Positional SHAP (PoSHAP) for Interpretation of Machine Learning Models Trained from Biological Sequences

Quinn Dickinson, Jesse G. Meyer*

Department of Biochemistry, Medical College of Wisconsin, Milwaukee, Wisconsin, 53226

Correspondence to jessegmeyer@gmail.com

Abstract (148/300)
Machine learning with multi-layered artificial neural networks, also known as “deep learning,” is effective for making biological predictions. However, model interpretation is challenging, especially for sequential input data used with recurrent neural network architectures. Here, we introduce a framework called “Positional SHAP” (PoSHAP) to interpret models trained from biological sequences by utilizing SHapely Additive exPlanations (SHAP) to generate positional model interpretations. We demonstrate this using three long short-term memory (LSTM) regression models that predict peptide properties, including binding affinity to major histocompatibility complexes (MHC), and collisional cross section (CCS) measured by ion mobility spectrometry. Interpretation of these models with PoSHAP reproduced MHC class I (rhesus macaque Mamu-A1*001 and human A*11:01) peptide binding motifs, reflected known properties of peptide CCS, and provided new insights into interpositional dependencies of amino acid interactions. PoSHAP should have widespread utility for interpreting a variety of models trained from biological sequences.

Keywords: Neural networks, deep learning, model interpretation, shapely additive explanations (SHAP), peptides, major histocompatibility complex (MHC), long-short term memory, LSTM

Introduction

Sequences are ubiquitous in biology. Nucleic acids and proteins encode information as sequences of monomeric building blocks. Sequence order is extremely important; the primary amino acid sequence of a protein uniquely determines the set of 3D structures formed after folding. Decades of effort by thousands of scientists has focused on measuring protein structures (1–3) and determining intermolecular binding (4). Significant efforts have been devoted to protein structure prediction (5,6). Recent advances in deep learning have achieved major milestones in protein structure prediction (7).

Deep learning is a type of machine learning that uses neural networks to learn relationships between pairs of input and output data (8). For example, deep learning models can take inputs of peptide sequences and predict chromatographic retention time (9). There are many types of neural network models that differ primarily in how neurons are connected. Each architecture is well suited for different types of input data. For example, convolutional neural networks (CNNs) are effective at using images as inputs (10), and recurrent neural networks (RNNs) are effective at using sequence data as input (11). RNNs have found extensive application to natural language processing (NLP) (12), and by extension as a similar type of data, predictions from biological sequences such as peptides or nucleic acids (13). A specific type of RNN called long short-term memory (LSTM) solves the vanishing gradient problem seen with backpropagation of RNNs (14), and thus LSTM has seen widespread use for biological sequence data.

One goal of building predictive models is to create an understandable and actionable relationship between the input and output data. Although deep learning with LSTM models is effective for making predictions from sequences, interpreting how inputs lead to specific outputs is not trivial. There are model-specific interpretation strategies, such as layer-wise relevance propagation (15) or the attention mechanism (16). There are also strategies to enable interpretation of an arbitrary model, such as permutation importance (17), and Shapley additive explanations (18,19). SHAP is a perturbation-based explanation method where the contribution of an input is calculated by hiding that input and determining the effect on the output. SHAP expands this using the game theoretic approach of Shapely values that ensures the contributions of the inputs sum to the predicted output plus a baseline. (18) SHAP is an attractive option because it can dissect interactions between inputs, for example even when inputs are correlated. SHAP is also beneficial in that it can be used with any arbitrary model. However, SHAP does not directly enable sequence-dependent model interpretation.

The major histocompatibility complex (MHC) is an array of closely related genes that encode cell surface proteins that form an essential part of the adaptive immune system (20–22). There are two main classes of MHC complexes, I and II. Peptides bound by the MHC I complex are primarily generated by the proteasome from intracellular proteins (23). Not all degradation products are bound into the MHC class I complex, nor are all peptides bound with equal frequency. Peptides suitable for the MHC class I complex are generally between eight and ten amino acids in length, although longer peptides have been reported (24). The sequence of the peptide is the primary determinant of binding affinity to each MHC class I complex allele. Given the polymorphism of MHC class I alleles in the human population, abundance of potential

binding peptides, and the low throughput of current binding assays, the direct testing of most peptides is infeasible. Therefore, the prediction of binding affinities through methods such as machine learning or molecular modelling could lead to improved development of vaccines against disease like cancer (25).

Extensive efforts have focused on prediction of peptide-MHC interactions (26,27). Both classification and regression models are used to learn which peptides bind to each MHC allele, for example see O'Donnell *et al.*, Zeng and Gifford, and Liu *et al.* (28–30) However, because many reports forgo model interpretation, the learned biochemical relationships remain unknown. Other works determine relationships learned by their model, for instance both Jin *et al.* (31), and Hu *et al.*(32) used CNNs with an attention mechanism to determine the weights of the inputs on the final prediction.

Attention mechanisms have been successful in recapitulating experimentally defined binding motifs, but require that the model be constructed with attention layers. This may limit the flexibility of model architecture when designing new models. For example, attention mechanisms are specific to neural networks. Simpler models, such as random forests and XGboost, may also be more suitable for some applications, and these cannot utilize attention. Also, while attention mechanisms are currently very effective, there is always a possibility that new architectures will emerge that make interpretations using attention infeasible. Beyond this, attention is a metric of the model itself, while SHAP values are calculated on a per input basis. By looking at the model through the lens of the inputs, we can understand the model's "reasoning" behind any peptide. Attention mechanisms also do not enable dissection of interpositional dependencies between amino acids. Thus, new methods for model agnostic interpretation are desirable.

In addition to predicting binding affinities of peptides, deep learning is useful for predicting peptide properties for proteomics applications (33) including: fragmentation patterns during tandem mass spectrometry (34–36), liquid chromatographic retention time (35,37,38), and ion mobility (39). However, attempts at model interpretation are uncommon in this body of literature. One recent paper (39) utilized SHAP to better understand the mechanics behind the collisional cross section (CCS) of peptides, but insight was limited to aggregate amino acid contributions without position context. Further work is needed to allow model-agnostic interpretation of neural networks trained from biological sequences to understand general patterns in the chemistry of peptides.

Many effective deep learning model architectures are available for making predictions from inputs of biological sequences, and there is currently no single correct choice. CNN models such as MHCflurry 2.0 (40) and LSTM models are effective at predicting MHC binding of peptides (41). Even simpler models, such as random forests, have been used to predict MHC binding (42,43). Prediction of other peptide properties like tandem mass spectra are often done with CNN or LSTM models (33). More recently, given the extraordinary performance of transformer models like BERT (44) and GPT-3 (45) for NLP, there is an interest in transformer models for biological sequences (46).

Here we demonstrate that LSTM models easily learn to perform regression directly from peptide sequence to that peptide's properties, including affinity to various MHC alleles (32,47) and CCS (39). Our main contribution is a strategy to interpret such models that we term "positional SHAP" (PoSHAP). Unlike other strategies that adapt the SHAP explainer (48) or another approach that enables visualization of SHAP interpretations from sequences (49), PoSHAP simply adds indexes to inputs and maintains positional context after SHAP kernelExplainer to reveal how each amino acid contributes to predicted properties. We show how this enables new analysis for specific peptides and generally across all peptide predictions. We extend the strategy to track interpositional dependence of amino acids for binding and CCS. This work therefore describes a general, broadly applicable framework for understanding notoriously abstruse deep learning models trained on biological sequences.

Methods

Data

Data used for training and testing the Mamu model was obtained from Haj *et al* 2020 (47), where all possible 8-, 9-, and 10-mer peptides from 82 SIV and SHIV strains were measured by fluorescent peptide array. The data consists of 61,066 entries containing the peptide sequence, peptide length, and five intensity values corresponding to the intensity obtained from the fluorescence assay for each of the five Mamu alleles tested (A001, A002, A008, B008, and B017). From the methods of Haj *et al.*, each intensity is the base 2 logarithm of the median intensity value of five replicates reported for each peptide as measured by an MS200 Scanner at a resolution of 2µm and a wavelength of 635nm (47). For training and testing of the model, the dataset was randomly split into three categories (**Supplemental Figure 1**). Because the dataset contains truncated forms of each core peptide sequence as 8-, 9-, and 10-mers, the data splitting grouped each core sequence into unique indexes and split based on those indexes. This core sequence-based splitting ensured that training and testing data would not have truncated versions of the same peptide. The training data had 43,923 entries (71.93% of all data). The validation data to assess overfitting during training had 10,973 entries (17.97% of all data). The test data to test the overall model performance had 6,170 entries (10.10% of all data).

Data for the human MHC allele was obtained from Hu *et al.* 2019 (32), and is a compilation of data from the IEDB MHC class I binding affinity dataset (Kim *et al*., 2014 (50), Vita *et al*., 2015 (51), and Pearson *et al.* 2016 (52)). This dataset consists of species, allele, peptide length, peptide sequence, and a binding affinity measurement as IC50. For A*11:01, a subset of the data was chosen by selecting only the peptides between eight and ten amino acids in length with binding data for the allele. The IC50 were transformed as described in Hu *et al.* 2019 (32) and Nielsen *et al.* 2007 (53) where score = 1-log(affinity)/log(50000). Data splitting into training, validation, and test data was performed as above, split by core peptide sequences (**Supplemental Figure 2**). The training data had 4,522 entries (71.97% of all data), the validation data consisted of 1,132 entries (18.02% of all data), and the test data consisted of 629 entries (10.01% of all data).

Data for the CCS was obtained from Meier *et al.* 2021 (39). The dataset consists of peptide sequences, peptide lengths, peptide modifications, retention times, and calculated CCS, among other values, for about 2,000,000 values. From this dataset, we removed all peptides that had any modifications, and for simplicity, kept only peptides with lengths of 8, 9, or 10 amino acids. The mean of the CCS were taken for remaining peptides that had the same sequence. The final dataset consisted of 45,990 entries. The data was split into training, validation, and test sets, split by core peptide sequences, as described above (**Supplemental Figure 2**). The training data consisted of 33,134 entries (72.04% of all data). The validation data consisted of 8,256 entries (17.95% of all data). The test data consisted of 4,600 entries (10.00% of all data).

Model architecture

The Keras(2.3.0-tf) (54) interface for Tensorflow(2.2.0) (55) was used to build and train the LSTM models (see layouts in **Supplemental Figure 3**). Peptide sequences were converted to

integers ranging from 0 to 20 where each integer corresponds to an amino acid or the special token “END”, which is used to pad peptides with length 8 or 9 to have length 10. The embedding layer takes these ten integer inputs corresponding to each position of the peptide. Each input is transformed by the embedding layer to a 10x50 dimensional matrix that is sent to the first LSTM layer (14). The LSTM layer outputs a 10x128 dimensional matrix to a dropout layer where a proportion of values are randomly “dropped”, or set to 0. For the MHC models, a second LSTM layer outputs a tensor with length 128 to a second dropout layer. Then in all models, a dense layer reduces the data dimensionality to 64. For the MHC models, the data is then passed through a leaky rectified linear unit (LeakyReLU) activation before a final dropout layer, present in all models. The final dense layer produces either one or five outputs, which are trained to predict the output values (intensity, IC50, CCS). The model is compiled with the Adam optimizer (56) and uses mean squared error (MSE) loss.

Hyperparameter search

For the Mamu MHC alleles, dropout, batch size, and the number of epochs were optimized using the hyperas wrapper for hyperopt (57). The hyperparameter search allowed a uniform range between 0 and 0.6 for each of the three dropout layers. The search for epoch and batch size hyperparameters had binary choices. Epochs were either 1,000 or 2,000. Batch size was either 2,500 or 5,000. To ensure that our unorthodox batch size was correct, we performed the hyperparameter search again, with options of 32, 64,128, or 5,000, and a batch size of 5,000 was again selected as optimal.

For the human MHC allele model, dropout, batch size, learning rate, and the number of epochs were optimized using the hyperas wrapper for hyperopt (57). The hyperparameter search used a uniform range between 0 and 0.8 for each of the three dropouts. The search for epochs, batch size, and learning rate hyperparameters had defined choices. Epochs were 100, 500, or 1,000. Batch size was 32, 64, 128, or 5000. Learning rate was a choice between 0.001, 0.005, 0.01, 0.05, and 0.1. To reduce overfitting, the number of epochs was fixed at 200.

For the CCS model, dropout, batch size, learning rate, and the total number of epochs were optimized using the hyperas wrapper for hyperopt (57). Each of the two dropout layers had ranges between 0 and 0.8. Learning rate was a choice of 0.001, 0.005, 0.01, 0.05, and 0.1. Batch size was a choice of 32, 64, 128, and 256. Epochs were a choice of 100, 500, and 1000. To reduce overfitting, we decided on a final epoch count of 200.

For each dataset, the hyperparameter search was run with the tree of parzen estimators algorithm (58) allowing a maximum of 100 evaluations. The optimal parameters from this search are in **Supplemental Figure 3**.

Final Model Training
For each dataset, the final models were re-trained using the best hyperparameters **(Supplemental Figure 3)**. Loss (as MSE) for training and validation data was plotted against the training epochs to monitor overfitting **(Supplemental Figure 4)**.

Model Performance - Regression Metrics
Test peptides were input to the final trained model and the predicted outputs were compared with the experimental data**.** Correlations between true and predicted values were assessed by MSE, Spearman's rank correlation coefficient (Spearman's ρ), and the correlation p-value.

Positional SHAP (PoSHAP)
Shapely Additive Explanations (SHAP) (18) were used to determine the contribution of each position on each peptide to the peptide's overall predicted value. As the baseline, training peptide sequence data was summarized as 100 weighted samples using the SHAP kmeans method. The summarized data, the test peptide sequence data, and the trained model were input into SHAP's KernelExplainer method. The contribution of each amino acid at each position was stored in an array. The mean SHAP value of each amino acid at each position was calculated for each input dataset**.** Exemplary plots of the top predicted peptides were generated using SHAP's force_plot method indexed with peptides and position (19). Dependence plots were generated using SHAP's dependence_plot method and modified with MatPlotLib (59).

PoSHAP compared to Summary Statistics
Heatmaps were created for each of the three training datasets, with the count of each amino acid at each position. An array was created with a value to represent each amino acid at each position. Each peptide was iterated over, and the value in the heatmap for each amino acid position was incremented to get the counts. To determine the top valued peptides for each dataset, the dataset was sorted by the experimental values of the training data and each peptide given a rank index. A linear regression was calculated between the rank index and experimental values of the training data. All values that were greater than the linear regression at the particular index and greater than the overall mean of the dataset were considered to be top valued peptides. The subset of top valued peptides were then arranged into an array as above to create a heatmap.

Dependence Analysis
To generate the dependence analysis tables, for each amino acid at each position, the SHAP values were split into two sets. The first set consists of the SHAP values where a different specified position contains a different specific amino acid. The second set consists of the remaining SHAP values for the amino acid at the position. For each position and amino acid, all sets of positions and amino acids are compared. The two sets are not normally distributed and were therefore compared with a Wilcoxon Rank Sum test (also known as Mann-Whitney U-test), and the p-values are adjusted with the Bonferroni correction.

