Abstract

Artificial neural networks show promising performance in detecting correlations within data that are associated with specific outcomes. However, the black-box nature of such models can hinder the knowledge advancement in research fields by obscuring the decision process and preventing scientists from fully conceptualizing predicted outcomes. Furthermore, domain experts like healthcare providers need explainable predictions to assess whether a predicted outcome can be trusted in high-stakes scenarios and to help them integrating a model into their own routine. Therefore, interpretable models play a crucial role for the incorporation of machine learning into high-stakes scenarios like healthcare. In this paper we introduce Convolutional Motif Kernel Networks, a neural network architecture that involves learning a feature representation within a subspace of the reproducing kernel Hilbert space of the position-aware motif kernel function. The resulting model enables to directly interpret and evaluate prediction outcomes by providing a biologically and medically meaningful explanation without the need for additional post-hoc analysis. We show that our model is able to robustly learn on small datasets and reaches state-of-the-art performance on relevant healthcare prediction tasks. Our proposed method can be utilized on DNA and protein sequences. Furthermore, we show that the proposed method learns biologically meaningful concepts directly from data using an end-to-end learning scheme.

1 Introduction

Biological sequences contain valuable information for a wide variety of biological processes. While this property makes them crucial for advances in related research fields, it also provides the potential to improve diagnosis and treatment decisions in healthcare systems. For this reason, a large amount of machine learning approaches that solve learning tasks on biological sequences were developed over the last years. Among others, these approaches include the prediction of splice sites (Degroeve et al., 2002) and translation initiation sites (Zien et al., 2000), predicting binding affinity between proteins and DNA/RNA (Alipanahi et al., 2015; Yang and Deng, 2020), drug resistance prediction (Döring et al., 2018), or the denoising of biological sequence data (Eraslan et al., 2019). However, trained models can only be safely incorporated into medical routines if their prediction outcomes
can be thoroughly interpreted and understood even by domain experts, e.g., healthcare providers like medical practitioners, without strong knowledge in the foundations of machine learning. Kernel methods and statistical models provide the possibility to interpret results within the data’s domain, hence, allowing domain experts to judge outcomes using their own expertise. Yet, scalability issues in terms of data size limit their utility considering the rapid increase of available data in medical and biological research. On the other hand, gradient-based learning approaches like neural networks can handle huge data pools with relative ease but are normally developed as black-box models. Although there are model-agnostic techniques to interpret these models, e.g., saliency maps (Kadir and Brady, 2001) or Shapley additive explanations (SHAP) (Lundberg and Lee, 2017), recent work by Rudin (Rudin, 2019) advises the use of inherently interpretable models for high stakes scenarios over post-hoc explaining black-box models. One problem of post-hoc ML explanation models identified by Rudin is their unfaithfulness regarding the original model’s computation, which can result in misleading explanations. Sixt and colleagues showed this unfaithfulness for attribution methods by proving that most methods ignored later layers of a model when computing explanations (Sixt et al., 2020). Furthermore, Bordt and colleagues showed the limitations of post-hoc explanations in adversarial contexts (Bordt et al., 2022). Lipton warned about the danger of optimizing post-hoc methods to produce plausible but misleading explanations (Lipton, 2018). In high stakes scenarios like healthcare, decisions made on misleading or wrong explanations can cause dangerous situations with the potential to further harm patients or other vulnerable groups.

In recent years, several efforts were published to combine kernel functions and neural networks (Cho and Saul, 2009; Bo et al., 2011; Mairal et al., 2014; Mairal, 2016). Combining these two approaches enhances neural networks with the interpretability and robustness of kernel methods. On the other hand, it allows to extend learning within a reproducing kernel Hilbert space (RKHS) to problems with massive numbers of data points. Recently, Chen and colleagues introduced these efforts into data mining on biological sequences by developing convolutional kernel networks based on a continuous relaxation of the mismatch kernel (Chen et al., 2019a). Although these models show promising performance, the choice of kernel resulted in the necessity of a post-hoc model for interpretation. Another limitation results from the fact that the mismatch kernel restricts considered k-mer occurrences to a position-independent representation (Eskin et al., 2003; Leslie and Kuang, 2004). In many medical tasks, however, positional and compositional variability provide key information. One kernel network approach that utilizes positional information is the recurrent kernel network (RKN) proposed by Chen and colleagues (Chen et al., 2019b). While this architecture showed promising performance capabilities, the chosen architecture resulted once again in a black-box models with the need for post-hoc interpretation. The oligo kernel proposed by Meinicke and colleagues is able to model positional variability and can additionally provide traditional monomer-based representations as well as position-independent k-mer representations as limiting cases (Meinicke et al., 2004). Furthermore, the oligo kernel allows for intuitive and simple interpretation of k-mer relevance and positional variability. However, the oligo kernel cannot be directly incorporated into a convolutional network architecture and does not take into account information provided by compositional variability of motifs. While k-mers are short sequences with fixed letters at each position, motifs are short sequence patterns that can represent more than one possible letter at each position. The above mentioned limitations motivated our work presented here.

This work is structured in the following way. Section 2 introduces the position-aware motif kernel function and details how to incorporate the position-aware motif kernel into a convolutional kernel layer and how to interpret a trained CMKN model. Section 3 provides details regarding the conducted experiments on synthetic and real-world data and the results. Finally, section 4 provides a discussion of presented prediction and interpretation results and section 5 completes this work with a conclusion.

In summary, our manuscript provides the following contributions:

- We extend convolutional kernel network models for biological sequences to incorporate positional information and make them inherently interpretable, which removes the necessity for post-hoc explanation models. The new models are called convolutional motif kernel networks (CMKNs).
- This extension is achieved by introducing a new kernel function, called position-aware motif kernel, that quantifies the position dependent similarity of motif occurrences.
- We use one synthetic and two real-world datasets to show how our method can be used as a research tool to gain insight into biological sequence data and how CMKNs can provide...
local interpretation that can help domain experts, e.g., healthcare providers, to quickly interpret and validate prediction outcomes of a trained CMKN model with their domain expertise.

