Supplementary Materials for

ACEP: improving recognition of antimicrobial peptide through an attention mechanism, convolutional neural network and embedding tensor

Haoyi Fu [1], Zicheng Cao [2], Mingyuan Li [1] and Shunfang Wang [1,*]

[1]School of Information Science and Engineering, Yunnan University, Kunming 650500, China
[2]School of Public Health (Shenzhen), Sun Yat-sen University, Guangzhou 510006, China.
*To whom correspondence should be addressed.

1 Length distributions of sequences

Sequence length distributions are shown for the training set (top), tuning set (middle), and testing set (bottom) partitions in Figure S1. All the sequences come from a benchmark dataset constructed by Veltri *et al.* (2018) using data from the APD (Wang *et al.*, 2015).

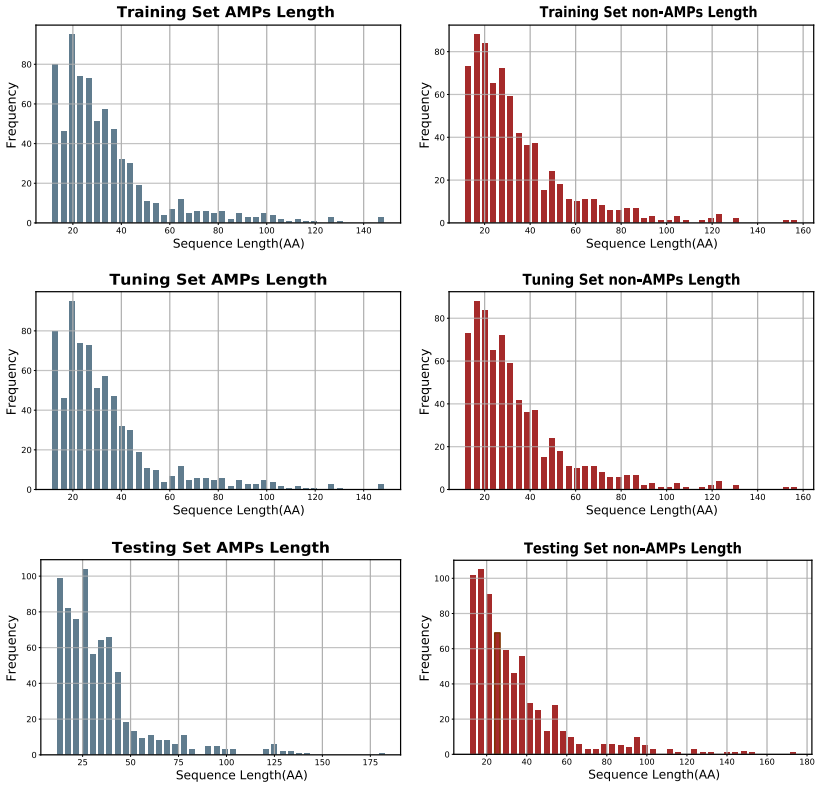

Figure S1: Sequence length distributions of AMPs and non-AMPs

2 Impact of training epochs on accuracy

To assess the stability of the model in the training process, we used the training history data recorded by Keras to construct the curve of accuracy and training epochs. This curve is shown in Figure S2. The red line is the accuracy of the training set, and the green line is the accuracy of the testing set. During the training, the accuracy of training set and testing set increased steadily with the number of training epochs, suggesting our model is steady but is not overfitting.

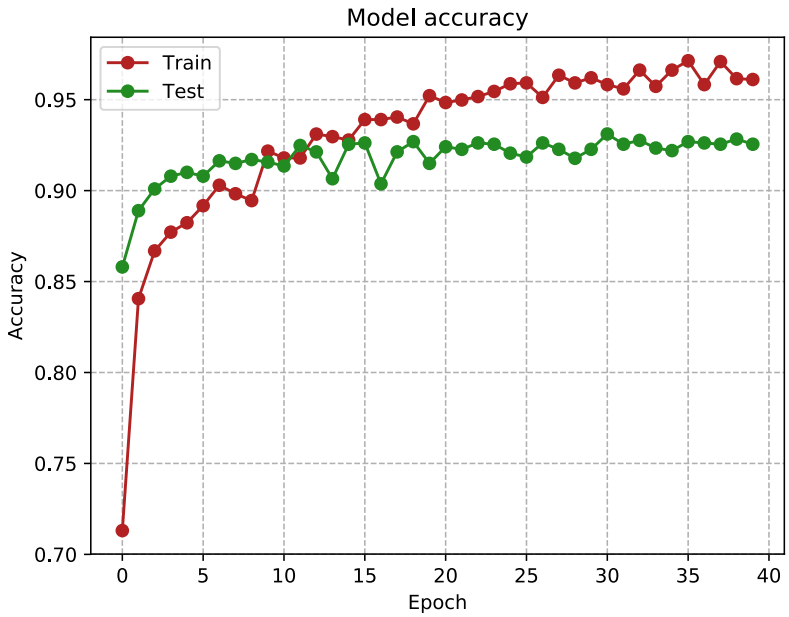

Figure S2: Training history curve

3 Experimental setup and runtime performance

The experiments are conducted on an Intel i7 laptop with an eight core 2.2GHz processor and 8GB of RAM. The deep neural network is built on Keras vr.2.1.5 using a GPU-based TensorFlow vr.1.6.0 backend. Training takes approximately 10 min with the training set, 15 min using all of the data and 3h for 10-fold CV. It takes $<$ 1 minute to run a trained network on a test set.

4 Comparison of functions in different regions

Table 1: Model performance on different training and evaluation data partitions

Region	Sequence length$<$30		Sequence length$\geq$30		All sequences	
	ACC(%)	MCC	ACC(%)	MCC	ACC(%)	MCC
R1	89.74	0.7946	93.11	0.8623	91.29	0.8258
R2	91.03	0.8214	90.06	0.8013	90.58	0.8124
R3	89.09	0.7816	90.36	0.8077	89.67	0.7938
R1+R2	89.61	0.7926	92.50	0.8500	90.94	0.8188
R1+R3	91.03	0.8206	94.18	0.8846	92.48	0.8500
R2+R3	91.42	0.8284	88.37	0.7716	90.02	0.8018
R1+R2+R3	91.16	0.8236	94.34	0.8867	92.62	0.8527

In ACEP architecture regions R1~R3 process sequence evolutionary information, raw information and amino acid compositional information, respectively. We list all the combinations of these regions to compare the impact of each region on the overall performance of the system. In these combinations, when only one region is used, we disabled fusion region R4 at the same time. When more than two regions are used, we integrate the output features of each region through R4. Table 1 shows the performance of the system in each case. In lines 1 to 3, the predicted performance using a single region is shown. Due to containnig the evolutionary information, R1 performs well in the long sequence,and the ACC reaches 93.11%; the performance of predicting the short sequence is poor, where the ACC is only 89.74%. On lines 4 to 6, the predicted performance of fusing two regions is shown. R3 contains the amino acid compositional information, which contributes to the recognition of short sequences; thus R1 + R3 is very effective for long sequence as well as a short sequence, and the overall ACC is 92.18%; R2 + R3 has the worst effect on the long sequence, and the ACC is 88.37% because R2 and R3 contain no evolutionary information.

We note that regions R1~R3 have different effects on the sequences of different lengths. R1 has a better effect on the long sequences, R2 is not sensitive to the length of sequences, and R3 can enhance the recognition performance of R1 on short sequences. R4 can integrates these heterogeneous features through the improved attention mechanism, so that the model can automatically adapt to the sequences of different length.

For the long sequences, ACEP tends to use the features containing evolutionary information; for the short sequences, the ACEP tends to use the features containing amino acid compositional information.

5 Amino acid clustering

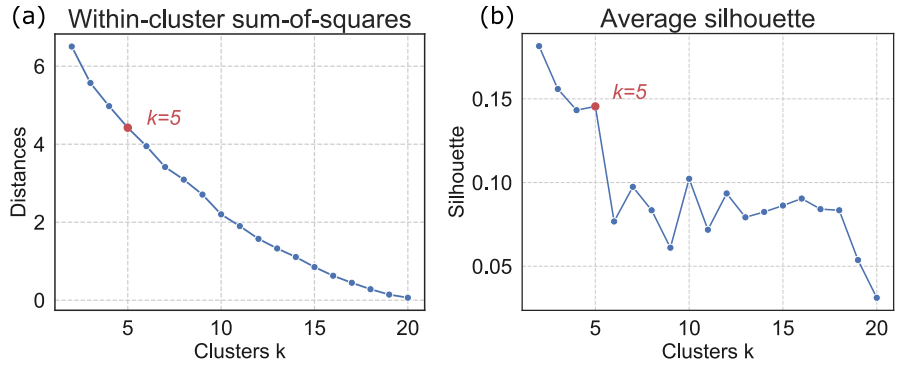

Figure S3: (a) shows the sum of squared distances under different k values, and (b) shows the average silhouette under different k values.

To further analyze how the model works, we extract the $\boldsymbol{E}$ from the model, and cluster the embedding tensor of 20 amino acids in $\boldsymbol{E}$ using the scikit-learn's k-means algorithm (Lloyd *et al.*, 1982). To find the natural number of clusters, we calculate the average silhouette and the sum of squared distances under different k values, as shown in Figure S3. The silhouette of a data instance is a measure of how closely it is matched to data within its cluster and how loosely it is matched to data of the neighboring cluster, i.e., the cluster whose average distance from the datum is lowest (Rousseeuw *et al.*, 1987). The sum of squared distances measures the distance between the sample and the cluster center.

By adjusting different k values, we draw the within-cluster sum-of-squares curve and the silhouette curve to find the k value that corresponds to the largest silhouette value under the premise of the smaller the sum of squared distances, the more likely that this k is the real cluster number. It can be noticed in the figure the value of the silhouette is the largest when $k = 3$, but the sum of squared distances is also very large, approximately 4.5, and the data instances are far from the cluster center. As the trade-off between the silhouette and the sum of squared distances, we choose $k = 5$ as the cluster number. We reduce 64 dimensions to 2 using t-distributed stochastic neighbor embedding (t-SNE) (Maaten and Hinton *et al.*, 2008) in scikit-learn.

6 The connections and shapes of each layer

Figure S4 shows the shapes and connections of each layer in the ACEP model. The yellow module, the blue module and the red module correspond to feature generating regions R1, R2 and R3, respectively. The green module corresponds to the feature fusion region R4; the purple module corresponds to the sigmoid node that outputs the prediction results.

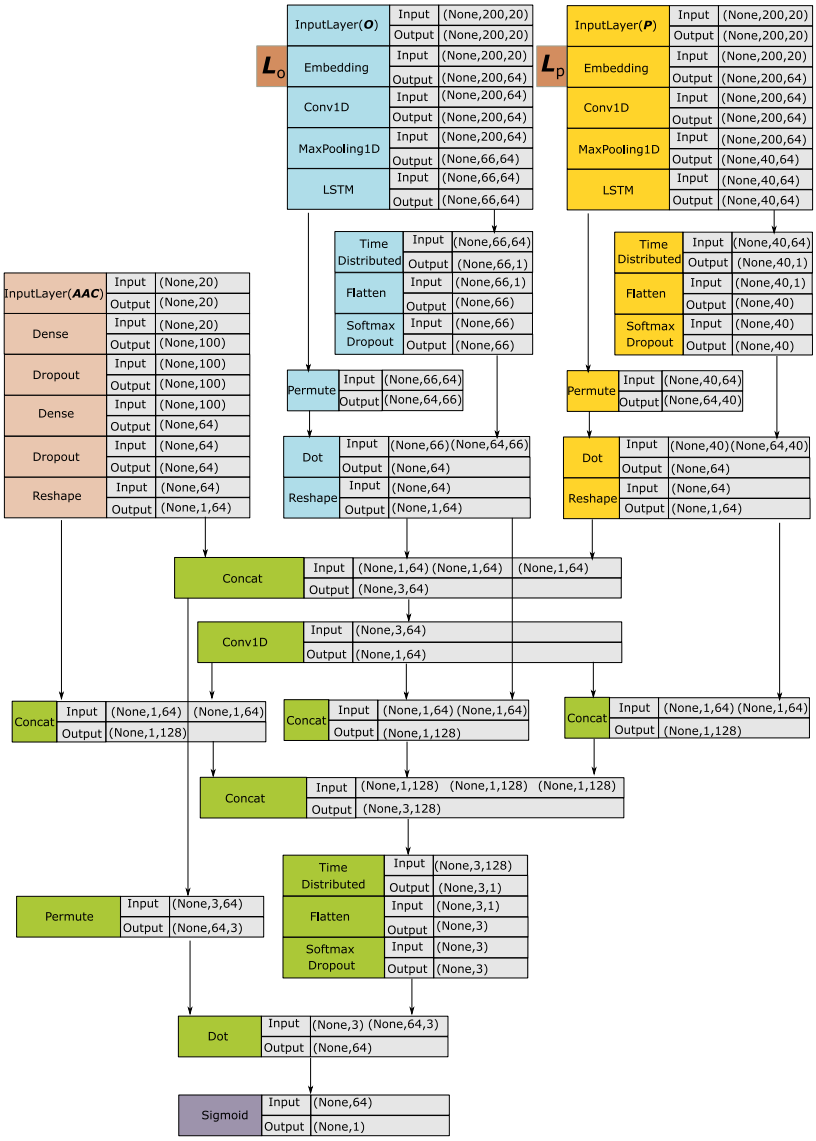

Figure S4: The shapes and connections of each layer in the ACEP model.

7 Schematic illustration of the ACEP model

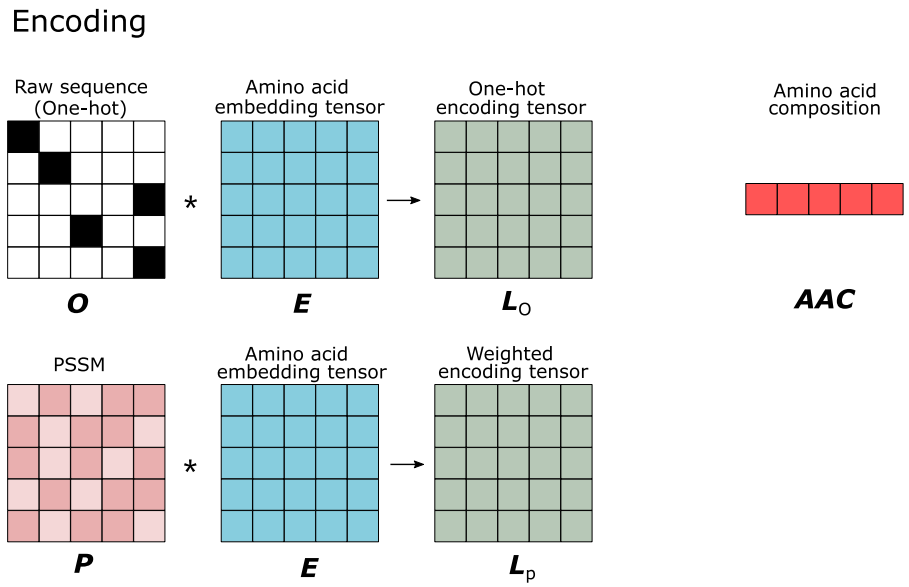

Figure S5: Encoding sequences

Figure S5 shows the encoding process of the sequences. Using position-specific scoring matrix (Altschul *et al.*, 1997), one-hot vectors and amino acid composition, one sequence is encoded as tensor $\boldsymbol{L}_p$, tensor $\boldsymbol{L}_o$ and vector $\boldsymbol{ACC}$. Each one-hot vector is a 20-dimensional vector used to distinguish each letter in a alphabet from every other letter in the alphabet. The vector consists of 0s in all cells with the exception of a single 1 in a cell used uniquely to identify the letter. The $\boldsymbol{ACC}$ vector is calculated through using amino acid composition (Qiang *et al.*, 2018). The $\boldsymbol{P}$ is obtained from the POSSUM online server (Wang *et al.*, 2017). More details of the attention module can be found in Section 2.2.

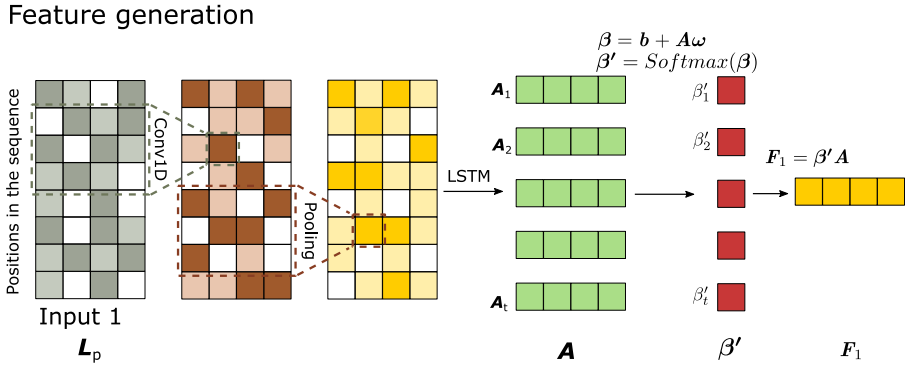

Figure S6: Schematic illustration of the feature generation.

In figure S6, the convolutional layer (LeCun *et al.*, 2015) extracts the features from

the input $\boldsymbol{L}_p$, the max pooling layer improves generalization ability, and the LSTM layer (Hochreiter and Schmidhuber *et al.*, 1997) separates the features of each time step. Each row vector of the feature map $\boldsymbol{A}$ corresponds to a position in the input sequence. $\boldsymbol{A}_i$ is then used as the input to the attention layer to generate its positional weight β_i , which becomes β_i' after softmax normalization. $\boldsymbol{F}_1$ stands for the final feature extracted from $\boldsymbol{L}_p$.

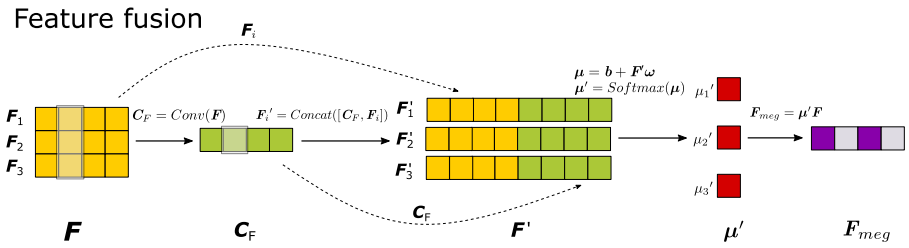

Figure S7: Convolution and attention mechanism module

In Figure S7, we vertically stack the feature vectors $\boldsymbol{F}_1$~$\boldsymbol{F}_3$ into the matrix $\boldsymbol{F}$ of 3×64, and put $\boldsymbol{F}$ into a 1D convolution with a size of 3 to calculate the convolutional feature vector $\boldsymbol{C}_F$. Next, we concatenate $\boldsymbol{C}_F$ with $\boldsymbol{F}_1$~$\boldsymbol{F}_3$ to obtain $\boldsymbol{F}_1'$~$\boldsymbol{F}_3'$. In the attention layer, the fusion weight u_i is predicted for each input $\boldsymbol{F}_i'$. The vector $\boldsymbol{\mu}$ reflects the fusion proportion of features $\boldsymbol{F}_1$~$\boldsymbol{F}_3$. Finally, the weighted average over the feature vectors $\boldsymbol{F}_1$~$\boldsymbol{F}_3$ is calculated with the weights in $\boldsymbol{\mu}$. $\boldsymbol{F}_{meg}$ stands for the fusion feature of the first three regions.

8 Misclassified AMPs

Table 2: AMPs classified by the production model as false negatives

APD Identifier	Sequence
AP02360	MVALLKSLERRRLMITISTMLQFGLFLIALIGLVIKLIELSNKK
AP01802	RPWAGNGSVHRYTVLSPRLKTQ
AP01343	TESYFVFSVGM
AP02702	LRHKVYGYCVLGP
AP01969	GPVGLLSSPGSLPPVGGAP
AP02351	QKIAEKFSGTRRG
AP01339	FLSFPTTKTYFPHFDLSHGSAQVKGHGAK
AP02805	VVYTLKRNGRTLYGF
AP02666	AVAGEKLWLLPHLLKMLLTPTP
AP02517	PPPVIKFNRPFLMWIVERDTRSILFMGKIVNPKAP
AP01975	KQIMTQFFNFARSPAVKD
AP02269	CVHWMTNTARTACIAP
AP02624	EVASFDKSKLK
AP02367	INLKAIAALARNY
AP02743	MGYGDIMKVDTSGASMKTAGQDRLTYAGVAASNTMAQTDLGRMNNYKAIIQRVGGKKDVDPAII AGIISRESRAGNVLVNGWGDNGNAWGLMQVDKRYHTPQGGWNSEEHLSQGTDIISFIKQVQGKF PSWTAEQQLKGGIAAYNIGLGGVQTYERMDVGTTGDDYSSDVVARAQWYKSQGGF
AP00140	SQLGDLGSGAGQGGGGGGSIRAAGGAFGKLEAAREEEFFYKKQKEQLERLKNDQIHQAEFHHQQI KEHEEAIQRHKDFLNNLHK
AP00520	DSHAKRHHGYKRKFHEKHHSHRGYRSNYLYDN
AP00480	VGIGTPIFSYGGGAGHVPEYF
AP01230	DGNDGQAELIAIGSLAGTFISPGFGSIAGAYIGDKVHSWATTATVSPSMSPSGIGLSSQFGSGRGTSSA SSSAGSGS
AP01233	QKKPPRPPQWAVGHFM
AP00806	HHQELCTKGDDALVTELECIRLRISPETNAAFDNAVQQLNCLNRACAYRKMCATNNLEQAMSVYF TNEQIKEIHDAATACDPEAHHEHDH
AP01831	ILPFVAGVAAMEMEHVYCAASKKC
AP01195	KRGSGWIATITDDCPNSVFVCC
AP01724	GTPGFQTPDARVISRFGFN
AP01205	STPVLASVAVSMELLPTASVLYSDVAGCFKYSAKHHC
AP00812	FAEPLPSEEEGESYSKEPPEMEKRYGGFM
AP01941	CVHWQTNTARTSCIGP
AP02895	SMATPHVAGAAALILSKHPTWTNAQVRDRLESTATYLGNSFYYGK
AP02250	MKTILRFVAGYDIASHKKKTGGYPWERGKA
AP01004	DWTAWSALVAAACSVELL
AP01326	SKGKKANKDVELARG
AP02783	ISQSDAILSAIWSGIKSLF
AP00560	TTLTLHNLCPYPVWWLVTPNNGGFPIIDNTPVVLG
AP01794	FVDLKKIANIINSIF
AP02197	PAAAAQAVAGLAPVAAEQ
AP00749	EADEPLWLYKGDNIERAPTTADHPILPSIIDDVKLDPNRRYA
AP02321	TNYGNGVGVPDAIMAGIIKLIFIFNIRQGYNFGKKAT
AP00666	EGGGPQWAVGHFM
AP00175	DSHEERHHGRHGHHKYGRKFHEKHHSHRGYRSNYLYDN
AP02028	KRKCPKTPFDNTPGAWFAHLILGC
AP02249	FISQIISTAHI
AP00027	ITPATPFTPAIITEITAAVIA
AP01624	HAEHKVKIGVEQKYGQFPQGTEVTYTCSGNYFLM
AP00998	ALPKKLKYLNLFNDGFNYMGVV
AP01379	ILENLLARSTNEDREGSIFDTGPIRRPKPRPRPRPEG
AP02858	GATPEDLNQKLS
AP00990	RNCESLSHRFKGPCTRDSN
AP01632	ATPATPTVAQFVIQGSTICLVC
AP00754	ETESTPDYLKNIQQQLEEYTKNFNTQVQNAFDSDKIKSEVNNFIESLGKILNTEKKEAPK
AP00741	PITYLDAILAAVRLLNQRISGPCILRLREAQPRPGWVGTLQRRREVSFLVEDGPCPPGVDCRSCEPGA LQHCVGTVSIEQQPTAELRCRPLRPQ
AP02193	YSKSLPLSVLNP
AP02030	MQIFVKTLTGKTITLEVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYNIQKESTLHL VLRLR
AP00996	ISLEICAIFHDN
AP02072	MSNTQAERSIIGMIDMFHKYTRRDDKIDKPSLLTMMKENFPNFLSACDKKGTNYLADVFEKKDKN EDKKIDFSEFLSLLGDIATDYHKQSHGAAPCSGGSQ

9 Supplementary Data

Supplementary Data 1. The amino acid embedding tensor $\boldsymbol{E}$ trained by ACEP model.
Supplementary Data 2. The average attention intensity calculated from 500 AMPs.
Supplementary Data 3. The fusion ratio of the features.
Supplementary Figures S8.

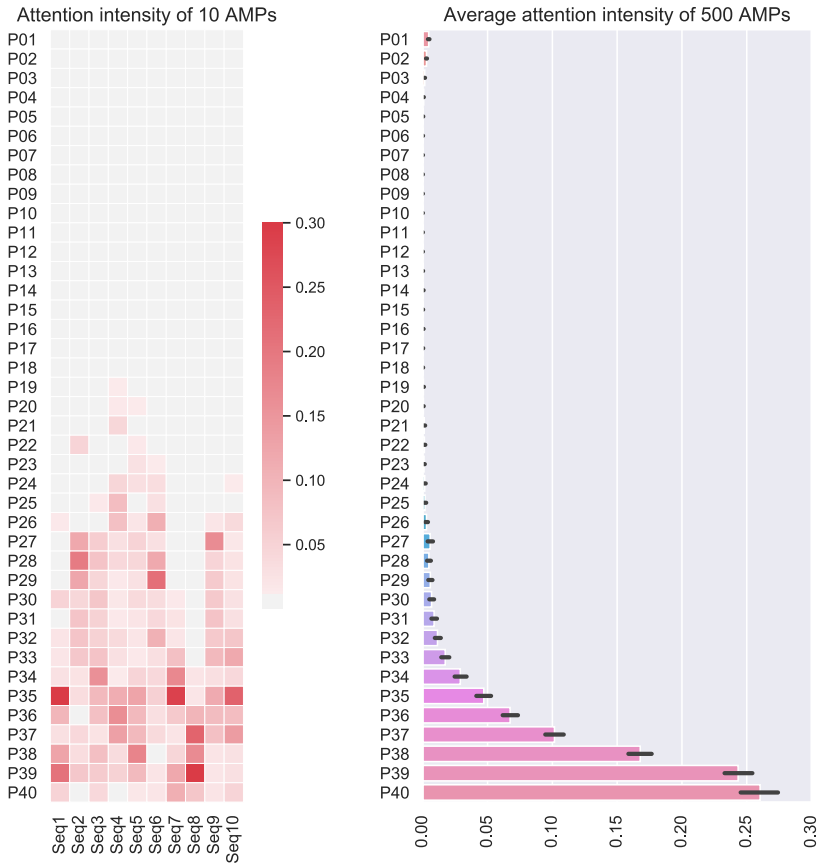

Figure S8: The attention intensity of different parts of the sequence

References

Altschul, S. F. *et al*. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**(17), 3389–3402.

Hochreiter, S. and Schmidhuber, J. *et al*. (1997). Long short-term memory. *Neural Comput.*, **9**(8), 1735–1780.

LeCun, Y. *et al*. (2015). Deep learning. *Nature*, **521**(7553), 436–444.

Lloyd, S. *et al*. (1982). Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, **28**(2), 129–137.

Van der Maaten, L. and Hinton, G. *et al*. (2008). Visualizing data using t-sne. *J. Mach. Learn. Res.*, **9**(Nov), 2579–2605.

Qiang, X. *et al*. (2018). CPPred-FL: a sequence-based predictor for large-scale identification of cell-penetrating peptides by feature representation learning. *Briefings Bioinf.*

Rousseeuw, P. J. *et al*. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, **20**, 53–65.

Veltri, D. *et al*. (2018). Deep learning improves antimicrobial peptide recognition. *Bioinformatics*, **34**(16), 2740–2747.

Wang, G. *et al*. (2015). APD3: the antimicrobial peptide database as a tool for research and education. *Nucleic Acids Res.*, **44**(D1), D1087–D1093.

Wang, J. *et al*. (2017). POSSUM: a bioinformatics toolkit for generating numerical sequence feature descriptors based on pssm profiles. *Bioinformatics*, **33**(17), 2756–2758.