To analyze the interdependent interactions between positions and amino acids, the subset of all significant (Bonferroni adjusted P-value < 0.05) interactions were taken from the CCS

dependence analysis tables. Interactions involving the “End” token were removed. The remaining interactions were grouped by distance or by expected interaction type. Interactions grouped by distance, were further grouped into either neighboring (distance = 1), near (distance = 2,3,4,5,6), or far (distance = 7,8,9). Each amino acid was grouped into the following categories: “Positive” for arginine, histidine, and lysine; “Negative” for aspartic acid and glutamic acid; “Polar” for serine, threonine, asparagine, and glutamine; “Hydrophobic” for alanine, valine, isoleucine, leucine, methionine, phenylalanine, tyrosine, tryptophan, cysteine, glycine, and proline; and “End” for interactions involving the end input. Expected interaction type was determined by the following: “Charge Attraction” by interactions between “Positive” and “Negative” categories. “Charge Repulsion” by interaction between “Positive” and “Positive” or “Negative” and “Negative” categories. “Polar” by interactions between “Polar” and “Polar,” “Polar” and “Negative,” or “Polar” and “Positive” categories. “Other” by interactions not noted here, including interactions such as “Polar” and “Hydrophobic.” As there were very few hydrophobic interactions i.e. hydrophobic and hydrophobic, they were included with “Other.” ANOVA with Tukey’s post hoc test was calculated amongst the distance groups and amino acids categories to determine significance. Finally, each amino acid category was split into the distance of interaction as above, neighboring (distance = 1), near (distance = 2,3,4,5,6), or far (distance = 7,8,9). ANOVA with Tukey’s post hoc test was calculated amongst the combined categories to determine significant difference.

Additional Model Testing

ExtraTreesRegressor from scikit_learn (60) and XGBRegressor from xgboost (61) were used to train models from the training data for each of the three datasets. SHAP values were calculated for the testing data using KernelExplainer with the training data summarized by SHAP’s kmeans method to 100 points. SHAP values were processed with PoSHAP as above.

Three additional LSTM models were trained from the training datasets with different hyperparameters and using RMSprop as the optimizer. The model architectures remained the same. For the Mamu model, the dropout rates were 0.4839, 0.1829, and 0.1177 for dropout layers one, two, and three respectively. It had a batch size of 32 and ran for 200 epochs. Learning rate was set at the default value. For the A*11:01 model, the dropout rates were 0.0500, 0.2953, and 0.3258 for dropout layers one, two, and three respectively. It had a batch size of 32 and ran for 200 epochs. Learning rate was set at 0.01. For the CCS model, the dropout rates were 0.5324 and 0.0865 for dropout layers one and two respectively. It had a batch size of 128 and ran for 100 epochs. Learning rate was set at the default value. SHAP values were calculated for the testing data using KernelExplainer with the training data summarized by SHAP’s kmeans method to 100 points. SHAP values were processed with PoSHAP as above.

Data and Code Availability

The data and code are available at https://github.com/jessegmeyerlab/positional-SHAP.

Results

Model Training and Prediction

Datasets from each source consisted of a peptide sequence and a corresponding measurement, including fluorescent intensity (47), IC50 (32,50–52), or CCS (39). For the Mamu dataset, each peptide in the table had values for five Mamu MHC class I alleles: A001, A002, A008, B008, and B017. For the human MHC and peptide ion mobility datasets, each peptide had a single value, representing IC50 and CCS, respectively. Data was split into training, validation and test sets in a manner that ensures truncated versions of the same core peptide are in the same set (**Supplemental Figure 1, Supplemental Figure 2**). The LSTM models used peptide sequences converted to integers as input to an embedding layer, and learned to perform either single-output regression, for the human MHC and CCS, or multi-output regression for the outputs of the five Mamu MHC alleles. (**Figure 1, Supplemental Figure 3).**

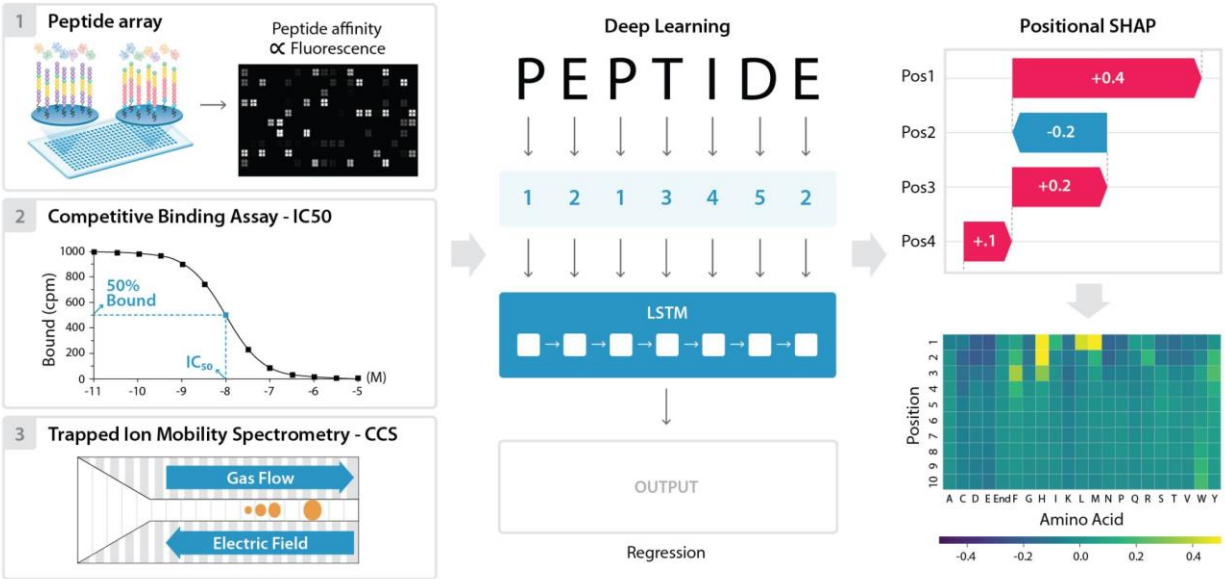


Figure 1: Overview of data, modeling, and positional SHAP analysis for model interpretation. Peptide sequence and output data was downloaded from Haj *et al.* 2020, Hu *et al.* 2019, and Meier *et al.* 2021 and used as an input for three separate deep learning models. The peptide sequences were numerically encoded, split to positional inputs, and Long Short-Term Memory (LSTM) models were trained to predict each of the outputs. These include the five peptide array intensities for the Mamu MHC allele data, IC50 binding data for the A*11:01 data, and CCS for the mass spectrometry data. The trained models were then used to make predictions on a separate test subset for each of the datasets. Finally, the model interpretation method SHAP was adapted to enable determination of each amino acid position's contribution to the final prediction. This PoSHAP analysis was visualized by plotting the mean SHAP value of each amino acid at each position as a heatmap.

Despite the limited sizes of the training sets, the LSTM models achieved excellent performance on these regression tasks as evidenced by scatterplots of true values versus model predictions for the held-out test set (**Figure 2**). Training versus validation loss for the final multi-output model (**Supplemental Figure 4**) demonstrates some overfitting but not to the detriment of the model's generalizability. To prevent overfitting of the single output regression models, epochs were limited to 200. All correlations between true and predicted values were significant with p-values less than 1.0E-145.

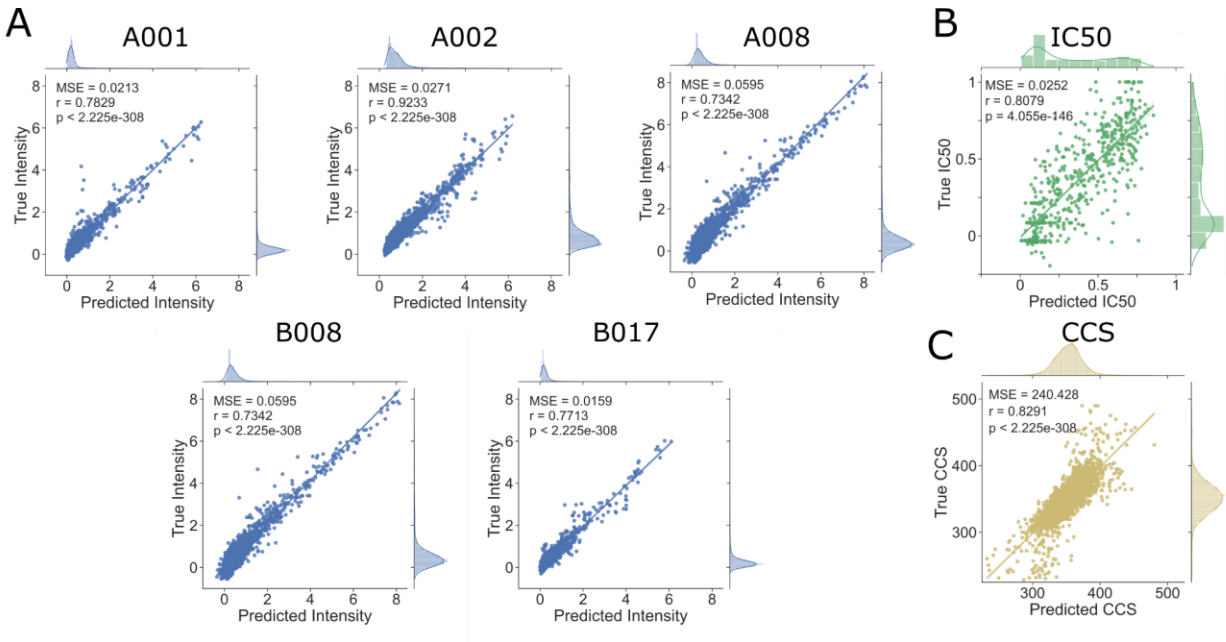


Figure 2. LSTM Model Performance. Held-out test peptides were input to the model and predictions were plotted against true experimental values. For the Mamu allele multi-output regression model (A), predicted and experimental intensities were compared. For the A*11:01 model (B), predicted and experimental IC50s were compared. For the collisional cross section model (C), predicted and experimental collisional cross sections were compared. For each model, predicted and experimental values were compared with the Spearman's rank correlation and all demonstrated a significant (p-value < 1E-145) positive correlation (rho>0.6).

Positional SHAP (PoSHAP)

PoSHAP utilizes the standard SHAP package but adapts the analysis by simply appending an index to each input and maintaining positional information after the kernelExplainer interpretation, which enables tracking of each input postion's contribution to an output prediction (**Supplementary figure 5**).

PoSHAP analysis revealed expected patterns of positional effects for experimentally supported interactions. For the Mamu allele A001, we found patterns similar to a prior publication that determined specificity experimentally with a library of peptides with single amino acid substitutions (62). This previous study determined a preference for "...S or T in position 2, P in position 3, and hydrophobic or aromatic residues at the C terminus". Our heatmap shows a similar preference (**Figure 3**), but we also note that F/I/L is preferred at position 1, and a proline at one of the positions between 2-5. The preference for a hydrophobic amino acid in position 1 was also seen using a substitution array in the original publication of the peptide array data used to train our models (47).

Figure 3. Heatmaps showing PoSHAP analysis to determine amino acid binding motifs from deep learning models. The mean SHAP values for each amino acid at each position across all peptides in the test set were arranged into a heatmap. The position in each peptide is along the y-axis and the amino acid is given along the x-axis. "End" is used in positions 9 and 10 to enable inputs of peptides with length 8, 9, or 10. For comparison, the SHAP force plot for the peptide with the highest binding prediction is shown for each allele.

For the human MHC allele A*11:01 model, PoSHAP analysis recapitulates positional relationships found through attention mechanism based models **(Figure 4A)** (32). This pattern is in congruence with the experimental data for the binding of A*11:01 (63). Jin *et al.* (31) reported anchor sites for MHC alleles from attention-based models. PoSHAP analysis matched these anchor sites based on the PoSHAP heatmap **(Figure 4A)** and the range of the SHAP values per position **(Supplemental Figure 6).** Remarkably, this was achieved for A*11:01 using a total dataset of only 4,522 examples, which shows that PoSHAP can be effective with even less than 10,000 training examples.

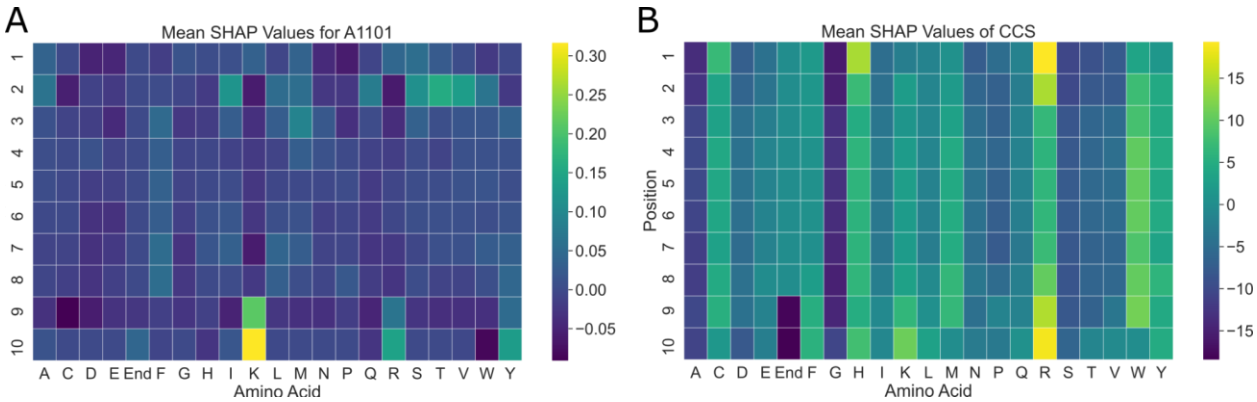


Figure 4. PoSHAP interpretation of models trained to predict A*11:01 binding or CCS. The mean SHAP value for each amino acid across all test peptides were calculated and arranged into heatmaps representing the values for A*11:01 (A) and CCS (B). The position along the peptide is along the y-axis and each amino acid is listed along the x-axis.

For the CCS model, PoSHAP analysis reflects results from experimental positional analysis performed in Meier *et al.* (39), which demonstrated the importance of the histidine residue position relative to the peptide's C- and N-terminus. Our PoSHAP analysis also shows the importance of histidine, with the highest PoSHAP values at the peptide's N-terminus, reflecting that peptides with n-terminal histidine have higher CCS. Meier *et al.* (39) also performed SHAP analysis on their own model that illustrates the contribution of each amino acid across all positions. They noticed that lysine, arginine, and histidine had the highest range of SHAP values, and suggested that this variation indicated the exact positions of these residues would influence the CCS. PoSHAP analysis agreed with this and showed that the amino acids with the greatest ranges **(Supplemental Figure 7)** also had the highest levels of positional dependence, with histidine, lysine, and arginine having the greatest overall ranges and the greatest dependence on position **(Figure 4B).**

Given the accuracy of PoSHAP in recapturing experimentally verified positional effects, it's use has promise in generating hypotheses about the analyzed systems. One example is with the CCS data. The PoSHAP analysis revealed that the three amino acids (H, K, R) that contribute the highest proportion to CCS when at the termini are all positively charged under physiological and mass spectrometry electrospray conditions. The contribution to CCS by the positively charged amino acids may be due to charge repulsion between the positive charges on adjacent

amino acids in the peptide, as the vast majority of peptides used to train the model had a greater charge than +1. When the amino acids are at the termini, they have the greatest freedom to extend away from the rest of the peptide, increasing the CCS. The opposite trend is seen for the negatively charged amino acids, aspartic acid and glutamic acid, which had a slight negative effect across PoSHAP with the greatest effects at the termini.