2 Methods

In the following section, we will introduce our new kernel function and show how this kernel can be used to create inherently interpretable kernel networks.

2.1 Position-Aware Motif Kernel

We introduce a new kernel function that incorporates the positional uncertainty of the oligo kernel (Meinicke et al., 2004) but is defined for arbitrary sequence motifs. Furthermore, our kernel function can be used to construct a convolutional kernel layer as described by Mairal (Mairal, 2016). Our kernel function is based on two main ideas: First, we introduce a mapping of sequence positions onto the unit circle, which allows us to represent the position comparison term by a linear operation followed by a non-linear activation function. Second, we introduce a $k$-mer comparison term. This extension enables the kernel function to deal with inexact $k$-mer matching, which capacitates our kernel function to handle arbitrary sequence motifs. We call our new kernel function position-aware motif kernel (PAM).

The first part of our position-aware motif kernel compares sequence positions. In prior work, e.g., Meinicke et al., 2004 (Meinicke et al., 2004) or Mialon et al., 2021 (Mialon et al., 2021), a quadratic term is usually employed to measure the similarity of positions. We utilize a linear comparison term instead. First, all positions are mapped onto the upper half of the unit circle to create unit $\ell_2$-norm vectors: $\tilde{p} = \left(\cos \left(\frac{p|\pi}{|x|}\right), \sin \left(\frac{p|\pi}{|x|}\right)\right)^T$, where $|x|$ denotes the length of the corresponding sequence. Due to the position vectors now having unit $\ell_2$-norm, the position comparison term can be written as follows: $-\frac{1}{4\sigma^2}||\tilde{p} - \tilde{q}||_2^2 = \frac{\beta}{2\sigma^2}(\tilde{p}^T \tilde{q} - 1)$. This allows us to define the following position comparison kernel function over pairs of sequence positions:

$$K_{\text{position}}(p, q) = \exp \left(\beta \frac{2}{\sigma^2} (\tilde{p}^T \tilde{q} - 1)\right),$$

(1)

where $\beta$ is a scaling parameter that compensates for the reduced absolute distance between sequence positions due to the introduced mapping and $\sigma$ is a positional uncertainty parameter similar to the homonymous $\sigma$ parameter of the oligo kernel.

The second part of our position-aware motif kernel compares sequence motifs. For biological sequences, a motif describes a nucleotide or amino acid pattern of a certain length. Sequence motifs can be written in form of a normalized position frequency matrix (nPFM), which is a matrix in $\mathbb{R}^{|A| \times k}$ with $|A|$ being the size of the alphabet over which the motif is created and $k$ being the length of the motif. An nPFM has to fulfill the additional constraint that each column has unit $\ell_2$-norm (see supplement for more details). For two motifs $\omega$ and $\omega'$ of length $k$ given as flattened nPFMs, i.e., the columns are concatenated to convert the matrix into a vector, we define the following motif comparison kernel function:

$$K_{\text{nPFM}}(\omega, \omega') = \exp \left(\alpha (\omega^T \omega' - k)\right),$$

(2)

This function will become one if the two motifs match exactly and will approach zero with increasing difference of the two motifs. The parameter $\alpha$ determines how fast the function approaches zero and, hence, specifies the influence of inexact matching motifs.

We define our position-aware motif kernel by forming the product kernel using the functions introduced in Equation 1 and 2 and aggregating the kernel evaluation of all motif-position pairs with a sum. In other words, the position-aware motif kernel for pairs of sequences $x$ and $x'$ over an alphabet $A$ is given by:

$$K_{\text{PAM}}(x, x') = C \sum_{p=1}^{|x|} \sum_{q=1}^{|x'|} K_{\text{0}}((\omega_p, p), (\omega_q, q))$$

(3)
The main intuition behind the position-aware motif kernel is to detect similarities between motifs, even if they occur at a certain distance from each other and even if the nPFM underlying the motif

\[ K_0((\omega_p, p), (\omega_q, q)) = K_{\text{nPFM}}(\omega_p, \omega_q) \cdot K_{\text{position}}(p, q) = \exp \left( \alpha \left( \omega_p^T \omega_q - k \right) + \frac{\beta}{2\sigma^2} \left( \hat{p}^T \hat{q} - 1 \right) \right). \]

Here, \(|x|\) and \(|x'|\) are the lengths of the respective sequences, \(\omega_p\) is the motif of length \(k\) starting at position \(p\) in sequence \(x\) represented as a flattened nPFM, and \(\omega_q\) is defined analogously to \(\omega_p\) but for sequence \(x'\). The constant \(C = \sqrt{\frac{\pi^2\sigma^2}{2\sigma^2}}\) results from the derivation of the motif kernel matrix elements as the inner product of two sequence representatives \(\phi_x, \phi_{x'}\) in the feature space of all motifs as detailed in the Supplement.

2.2 Extracting a Feasible Kernel Layer using Nyström’s Method

Mairal and colleagues showed that a variant of the Nyström method ([Williams and Seeger 2001, Zhang et al. 2008]) can be used to incorporate learning within a reproducing kernel Hilbert space (RKHS) into neural networks (Mairal et al. 2014, Mairal 2016). We use the same approach to construct a finite-dimensional subspace of the RKHS \(\mathcal{H}\) over motif-position pairs that is implicitly defined by \(K_0\) and incorporate learning within this subspace into a neural network architecture.

Consider a set of \(n\) anchor points \(z_1, \ldots, z_n\), where each anchor point is a motif-position pair \(z_i = (\omega_{z_i}, p_{z_i})\). We define an \(n\)-dimensional subspace \(\mathcal{E}\) of \(\mathcal{H}\) that is spanned by a set of anchor points, i.e.

\[ \mathcal{E} = \text{Span}(\phi_{z_1}, \ldots, \phi_{z_n}), \tag{4} \]

where \(\phi_{z_i}\) denotes the projection of each anchor point into the RKHS \(\mathcal{H}\). Utilizing the kernel trick, a motif-position pair can be projected onto \(\mathcal{E}\) without explicitly calculating the images of the anchor points \(\phi_{z_1}, \ldots, \phi_{z_n}\). This natural parametrization is given by [Mairal 2016]

\[ \psi((\omega, p)) = K_{\mathcal{ZZ}}^{-\frac{1}{2}} K_{\mathcal{Z}}((\omega, p)). \tag{5} \]

Here, \(K_{\mathcal{ZZ}} = (K_0(z_i, z_j))_{i=1,\ldots,n; j=1,\ldots,n}\) is the Gram matrix formed over the anchor points, \(K_{\mathcal{ZZ}}^{-\frac{1}{2}}\) is the (pseudo)-inverse square root of the Gram matrix, and \(K_{\mathcal{Z}}((\omega, p)) = (K_0(\omega_{z_1}, \omega, p), \ldots, K_0(\omega_{z_n}, \omega, p))^T\). We follow the procedure proposed in prior work (Mairal 2016, Chen et al. 2019) to initialize anchor points. First, we sample a set of \(m >> n\) motif-position pairs from the training data. Afterwards, we perform k-means clustering with euclidean distance metric using k-means++ initialization to get \(n\) cluster centers of the sampled set. After convergence, we enforce the nPFM constraints onto the cluster centers. With initialized anchor points, CMKN models can be trained by a simple end-to-end learning scheme. A schematic overview over a CMKN model for DNA input together with a visualization of the information flow within the network is shown in Figure 1.

2.3 Interpreting a CMKN Model

The main intuition behind the position-aware motif kernel is to detect similarities between motifs, even if they occur at a certain distance from each other and even if the nPFM underlying the motif
is different to a certain degree. In this way, our approach extends previous approaches like the oligo kernel (Meinicke et al., 2004) and the weighted degree kernel with shifts (Rätsch et al., 2005), which only evaluated exact \( k \)-mer matches. However, our kernel is based on a concept that we call motif functions which are extensions of the oligo functions introduced by Meinicke and colleagues (Meinicke et al., 2004) (see Supplement for details). A motif function represents the \( nPFM \) and position(s) of occurrence of the corresponding motif with a smoothing of the position to account for positional uncertainty. Apart from providing a biologically meaningful feature representation, the use of a kernel based on motif functions allows for a direct interpretation of a trained CMKN model without the need for post-hoc methods. If the CMKN model consists only of linear fully-connected layers after the kernel layer, as strictly applied throughout this study, important sequence positions and corresponding motifs can be directly inferred from the learned weights and anchor points, since this ensures that only linear combinations of the learned feature representations are considered. The importance of a sequence position for a certain class can be assessed by calculating the mean positive weight of the edges that connect the position with the output state that corresponds to the class. The importance \( \iota \) of position \( p \) for class \( c \) can thereby be expressed as:

\[
\iota^p_c = \frac{1}{|N_p|} \sum_{n \in N_p} \tilde{\iota}^n_{n,c}, \quad \tilde{\iota}^n_{n,c} = \begin{cases} \sum_{m \mid m \in N^{(n)}} w_{n,m} \tilde{\iota}^m_{m,c}, & \text{if } N^{(n)} \cap N^{(O)} = \emptyset \\ 1, & \text{if } N^{(n)} \cap N^{(O)} = o_c \\ 0, & \text{otherwise.} \end{cases}
\]

Here, \( N_p \) denotes the set of neurons contributing to the importance of position \( p \), \( N^{(n)} \) denotes the set of neurons from the next layer connected by an edge with positive weight to neuron \( n \in N_p \), \( w_{n,m} \) denotes the weight of the edge connecting neuron \( n \) with neuron \( m \in N^{(n)} \), and \( N^{(O)} \) denotes the set of \( c \) neurons \( o_c \), each representing a single class, in the output layer. Furthermore, the motif associated with the class at that position is retrieved by identifying all learned motifs with positive weights and calculating the weighted mean motif using the learned weights. This procedure is similar to inferring feature importances from the primal representation using the learned parameters of a SVM. Said utilization of the primal representation is possible for linear kernels and most string kernels (Meinicke et al., 2004). The importance of each amino acid at each position of the motif can be directly accessed by sorting the rows of each column of the associated \( nPFM \) in decreasing order. Additionally, motif functions enhance CMKN models with the ability to compute local interpretations, i.e., an explanation of prediction results for single inputs within the data’s domain. For an input sequence and a learned motif-position pair, we can estimate the importance of that pair by calculating the \( \ell_2 \)-norm of the corresponding motif function. To assess the class that a model associates with an important position, the class-specific motifs that were learned by a model at that position can be retrieved and ranked by the \( \ell_2 \)-norm of the motif functions on the input sequence. The motif with the highest \( \ell_2 \)-norm determines which class a model assigns to the position. We show an exemplary visualization for domain experts of this procedure in Figure 3b.

3 Experiments

We used synthetic data to evaluate CMKN’s ability to recover meaningful sequence patterns. Furthermore, we evaluated the performance capability of our proposed method on two different prediction tasks: antiretroviral drug resistance prediction and splice site recognition.

3.1 Recovering Meaningful Patterns in Synthetic Data

In order to assess whether CMKN models can reliably recover distinct biological patterns from sequences, we created a synthetic dataset containing 1000 randomly generated DNA sequences of length 100. The set was equally split into negative and positive sequences, with a distinct motif embedded into each class of sequences at a specific position (see Figure 2 for the embedded motifs). For negative sequences, the motif was embedded at position 20 with a positional uncertainty of \( \pm 5 \) positions. For positive sequences, the motif was embedded at position 80 with a positional uncertainty of \( \pm 5 \) positions. The compositional variability shown in Figure 2 can be understood in a way that one-third of the 5-mers embedded into negative sequences had a thymine at position 2 while two-third of the k-mers had a cytosine. This is equivalent for the other motif positions with compositional variability. By creating a synthetic dataset with this procedure, we made sure that the data contains positional and compositional variability that are important for the prediction task. We
trained a CMKN model using a motif length of 5 and a positional uncertainty parameter of 4. For the kernel layer, we chose 50 anchor points. The other kernel hyperparameters were set to $\alpha = 1$ and $\beta = 1000$. The model was trained for 50 epochs using the binary cross-entropy with logits loss function.

Figure 2 shows the results of our experiment with the synthetic dataset. We recovered the positional feature importance values as well as the learned motifs at position 20 and 80 using the procedure described in section 2.3. As clearly visible on the left side, CMKN is able to recover the two embedded motifs with high similarity using simple end-to-end learning without post-hoc model optimization. Furthermore, the right side of Figure 2 shows that CMKN is able to detect the relevant areas of biological sequences.

3.2 Prediction of antiretroviral drug resistance

When choosing a personalized treatment combination for HIV-infected people, it is crucial to know the resistance profile of the viral variants against available drugs. It has been shown that the genetic sequence of a virus can be used to predict resistance against certain antiretroviral drugs (Döring et al., 2018). We performed resistance prediction for drugs representing the three most commonly used antiretroviral drug classes against HIV infections: Nucleoside reverse-transcriptase inhibitors (NRTIs), non-nucleoside reverse-transcriptase inhibitors (NNRTIs), and protease inhibitors (PIs). This prediction task was chosen for evaluation of the proposed method, since it remains an highly important problem in the treatment of HIV infections and the acquired immune deficiency syndrome (AIDS) and is often considered as a role model for precision medicine.

Amino acid sequences of virus protein variants with corresponding drug resistance information were extracted from Stanford University’s HIV drug resistance database (HIVdb) (Rhee et al., 2003; Shafer, 2006). An overview of the available data for each of the drugs included in the evaluation can be found in the Supplement. The network architecture used for HIV drug resistance prediction consists of a single convolutional motif kernel layer followed by two fully-connected layers. The first fully-connected layer projected the flattened output of the kernel layer onto 200 nodes and the second fully-connected layer had two output states, one for the susceptible class and one for the resistant class. The motif length and the hyperparameter $\alpha$ of the kernel function were both fixed to 1 based on prior biological knowledge (for details see supplement material). The scaling hyperparameter $\beta$ was fixed to $\frac{|x|^2}{\sigma^2}$ with $|x| = 99$ for PI datasets and $|x| = 240$ for NRTI/NNRTI datasets. This compensates for the transformation of sequence positions (for details see supplement material). The number of anchor points and the positional uncertainty parameter $\sigma$ were optimized using a grid search (for details see supplement material). Due to the limited number of available samples, each model was trained using a 5-fold stratified cross-validation. The data splits for each fold were fixed across models to ensure the same training environment for each hyperparameter combination. Training success was evaluated using the performance measures accuracy, F1 score, and area under the receiver operating characteristic curve (auROC). Due to the fact that some datasets
Table 1: Mean performance and standard derivation of prediction models for three different HIV drug classes: PIs, NRTIs, NNRTIs. Models include polynomial kernel SVMs (SVM\text{poly}), oligo kernel SVMs (SVM\text{oligo}), random forests (RF), convolutional neural networks (CNN), convolutional kernel networks (CKN seq), and convolutional motif kernel networks (CMKN). Highest values are displayed in bold.

| Drug Class | Model     | Accuracy   | F1 Score   | auROC     | MCC     |
|-----------|-----------|------------|------------|-----------|---------|
| PI        | SVM\text{poly} | 0.90 ± 0.04 | 0.83 ± 0.09 | 0.95 ± 0.03 | 0.75 ± 0.10 |
|           | SVM\text{oligo} | 0.92 ± 0.03 | 0.86 ± 0.09 | 0.97 ± 0.03 | 0.81 ± 0.09 |
|           | RF         | 0.92 ± 0.04 | 0.85 ± 0.13 | 0.97 ± 0.03 | 0.79 ± 0.13 |
|           | CNN        | 0.91 ± 0.3  | 0.84 ± 0.11 | 0.94 ± 0.05 | 0.77 ± 0.11 |
|           | CKN seq    | 0.84 ± 0.05 | 0.72 ± 0.12 | 0.88 ± 0.05 | 0.60 ± 0.11 |
|           | CMKN       | 0.92 ± 0.03 | 0.87 ± 0.09 | 0.96 ± 0.03 | 0.81 ± 0.10 |
| NRTI      | SVM\text{poly} | 0.86 ± 0.06 | 0.82 ± 0.09 | 0.90 ± 0.05 | 0.70 ± 0.12 |
|           | SVM\text{oligo} | 0.88 ± 0.05 | 0.85 ± 0.09 | 0.94 ± 0.03 | 0.75 ± 0.10 |
|           | RF         | 0.88 ± 0.06 | 0.84 ± 0.12 | 0.94 ± 0.04 | 0.74 ± 0.15 |
|           | CNN        | 0.88 ± 0.05 | 0.85 ± 0.09 | 0.93 ± 0.04 | 0.74 ± 0.12 |
|           | CKN seq    | 0.79 ± 0.06 | 0.73 ± 0.12 | 0.85 ± 0.05 | 0.54 ± 0.13 |
|           | CMKN       | 0.89 ± 0.05 | 0.86 ± 0.09 | 0.93 ± 0.05 | 0.76 ± 0.11 |
| NNRTI     | SVM\text{poly} | 0.82 ± 0.06 | 0.76 ± 0.11 | 0.84 ± 0.06 | 0.63 ± 0.14 |
|           | SVM\text{oligo} | 0.89 ± 0.05 | 0.86 ± 0.11 | 0.94 ± 0.05 | 0.79 ± 0.12 |
|           | RF         | 0.88 ± 0.05 | 0.85 ± 0.09 | 0.93 ± 0.07 | 0.75 ± 0.12 |
|           | CNN        | 0.89 ± 0.04 | 0.86 ± 0.08 | 0.94 ± 0.06 | 0.78 ± 0.10 |
|           | CKN seq    | 0.73 ± 0.06 | 0.63 ± 0.16 | 0.78 ± 0.08 | 0.42 ± 0.15 |
|           | CMKN       | 0.91 ± 0.03 | 0.89 ± 0.06 | 0.95 ± 0.05 | 0.81 ± 0.08 |

were highly unbalanced, we also included the Matthew’s correlation coefficient (MCC) (Matthews [1975]) in the performance assessment.

Mean performances achieved for each of the three investigated drug classes can be found in Table 1. Our method was able to achieve high accuracy, F1 score, and auROC values for each drug class. Even though the classification problem is highly imbalanced for some of the tested drugs, our model is still able to achieve a high Matthew’s correlation coefficient (MCC) value with mean MCC performance exceeding 0.75 for each of the three investigated drug classes. We compared CMKN’s performance to previously used models for HIV drug resistance prediction: SVMs with polynomial kernel (Döring et al. [2018]) and random forest (RF) classifiers (Raposo et al. [2020]). Furthermore, we included a SVM utilizing the oligo kernel and the CKN seq model (Chen et al. [2019a]) into our analysis. Additionally, we performed an ablation test by replacing the kernel layer with a standard convolutional layer to investigate the influence of our kernel architecture onto prediction performance (denoted by CNN in Table 1). The results for all models can be found in Table 1. Our method either outperformed the competitors or achieved similar performance.

3.2.1 Utilizing CMKN’s interpretation capabilities to identify resistance mutation positions and motifs

Apart from assessing CMKN’s prediction performance, we investigated how well our models were able to learn biologically meaningful patterns from drug resistance data. For each sequence position, we calculated the position importance for each class as described in Section 2.3 and identified peaks with a sliding window approach, i.e., the mean importance of a window of length 11 around each position was calculated and subtracted from the position importance. We selected the 10 highest peaks identified using this sliding window approach. For each peak position, the associated mean motif (of length one) as well as the two most important amino acids of this mean motif were retrieved using the approach described in Section 2.3. To get position importance and mean motifs for one of the three investigated drug classes (PIs, NRTIs, and NNRTIs), we averaged the importance values as well as the mean motifs over all models that belong to drugs of the same drug class (8 models for PIs, 6 models for NRTIs, and 3 models for NNRTIs). Figure 3 displays the top ten position of the resistant and susceptible class together with the top two amino acids of the corresponding mean
motif for each of the three investigated drug classes. The results indicate that CMKN models are able to learn biologically meaningful patterns from real-world datasets. The most important positions identified by CMKN models correspond mainly to known drug resistance mutation (DRM) positions while the corresponding learned motifs are focused on DRMs. This result is consistent for all three tested drug types. However, CMKN models provide more than a global interpretation. Figure 3b shows the result of CMKN’s local interpretation capabilities (as described in section 2.3) for the model trained on nelfinavir (NFV) data and three randomly selected isolates. First, we identified the ten most important positions learned by the model. Afterwards, we retrieved the resistant and susceptible motifs for each position from the trained model. Using the motif functions, we were able to identify which positions the model indicated to be informative for the susceptible class and which positions were indicated to be informative for the resistant class using the procedure described in section 2.3. This local interpretation shows biologically meaningful patterns and can be used by domain experts to verify a prediction made by the model. For a more detailed discussion of the visualization results, see section 4.

Figure 3: a (Global Interpretation): CMKNs can be used for data mining on biological sequences. The ten most important positions learned by the model, together with the top two contributing amino acids, are displayed. The height of the bar plot at each position indicates the normalized feature importance of that position, i.e., the mean position feature importance was subtracted from the feature importance of the specific position. Higher bars indicate more important positions. The importance of each sequence position was calculated as described in section 2.3 and peaks were identified using a sliding window approach with a window length of 11. Afterwards, the model’s learned motifs associated with the ten highest peaks were calculated (see section 2.3) and the two amino acids with the highest contribution to these motifs were selected. Positions displayed in red (blue) are associated with the resistant (susceptible) class. b (Local Interpretation): We created an exemplary visualization of CMKN’s explanation capabilities. Prediction results of the nelfinavir (NFV) model for three randomly chosen input sequences are visualized by showing the learned top ten positions together with the amino acid occurring at the respective position in the input. For each position, the motif functions of the learned motifs are evaluated to identify the one with the highest $\ell_2$-norm on the input (see Section 2.3). If the corresponding motif is a learned resistance (susceptibility) associated motif, the position-amino-acid pair is highlighted in red (blue). The height of the bars above each position corresponds to the $\ell_2$-norm of the corresponding susceptible (blue) and resistant (red) motif functions (scaled between 0 and 1). For each isolate, the true and predicted label is displayed.
Table 2: Test performance on splice site benchmarks. The displayed methods include higher order Markov Chain (MC) classifiers (Sonnenburg et al., 2007), a combination of higher order Markov Chains and SVMs with polynomial kernel (MC-SVM) (Baten et al., 2006), SVMs with the locality improved kernel (LIK) (Sonnenburg et al., 2007), SVMs with the weighted degree kernel (WD) (Sonnenburg et al., 2007), SVMs with the weighted degree kernel with shifts (WDS) (Sonnenburg et al., 2007), SpliceRover (Zuallaert et al., 2018), and our CMKN. Highest numbers are shown in bold. Dashes indicate missing values in the original manuscripts.

| Model         | NN269   |          |          | DGSplicer |          |          |
|---------------|---------|----------|----------|-----------|----------|----------|
|               | Acceptor| Donor    | Acceptor| Donor     | Acceptor| Donor    |
|               | auROC   | auPRC    | auROC   | auPRC     | auROC   | auPRC    |
| MC            | 0.97    | 0.88     | 0.98    | 0.92      | 0.97    | 0.31     |
| MC-SVM        | 0.97    | 0.88     | 0.98    | 0.90      | -       | -        |
| LIK           | 0.98    | 0.92     | 0.98    | 0.93      | -       | -        |
| WD            | 0.98    | 0.93     | 0.99    | 0.93      | 0.98    | 0.32     |
| WDS           | 0.99    | 0.94     | 0.98    | 0.93      | 0.97    | 0.29     |
| SpliceRover   | 0.99    | -        | 0.98    | -         | 0.97    | 0.65     |
| CMKN          | 0.97    | 0.94     | 0.98    | 0.96      | 0.97    | 0.65     |

3.3 Splice Site Prediction

The recognition of splice sites is an important task in healthcare, since it can uncover genetic variants and differences in protein composition in individual patients. It consists of two classification problems: distinguishing decoys from true targets for acceptor sites and for donor sites.

We used two benchmarks to assess performance of our model on the splice site recognition task: NN269 (Reese et al., 1997) and DGSplicer (Chen et al., 2005). Both benchmarks provide test sets and are highly imbalanced. Details on training and test sets for both benchmarks can be found in the Supplement. For splice site recognition, we used the same architecture that was used for the HIV drug resistance prediction. The hyperparameter $\alpha$ was again fixed to 1. We similarly fixed the scaling parameter to $\beta = \frac{|x|^2}{10}$ with $|x| = 90$ for acceptor sequences and $|x| = 15$ for donor sequences on the NN269 benchmark and $|x| = 36$ for acceptor sequences and $|x| = 18$ for donor sequences on the DGSplicer benchmark. The number of anchor points, the motif length $k$, and the positional uncertainty parameter were optimized using a grid search with 5-fold stratified cross-validation on the training data (details can be found in the Supplement). The model with the best hyperparameter combination was retrained on the whole training set and evaluated using the test set. Training success was evaluated using the area under the precision-recall curve (auPRC), to account for class imbalance, and the auROC to enable comparison with previously published models.

We compared our method to several methods that were previously applied on splice site recognition. These included higher order Markov Chain (MC) classifiers, SVMs with the locality improved kernel (LIK), the weighted degree kernel (WD), and the weighted degree kernel with shifts (WDS) published in (Sonnenburg et al., 2007), a method combining higher order Markov Chains and SVMs with polynomial kernel (MC-SVM) published in (Baten et al., 2006), and a CNN architecture called SpliceRover (Zuallaert et al., 2018). On the NN269 benchmark, our method performed comparable to other methods in terms of auROC and outperformed almost all competitors in terms of auPRC (see Table 2). On the DGSplicer benchmark, our method performed comparable to other methods in terms of auROC, while substantially outperforming all competitors in terms of auPRC (see Table 2). An evaluation of CMKN’s interpretation on the splice site prediction task can be found in the Supplement.

4 Discussion

In this work, we introduced convolutional motif kernel networks (CMKNs), a convolutional network architecture that allows for end-to-end learning within a subspace of our proposed position-aware motif kernel’s RKHS.

By combining a convolutional network architecture with a kernel function, our model is able to perform robust end-to-end learning on relatively small datasets as was shown on data from Standford’s
HIVdb. Our model was able to generalize to validation data with only a few hundred training samples even in highly unbalanced scenarios. However, due to the fact that our model is based on a standard convolutional network architecture, CMKNs can easily be used on datasets with several hundreds of thousands of samples, as shown on the splice site prediction benchmarks. This allows to utilize our proposed kernel function on very large datasets, something that would be notoriously hard using standard kernel methods like SVMs, since the calculation of a large Gram matrix for our position-aware motif kernel is computationally very demanding.

We included accuracy and auROC as performance measures in our evaluation, since both measures are often used in the ML literature. However, on imbalanced data their informative value is decreased due to a bias towards the majority class (Chicco [2017]), as can be seen by considering the auROC vs. auPRC performances on the DGSplicer benchmark in Table 2. Therefore, we included measures that provide better insights on imbalanced data with few positives: F1 and MCC for HIV drug resistance prediction and auPRC for splice site prediction. Considering F1, MCC, and auPRC, our model performed similar or better compared to all other models.

Another advantage of introducing kernel function evaluation into a neural architecture is the possibility to overcome the black-box nature of neural networks. Since learning within our proposed kernel layer admits a projection onto a subspace of the RKHS of our position-aware motif kernel, each output node of the kernel layer is associated with a position-motif pair. This allows for a biological interpretation of the learned weights associated with each node of the kernel layer. With these global interpretation capabilities, our model can be used as a tool for data mining on biological sequence data. We showed on HIV drug resistance data that our model is able to learn biologically meaningful patterns using standard end-to-end learning methods (see Figure 3). The majority of the ten most important positions correspond to known DRM positions (nine for PI drugs, eight for NRTI drugs, seven for NNRTI drugs). Furthermore, the top amino acids in the learned resistant motifs reflect known DRMs while the top amino acids in the learned susceptible motifs either reflect the wildtype or none DRMs. There are three exceptions where the susceptible motif features amino acids that lead to an increased drug resistance. These exceptions are leucine (L) and valine (V) at position 50 for PI drugs, valine (V) at position 184 for NRTI drug, and aspartic acid (D) at position 215 for NRTI drugs. However, these exceptions appear to occur due to the averaging of motifs over all drugs for a specific drug class. While all of the four mentioned mutations cause an increase resistance against a subset of drugs (Rhee et al. [2010] Colombo et al. [2004] Goudsmit et al. [1996]), they are also a cause of increased susceptibility or have no effect for other drugs (Rhee et al. [2003] Bethell et al. [2012] Larder et al. [1995]). Valine at position 50 reduces susceptibility to fosamprenavir (FPV), lopinavir (LPV), and darunavir (DRV) but increases susceptibility to tipranavir (TPV). Leucine at position 50 confers high-level resistance to atazanavir (ATV) but increases susceptibility to all other PI drugs. For NRTI drugs, valine at position 184 reduces susceptibility to lamivudine (3TC) but increases susceptibility to zidovudine (AZT), stavudine (d4T), and tenofovir (TDF). At position 215, a mutation to aspartic acid is a so-called thymidine analog mutation that reduces susceptibility to AZT and d4T but has no effect on susceptibility to all other NRTI drugs.

Apart from the data mining capabilities of our proposed CMKN model, the motif functions enrich our model with the capability to provide local interpretations for prediction results within the data’s domain. Figure 3 shows an example of the visualization capabilities of our CMKN model using nelfinavir (NFV) data, one of the PI drugs. The figure was created with the following steps. First, the trained NFV model was used to build the susceptible and resistant motifs for each of the ten most informative resistance positions learned for the NFV drug, as described in Section 2.3 and 3.2. Afterwards, for each position if the model relates the position to the susceptible or resistant class, as described in Section 2.3. For the first input, which was correctly classified as susceptible, the visualization shows that the model associated susceptible motifs with each of the positions except for position 63 and 88. However, a domain expert can quickly verify that the model falsely classified that the amino acid asparagine (N) at position 88 indicates resistance, since asparagine corresponds to the wildtype and is therefore in accordance with a susceptible isolate. Furthermore, there is no experimental evidence supporting that position 63 is associated with a resistance causing mutation. Using this knowledge, a domain expert can make an educated decision that the prediction is correct. For the correctly classified resistant input, the model associates resistant motifs with positions 63, 71, 90, and, again falsely, with position 88. Since a mutation to methionine (M) at position 90 causes a strong resistance against NFV (Rhee et al. [2010] Schapiro et al. [1996] Craig et al. [1998] Zolopa et al. [1999]), a domain expert could again directly validate the prediction.
result. The interpretation capabilities gain importance in case of a wrongly classified input as shown in the bottom part of Figure 3b. Here a domain expert would see that a susceptibility to NFV was predicted while three positions, 10, 84, and 88, are associated with resistant motifs. We again have the previously described, apparently systematic, error at position 88, but a mutation to valine (V) at position 84 causes a moderate resistance against NFV [Kempf et al., 2001]. Additionally, a mutation to phenylalanine (F) at position 10 is known to be associated with reduced in vitro susceptibility to NFV [Rhee et al., 2010; Van Mark et al., 2009]. Thus, the visualization provides the domain expert with all information needed to treat the prediction outcome with the adequate caution. This shows that utilizing the proposed kernel formulation in our model’s architecture, together with the proposed motif functions, can provide a visualization of a trained model’s output that helps domain experts to validate the predictions.

5 Conclusion

Our convolutional motif kernel network architecture provides inherently interpretable end-to-end learning on biological sequence data and achieves state-of-the-art performance on relevant healthcare prediction tasks, namely predicting antiretroviral drug resistance of HIV isolates and distinguishing decoys from real splice sites.

We show that CMKN is able to learn biologically meaningful motif and position patterns on synthetic and real-world datasets. CMKN’s global interpretation can foster data mining and knowledge advancement on biological sequence data. On the other hand, CMKN’s local interpretation can be utilized by domain experts to judge the validity of a prediction.

Possible future improvements include investigating a combination of different motif kernel layers to combine different motif lengths and extend the architecture to utilize meaningful combinations of motifs. Another improvement that we want to explore in future work is the extension of the kernel formulation to multi-layer networks while securing the interpretation capabilities.

Data and Code Availability

Source code, pre-processing scripts, and experimental scripts are available on GitHub [here]. HIV drug resistance data can be found online on HIVdb. The NN269 benchmark can be downloaded from the George Manson University. The DGSplicer benchmark was provided by the original author, Prof. Chung-Chin Lu.

Acknowledgments and Disclosure of Funding

The authors would like to thank Prof. Chung-Chin Lu for providing the DGSplicer benchmark. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy – EXC number 2064/1 – Project number 390727645. This research was supported by the German Federal Ministry of Education and Research (BMBF) project ‘Training Center Machine Learning, Tübingen’ with grant number 01IS17054. This work was supported by the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039A.

References

Alipanahi, B., Delong, A., Weirauch, M. T., and Frey, B. J. (2015). Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. Nature biotechnology, 33(8):831–838.

Baten, A. K., Chang, B. C., Halgamuge, S. K., and Li, J. (2006). Splice site identification using probabilistic parameters and svm classification. In BMC bioinformatics, volume 7, pages 1–15. BioMed Central.

Bethell, R., Scherer, J., Witvrouw, M., Paquet, A., Coakley, E., and Hall, D. (2012). Phenotypic protease inhibitor resistance and cross-resistance in the clinic from 2006 to 2008 and mutational
prevalences in HIV from patients with discordant tipranavir and darunavir susceptibility phenotypes. *AIDS research and human retroviruses*, 28(9):1019–1024.

Bo, L., Lai, K., Ren, X., and Fox, D. (2011). Object recognition with hierarchical kernel descriptors. In *CVPR 2011*, pages 1729–1736. IEEE.

Bordt, S., Finck, M., Raidl, E., and von Luxburg, U. (2022). Post-hoc explanations fail to achieve their purpose in adversarial contexts. *arXiv preprint arXiv:2201.10295*.

Chen, D., Jacob, L., and Mairal, J. (2019a). Biological sequence modeling with convolutional kernel networks. *Bioinformatics*, 35(18):3294–3302.

Chen, D., Jacob, L., and Mairal, J. (2019b). Recurrent kernel networks. In *Advances in Neural Information Processing Systems*, pages 13431–13442.

Chen, T.-M., Lu, C.-C., and Li, W.-H. (2005). Prediction of splice sites with dependency graphs and their expanded bayesian networks. *Bioinformatics*, 21(4):471–482.

Chicco, D. (2017). Ten quick tips for machine learning in computational biology. *BioData mining*, 10(1):1–17.

Cho, Y. and Saul, L. (2009). Kernel methods for deep learning. *Advances in neural information processing systems*, 22:342–350.

Colonno, R., Rose, R., McLaren, C., Thiry, A., Parkin, N., and Friborg, J. (2004). Identification of i50l as the signature atazanavir (atv)-resistance mutation in treatment-naive hiv-1-infected patients receiving atv-containing regimens. *Journal of Infectious Diseases*, 189(10):1802–1810.

Craig, C., Race, E., Sheldon, J., Whittaker, L., Gilbert, S., Moffatt, A., Rose, J., Dissanayake, S., Chirn, G.-W., Duncan, I. B., et al. (1998). Hiv protease genotype and viral sensitivity to hiv protease inhibitors following saquinavir therapy. *Aids*, 12(13):1611–1618.

Cui, Y., Jia, M., Lin, T.-Y., Song, Y., and Belongie, S. (2019). Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9268–9277.

Degroeve, S., De Baets, B., Van de Peer, Y., and Rouzé, P. (2002). Feature subset selection for splice site prediction. *Bioinformatics*, 18(suppl_2):S75–S83.

Döring, M., Büch, J., Friedrich, G., Pironti, A., Kalaghdti, P., Knops, E., Heger, E., Obermeier, M., Däumer, M., Thiel, A., et al. (2018). geno2pheno [ngs-freq]: a genotypic interpretation system for identifying viral drug resistance using next-generation sequencing data. *Nucleic acids research*, 46(W1):W271–W277.

Eraslan, G., Simon, L. M., Mircea, M., Mueller, N. S., and Theis, F. J. (2019). Single-cell rna-seq denoising using a deep count autoencoder. *Nature communications*, 10(1):1–14.

Eskin, E., Weston, J., Noble, W. S., and Leslie, C. S. (2003). Mismatch string kernels for svm protein classification. In *Advances in neural information processing systems*, pages 1441–1448.

Goudsmit, J., De Ronde, A., Ho, D. D., and Perelson, A. S. (1996). Human immunodeficiency virus fitness in vivo: calculations based on a single zidovudine resistance mutation at codon 215 of reverse transcriptase. *Journal of virology*, 70(8):5662–5664.

Kadir, T. and Brady, M. (2001). Saliency, scale and image description. *International Journal of Computer Vision*, 45(2):83–105.

Kempf, D. J., Isaacson, J. D., King, M. S., Brun, S. C., Xu, Y., Real, K., Bernstein, B. M., Japour, A. J., Sun, E., and Rode, R. A. (2001). Identification of genotypic changes in human immunodeficiency virus protease that correlate with reduced susceptibility to the protease inhibitor lopinavir among viral isolates from protease inhibitor-experienced patients. *Journal of Virology*, 75(16):7462–7469.

Larder, B. A., Kemp, S. D., and Harrigan, P. R. (1995). Potential mechanism for sustained antiretroviral efficacy of azt-3tc combination therapy. *Science*, 269(5224):696–699.
Leslie, C. and Kuang, R. (2004). Fast string kernels using inexact matching for protein sequences. *Journal of Machine Learning Research*, 5(Nov):1435–1455.

Lipton, Z. C. (2018). The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777.

Mairal, J. (2016). End-to-end kernel learning with supervised convolutional kernel networks. In *Advances in neural information processing systems*, pages 1399–1407.

Mairal, J., Koniusz, P., Harchaoui, Z., and Schmid, C. (2014). Convolutional kernel networks. In *Advances in neural information processing systems*, pages 2627–2635.

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451.

Meinicke, P., Tech, M., Morgenstern, B., and Merkl, R. (2004). Oligo kernels for datamining on biological sequences: a case study on prokaryotic translation initiation sites. *BMC Bioinformatics*, 5(1):169.

Mialon, G., Chen, D., d’Aspremont, A., and Mairal, J. (2021). A trainable optimal transport embedding for feature aggregation and its relationship to attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Morley, J., Machado, C. C., Burr, C., Cowls, J., Joshi, I., Taddeo, M., and Floridi, L. (2020). The ethics of ai in health care: a mapping review. *Social Science & Medicine*, 260:113172.

Raposo, L. M., Rosa, P. T. C., and Nobre, F. F. (2020). Random forest algorithm for prediction of hiv drug resistance. In *Pattern Recognition Techniques Applied to Biomedical Problems*, pages 109–127. Springer.

Rätsch, G., Sonnenburg, S., and Schölkopf, B. (2005). Rase: recognition of alternatively spliced exons in c.elegans. *Bioinformatics*, 21(suppl_1):369–377.

Reese, M. G., Eeckman, F. H., Kulp, D., and Haussler, D. (1997). Improved splice site detection in genie. *Journal of computational biology*, 4(3):311–323.

Rhee, S.-Y., Gonzales, M. J., Kantor, R., Betts, B. J., Ravela, J., and Shafer, R. W. (2003). Human immunodeficiency virus reverse transcriptase and protease sequence database. *Nucleic acids research*, 31(1):298–303.

Rhee, S.-Y., Taylor, J., Fessell, W. J., Kaufman, D., Towner, W., Troia, P., Ruane, P., Hellinger, J., Shirvani, V., Zolopa, A., et al. (2010). Hiv-1 protease mutations and protease inhibitor cross-resistance. *Antimicrobial agents and chemotherapy*, 54(10):4253–4261.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Schapiro, J. M., Winters, M. A., Stewart, F., Efron, B., Norris, J., Kozal, M. J., and Merigan, T. C. (1996). The effect of high-dose saquinavir on viral load and cd4+ t-cell counts in hiv-infected patients. *Annals of Internal Medicine*, 124(12):1039–1050.

Shafer, R. W. (2006). Rationale and uses of a public hiv drug-resistance database. *The Journal of infectious diseases*, 194(Supplement_1):S51–S58.

Sixt, L., