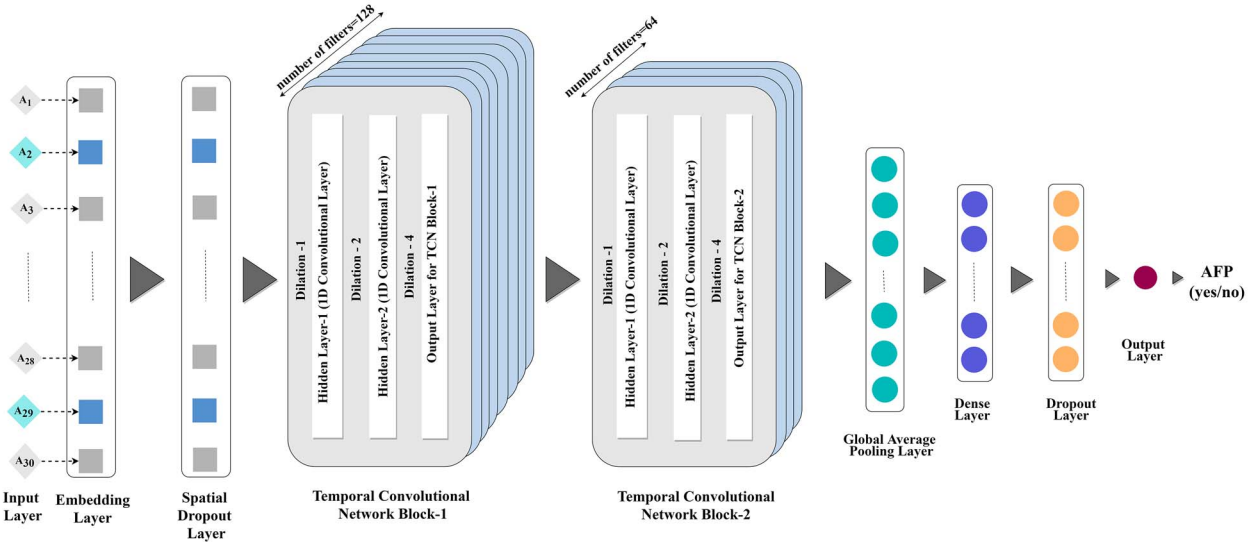

Figure 3. The overall architecture of the TCN-AFPpred model. The hidden layers of both TCN block-1 and TCN block-2 consist of 30 steps. The embedding layer generates a feature vector of size (30,200). TCN block-1 uses 128 filters and TCN block-2 uses 64 filters. The global average pooling layer is of size (64,1) and the subsequent dense layer has size (16,1).

3) **Pre-trained word embeddings**- One may also use pre-existing word embeddings instead of using their own feature matrix. This is a type of TL approach that involves the use of pre-formed word embeddings that were obtained using some natural language processing model on peptides available in public databases (e.g. SeqVec word embeddings [64]).

We have constructed our embedding layer using the Word2vec technique. Each AA residue was represented using a feature vector of length 200. Hence, each peptide gets transformed into a two-dimensional feature matrix of size (30,200) before being fed into the model (the first element of this tuple denotes the size of the numerical string that is obtained after the preprocessing stage, as discussed in Section 2.1). The reason behind using these embeddings was that they are tailored according to our own dataset.

Dropout and spatial dropout layer

The dropout layers prevent overfitting (a situation when the difference between the performance on the training and test sets differs significantly) by regularising the preceding layers. Regular dropout layers drop random elements (neurons) from the prior layers. In contrast, spatial dropout layers are advanced dropout mechanisms that drop an entire column of the feature matrix. In our case, a one-dimensional spatial dropout layer was applied after the embedding layer, which drops those feature frames (columns in the feature matrix) that are highly similar to each other.

The dropout rate decides the extent of dropout. Hence, it is chosen carefully after observing the model's performance on the validation set. If this rate is too high or too low, it may cause the model to underfit (when the model performs poorly on the training set) or overfit the training data, respectively.

Global average pooling layer

The one-dimensional global average pooling layer in our model calculates the average value of each feature map generated by the TCN block (a feature map is a vector generated on application of a single filter). Likewise, we have *F* (number of filters applied in the constituent layers of a TCN Block) feature maps that are generated by a TCN block. This layer compresses them into *F* units (by finding the average value of each feature map) and prepares the model for final classification.

Overall architecture of TCN-AFPpred model

The model TCN-AFPpred is a deep TCN architecture built using various layers (as discussed earlier), as shown in Figure 3. In the beginning, we have our input layer where the AA residues of a given peptide (in numerical string format) are fed into the model. After that, a feature matrix is generated by the embedding layer. This is followed by a one-dimensional spatial dropout layer, which is used to prevent overfitting. The output of this layer is fed into TCN block-1, which has 128 filters and uses dilations of sizes 1,2 and 4. This block is connected to TCN block-2, which has 64 filters, and the same number and size of dilations as used in the preceding TCN block. After this, we used a 1-D global average pooling layer followed by a dense layer and a dropout layer. The rectified linear unit activation function was used in these layers. The output layer was fed with the result of the dropout layer to classify a given peptide as AFP and non-AFP using the sigmoid activation function. The result given by this output layer is a probability value, and a peptide is classified as an AFP if this value is greater than or equal to 0.5.

Experiments and results

The TCN-AFPpred model has been coded in python language and trained and tested using a compute node

Table 1. Performance of TCN-AFPpred model with and without TL on the primary test set

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	AUROC (%)
TCN-AFPpred (with TL)	94.05	94.08	94.08	94.08	98.06
TCN-AFPpred (without TL)	92.13	92.83	91.39	92.11	96.92

Table 2. Performance on the primary test set

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	AUROC (%)
TCN-AFPpred	94.05	94.08	94.08	94.08	98.06
Deep-AFPpred	92.91	93.06	92.80	92.93	97.34
HAFPD	81.71	80.18	84.49	82.28	81.71
AFPD	79.38	75.61	87.02	80.92	79.34
AMPFUN	78.10	74.88	84.91	76.58	87.72
AMAP	77.46	73.25	86.88	79.48	77.42
iAMPpred	77.04	79.30	73.48	76.28	86.16
RNN-AFPD	71.16	67.85	80.96	73.83	71.11
MLAMP	57.69	76.92	22.57	34.90	57.86
iAMP-CA2L	55.77	59.49	33.62	42.96	55.57
Antifp	53.73	57.52	30.18	39.59	53.81

Table 3. Performance on the secondary test set

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	AUROC (%)
TCN-AFPpred	92.00	92.79	88.54	90.62	95.43
HAFPD	81.25	75.32	84.81	79.78	81.65
AFPD	76.13	67.95	85.67	75.79	77.20
AMPFUN	77.63	70.53	83.67	76.54	85.70
AMAP	75.75	67.03	87.39	75.87	77.06
RNN-AFPD	71.13	63.11	81.38	71.09	72.28
iAMP-CA2L	59.39	56.39	34.10	42.50	56.85
Antifp	56.63	50.53	27.50	35.62	53.33

equipped with 2.4 GHz Intel-Xeon Skylake 6148 CPU cores with 192 GB RAM and NVIDIA V100 GPU cores with 16 GB RAM. To implement our model, we used Keras library with Tensorflow [65] as backend. The model was evaluated and compared with various state-of-the-art models such as MLAMP [38], iAMPpred [36], AMPFUN [40], AMAP [37], Antifp [33], Deep-AFPpred [42], iAMP-CA2L [43], AFPD, RNN-AFPD and HAFPD [39] on the primary and secondary test sets. We also did rigorous statistical analysis using the KWH and Tukey HSD test to compare its performance with other models.

Performance metrics

Our model was evaluated on various performance metrics such as accuracy, precision, recall, f1-score and area under the receiver operating characteristic curve (AUROC), explained as follows.

1) **Accuracy**: For a given dataset, accuracy is the ratio of correctly classified data points to the total number of data points.
2) **Precision**: It is the ratio of correctly classified AFPs to the total number of data points that were classified as AFPs.
3) **Recall**: It is the ratio of correctly classified AFPs to the actual number of AFPs present in the dataset.
4) **F1-score**: F1-score is a combination (harmonic mean) of precision and recall.
5) **AUROC**: An ROC curve is a plot that shows the relationship between recall and false positive rate (ratio of incorrectly classified non-AFPs to the total number of non-AFPs present in the dataset).

Performance evaluation

The TCN-AFPpred model was trained and tuned extensively using the training and the validation sets (both primary and secondary datasets were used for training the model), respectively. Then, primary and secondary test sets were used to demonstrate the performance of the model after being trained on primary and secondary training sets, respectively. This gave an unbiased estimate of the model's performance and its ability to generalize the data points on which it had not been trained. Firstly, we observed the difference in the performance of our model on the primary test set, with and without employing TL, using various performance metrics like accuracy, recall, precision, f1-score and

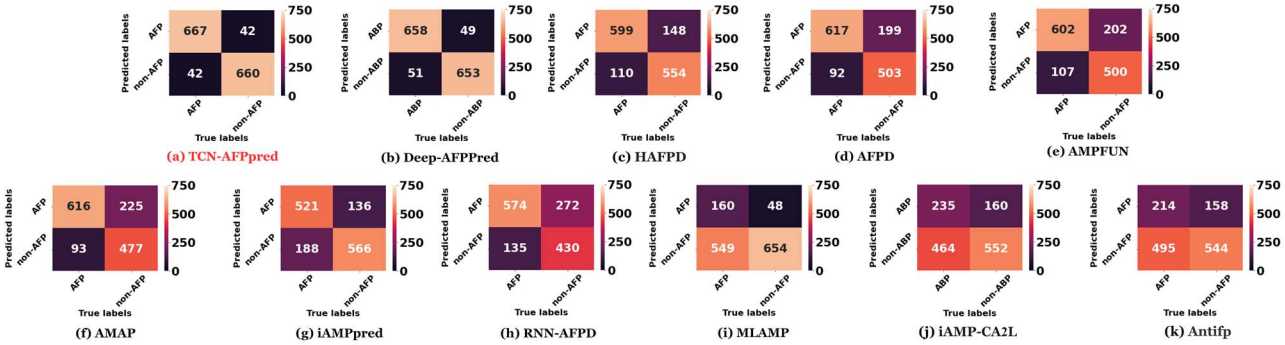

Figure 4. Confusion matrices obtained after running the models on the primary test set.

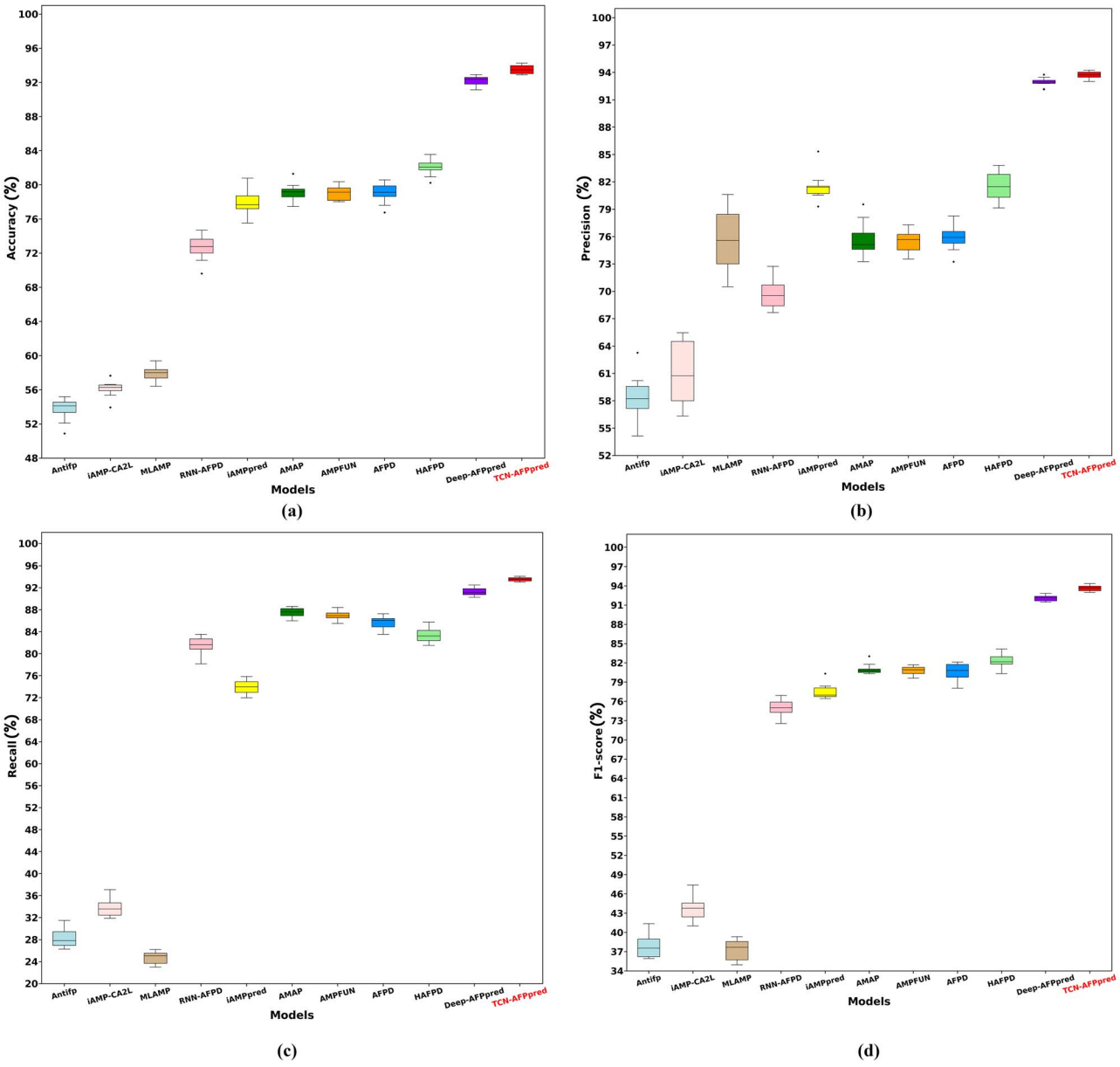

Figure 5. Box and whisker plots of different models for (A) Accuracy, (B) Precision, (C) Recall and (d) F1-score.

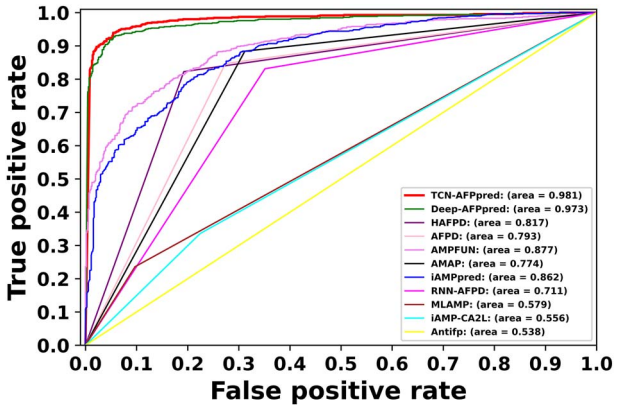

Figure 6. ROC curves of different models on the primary test set.

AUROC. As is evident from Table 1, we found that use of TL led to better performance of our model. Therefore, we selected the model that was built using TL (TCN-AFPpred (with TL), which will be henceforth known as TCN-AFPpred) and compared its performance with various state-of-the-art models like MLAMP, iAMPpred, AMPFUN, AMAP, Antifp, Deep-AFPpred, iAMP-CA2L, AFPD, RNN-AFPD and HAFPD and analyzed their results. The results on the primary test set have been given in Table 2, which shows that TCN-AFPpred performed better than all these models. We also compared the performance of our model with other models (AMPFUN, AMAP, Antifp, iAMP-CA2L, AFPD, RNN-AFPD and HAFPD) using the secondary test set, wherein the similar sequences between training and test sets have been removed using the CD-HIT tool to make a fair and unbiased comparison. We observed that our model outperformed others on the secondary test set as well. The results have been presented in Table 3. A detailed analysis of the results has been done in Sections 4.3 and 4.4.

Figurative analysis of the performance

This section compares our model's performance with other models on the test set using confusion matrices, box and whisker plots and ROC curves. Confusion matrices are essential tools that show the actual classification capability of a model. Figures 4 and 7 show the confusion matrices that resulted after running various models on the primary and secondary test sets, respectively. It is evident that TCN-AFPpred shows superior performance as the number of true positives (correctly classified AFPs) and true negatives (correctly classified non-AFPs) are higher than any other model. Also, the number of false positives (misclassified non-AFPs) and false negatives (misclassified AFPs) in the case of our model is very less than others. This shows that our model is more accurate and precise than others.

The value of the AUROC shows a classifier's capability to distinguish between the classes under consideration (AFPs and non-AFPs). A high value of AUROC implies that a given classifier can differentiate efficiently between the classes. In Figures 6 and 8, we have given the ROC curves for various classifiers on the primary and secondary test sets, respectively. The AUROC curve of TCN-AFPpred is much larger than any other model. This indicates that it is more efficient in discriminating between AFPs and non-AFPs than others. Moreover, it was observed that despite a slight drop in performance of TCN-AFPpred when trained using the secondary dataset, it outperforms all other models by a significant margin. Hence, this proves that the underlying model does not suffer from the loss of generalizability when trained and tested on non-similar sequences.

In case of primary test set, the performance of all the models has been illustrated using box and whisker plots for the values of accuracy, precision, recall and f1-score in Figure 5. This plot consists of five components—median (the middle value), the lower quartile or Q1 (the median of values lying to the left of the median), the upper quartile or Q3 (the median of values lying to the right of the median), the maximum value and the minimum value (both denoted by upper and lower caps of the whiskers, respectively). The maximum and minimum values are calculated using the inter-quartile range (IQR) (Q3–Q1). The points beyond the maximum and minimum values are called outliers. The outliers (represented by the isolated circles or fliers that lie beyond the whiskers) denote the exceptions to the general performance. In other words, their presence implies that the model might show erratic performance(s). For constructing these plots, we ran TCN-AFPpred as well as other models on 10 independently sampled test sets (discussed later) and obtained their results on the aforementioned metrics. The obtained box plots show that results given by TCN-AFPpred are better than others because of the following reasons.

1) The median performance of TCN-AFPpred is greater than other models, which shows that it is better than them.
2) The IQR for TCN-AFPpred is less and the whiskers are short, which suggests that it gives more or less similar performance on all the test sets. In other words, the performance values are close to the median, which implies that the performance of TCN-AFPpred is consistent and reliable.
3) We can easily note the presence of outliers in case of most of the models for one or the other performance metric. The absence of outliers for every metric in the case of TCN-AFPpred shows that its performance is non-erratic.

All the aforementioned figures suggest that our model outperforms others in terms of accuracy, recall, precision, F1-score and AUROC. In Section 4.4, we present a statistical proof to this claim.

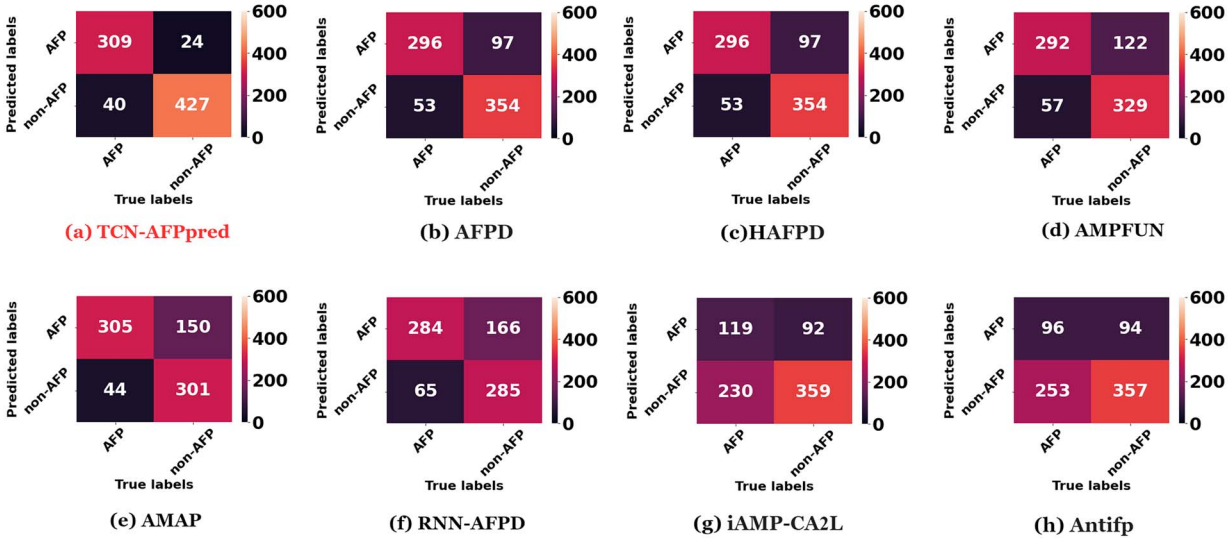

Figure 7. Confusion matrices obtained after testing the models on the secondary test set.

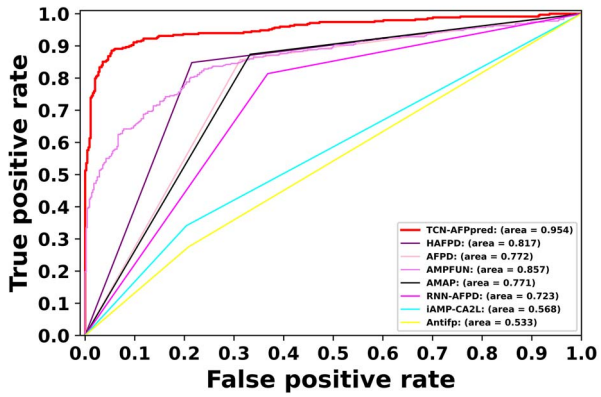

Figure 8. ROC curves of different models on the secondary test set.

Statistical analysis of the performance

We performed a thorough statistical analysis by forming ten independently and randomly sampled pairs of training and test sets (using the primary dataset). Then we trained and tested our model on each pair. Note that the tuning was performed only using the first sample. After that, the hyper-parameters were re-used for the rest. We also ran each state-of-the-art model (as mentioned earlier) on these test sets.

Kruskal–Wallis H test

After obtaining the results for all the models, we performed KWH [48] on the accuracy, precision and AUROC. This test was used to examine whether the difference in the medians of performance of different groups (models) is statistically significant or not. The proposed null and the alternative hypothesis for accuracy are given as follows.

H0: Median accuracy for all the models are equal. (3)

H1: Median accuracy for all the models are not equal. (4)

Table 4. KWH test results

Metric	*p*-value
Accuracy	1.29e-17
Precision	2.90e-17
Recall	2.97e-18
F1-score	1.04e-17
AUROC	2.37e-18

Similarly, we propose null and alternative hypothesis for precision, recall, f1-score and AUROC as well. The KWH test is a non-parametric alternative of the analysis of variance (ANOVA) [66] test. The main advantage of the KWH test over ANOVA is that it does not assume that the data values (performance values) of a particular group (model) follow a normal distribution. It is also known as one-way ANOVA on ranks, as it works on ranks instead of the actual data values themselves. We calculate the H-statistic and the corresponding *p*-value for these ranks. We reject the null hypothesis if the *p*-value is less than a certain cut-off point, which is also known as the alpha or significance level (here, 0.05).

On running this test, we found the *p*-values for precision, accuracy, recall, f1-score and AUROC as shown in Table 4. Since the *p*-values are less than 0.05 in case of all three metrics, we have sufficient evidence to reject the null hypothesis and conclude that the difference in the means of performance of given models is statistically significant. However, the KWH test does not inform us which model's performance is better than others.

To establish that the performance of TCN-AFPpred is significantly different and better than others, we did a post hoc analysis on the results.

Tukey HSD test

To find out which model's performance is the best, we used Tukey HSD test [49]. It compares all the models

Table 5. Tukey HSD test on accuracy (%) of various models

(a) Input summary

Group	Count	Sum	Average
TCN-AFPpred	10	932.59	93.26
Deep-AFPpred	10	922.31	92.23
HAFPD	10	820.19	82.02
AFPD	10	790.29	79.03
AMPFUN	10	790.19	79.02
AMAP	10	791.72	79.17
iAMPpred	10	779.83	77.98
RNN-AFPD	10	726.22	72.62
MLAMP	10	578.58	57.86
iAMP-CA2L	10	562.07	56.21
Antifp	10	537.07	53.71

(b) Post hoc analysis

Model compared with TCN-AFPpred	Difference in means	*p*-adjusted	Lower-bound	Upper-Bound
Deep-AFPpred	1.028	0.043	0.0236	2.112
HAFPD	11.240	0.001	9.631	12.849
AFPD	14.230	0.001	12.621	15.839
AMPFUN	14.240	0.001	12.631	15.849
AMAP	14.087	0.001	12.478	15.696
iAMPpred	15.276	0.001	13.668	16.885
RNN-AFPD	20.637	0.001	19.028	22.246
MLAMP	35.401	0.001	33.792	37.009
iAMP-CA2L	37.052	0.001	35.443	38.661
Antifp	39.552	0.001	37.943	41.161

Table 6. Tukey HSD test on recall (%) of various models

(a) Input summary

Group	Count	Sum	Average
TCN-AFPpred	10	931.49	93.15
Deep-AFPpred	10	912.72	91.27
HAFPD	10	833.19	83.32
AFPD	10	855.83	85.58
AMPFUN	10	868.89	86.89
AMAP	10	875.25	87.52
iAMPpred	10	739.67	73.97
RNN-AFPD	10	813.25	81.32
MLAMP	10	246.79	24.68
iAMP-CA2L	10	339.28	33.93
Anti-fp	10	282.78	28.28

(b) Post hoc analysis

Model compared with TCN-AFPpred	Difference in means	*p*-adjusted	Lower-bound	Upper-Bound
Deep-AFPpred	1.877	0.011	0.285	3.469
HAFPD	9.830	0.001	7.906	11.276
AFPD	7.566	0.001	5.642	9.490
AMPFUN	6.260	0.001	4.336	8.184
AMAP	5.624	0.001	3.699	7.548
iAMPpred	19.182	0.001	17.258	21.106
RNN-AFPD	11.824	0.001	9.899	13.748
MLAMP	68.470	0.001	66.546	70.394
iAMP-CA2L	59.221	0.001	61.145	57.296
Anti-fp	64.871	0.001	62.947	66.795

Table 7. Tukey HSD test on f1-score (%) of various models

(a) Input summary

Group	Count	Sum	Average
TCN-AFPpred	10	933.24	93.32
Deep-AFPpred	10	921.17	92.12
HAFPD	10	823.22	82.32
AFPD	10	805.98	80.59
AMPFUN	10	808.16	80.82
AMAP	10	810.26	81.03
iAMPpred	10	775.26	77.53
RNN-AFPD	10	749.94	74.99
MLAMP	10	372.36	37.24
iAMP-CA2L	10	439.52	43.95
Antifp	10	379.39	37.94

(b) Post hoc analysis

Model compared with TCN-AFPpred	Difference in means	*p*-adjusted	Lower-bound	Upper-Bound
Deep-AFPpred	1.207	0.041	0.029	2.386
HAFPD	11.002	0.001	9.087	12.917
AFPD	12.726	0.001	10.811	14.641
AMPFUN	12.508	0.001	10.593	14.423
AMAP	12.298	0.001	10.383	14.213
iAMPpred	15.798	0.001	13.883	17.713
RNN-AFPD	18.330	0.001	16.415	20.245
MLAMP	56.088	0.001	54.173	58.003
iAMP-CA2L	49.372	0.001	47.457	51.287
Antifp	55.385	0.001	53.470	57.299

Table 8. Tukey HSD test on AUROC (%) of various models

(a) Input summary

Group	Count	Sum	Average
TCN-AFPpred	10	974.42	97.44
Deep-AFPpred	10	964.51	96.45
HAFPD	10	818.76	81.88
AFPD	10	789.63	78.96
AMPFUN	10	889.97	89.00
AMAP	10	789.20	78.92
iAMPpred	10	875.35	87.54
RNN-AFPD	10	725.49	72.55
MLAMP	10	582.02	58.20
iAMP-CA2L	10	565.02	56.50
Antifp	10	539.20	53.92

(b) Post hoc analysis

Model compared with TCN-AFPpred	Difference in means	*p*-adjusted	Lower-bound	Upper-Bound
Deep-AFPpred	0.991	0.044	0.018	1.984
HAFPD	15.566	0.001	14.066	17.066
AFPD	18.479	0.001	16.979	19.979
AMPFUN	8.445	0.001	6.945	9.945
AMAP	18.522	0.001	17.022	20.022
iAMPpred	9.907	0.001	8.407	11.407
RNN-AFPD	24.893	0.001	23.393	26.393
MLAMP	39.240	0.001	37.734	40.740
iAMP-CA2L	40.94	0.001	39.432	42.440
Antifp	43.522	0.001	42.022	45.022

Table 9. Tukey HSD test on precision (%) of various models

(a) Input summary

Group	Count	Sum	Average
TCN-AFPpred	10	936.64	93.66
Deep-AFPpred	10	931.23	93.12
HAFPD	10	814.99	81.50
AFPD	10	758.93	75.89
AMPFUN	10	754.61	75.46
AMAP	10	757.05	75.71
iAMPpred	10	814.81	81.48
RNN-AFPD	10	696.55	69.66
MLAMP	10	756.48	75.65
iAMP-CA2L	10	609.36	60.94
Antifp	10	583.84	58.38

(b) Post hoc analysis

Model compared with TCN-AFPpred	Difference in means	*p*-adjusted	Lower-bound	Upper-Bound
Deep-AFPpred	0.541	0.900	-2.271	3.743
HAFPD	12.165	0.001	9.158	15.172
AFPD	17.771	0.001	14.764	20.778
AMPFUN	18.203	0.001	15.196	21.210
AMAP	17.959	0.001	14.952	20.966
iAMPpred	12.183	0.001	9.176	15.190
RNN-AFPD	24.009	0.001	21.002	27.016
MLAMP	18.016	0.001	15.009	21.023
iAMP-CA2L	32.728	0.001	29.721	35.735
Antifp	35.280	0.001	32.273	38.287

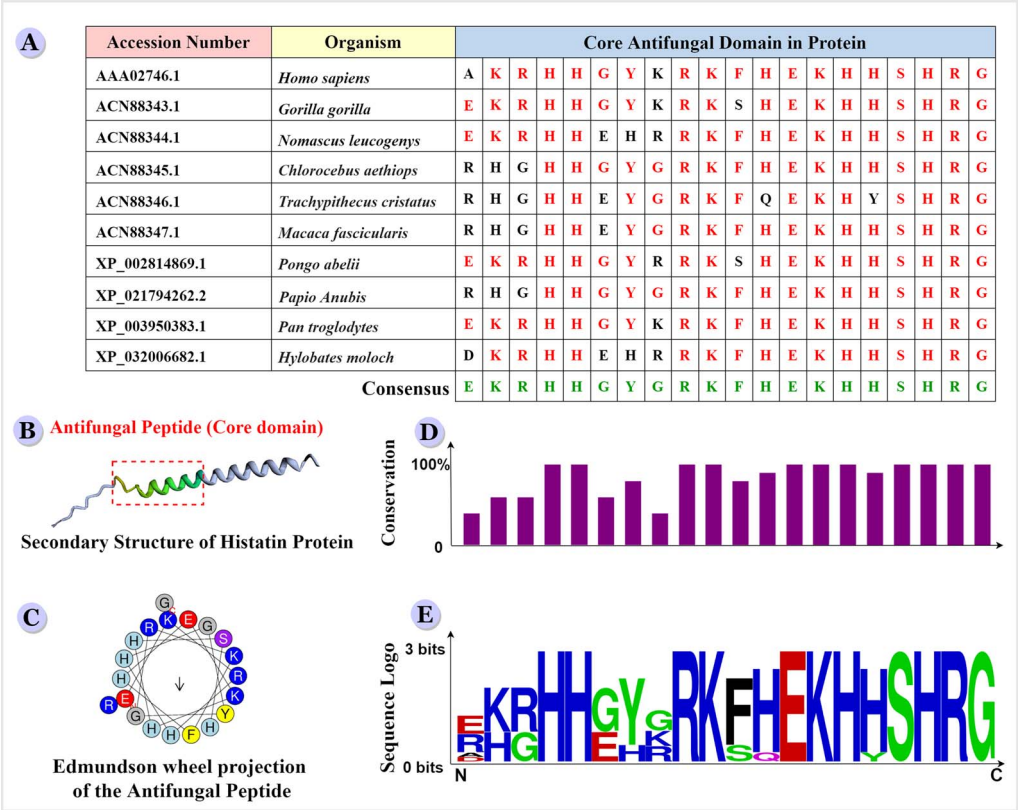

Figure 9. Identification of novel AFPs in the Histatin protein of different animals: (**A**) The alignment of AFPs predicted in ten Histatin proteins obtained from different animals and the consensus sequence among these AFPs. (**B**) The secondary structure of the Histatin protein, wherein the core domain (AFP) is highlighted (generated using a web tool available at https://robetta.bakerlab.org/ [67]). (**C**) The Edmundson Wheel projection of the AFPs, which shows the alpha helical projection of the consensus sequence (generated using a free web tool available at https://heliquest.ipmc.cnrs.fr/ [68]). (**D**) Bar chart showing the conservation percentage of different AAs at each position. (**E**) The sequence logo plot depicting the distribution of different AAs at each position (generated using the web tool available at https://weblogo.berkeley.edu/logo.cgi [69].

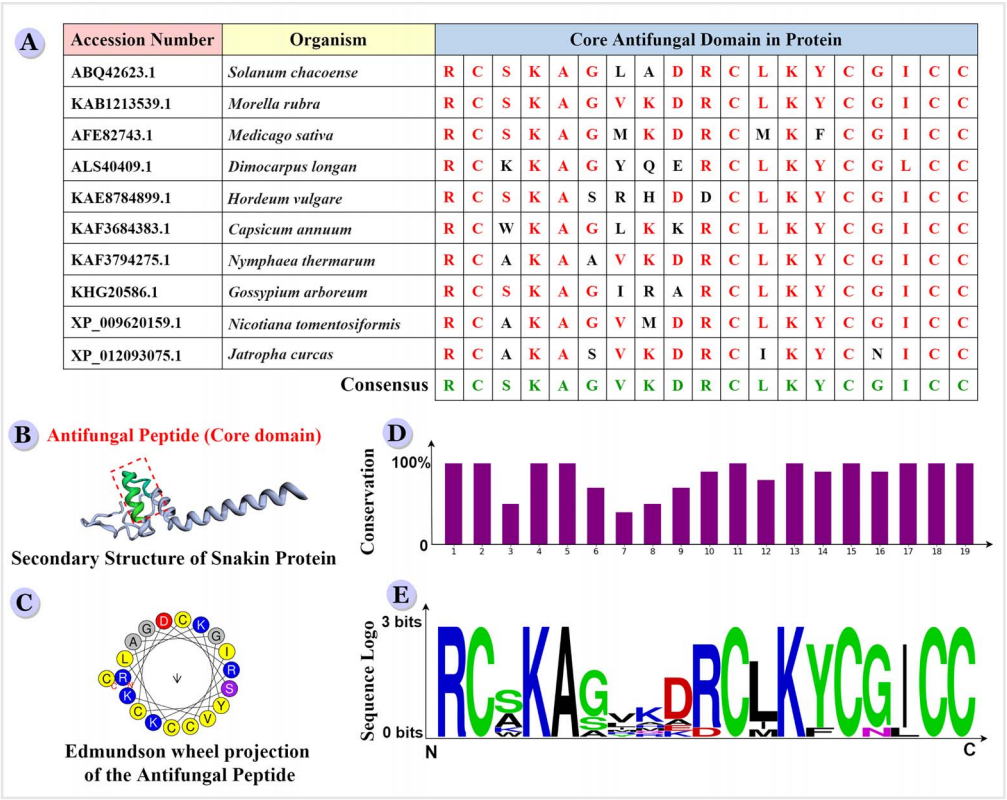

Figure 10. Identification of novel AFPs in the Snakin protein of different plants: (**A**) The alignment of AFPs predicted in ten Snakin proteins obtained from different plants and the consensus sequence among these AFPs. (**B**) The secondary structure of the Snakin protein, wherein the core domain (AFP) is highlighted (generated using a web tool available at https://robetta.bakerlab.org/ [67]). (**C**) The Edmundson Wheel projection of the AFPs, which shows the alpha helical projection of the consensus sequence (generated using a free web tool available at https://heliquest.ipmc.cnrs.fr/ [68]). (**D**) Bar chart showing the conservation percentage of different AAs at each position. (**E**) The sequence logo plot depicting the distribution of different AAs at each position (generated using the web tool available at https://weblogo.berkeley.edu/logo.cgi [69]).

in pairs and finds out whether the difference in means of their performance is statistically significant or not. The output of this test gives the difference in means, the adjusted *p*-value and the lower and upper bounds of the confidence interval (CI) for all possible pairs of models under consideration. However, we have shown only those results in which our model was a part of the pair. For a given pair of models, the difference in means is significant, only if the adjusted *p*-value is less than alpha level (0.05) and CI (here we considered 95% CI), does not contains zero. We performed Tukey HSD test for accuracy, precision, recall, f1-score, and AUROC to check whether TCN-AFPpred's performance is better than others. The results given in Tables 5–9 show that the difference in means of TCN-AFPpred and other models' performance is statistically significant for accuracy, recall, f1-score and AUROC. This is due to the fact that the adjusted *p*-values are less than 0.05, and none of the CIs contain zero. Furthermore, because our model's mean performance is higher than that of other models, we can conclude that our model outperforms them with respect to these metrics.

Identification of AFPs using the web app

We built and deployed a freely accessible web app using the TCN-AFPpred model at https://tcn-afppred.anvil.app/ for classification and identification of AFPs. It takes a protein sequence as an input, checks the nature of all possible peptides that may be derived from it (i.e. having a length in the interval [4, 30]) and assigns a probability score to them. An assigned probability score is directly proportional to the classifier's confidence in the antifungal potential of a given peptide. Further, we used this app to identify AFPs in the Snakin and Histatin protein sequences. These are well-known antifungal proteins found in multiple organisms across the plant and animal kingdom. We selected ten Histatin proteins (belonging to different animals) and fed them as input to our model. The model identified several AFPs with a probability score $\geq$ 0.90 and out of these AFPs, one representative AFP with probability score $\geq$ 0.99 was selected for each protein sequence. This set of 10 AFPs was aligned to identify the core antifungal domain as shown in Figure 9. After this, we identified a consensus sequence among AFPs belonging to Histatin protein of different animals. A similar analysis was performed on the Snakin proteins found in various plants, and the results are shown in Figure 10. Although the peptides identified in Histatins and Snakins of different animals and plants, respectively, are predicted as antifungal, the consensus sequence thus obtained among these AFPs is purported to possess potentially better antifungal

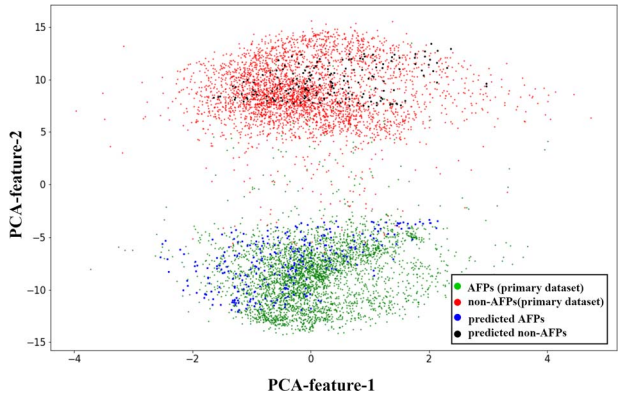

Figure 11. Scatter plot showing the distribution of AFPs and non-AFPs predicted in Histatin protein along with the AFPs and non-AFPs of the primary dataset.

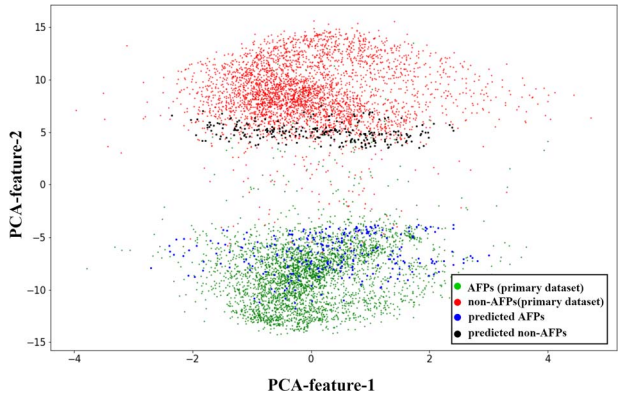

Figure 12. Scatter plot showing the distribution of AFPs and non-AFPs predicted in Snakin protein along with the AFPs and non-AFPs of the primary dataset.

activity. This sequence can be chemically synthesized and experimentally validated for its activity against different fungal strains.

Furthermore, we visualized the distribution of AFPs and non-AFPs (in two dimensions) present in the primary dataset, along with the AFPs (with probability score $\geq$ 0.90) and non-AFPs (with probability score $\leq$ 0.10) predicted by TCN-AFPpred (in the Snakin and Histatin proteins) by reducing the dimensionality of the feature space using principal component analysis. This technique is used to visualize high dimensional data by reducing an *n*-dimensional feature space to two or three dimensions in order to get a clear picture of the distribution of data points. As illustrated in Figures 11 and 12, it is clear that in the case of both the proteins (Histatin and Snakin), the predicted AFPs lie within the distribution of known AFPs, and similarly, the predicted non-AFPs lie within the distribution of known non-AFPs. This implies that the AFPs for which our classifier assigns high scores follow the same distribution as the AFPs of the primary dataset. Similar inference can be drawn in the case of predicted non-AFPs.

Conclusion

We proposed a TCN-based predictor for identifying peptides that exhibit antifungal activity. As input, we used the AA sequences of peptides to perform AFP/non-AFP classification. Since our primary dataset was small, we pre-trained our model on a base dataset consisting of ABPs. The model built after this initial round of training was referred to as the base model. The base model was utilized to boost the training of the primary model. This trained primary model was termed as TCN-AFPpred. We tested our model's efficacy on the primary test set using various metrics like accuracy, precision, recall, f1-score and AUROC, which came out to be 94%, 94%, 94%, 94% and 98%, respectively. Thus, our model was found to be very precise and accurate in predicting AFPs. On comparing TCN-AFPpred with the state-of-the-art models on primary and secondary test sets, we found that it outperforms them. We also performed a thorough statistical analysis to check the statistical difference of our model's results with respect to others.

Moreover, we built and deployed a web app called TCN-AFPpred (https://tcn-afppred.anvil.app/), which identifies AFPs in protein sequences of various organisms and classifies peptides as AFPs and non-AFPs. In this paper, we used this app to predict AFPs in plant (Snakin) and animal (Histatin) antifungal proteins. We also identified the core antifungal domain in the Snakin and Histatin proteins. The antifungal potential of these predicted AFPs can be verified after chemical synthesis and testing their activity against a range of fungi that infect plants and animals.

In this work, we have only used ABPs to pre-train our model. In the future, we would like to analyse the effect of pre-training the base model using antibacterial, antiviral as well as antiparasitic peptides. Also, we would consider using pre-trained word embeddings in our model and observe the difference in its final performance. Apart from this, a multi-label classifier may be developed to predict AFPs on other important parameters like haemotoxicity and cytotoxicity. Yet another multi-label classifier may also be built to predict AFPs, which show considerable activity against fungal biofilms.

Key Points

- We proposed TCN-AFPpred, a temporal convolutional network-based deep learning model using techniques like transfer learning to identify AFPs in proteins of various organisms.
- We compared the performance of our model with other state-of-the-art classifiers and statistically proved that our model outperforms others in terms of precision, accuracy, AUROC, etc.
- We analyzed animal (Histatin) and plant (Snakin) antifungal proteins using TCN-AFPpred and identified their consensus antifungal domains.

- We built a user-friendly web app that is freely accessible at https://tcn-afppred.anvil.app/, wherein users can provide a protein sequence as input and obtain a list of predicted AFPs as output.

Acknowledgments

The support and resources provided by the ICAR-National Agricultural Science Fund (NASF) at ICAR-Indian Veterinary Research Institute, Izatnagar and PARAM Shivay facility under the National Supercomputing Mission, Government of India at Indian Institute of Technology (IIT(BHU)), Varanasi, are thankfully acknowledged.

Data availability

All the datasets used in this paper are available at https://tcn-afppred.anvil.app/.

References

1. Bongomin F, Gago S, Oladele RO, *et al.* Global and multi-national prevalence of fungal diseases-estimate precision. *J Fungi* 2017;**3**(4):57.
2. Kaushik N, Pujalte G, Reese ST. Superficial fungal infections. *Prim Care* 2015;**42**(4):501–16.
3. Rautemaa-Richardson R, Richardson MD. Systemic fungal infections. *Medicine* 2017;**45**(12):757–62.
4. Fernández de Ullivarri M, Arbulu S, Garcia-Gutierrez E, *et al.* Antifungal peptides as therapeutic agents. *Front Cell Infect Microbiol* 2020;**10**:105.
5. Fisher MC, Gurr SJ, Cuomo CA, *et al.* Threats posed by the fungal kingdom to humans, wildlife, and agriculture. *MBio* 2020;**11**(3):e00449–20.
6. Tao H, Bao Z, Jin C, *et al.* Toxic effects and mechanisms of three commonly used fungicides on the human colon adenocarcinoma cell line Caco-2. *Environ Pollut* 2020;**263**:114660.
7. Sanglard D. Finding the needle in a haystack: mapping antifungal drug resistance in fungal pathogen by genomic approaches. *PLoS Pathog* 2019;**15**(1):e1007478.
8. Seyedmousavi S, SdM B, De Hoog S, *et al.* Fungal infections in animals: a patchwork of different situations. *Med Mycol* 2018;**56**(suppl_1):S165–87.
9. Weisskopf L. The potential of bacterial volatiles for crop protection against phytophathogenic fungi. *Microbial pathogens and strategies for combating them: science, technology and education* 2013;**2**:1352–63.
10. De Lucca AJ, Walsh TJ. Antifungal peptides: novel therapeutic compounds against emerging pathogens. *Antimicrob Agents Chemother* 1999;**43**(1):1–11.
11. Oshiro KG, Rodrigues G, Monges BED, *et al.* Bioactive peptides against fungal biofilms. *Front Microbiol* 2019;**10**:2169.
12. Wang J, Yin T, Xiao X, *et al.* StraPep: a structure database of bioactive peptides. *Database* 2018;**2018**:1–7.
13. Aguilera-Mendoza L, Marrero-Ponce Y, Tellez-Ibarra R, *et al.* Overlap and diversity in antimicrobial peptide databases: compiling a non-redundant set of sequences. *Bioinformatics* 2015;**31**(15):2553–9.
14. Lu T, Zhang Z, Zhu J, *et al.* Deep learning-based prediction of the T cell receptor–antigen binding specificity. *Nat Mach Intell* 2021;**3**(10):864–75.
15. Cheng S, Liu S, Yu J, *et al.* Robust whole slide image analysis for cervical cancer screening using deep learning. *Nat Commun* 2021;**12**(1):1–10.
16. Kumar A, Singh SK, Saxena S, *et al.* Deep feature learning for histopathological image classification of canine mammary tumors and human breast cancer. *Inform Sci* 2020;**508**:405–21.
17. Kumar A, Singh SK, Saxena S, *et al.* CoMHisP: A novel feature extractor for histopathological image classification based on fuzzy SVM with within-class relative density. *IEEE Trans Fuzzy Syst* 2021;**29**(1):103–17.
18. Kumar A, Singh SK, Lakshmanan K, *et al.* A novel cloud-assisted secure deep feature classification framework for cancer histopathology images. *ACM Trans Internet Technol* 2021;**21**(2):1–22.
19. Kumar A, Sharma A, Bharti V, *et al.* MobiHisNet: a lightweight CNN in mobile edge computing for histopathological image classification. *IEEE Internet Things J* 2021;**1–1**. https://doi.org/10.1109/JIOT.2021.3119520.
20. Singh R, Bharti V, Purohit V, *et al.* MetaMed: few-shot medical image classification using gradient-based meta-learning. *Pattern Recognit* 2021;**120**:1–13.
21. Bharti V, Biswas B, Shukla KK. A novel multiobjective gdwcn-pso algorithm and its application to medical data security. *ACM Trans Internet Technol* 2021;**21**(2):1–28.
22. Kavousi K, Bagheri M, Behrouzi S, *et al.* IAMPE: NMR-assisted computational prediction of antimicrobial peptides. *J Chem Inf Model* 2020;**60**(10):4691–701.
23. Li C, Sutherland D, Hammond SA, *et al.* AMPlify: attentive deep learning model for discovery of novel antimicrobial peptides effective against WHO priority pathogens bioRxiv. 2020.
24. Singh V, Shrivastava S, Kumar Singh S, *et al.* StaBle-ABPpred: a stacked ensemble predictor based on biLSTM and attention mechanism for accelerated discovery of antibacterial peptides. *Brief Bioinform* 2021. https://doi.org/10.1093/bib/bbab439.
25. Sharma R, Shrivastava S, Kumar Singh S, *et al.* Deep-ABPpred: identifying antibacterial peptides in protein sequences using bidirectional LSTM with word2vec. *Brief Bioinform* 2021. https://doi.org/10.1093/bib/bbab065.
26. Burdukiewicz M, Sidorczuk K, Rafacz D, *et al.* Proteomic screening for prediction and design of antimicrobial peptides with AmpGram. *Int J Mol Sci* 2020;**21**(12):4310.
27. Sharma R, Shrivastava S, Singh SK, *et al.* Deep-AVPpred: Artificial intelligence driven discovery of peptide drugs for viral infections. *IEEE J Biomed Health Inform* 2021;1–1. https://doi.org/10.1109/JBHI.2021.3130825.
28. Ahmad A, Akbar S, Khan S, *et al.* Deep-AntiFP: prediction of antifungal peptides using distanct multi-informative features incorporating with deep neural networks. *Chemom Intel Lab Syst* 2021;**208**:104214.
29. Yan J, Bhadra P, Li A, *et al.* Deep-AmPEP30: improve short antimicrobial peptides prediction with deep learning. *Mol Ther Nucleic Acids* 2020;**20**:882–94.
30. Veltri D, Kamath U, Shehu A. Deep learning improves antimicrobial peptide recognition. *Bioinformatics* 2018;**34**(16):2740–7.
31. Sharma R, Shrivastava S, Kumar Singh S, *et al.* AniAMPpred: artificial intelligence guided discovery of novel antimicrobial peptides in animal kingdom. *Brief Bioinform* 2021. https://doi.org/10.1093/bib/bbab242.
32. Joseph S, Karnik S, Nilawe P, *et al.* ClassAMP: a prediction tool for classification of antimicrobial peptides. *IEEE/ACM*

Trans Comput Biol Bioinform 2012;**9**(5):1535–8. https://doi.org/10.1109/TCBB.2012.89.
33. Agrawal P, Bhalla S, Chaudhary K, *et al.* In silico approach for prediction of antifungal peptides. *Front Microbiol* 2018;**9**:323.
34. Mousavizadegan M, Mohabatkar H. An evaluation on different machine learning algorithms for classification and prediction of antifungal peptides. *Med Chem* 2016;**12**(8):795–800.
35. Tyagi A, Roy S, Singh S, *et al.* PhytoAFP: in silico approaches for designing plant-derived antifungal peptides. *Antibiotics* 2021;**10**(7):815.
36. Meher PK, Sahu TK, Saini V, *et al.* Predicting antimicrobial peptides with improved accuracy by incorporating the compositional, physico-chemical and structural features into Chou's general PseAAC. *Sci Rep* 2017;**7**(1):1–12.
37. Gull S, Shamim N, Minhas F. AMAP: Hierarchical multi-label prediction of biologically active and antimicrobial peptides. *Comput Biol Med* 2019;**107**:172–81.
38. Lin W, Xu D. Imbalanced multi-label learning for identifying antimicrobial peptides and their functional types. *Bioinformatics* 2016;**32**(24):3745–52.
39. Pinacho-Castellanos SA, García-Jacas CR, Gilson MK, *et al.* Alignment-free antimicrobial peptide predictors: improving performance by a thorough analysis of the largest available data set. *J Chem Inf Model* 2021;**61**(6):3141–57.
40. Chung CR, Kuo TR, Wu LC, *et al.* Characterization and identification of antimicrobial peptides with different functional activities. *Brief Bioinform* 2020;**21**(3):1098–114.
41. Xiao X, Wang P, Lin WZ, *et al.* iAMP-2L: a two-level multi-label classifier for identifying antimicrobial peptides and their functional types. *Anal Biochem* 2013;**436**(2):168–77.
42. Sharma R, Shrivastava S, Kumar Singh S, *et al.* Deep-AFPpred: identifying novel antifungal peptides using pretrained embeddings from seq2vec with 1DCNN-BiLSTM. *Brief Bioinform* 2021. https://doi.org/10.1093/bib/bbab422.
43. Xiao X, Shao YT, Cheng X, *et al.* iAMP-CA2L: a new CNN-BiLSTM-SVM classifier based on cellular automata image for identifying antimicrobial peptides and their functional types. *Brief Bioinform* 2021;**22**(6):1–10.
44. Lea C, Vidal R, Reiter A, *et al.* Temporal convolutional networks: a unified approach to action segmentation. In: *European Conference on Computer Vision*. Amsterdam, The Netherlands: Springer, 47–54.
45. Bai S, Kolter JZ, Koltun V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling arXiv preprint arXiv:180301271. 2018.
46. Blotnick E, Sol A, Bachrach G, *et al.* Interactions of histatin-3 and histatin-5 with actin. *BMC Biochem* 2017;**18**(1):1–13.
47. Oliveira-Lima M, Maria Benko-Iseppon A, Ribamar Costa Ferreira Neto J, *et al.* Snakin: structure, roles and applications of a plant antimicrobial peptide. *Curr Protein Peptide Sci* 2017;**18**(4):368–74.
48. Kruskal WH, Wallis WA. Use of ranks in one-criterion variance analysis. *J Am Stat Assoc* 1952;**47**(260):583–621.
49. Tukey JW. Comparing individual means in the analysis of variance. *Biometrics* 1949;**5**(2):99–114.
50. Waghu FH, Gopi L, Barai RS, *et al.* CAMP: Collection of sequences and structures of antimicrobial peptides. *Nucleic Acids Res* 2014;**42**(D1):D1154–8.
51. Nielsen SD, Beverly RL, Qu Y, *et al.* Milk bioactive peptide database: a comprehensive database of milk protein-derived bioactive peptides and novel visualization. *Food Chem* 2017;**232**:673–82.
52. Brahmachary M, Krishnan S, Koh JLY, *et al.* ANTIMIC: a database of antimicrobial sequences. *Nucleic Acids Res* 2004;**32**(suppl_1):D586–9.
53. Kang X, Dong F, Shi C, *et al.* DRAMP 2.0, an updated data repository of antimicrobial peptides. *Scientific data* 2019;**6**(1):1–10.
54. Wang G, Li X, Wang Z. APD3: the antimicrobial peptide database as a tool for research and education. *Nucleic Acids Res* 2016;**44**(D1):D1087–93.
55. Théolier J, Fliss I, Jean J, *et al.* MilkAMP: a comprehensive database of antimicrobial peptides of dairy origin. *Dairy Sci Technol* 2014;**94**(2):181–93.
56. Aguilera-Mendoza L, Marrero-Ponce Y, Beltran JA, *et al.* Graph-based data integration from bioactive peptide databases of pharmaceutical interest: toward an organized collection enabling visual network analysis. *Bioinformatics* 2019;**35**(22):4739–47.
57. Aguilera-Mendoza L, Marrero-Ponce Y, García-Jacas CR, *et al.* Automatic construction of molecular similarity networks for visual graph mining in chemical space of bioactive peptides: an unsupervised learning approach. *Sci Rep* 2020;**10**(1):1–23.
58. Consortium U. UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Res* 2019;**47**(D1):D506–15.
59. Fu L, Niu B, Zhu Z, *et al.* CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics* 2012;**28**(23):3150–2.
60. Li W, Godzik A. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics* 2006;**22**(13):1658–9.
61. Huang Y, Niu B, Gao Y, *et al.* CD-HIT suite: a web server for clustering and comparing biological sequences. *Bioinformatics* 2010;**26**(5):680–2.
62. Pan SJ, Yang Q. A Survey on Transfer Learning. *IEEE Trans Knowl Data Eng* 2010;**22**(10):1345–59. 10.1109/TKDE.2009.191.
63. Lea C, Flynn MD, Vidal R, *et al.* Temporal convolutional networks for action segmentation and detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA: IEEE Computer Society, 156–65.
64. Heinzinger M, Elnaggar A, Wang Y, *et al.* Modeling aspects of the language of life through transfer-learning protein sequences. *BMC Bioinformatics* 2019;**20**(1):1–17.
65. Abadi M, Agarwal A, Barham P, *et al.* Tensorflow: large-scale machine learning on heterogeneous distributed systems arXiv preprint arXiv:160304467. 2016.
66. Singh V, Gupta I, Jana PK. A novel cost-efficient approach for deadline-constrained workflow scheduling by dynamic provisioning of resources. *Fut Gener Comput Syst* 2018;**79**:95–110.
67. Kim DE, Chivian D, Baker D. Protein structure prediction and analysis using the Robetta server. *Nucleic Acids Res* 2004;**32**(suppl_2):W526–31.
68. Gautier R, Douguet D, Antonny B, *et al.* HELIQUEST: a web server to screen sequences with specific α-helical properties. *Bioinformatics* 2008;**24**(18):2101–2.
69. Crooks GE, Hon G, Chandonia JM, *et al.* WebLogo: a sequence logo generator. *Genome Res* 2004;**14**(6):1188–90.