Another application of PoSHAP is to be able to make hypotheses about the binding characteristics of the uncharacterized MHC alleles **(Figure 3)**. We found that the model predicts that Mamu A001 prefers F, I, L,S, T, V, or Y in the first position, with a strong preference for S, T, or P in the second position; S and T are very similar chemically, with small, polar side chains containing hydroxyl functional groups. The heatmap of SHAP values also showed that A001 had a preference for proline between positions two and six. The preference of Mamu A002 was similar to A001 in that n-terminal serine or threonine resulted in high binding, but the preference for proline was absent. The preference map of Mamu A008 showed an opposite trend, where only the preference for early proline between positions one and four is readily apparent and the contribution of S or T is absent. Mamu B008 appears to be most selective for the amino acids near the N-terminus, with a strong preference for arginine or methionine and strong negative SHAP values for many amino acids. Finally, B017 showed a preference for L, M, or H followed by F near the N-terminus. The heatmap of SHAP values for B017 also showed a positive contribution to binding from tryptophan near the C-terminus, suggesting that the entire peptide length may play a bigger role in binding to the B017 MHC protein.

PoSHAP analysis also reveals the amino acids at each position that decrease peptide binding. All MHC alleles except for A002, and most pronounced in B008, have a strong negative contribution to binding prediction if there is an acidic amino acid in position one or two (i.e. D or E). For all alleles except B017, histidine near the peptide N-terminus also predicts low binding affinity.

We further show that PoSHAP can reveal the pooled binding contributions for any subsets of peptides. When the PoSHAP heatmap is filtered for the eight peptides with the highest binding predictions (top 0.013%), distinct patterns emerge **(Supplemental Figure 8)**. The serine or threonine at position two remains important for the A001 and A002 alleles and can also be important to A008 binding. We also performed the same analysis with the eight peptides with the lowest binding predictions (**Supplemental Figure 8)**. These PoSHAP heatmaps are primarily composed of negative SHAP values, suggesting that using this subset reveals amino acids at certain positions that are detrimental to MHC binding. Of note is the negative SHAP values for aspartic acid and glutamic acid along peptide, suggesting that positive charge may inhibit binding.

PoSHAP versus Simple Summary Statistics

We wondered whether the patterns revealed by PoSHAP simply reflect the summary statistics for the high-binding or high-CCS subset of peptides. As expected, due to known differences in amino acid abundance across the proteome, the prevalence of amino acids was different across the training data and were also heterogeneous across positions (**Figure 5A**). To determine the

subset of high CCS peptides, peptides were ordered in the training set by their CCS rank and then linear regression was performed to get the average trend line (**Figure 5B**). Any peptide above that trendline and the overall mean was defined as “high CCS”, and the frequency of amino acids at each position in this set was summarized using a heatmap (**Figure 5C**). Compared to the statistical amino acid frequencies, PoSHAP suggests a greater importance of arginine at both termini, the importance of tryptophan to increase CCS becomes apparent, and interior glutamic acid contributes less to high CCS than the frequencies would suggest (**Figure 5D**). The same analysis was repeated for MHC data (**Supplementary Figures 9 and 10**). This demonstrates that PoSHAP finds non-linear relationships between the inputs and the outputs that are not present by simple correlation.

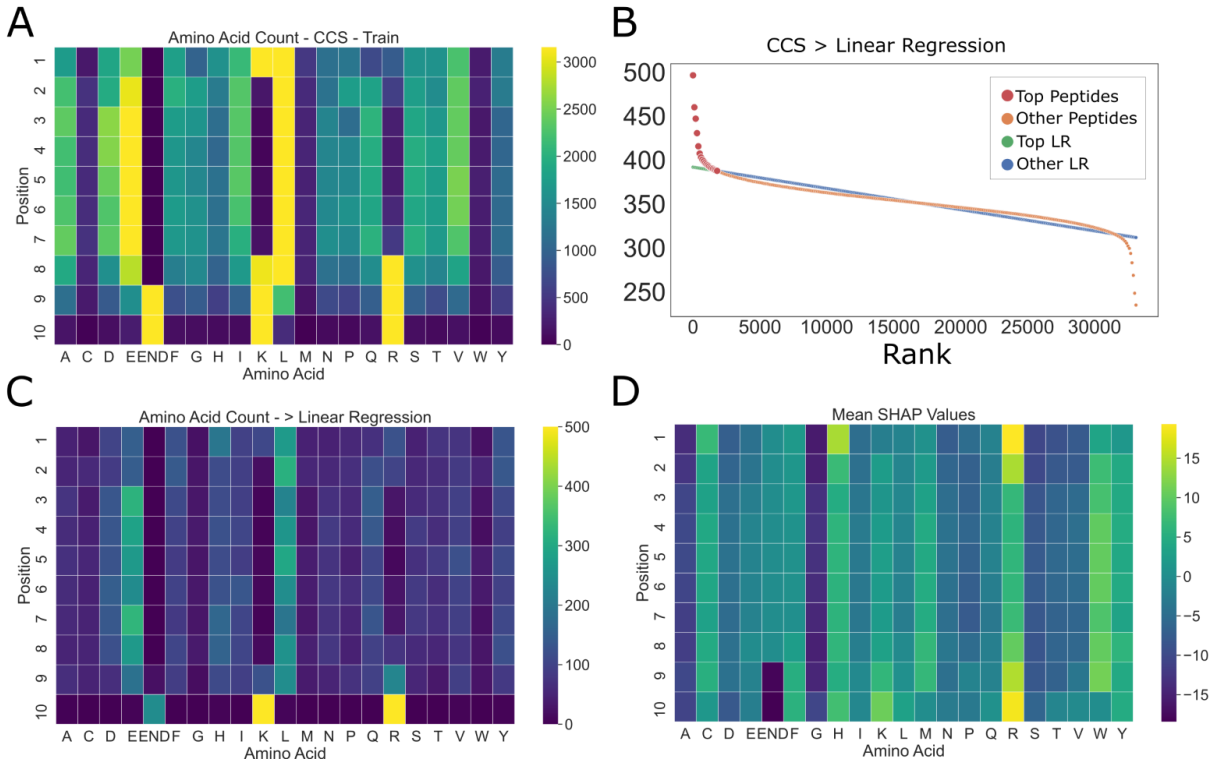


Figure 5: Amino acid summary statistics differ from PoSHAP values for the CCS data. (A) Amino acid counts as a function of position for training data. (B) Procedure for picking the ‘top peptides’ with the highest CCS. Linear regression was performed on the peptides ranked by their actual CCS value. Any peptide that fell above the trendline and overall mean were defined as ‘top peptides’. (C) Counts of amino acids for the top peptides were summarized in a heatmap. (D) Mean SHAP values across amino acids and positions from PoSHAP analysis.

PoSHAP Interpositional Dependence Analysis

The SHAP value of any position is dependent on the values of all other positions in the peptide. PoSHAP values for each amino acid at each position were split based off of the amino acid at another position across all peptides. This enabled the determination of the dependence of a PoSHAP value on the presence of an amino acid at any and all other positions. This method also enabled determination of the significance and magnitude of the dependence by comparing the means and calculating a Wilcoxon Rank Sum test with a Bonferroni correction (**Supplemental Tables 1-3**). The original SHAP package provides a means to illustrate these relationships through its dependence plots.

Using the dependence analysis, we were able to discover significant positional relationships for each of the three models we trained. For the Mamu alleles, the most striking relationship was observed with the model trained on the A001 dataset (**Figure 6**). As previously mentioned, it can be seen in the heatmaps that the highest SHAP values are observed for serine and threonine near the N-terminus of the peptide **(Figure 3).** However, the top predicted peptides do not show the same pattern, instead beginning with either a phenylalanine or a leucine, and continuing with a serine or threonine **(Supplemental Figure 8).** Among the calculated interpositional relationships with the greatest significance and magnitude are between the leucine and threonine (**Supplemental Table 1**, Bonferroni adj. p-value 1.72E-22) and the phenylalanine and serine (**Supplemental Table 1,** Bonferroni adj. p-value 6.44E-8) between the first and second positions **(Figure 6).** Additionally, threonine or serine followed by a proline between positions one and two (**Supplemental Table 1**, Bonferroni adj. p-values 3.61E-6, 6.62E-8, respectively), or two and three (Bonferroni adj. p-values 1.16E-7, 4.24E-5, respectively) were significant. This suggests that the most important motif for binding is Thr-Pro or Ser-Pro and that the ideal binding motif for A001 is Leu-Thr-Pro or Phe-Ser-Pro at the N-terminus of the peptide.

For the human allele A*11:01 model, there were very few significant interactions, potentially due to the smaller size of the dataset. However, there were still a couple notable interactions. These are those between lysine at the ninth position and serine or leucine at position two (**Supplemental Table 2**, Bonferroni adj. p-values 0.001, 1.43E-5, respectively). Both of these have significantly greater SHAP contributions when lysine is at position nine. This may reflect that there is some flexibility with the earlier root site when lysine is bound, and may demonstrate that the model had learned the length of the binding motif between the second position and the C-terminus (31) **(Supplemental Table 2, Supplemental Figure 6B).**

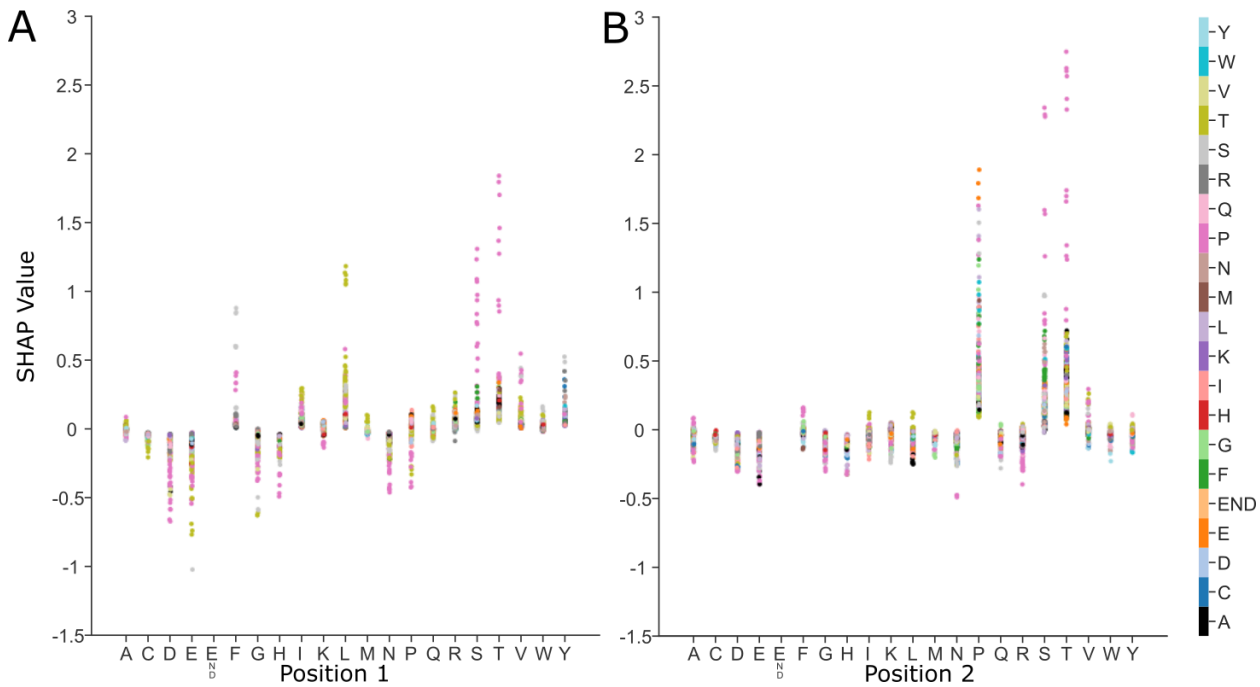


Figure 6. SHAP dependence plots for allele Mamu-A001 show how relationships between sequential amino acids contribute to binding. Each graph represents a pair of positions in the peptide, here positions one and two (A) and positions two and three (B). The x-axis lists each possible amino acid for that position and the y-axis shows the SHAP value. Each point represents a peptide with the listed amino acid at that position on the x-axis and the amino acid in the subsequent position is shown by color. This shows how the range of SHAP values for a particular amino acid at a specific location is reflective of the dependence of other amino acid positions.

For the CCS model, the interpositional interactions are different as they rely on the chemical interactions within the peptide itself, rather than interacting with a binding site. That is, interactions which promote peptide compaction will reduce CCS, and those that promote extension will generally increase CCS. To further determine how PoSHAP can reveal the important amino acids and positions generally, additional metrics were derived from the subset of significant amino acid pairs from the PoSHAP dependence analysis. All significant interactions (Bonferroni adj. p-value < 0.05) from the CCS model interpretation **(Supplemental Table 3)** were used to compute the absolute magnitude of the difference in SHAP value as a function of the distance between those residues **(Figure 7).** Absolute SHAP differences between interdependent amino acids were higher when the interaction was with the neighbor amino acid (ANOVA with Tukey's posthoc test p-value = 0.0426), or distant amino acids (distance 7-9, ANOVA with Tukey's posthoc adjusted p-value = 0.001) versus intermediate amino acids (distance 2-6) (**Figure 7A**). This suggests that amino acids interact more strongly with their neighbors because the R groups are adjacent, and have stronger interactions with those further along the peptide because of the flexibility of the chain, but interactions at an intermediate distance have a lesser magnitude of effect.

Additionally, there are interesting differences in the interactions of the amino acid among the significant set of interactions **(Figure 7B).** All significant interactions from the CCS data (**Supplemental Table 3**, Bonferroni adj. p-value < 0.05) within the peptides were grouped by the expected interaction type occurring between the amino acids based off of the chemistry of the R groups. Interestingly, this analysis revealed that generally attractive molecular interactions, including charge attraction and polar interactions on average decreased predicted CCS while generally

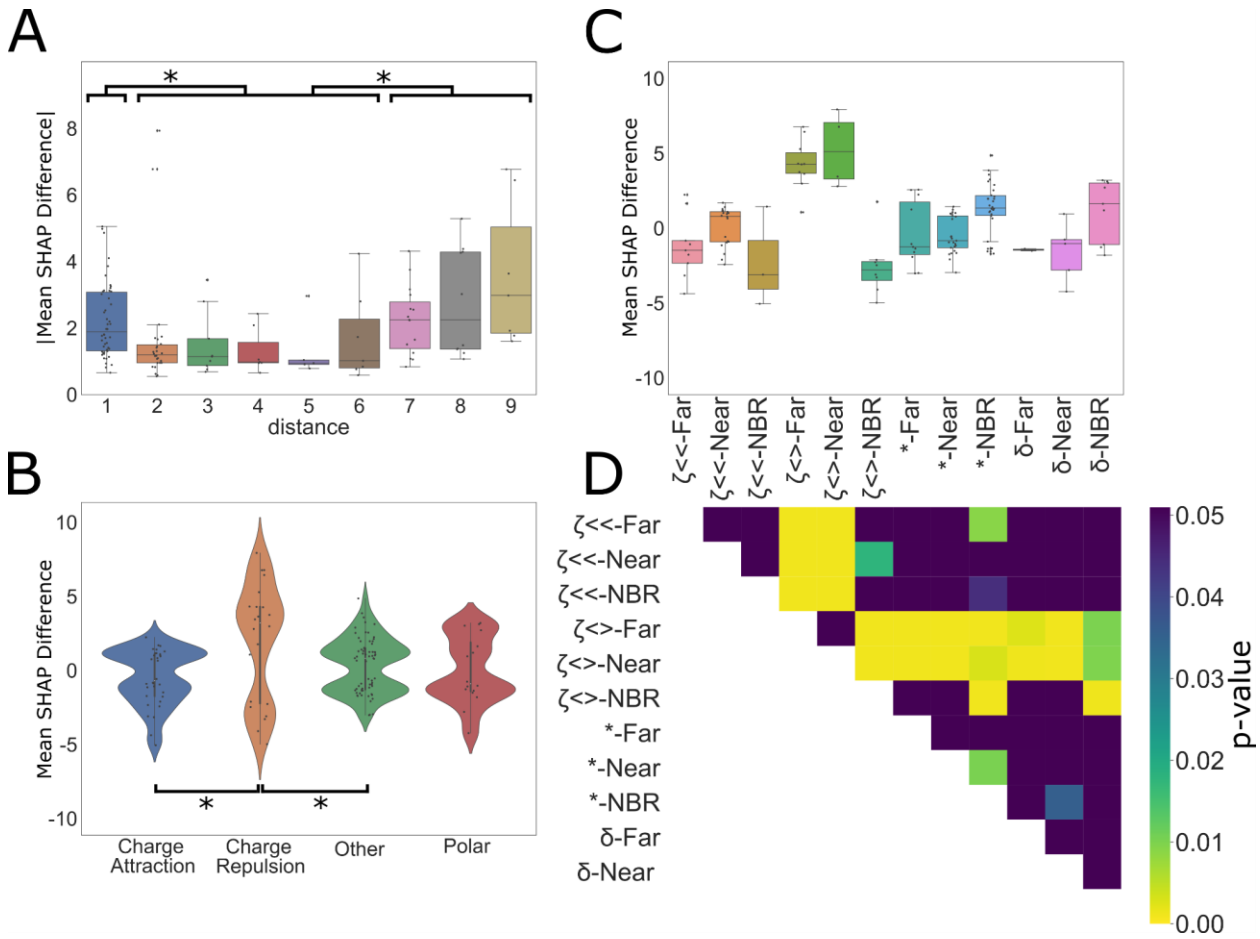


Figure 7. Dependence analysis of CCS model. (A) Significant (Bonferroni corr. P-value < 0.05) values were taken from the interpositional dependence analysis and the difference in the mean between the interdependent amino acids SHAP values and the remaining amino acids at each compared position pair were grouped based on the distance between the dependent interaction, **(B)** the category of interaction, or **(C)** distance and interaction category. Categories are labelled by the following for the combined bar plot and heatmap: ζ<< = charge attraction, ζ<> = charge repulsion, * = other, and δ = polar. For the distance analysis, interactions were grouped into three categories, neighboring (distance = 1), near (distance = 2, 3, 4, 5,6), and far (distance = 7, 8, 9). * indicates significance (ANOVA with Tukey's post hoc test p-value < 0.05). For the interaction categories in **(B)** and **(C)**, each interaction was grouped by the expected type of interaction between the two amino acids. Significant differences between interaction types are noted by the pairing by lines (ANOVA with Tukey's post hoc test p-value < 0.05). **(D)** Significant differences between combined categories are illustrated by the heatmap where significant values (ANOVA with Tukey's post hoc test p-value < 0.05) are designated by colors other than purple. Exact p-values for each are provided in **Supplemental Table 4.**

repulsive molecular interactions, including charge repulsion and “other” interactions (likely steric interactions or interactions between the termini) increased predicted CCS. Notably, there were very few significant hydrophobic interactions. This may reflect that hydrophobic interactions between amino acids in a peptide act to minimize contact with a polar solvent, rather than acting as an attractive force itself. Peptides lose polar solvent (water) during the electrospray process, which may prevent significant hydrophobic interactions, which might contradict prior work (64).

Though it is evident that the mean of each interaction type corresponds to the expected impact those interactions would have on CCS, each of the interaction dependence plots are bimodal, with some interactions increasing CCS and some decreasing it. To dissect this observation further, we combined the two methods of splitting the data to see if the bimodality of interaction types would be resolved by distance (**Figure 7C**). Though definitive conclusions cannot be made for most categories, likely due to the ever decreasing sample size by splitting, of note is the difference between neighboring charge repulsion and non-neighboring charge repulsion. Neighboring charge repulsion seems to decrease CCS while distant charge repulsion increases CCS (see adjusted p-value from Tukey’s posthoc test in **Figure 7D**). When distant, charge repulsion makes intuitive sense as the amino acids are forced apart, linearizing the peptide and increasing the surface area. When neighboring, it is possible that the repulsion causes a kink in the linear peptide, decreasing the cross section. Overall, these analyses demonstrate that the models were able to learn fundamental chemical properties of the amino acids and through PoSHAP analysis we were able to uncover them.

Finally, to ask if the absolute positions of amino acids in the peptide are relevant for the interaction, the data was split into 8, 9, or 10mers before analysis (**Supplemental Figure 11**). This revealed that there may be interactions between the termini, but this effect may be difficult to observe because there are significantly fewer 8mers and 9mers in the CCS dataset.

PoSHAP results are model-dependent

PoSHAP uses the SHAP KernelExplainer method, which is based on Local interpretable model-agnostic explanations (LIME). Using the general KernelExplanner method enables direct comparison of interpretations produced by different models trained from the same data. To ask whether PoSHAP interpretation changes based on the model used, the CCS data was used to train XGboost or ExtraTrees models. Surprisingly, the XGboost model performed better than the LSTM model with regard to MSE and spearman rho between true and predicted values in the test set (**Figure 8A**). ExtraTrees was slightly worse than the other two models. The model interpretation heatmaps from PoSHAP were similar between the LSTM and XGboost, but the interpretation from the ExtraTrees model was missing the high average SHAP due to n-terminal histidine or arginine (**Figure 8B**). Even though XGboost produced a similar PoSHAP heatmap, the interpositional dependence with regard to distance (**Figure 8C**) and chemical interactions (**Figure 8D**) were muted. This shows that the choice of model is important for revealing positional interactions.

Given the dependence of the model interpretation results on the model used, the same model architecture trained with different parameters might result in different model interpretation.

Models for each of the three tasks mentioned here were retrained with different hyperparameters including the “RMS prop” optimizer instead of Adam. Each model produces similar or better prediction performance compared to the initial version, and the model interpretation by PoSHAP was almost identical to the previous results in all three cases (**Supplementary figures 12, 13, 14**). This suggests that the model architecture drives the differences in interpretation, not the model training process.

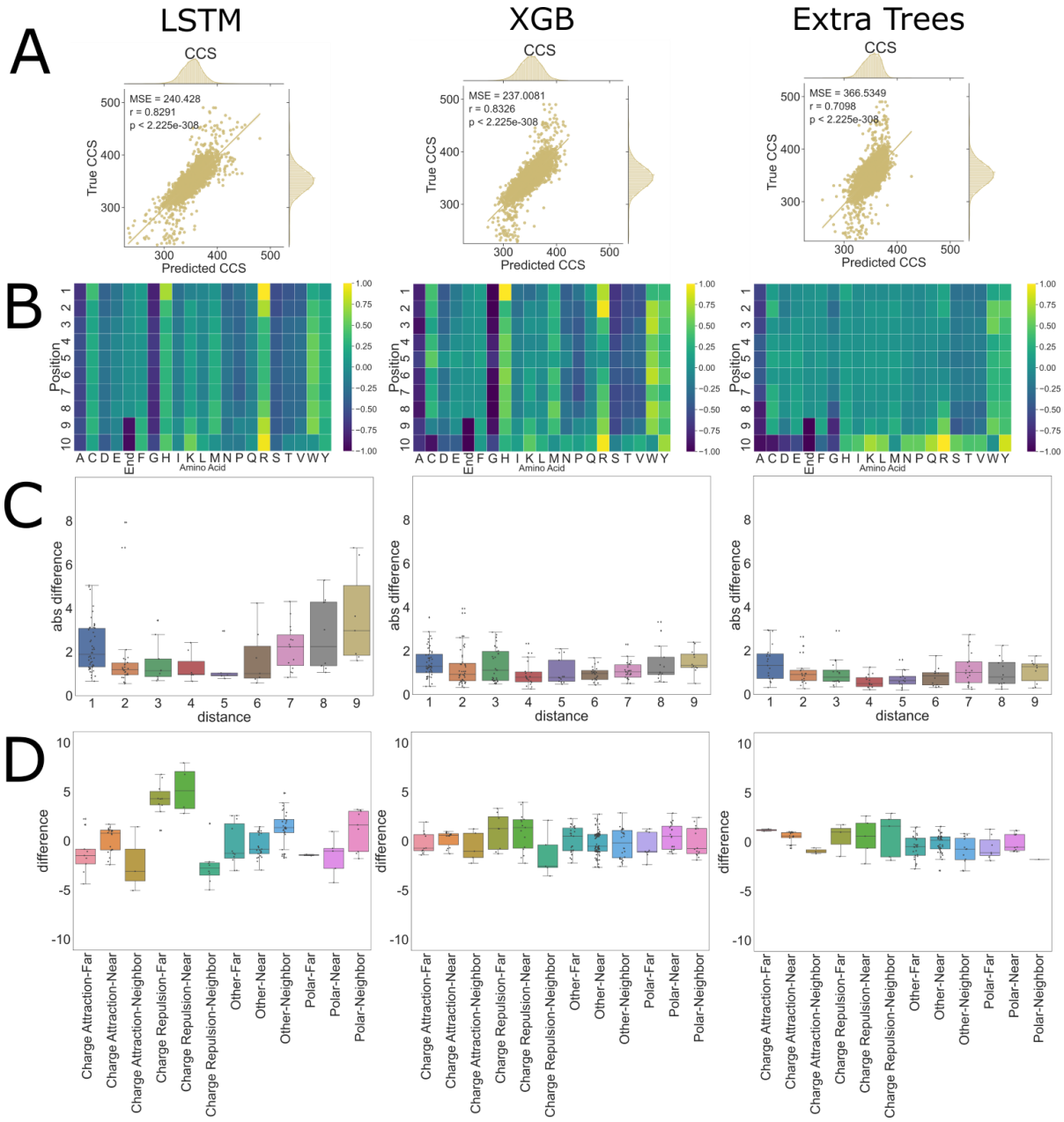


Figure 8. CCS PoSHAP of Various Machine Learning Models. PoSHAP analysis was performed on two additional machine learning models, Extra Trees, and Extreme Gradient Boosting (XGB). Predictions were plotted against experimental values and the Mean Squared Error and r values are reported for each model (A). PoSHAP heatmaps were created for each model, standardized by the highest value in each heatmap (B), illustrating an increase in model complexity as more sophisticated models are used. Dependence analysis was performed on each model and the significant interactions are plotted by distance (C) and by combined distance and interaction type (D).

Discussion

Here we demonstrate the concept of PoSHAP analysis to interpret machine learning models trained from inputs of biological sequences. We show how PoSHAP can reveal amino acid motifs that influence peptide MHC I binding or CCS. We further describe how PoSHAP enables understanding of interpositional dependence of amino acids that result in high MHC I affinity. Finally, we show how PoSHAP reveals the chemical interactions within peptides that alter their CCS. Overall, the three modeling examples laid out herein serve as a tutorial for PoSHAP interpretation of almost any model trained from almost any biological sequence.

Although there are many effective neural network models for biological sequences, there are a dearth of methods to understand those models. Thus, PoSHAP fills a gap in the biological machine learning community. Prior studies have used sequence logos from the top predictions (65), but this method doesn't ask the model what was learned and instead is observed-sequence centric. Another approach used by DeepLigand (29) is to apply Sufficient Input Subset (SIS) analysis (66), which attempts to reduce inputs to the minimal values required for prediction. While useful in many contexts, SIS is only amenable to classification models and does not provide contribution values for each input. A third approach is to create an interpretable model through the construction of the model itself. One example of this is by using attention mechanism based models, such as what has been done with ACME (32). However, this method of model interpretation limits the architecture of the model.

There are several benefits of PoSHAP over competing methods. First, PoSHAP determines important residues despite biases in the frequencies of amino acids (**Figure 5, Supplementary Figures 9 and 10**). PoSHAP is also applicable to any model trained from sequential data (**Figure 8**), and enables dissection of interpositional dependencies (**Figures 6 and 7**). Finally, we include a clearly explained jupyter notebook on Github that will take any model and dataset and perform PoSHAP analysis.

Altogether the advances described herein are likely to find widespread use for interpreting models trained from biological sequences, including models not covered here such as those to predict tandem mass spectra (reviewed in (33)).

Author Contributions

Conceptualization, QD, JGM; Methodology, QD, JGM; Software, QD, JGM; Validation, QD, JGM; Formal Analysis, QD, JGM; Investigation, QD, JGM; Resources, JGM; Data Curation, QD, JGM; Writing - Original Draft, QD, JGM; Writing - Reviewing and Editing, QD, JGM; Visualization, QD, JGM; Supervision, JGM; Project Administration, JGM; Funding Acquisition, JGM.

Acknowledgements

This research was completed in part with computational resources and technical support provided by the Research Computing Center at the Medical College of Wisconsin. We thank

Dasom Hwang for help with graphic design. This work was supported by the NIH (T15LM007359 and R35GM142502).

The authors declare that they have no conflict of interest.

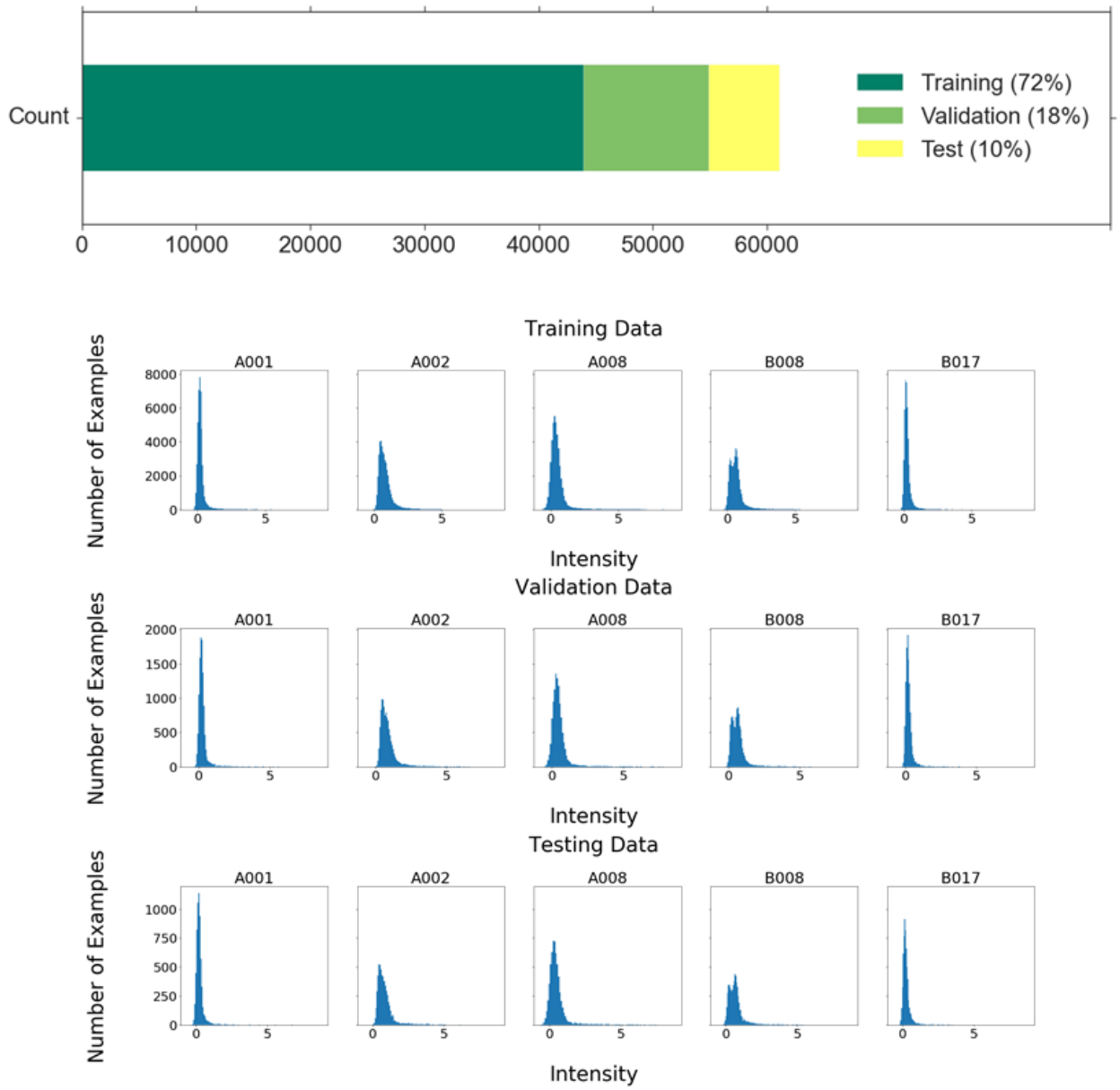
References

1. Wüthrich K. Protein structure determination in solution by NMR spectroscopy. J Biol Chem. 1990 Dec 25;265(36):22059–62.
2. Developments, applications, and prospects of cryo-electron microscopy - Benjin - 2020 - Protein Science - Wiley Online Library [Internet]. [cited 2021 Mar 2]. Available from: https://onlinelibrary.wiley.com/doi/10.1002/pro.3805
3. Protein crystallography from the perspective of technology developments: Crystallography Reviews: Vol 21, No 1-2 [Internet]. [cited 2021 Mar 2]. Available from: https://www.tandfonline.com/doi/abs/10.1080/0889311X.2014.973868
4. Huttlin EL, Ting L, Bruckner RJ, Gebreab F, Gygi MP, Szpyt J, et al. The BioPlex Network: A Systematic Exploration of the Human Interactome. Cell. 2015 Jul 16;162(2):425–40.
5. Lee J, Freddolino PL, Zhang Y. Ab Initio Protein Structure Prediction. In: J. Rigden D, editor. From Protein Structure to Function with Bioinformatics [Internet]. Dordrecht: Springer Netherlands; 2017 [cited 2021 Feb 26]. p. 3–35. Available from: https://doi.org/10.1007/978-94-024-1069-3_1
6. Kuhlman B, Bradley P. Advances in protein structure prediction and design. Nat Rev Mol Cell Biol. 2019 Nov;20(11):681–97.
7. Senior AW, Evans R, Jumper J, Kirkpatrick J, Sifre L, Green T, et al. Improved protein structure prediction using potentials from deep learning. Nature. 2020 Jan;577(7792):706–10.
8. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015 May;521(7553):436–44.
9. Petritis K, Kangas LJ, Ferguson PL, Anderson GA, Paša-Tolić L, Lipton MS, et al. Use of Artificial Neural Networks for the Accurate Prediction of Peptide Liquid Chromatography Elution Times in Proteome Analyses. Anal Chem. 2003 Mar 1;75(5):1039–48.
10. Fukushima K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol Cybern. 1980 Apr;36(4):193–202.
11. Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature. 1986 Oct;323(6088):533–6.
12. Young T, Hazarika D, Poria S, Cambria E. Recent Trends in Deep Learning Based Natural Language Processing [Review Article]. IEEE Comput Intell Mag. 2018;13(3):55–75.
13. Jurtz VI, Johansen AR, Nielsen M, Almagro Armenteros JJ, Nielsen H, Sønderby CK, et al. An introduction to deep learning on biological sequence data: examples and solutions. Bioinformatics. 2017 Nov 15;33(22):3685–90.
14. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997 Nov 15;9(8):1735–80.
15. Arras L, Arjona-Medina J, Widrich M, Montavon G, Gillhofer M, Müller K-R, et al. Explaining and Interpreting LSTMs. In: Samek W, Montavon G, Vedaldi A, Hansen LK, Müller K-R, editors. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning [Internet]. Cham: Springer International Publishing; 2019 [cited 2021 Feb 26]. p. 211–38. (Lecture Notes in Computer Science). Available from: https://doi.org/10.1007/978-3-030-28954-6_11
16. Bahdanau D, Cho K, Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate. ArXiv14090473 Cs Stat [Internet]. 2016 May 19 [cited 2021 Jun 16]; Available from: http://arxiv.org/abs/1409.0473

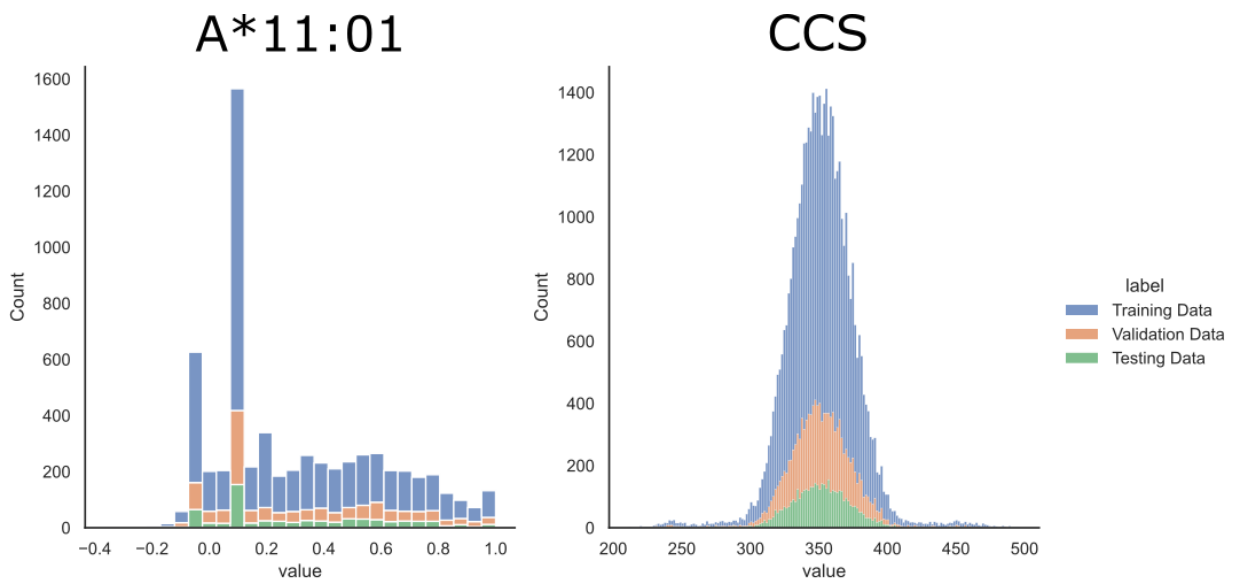
17. Breiman L. Random Forests. Mach Learn. 2001 Oct 1;45(1):5–32.
18. Lundberg SM, Lee S-I. A unified approach to interpreting model predictions. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc.; 2017. p. 4768–77. (NIPS'17).
19. Lundberg SM, Nair B, Vavilala MS, Horibe M, Eisses MJ, Adams T, et al. Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. Nat Biomed Eng. 2018 Oct;2(10):749–60.
20. Cresswell P, Bangia N, Dick T, Diedrich G. The nature of the MHC class I peptide loading complex. Immunol Rev. 1999;172(1):21–8.
21. Neefjes J, Jongsma MLM, Paul P, Bakke O. Towards a systems understanding of MHC class I and MHC class II antigen presentation. Nat Rev Immunol. 2011 Dec;11(12):823–36.
22. Matzaraki V, Kumar V, Wijmenga C, Zhernakova A. The MHC locus and genetic susceptibility to autoimmune and infectious diseases. Genome Biol. 2017 Apr 27;18(1):76.
23. Sijts EJAM, Kloetzel P-M. The role of the proteasome in the generation of MHC class I ligands and immune responses. Cell Mol Life Sci. 2011 May;68(9):1491–502.
24. Burrows SR, Rossjohn J, McCluskey J. Have we cut ourselves too short in mapping CTL epitopes? Trends Immunol. 2006 Jan;27(1):11–6.
25. Bonnal SC, López-Oreja I, Valcárcel J. Roles and mechanisms of alternative splicing in cancer — implications for care. Nat Rev Clin Oncol. 2020 Aug;17(8):457–74.
26. Nielsen M, Andreatta M, Peters B, Buus S. Immunoinformatics: Predicting Peptide–MHC Binding. Annu Rev Biomed Data Sci. 2020;3(1):191–215.
27. Antunes DA, Abella JR, Devaurs D, Rigo MM, Kavraki LE. Structure-based Methods for Binding Mode and Binding Affinity Prediction for Peptide-MHC Complexes. Curr Top Med Chem. 2018 Oct 1;18(26):2239–55.
28. O'Donnell TJ, Rubinsteyn A, Bonsack M, Riemer AB, Laserson U, Hammerbacher J. MHCflurry: Open-Source Class I MHC Binding Affinity Prediction. Cell Syst. 2018 Jul 25;7(1):129-132.e4.
29. Zeng H, Gifford DK. DeepLigand: accurate prediction of MHC class I ligands using peptide embedding. Bioinformatics. 2019 Jul 15;35(14):i278–83.
30. Liu Z, Cui Y, Xiong Z, Nasiri A, Zhang A, Hu J. DeepSeqPan, a novel deep convolutional neural network model for pan-specific class I HLA-peptide binding affinity prediction. Sci Rep. 2019 Jan 28;9(1):794.
31. Jin J, Liu Z, Nasiri A, Cui Y, Louis S-Y, Zhang A, et al. Deep learning pan-specific model for interpretable MHC-I peptide binding prediction with improved attention mechanism. Proteins Struct Funct Bioinforma [Internet]. [cited 2021 May 25];n/a(n/a). Available from: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.26065
32. Hu Y, Wang Z, Hu H, Wan F, Chen L, Xiong Y, et al. ACME: pan-specific peptide–MHC class I binding prediction through attention-based deep neural networks. Bioinformatics. 2019 Dec 1;35(23):4946–54.
33. Meyer JG. Deep learning neural network tools for proteomics. Cell Rep Methods. 2021 May 17;100003.
34. Lin Y-M, Chen C-T, Chang J-M. MS2CNN: predicting MS/MS spectrum based on protein sequence using deep convolutional neural networks. BMC Genomics. 2019 Dec 24;20(9):906.
35. Guan S, Moran MF, Ma B. Prediction of LC-MS/MS Properties of Peptides from Sequence by Deep Learning*[S]. Mol Cell Proteomics. 2019 Oct 1;18(10):2099–107.
36. Liu K, Li S, Wang L, Ye Y, Tang H. Full-Spectrum Prediction of Peptides Tandem Mass Spectra using Deep Neural Network. Anal Chem. 2020 Mar 17;92(6):4275–83.
37. Yang Y, Liu X, Shen C, Lin Y, Yang P, Qiao L. In silico spectral libraries by deep learning facilitate data-independent acquisition proteomics. Nat Commun. 2020 Jan 9;11(1):146.
38. Wen B, Li K, Zhang Y, Zhang B. Cancer neoantigen prioritization through sensitive and

reliable proteogenomics analysis. Nat Commun. 2020 Apr 9;11(1):1759.
39. Deep learning the collisional cross sections of the peptide universe from a million experimental values | Nature Communications [Internet]. [cited 2021 Feb 26]. Available from: https://www.nature.com/articles/s41467-021-21352-8
40. O'Donnell TJ, Rubinsteyn A, Laserson U. MHCflurry 2.0: Improved Pan-Allele Prediction of MHC Class I-Presented Peptides by Incorporating Antigen Processing. Cell Syst. 2020 Jul 22;11(1):42-48.e7.
41. Shao XM, Bhattacharya R, Huang J, Sivakumar IKA, Tokheim C, Zheng L, et al. High-Throughput Prediction of MHC Class I and II Neoantigens with MHCnuggets. Cancer Immunol Res. 2020 Mar 1;8(3):396–408.
42. Wilson EA, Krishna S, Anderson KS. A Random Forest based approach to MHC class I epitope prediction and analysis. J Immunol. 2018 May 1;200(1 Supplement):99.11-99.11.
43. Boehm KM, Bhinder B, Raja VJ, Dephoure N, Elemento O. Predicting peptide presentation by major histocompatibility complex class I: an improved machine learning approach to the immunopeptidome. BMC Bioinformatics. 2019 Jan 5;20(1):7.
44. Devlin J, Chang M-W, Lee K, Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv181004805 Cs [Internet]. 2019 May 24 [cited 2021 Sep 14]; Available from: http://arxiv.org/abs/1810.04805
45. Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, et al. Language Models are Few-Shot Learners. ArXiv200514165 Cs [Internet]. 2020 Jul 22 [cited 2021 Sep 14]; Available from: http://arxiv.org/abs/2005.14165
46. Vig J, Madani A, Varshney LR, Xiong C, Socher R, Rajani NF. BERTology Meets Biology: Interpreting Attention in Protein Language Models. ArXiv200615222 Cs Q-Bio [Internet]. 2021 Mar 28 [cited 2021 Sep 14]; Available from: http://arxiv.org/abs/2006.15222
47. Haj AK, Breitbach ME, Baker DA, Mohns MS, Moreno GK, Wilson NA, et al. High-Throughput Identification of MHC Class I Binding Peptides Using an Ultradense Peptide Array. J Immunol Baltim Md 1950. 2020 Mar 15;204(6):1689–96.
48. Bento J, Saleiro P, Cruz AF, Figueiredo MAT, Bizarro P. TimeSHAP: Explaining Recurrent Models through Sequence Perturbations. Proc 27th ACM SIGKDD Conf Knowl Discov Data Min. 2021 Aug 14;2565–73.
49. Kokalj E, Škrlj B, Lavrač N, Pollak S, Robnik-Šikonja M. BERT meets Shapley: Extending SHAP Explanations to Transformer-based Classifiers. In: Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation [Internet]. Online: Association for Computational Linguistics; 2021 [cited 2021 Sep 14]. p. 16–21. Available from: https://aclanthology.org/2021.hackashop-1.3
50. Kim Y, Sidney J, Buus S, Sette A, Nielsen M, Peters B. Dataset size and composition impact the reliability of performance benchmarks for peptide-MHC binding predictions. BMC Bioinformatics. 2014 Jul 14;15:241.
51. Vita R, Overton JA, Greenbaum JA, Ponomarenko J, Clark JD, Cantrell JR, et al. The immune epitope database (IEDB) 3.0. Nucleic Acids Res. 2015 Jan;43(Database issue):D405-412.
52. Pearson H, Daouda T, Granados DP, Durette C, Bonneil E, Courcelles M, et al. MHC class I–associated peptides derive from selective regions of the human genome. J Clin Invest. 2016 Dec 1;126(12):4690–701.
53. Nielsen M, Lundegaard C, Worning P, Lauemøller SL, Lamberth K, Buus S, et al. Reliable prediction of T-cell epitopes using neural networks with novel sequence representations. Protein Sci Publ Protein Soc. 2003 May;12(5):1007–17.
54. Chollet, Fran\c{c}ois. Keras [Internet]. [cited 2021 Jan 12]. Available from: https://keras.io/
55. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. :19.
56. Kingma DP, Ba J. Adam: A Method for Stochastic Optimization. ArXiv14126980 Cs

[Internet]. 2017 Jan 29 [cited 2021 Jan 12]; Available from: http://arxiv.org/abs/1412.6980
57. Bergstra J, Yamins D, Cox DD. Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures. In: Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28. Atlanta, GA, USA: JMLR.org; 2013. p. I-115-I–123. (ICML'13).
58. Bergstra JS, Bardenet R, Bengio Y, Kégl B. Algorithms for Hyper-Parameter Optimization. :9.
59. Hunter JD. Matplotlib: A 2D Graphics Environment. Comput Sci Eng. 2007;9(3):90–5.
60. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. J Mach Learn Res. 2011;12:2825–30.
61. Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining [Internet]. New York, NY, USA: ACM; 2016. p. 785–94. (KDD '16). Available from: http://doi.acm.org/10.1145/2939672.2939785
62. Sidney J, Dzuris JL, Newman MJ, Johnson RP, Amitinder K, Walker CM, et al. Definition of the Mamu A*01 Peptide Binding Specificity: Application to the Identification of Wild-Type and Optimized Ligands from Simian Immunodeficiency Virus Regulatory Proteins. J Immunol. 2000 Dec 1;165(11):6387–99.
63. Yusim K, Korber BT, Brander C, Barouch D, de Boer R, Haynes BF, et al. HIV Molecular Immunology 2015 [Internet]. Los Alamos National Lab. (LANL), Los Alamos, NM (United States); 2016 Apr [cited 2021 Jun 15]. Report No.: LA-UR-16-22283. Available from: https://www.osti.gov/biblio/1248095
64. Barylyuk K, Balabin RM, Grünstein D, Kikkeri R, Frankevich V, Seeberger PH, et al. What Happens to Hydrophobic Interactions during Transfer from the Solution to the Gas Phase? The Case of Electrospray-Based Soft Ionization Methods. J Am Soc Mass Spectrom. 2011 May 10;22(7):1167–77.
65. Jensen KK, Andreatta M, Marcatili P, Buus S, Greenbaum JA, Yan Z, et al. Improved methods for predicting peptide binding affinity to MHC class II molecules. Immunology. 2018;154(3):394–406.
66. Carter B, Mueller J, Jain S, Gifford D. What made you do this? Understanding black-box decisions with sufficient input subsets. ArXiv181003805 Cs Stat [Internet]. 2019 Feb 8 [cited 2021 Mar 2]; Available from: http://arxiv.org/abs/1810.03805



Supplemental Figure 1. Details of the data distributions and splitting. The data was split into three subsets. Training data made up 72% of the overall data and was used directly to train the model. Validation data made up 18% of the overall data and was used to monitor overfitting. Test data made up 10% of the total data and was used to test the final model's performance. The intensity distributions for each data subset were plotted for each allele to ensure that each maintained the same distribution. Note the difference in y-axis scales.



Supplemental Figure 2. Details of the data distributions and splitting for A*11:01 and CCS. The data was split into three subsets. Training data made up 72% of the overall data and was used directly to train the model. Validation data made up 18% of the overall data and was used to monitor overfitting. Test data made up 10% of the total data and was used to test the final model's performance. The output distributions for each data subset were plotted for each allele to ensure that each maintained the same distribution.

A

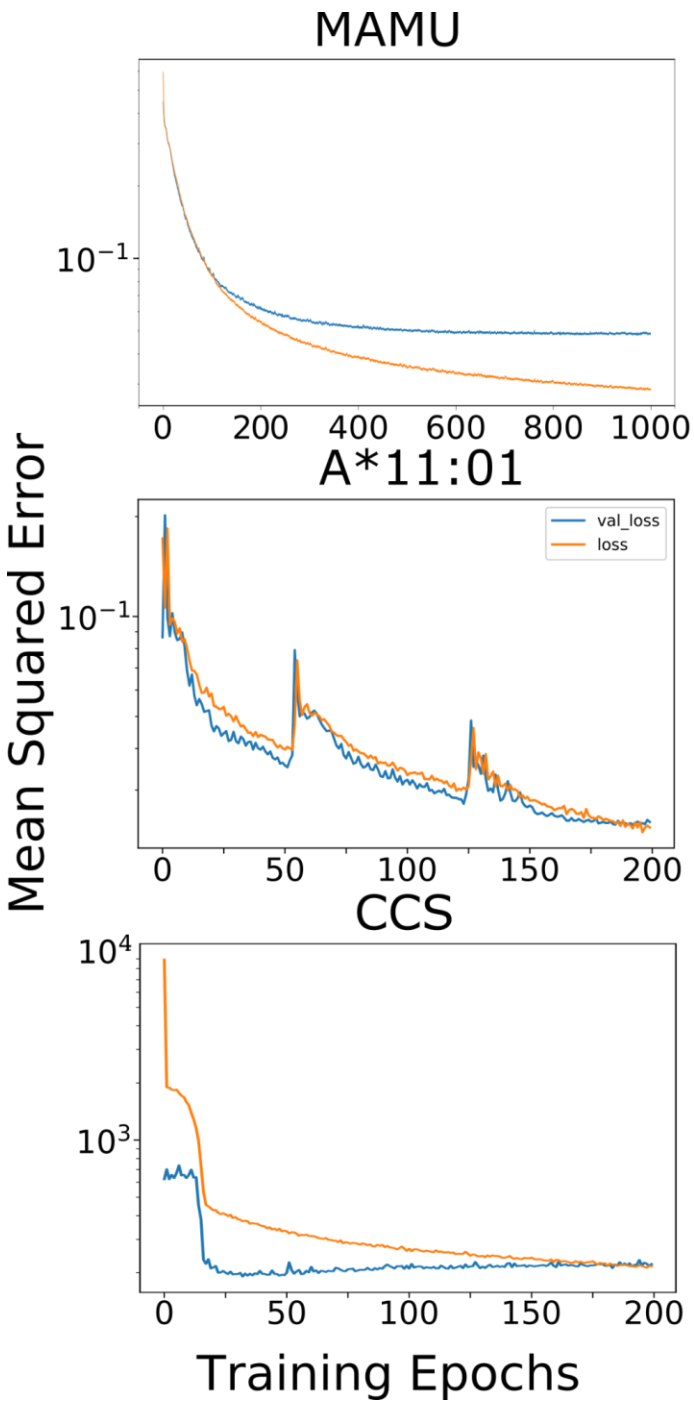
	MAMU		**A*11:01**		**CCS**	
Layer (type)	Output Shape	Param #	Output Shape	Param #	Output Shape	Param #
embedding (Embedding)	(None, 10, 50)	1,050	(None, 10, 50)	1,050	(None, 10, 50)	1,050
lstm (LSTM)	(None, 10, 128)	91,648	(None, 10, 128)	91,648	(None, 128)	91,648
dropout (Dropout)	(None, 10, 128)	0	(None, 10, 128)	0	(None, 128)	0
lstm_1 (LSTM)	(None, 128)	131,584	(None, 128)	131,584	n/a	n/a
dropout_1 (Dropout)	(None, 128)	0	(None, 128)	0	n/a	n/a
dense (Dense)	(None, 64)	8,256	(None, 64)	8,256	(None, 64)	8,256
leaky_re_lu (LeakyReLU)	(None, 64)	0	(None, 64)	0	n/a	n/a
dropout_2 (Dropout)	(None, 64)	0	(None, 64)	0	(None, 64)	0
dense_1 (Dense)	(None, 5)	325	(None, 1)	65	(None, 1)	65
Total params:		232,863		232,603		101,019
Trainable params:		232,863		232,603		101,019
Non-trainable params:		0		0		0

B

	MAMU	**A*11:01**	**CCS**
Dropout Layer 1	**0.44**	**0.53**	**0.53**
Dropout Layer 2	**0.2**	**0.56**	**n/a**
Dropout Layer 3	**0.05**	**0.8**	**0.09**
Learning Rate	**0.001***	**0.01**	**0.005**
Batch Size	**5000**	**5000**	**128**
Epochs	**1000**	**200***	**200***

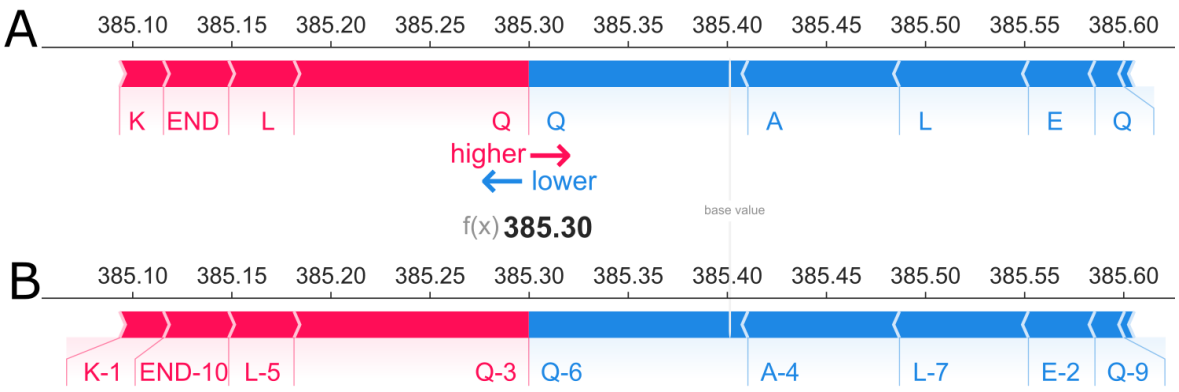
Supplemental Figure 3: Summary of LSTM model architecture. The architecture of the model consists of an embedding layer with 10 inputs with 21 dimensions each, representing each position of the peptide and each of the numeric representations of the possible amino acids and the end marker. (A) This is followed by a pair of LSTM and dropout layers, with the dropout ratios determined by a hyperparameter search. Following the LSTM layers are a dense layer, a leaky ReLU activation layer, a final dropout layer, and a final dense layer with five outputs, each representing the intensity of the corresponding allele. The model was trained with a batch size of 5000 for 1000 epochs. Hyperopt was

used to determine the ideal hyperparameters for the model using a tree of parzen estimators algorithm over 100 evaluations. (B) * indicates a hardcoded hyperparameter.

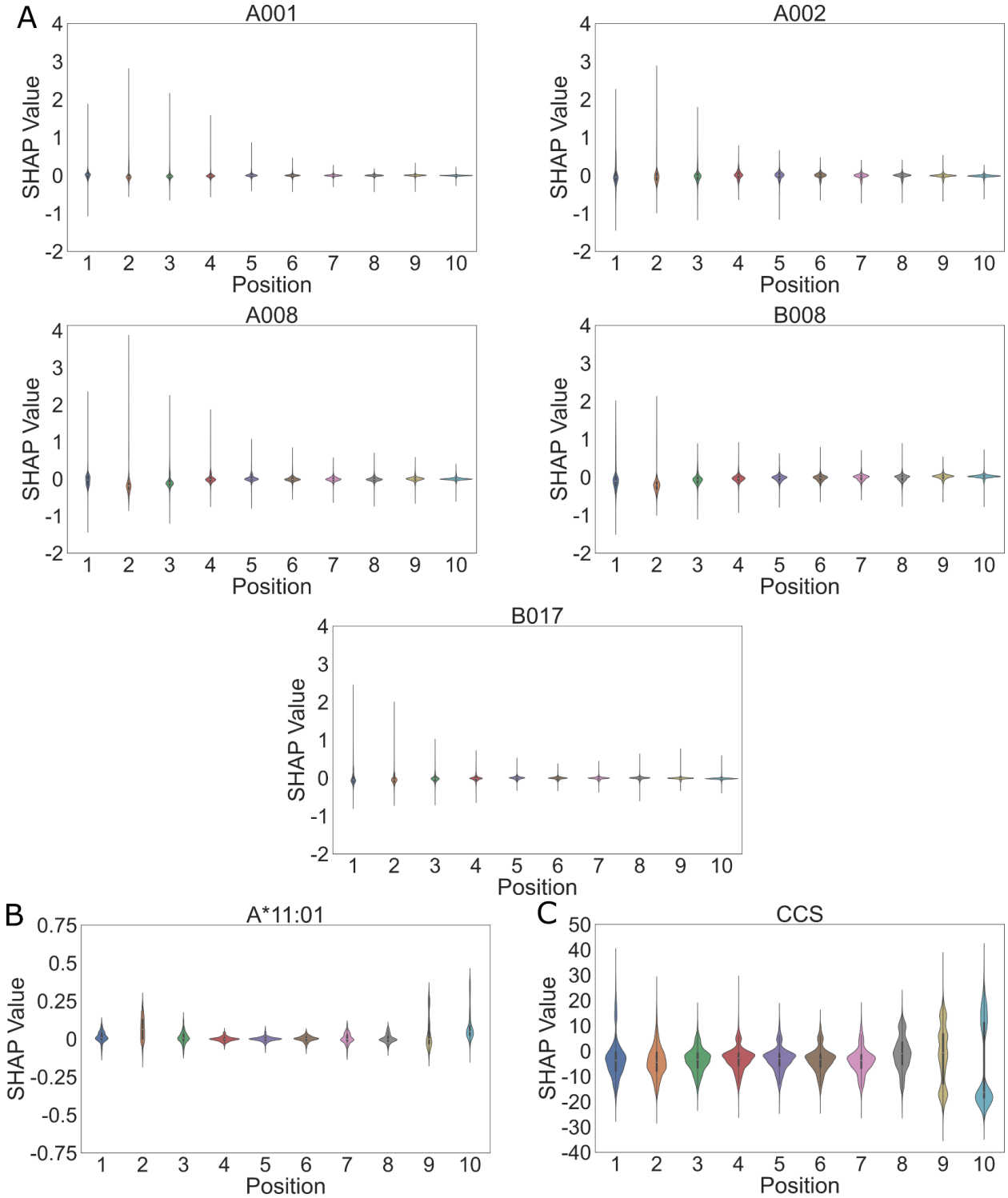


Supplemental Figure 4. Mean squared error loss over training. The models were trained for either 1000 or 200 epochs and the loss from mean squared error between predictions and true, known values were plotted for both the training data and the validation data. For the MAMU model, validation loss diverges from the test loss around epoch 175, indicating some amount of overfitting, however, the MSE of both

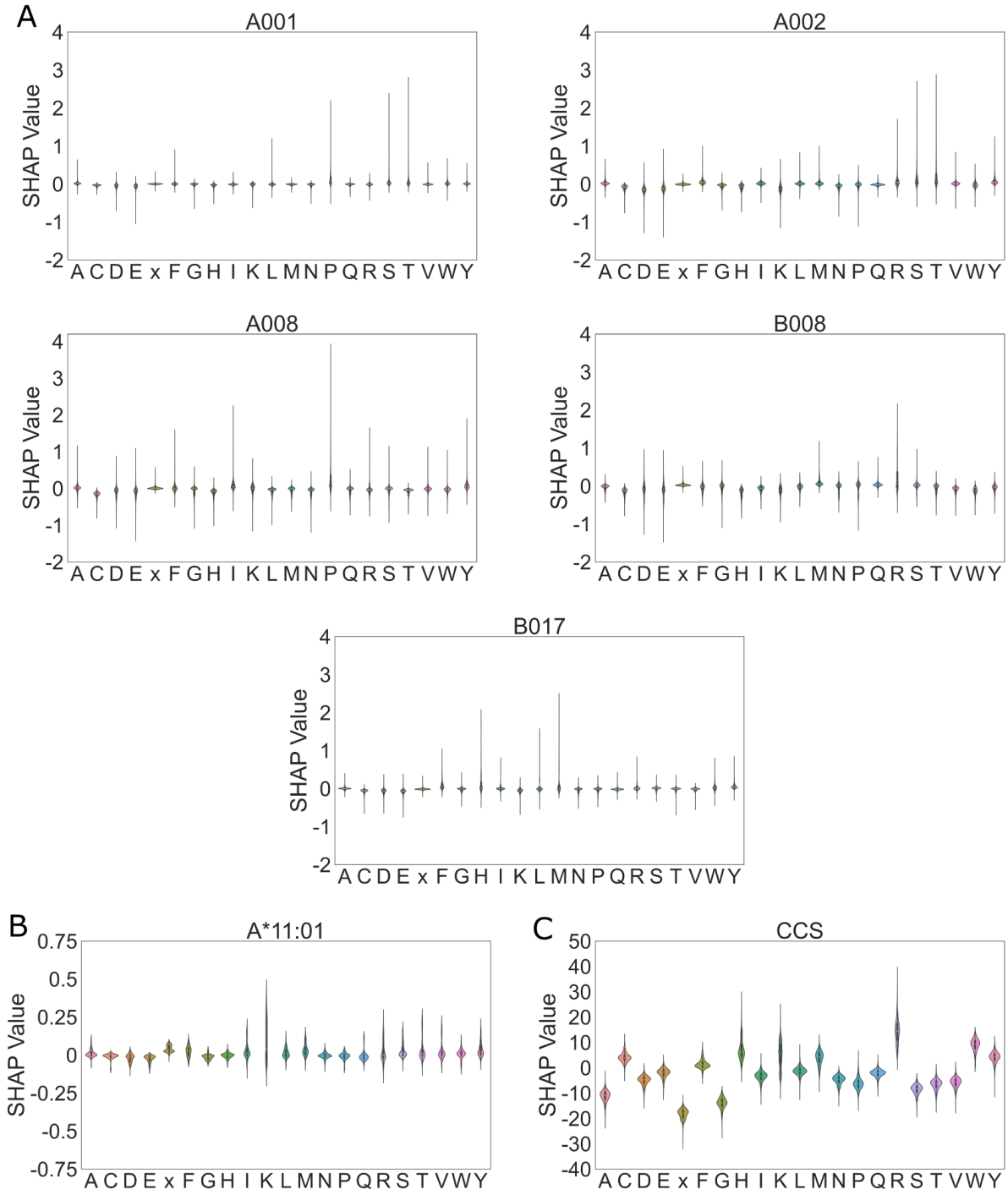
datasets continues to decrease as the model is trained over 1000 epochs. For the A*11:01 model, test and validation loss were similar, until around epoch 200 when the model was finished training. For the CCS dataset, the validation loss started and remained lower throughout the training.



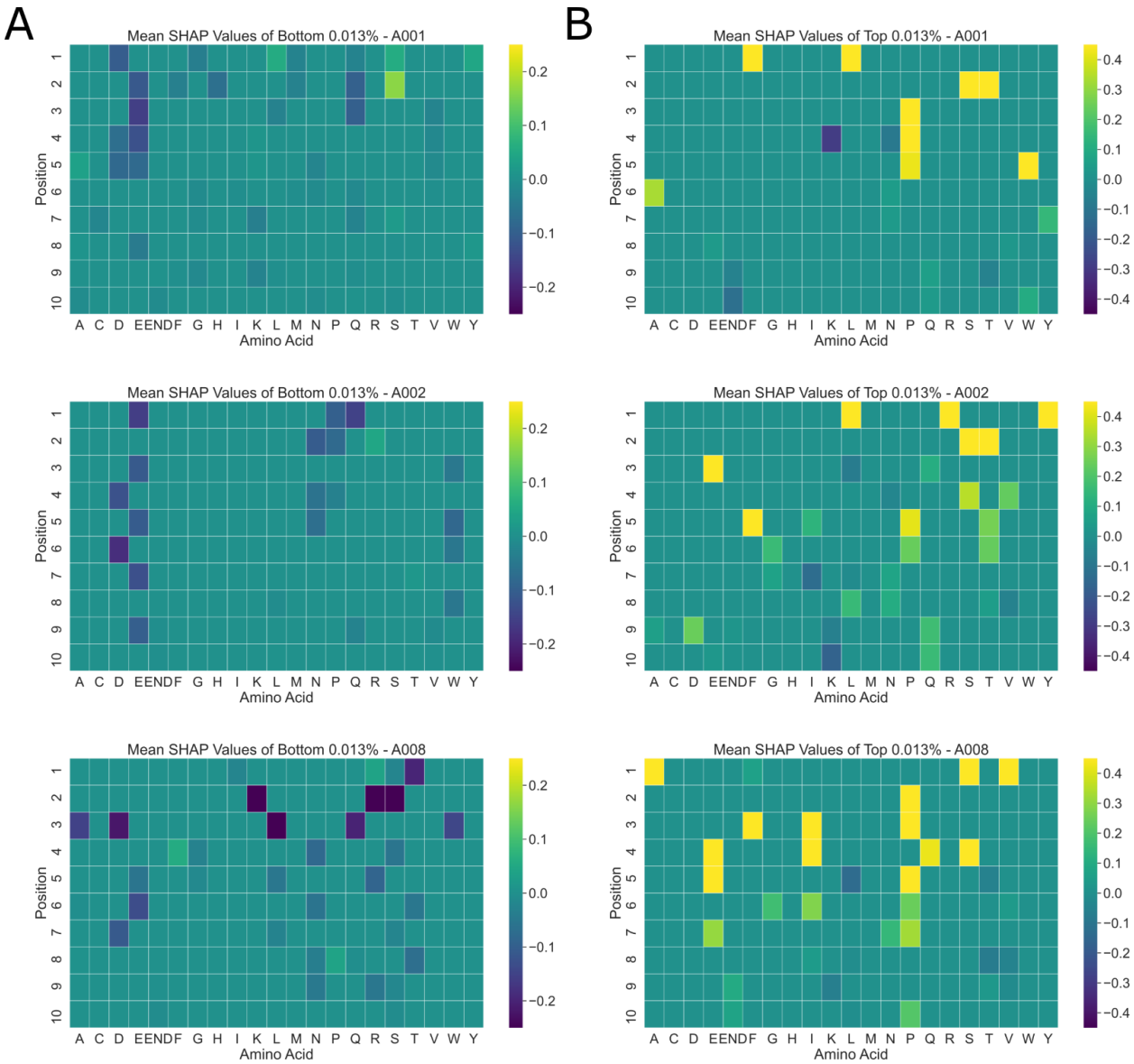
Supplemental Figure 5. SHAP Forceplots Demonstrating PoSHAP Indexing. Two forceplots were created with the SHAP forceplot method of the third peptide in the CCS testing set. (A) shows the plot with encoded inputs mapped to their amino acid. (B) shows the plot with the encoded inputs mapped to their amino acid and position. The addition of positional indexing removes the ambiguity of contributions, for example, glutamine having both a positive and a negative SHAP contribution to the prediction of the third peptide.



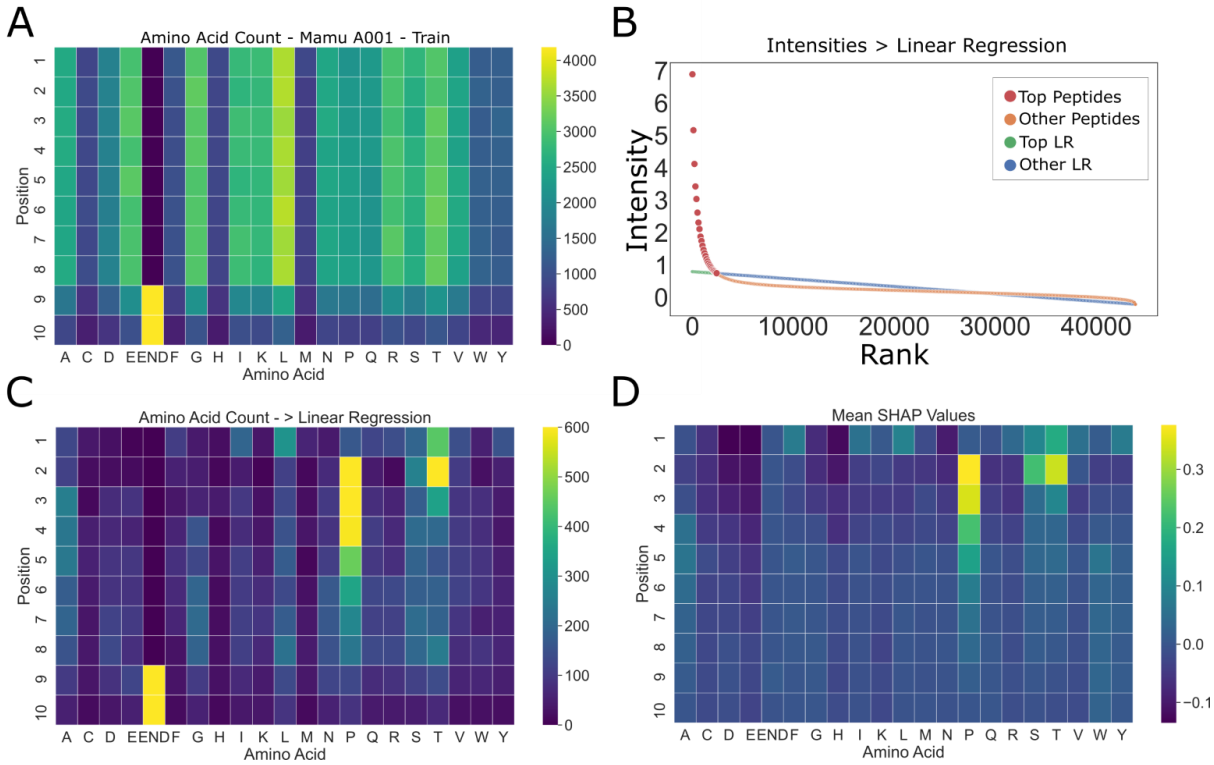
Supplemental Figure 6. SHAP value ranges by position. SHAP values were arranged by position in the peptides and their distributions were plotted as violin plots, with the quartile ranges and total range illustrated by the box and whisker plot within each. Each of the five modeled MAMU alleles (A), human MHC A*11:01 (B), and CCS (C) are displayed.



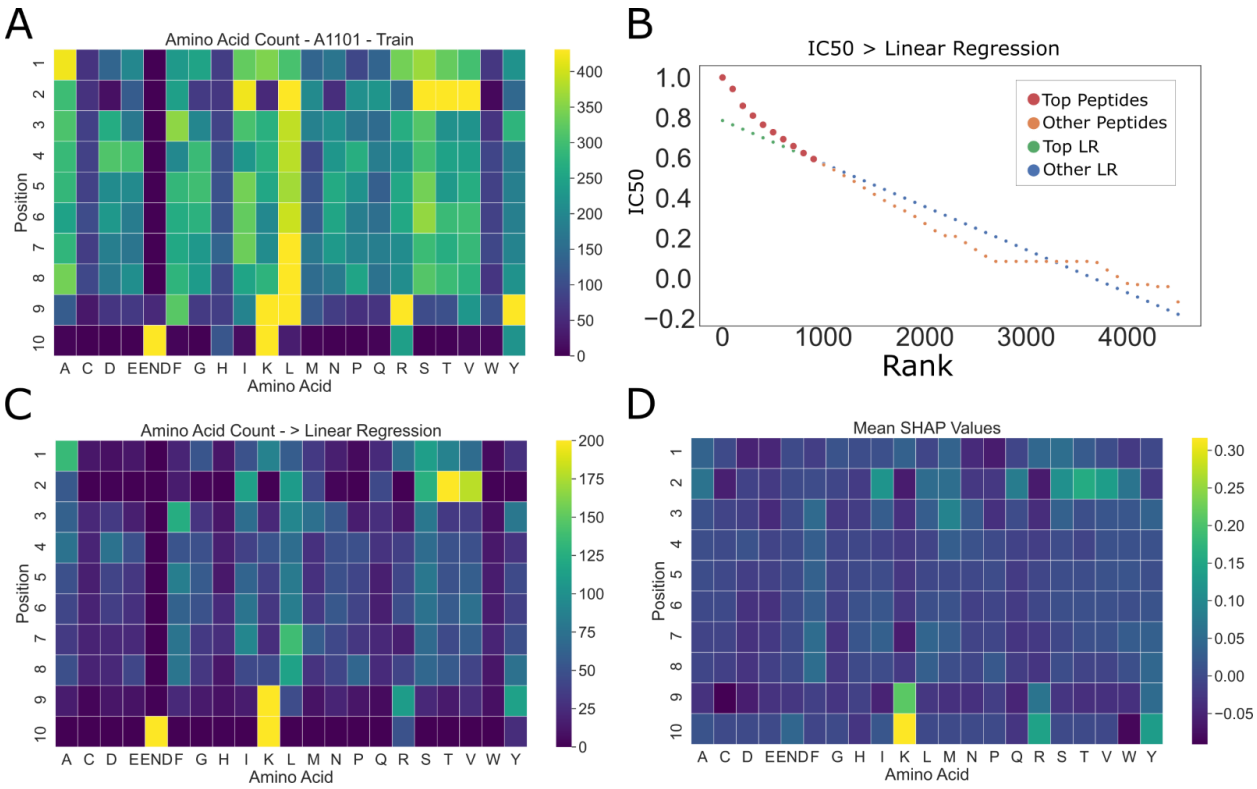
Supplemental Figure 7. SHAP value ranges by amino acid. SHAP values were arranged by amino acids across all positions in the peptides and their distributions were plotted as violin plots, with the quartile ranges and total range illustrated by the box and whisker plot within each. Each of the five modeled MAMU alleles (A), human MHC A*11:01 (B), and CCS (C) are displayed. End input is represented by x.



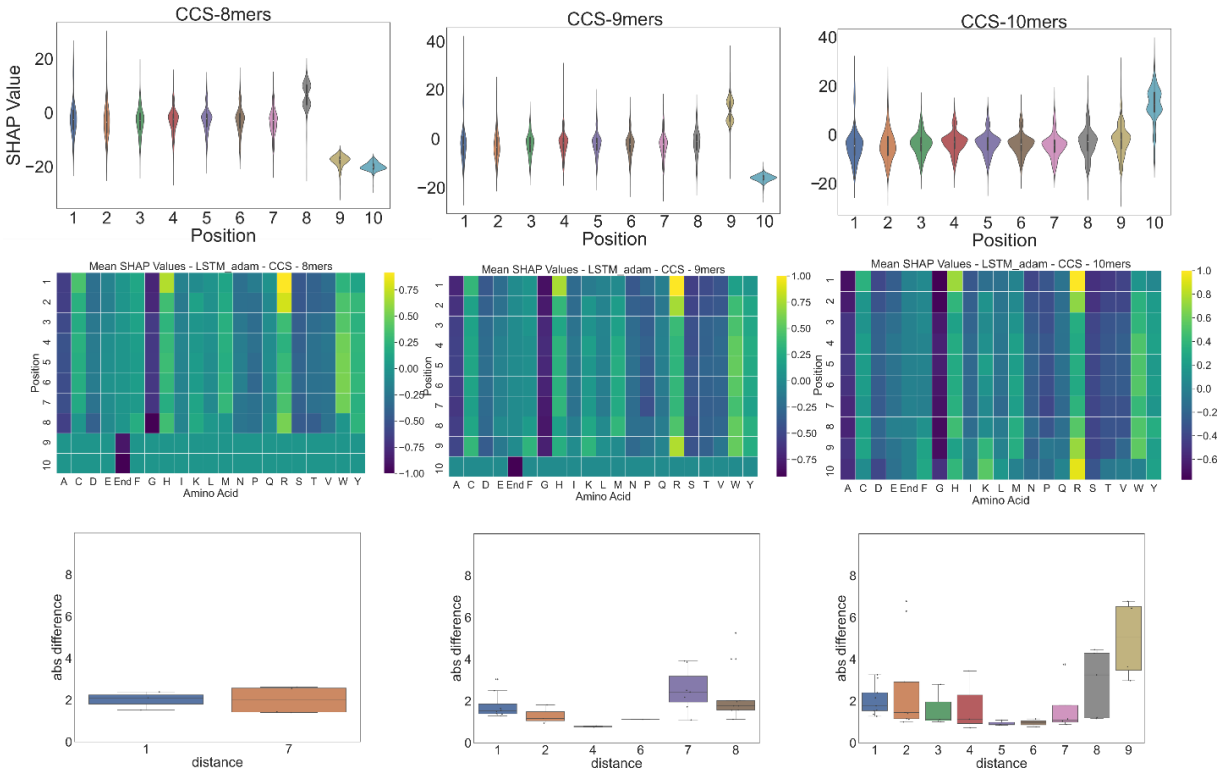
Supplemental Figure 8. Pooled PoSHAP for bottom and top predicted subsets of the data. The mean SHAP values for each amino acid at each position were calculated for the peptides with the bottom (A) or top (B) 0.013% predicted intensity (top 8 peptides) for the "A" Mamu alleles. Due to the small sample size, most of the amino acid positions have a value of zero. The positions with extreme values, however, illustrate important amino acids for prediction. Notably for A001 and A002, aspartic acid and glutamic acid contribute to low prediction along the peptide, suggesting charge may inhibit binding. For the top predictions, phenylalanine or leucine are important at the first position for both A001 and A008. A serine or threonine at position two is important for A001, A002, and A008. All alleles demonstrate the importance of a proline near the middle of the peptide.



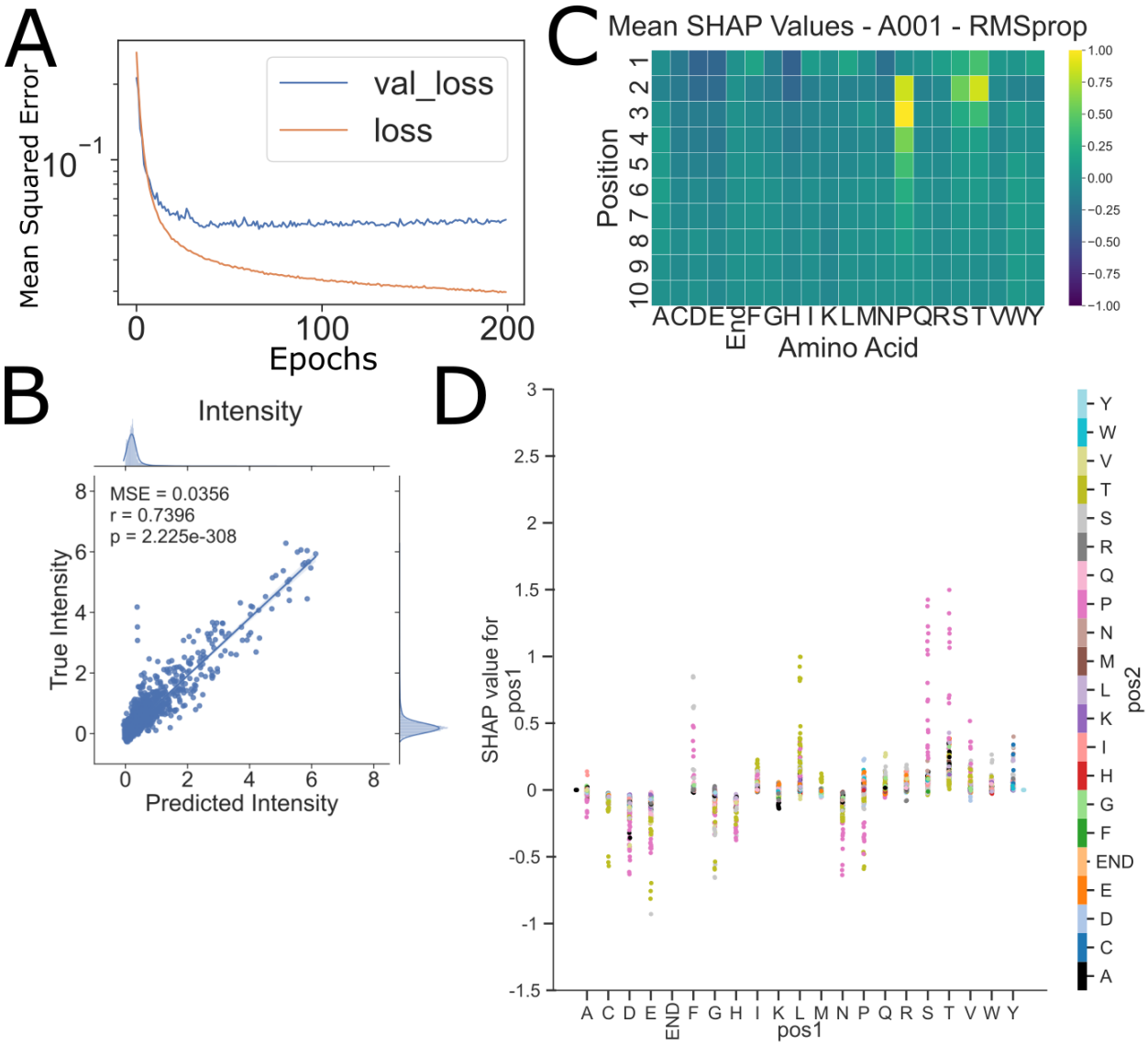
Supplemental Figure 9: Amino acid summary statistics differ from PoSHAP values for the A001 MAMU MHC I data. (A) Amino acid counts as a function of position for training data. (B) Procedure for picking the 'top peptides' with the highest CCS. Linear regression was performed on the peptides ranked by their actual CCS value. Any peptide that fell above the trendline and overall mean were defined as 'top peptides'. (C) Counts of amino acids for the top peptides were summarized in a heatmap. (D) Mean SHAP values across amino acids and positions from PoSHAP analysis. For the MAMU model, the amino acid frequencies of the input peptides show no obvious preference for amino acid position, but some amino acids are over-represented overall . The presence of the "end" token is more likely to be a high binder statistically (C), but the PoSHAP reveals that this end token is not the main determinant of binding (D).



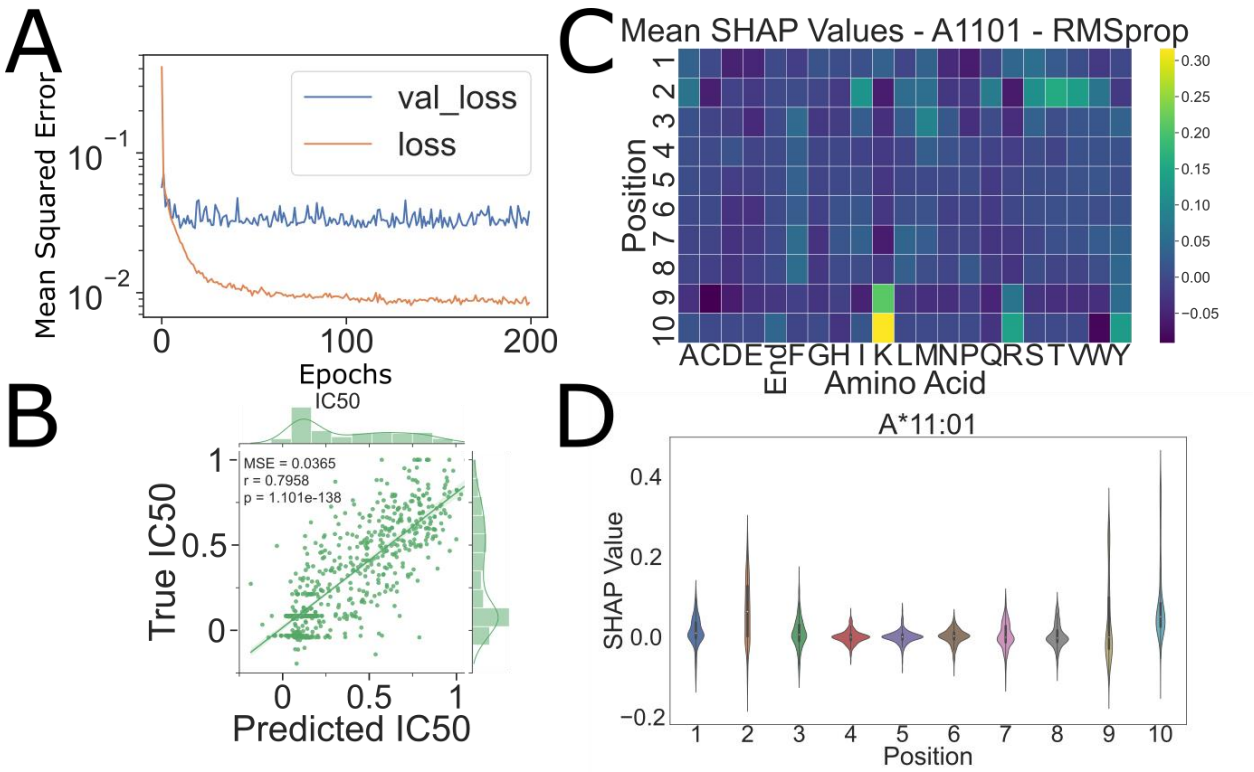
Supplemental Figure 10: Amino acid summary statistics differ from PoSHAP values for the human A1101 MHC I data. (A) Amino acid counts as a function of position for training data. The distribution of amino acids in this data. **(B)** Procedure for picking the 'top peptides' with the highest CCS. Linear regression was performed on the peptides ranked by their actual CCS value. Any peptide that fell above the trendline and overall mean were defined as 'top peptides'. **(C)** Counts of amino acids for the top peptides were summarized in a heatmap. **(D)** Mean SHAP values across amino acids and positions from PoSHAP analysis. There are clear differences between the summary statistics of top peptides **(C)** and PoSHAP heatmap **(D)**. For example, the end token is prominent in the summary statistics absent from the PoSHAP interpretation. Also, the preference for S/T/V at position two is tempered according to PoSHAP, but would be determined to be very important by the summary statistics.



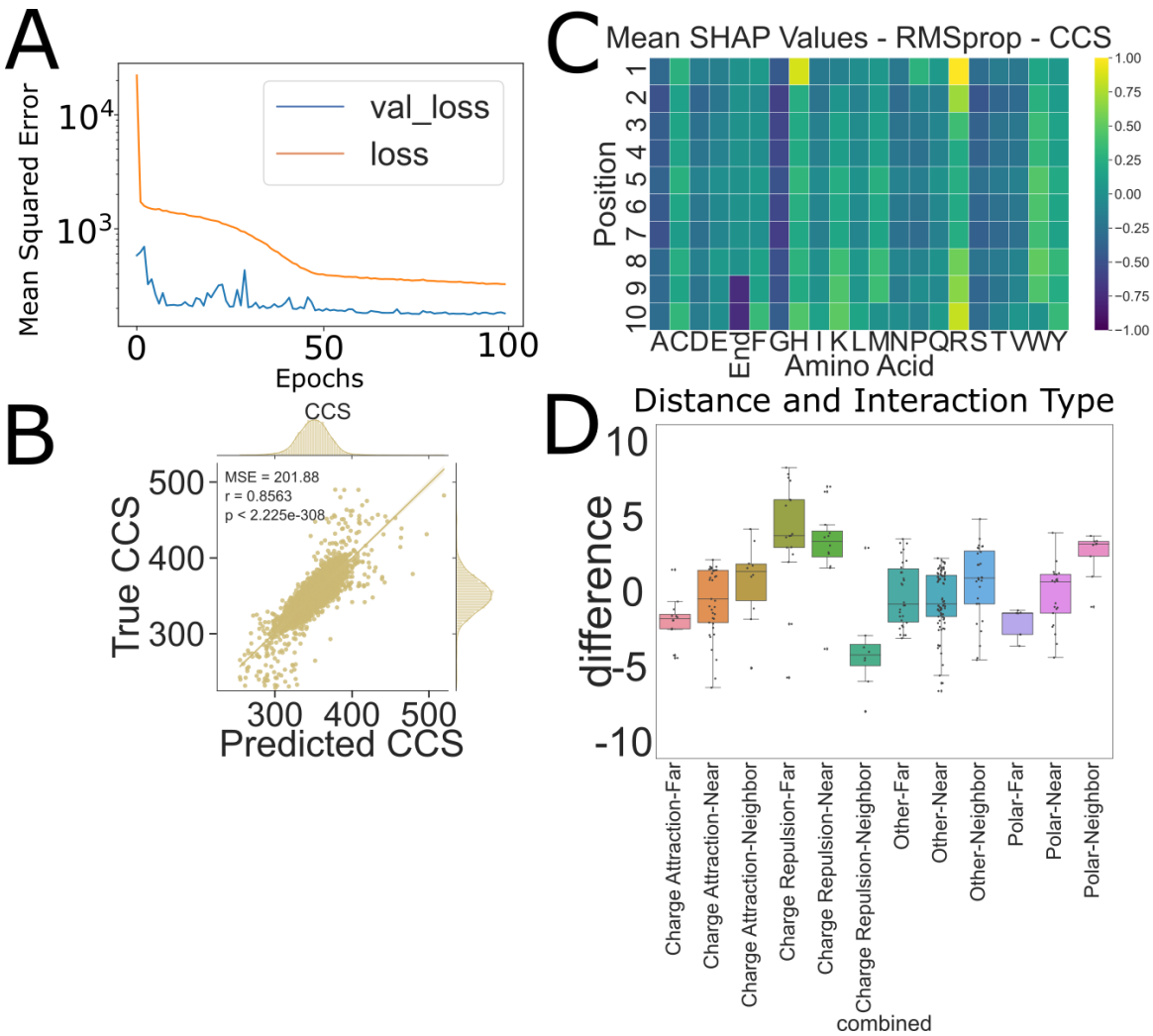
Supplemental Figure 11. SHAP Values of Collisional Cross Section by Peptide Length. The impact of peptide length on SHAP values was explored for the CCS data. The dataset was split into peptides of length 8, 9, and 10. All SHAP values were plotted as violin plots. The mean SHAP values were plotted in heatmaps by position and amino acid and standardized. Significant interactions by dependence analysis were plotted in bar charts by distance between interactions.



Supplemental Figure 12. PoSHAP Analysis of Mamu A001 With Unoptimized Hyperparameters and RMSprop. A new model for the Mamu data was trained using the same architectures but with different hyperparameters and RMSprop as the optimization algorithm. Loss was plotted as mean squared error compared to the validation data. (A) Similar metrics for MSE, r, and p-values were obtained (B). Similar patterns are also observed for the PoSHAP heatmap of A001. (C) A dependence plot for A001 shows similar patterns to the Adam optimized model, including the positional dependence of proline at position two for high SHAP values of serine and threonine.



Supplemental Figure 13. PoSHAP Analysis of A:11*01 With Unoptimized Hyperparameters and RMSprop. A new model for the A:11*01 data was trained using the same architectures but with different hyperparameters and RMSprop as the optimization algorithm. Loss was plotted as mean squared error compared to the validation data. (A) Similar metrics for MSE, r, and p-values were obtained (B). Similar patterns are also observed for the PoSHAP heatmap of A:11*01. (C) The SHAP ranges by position plot for A:11*01 shows similar patterns to the Adam optimized model, including the largest range of SHAP values at position two, nine, and ten.



Supplemental Figure 14. PoSHAP Analysis of CCS With Unoptimized Hyperparameters and RMSprop. A new model for the CCS data was trained using the same architectures but with different hyperparameters and RMSprop as the optimization algorithm. Loss was plotted as mean squared error compared to the validation data. (A) Similar metrics for MSE, r, and p-values were obtained (B). Similar patterns are also observed for the PoSHAP heatmap of CCS. (C) Dependence analysis was performed on the dataset and the combined distance-interaction type bar plot shows similar relationships between the groupings, notably charge repulsion's split.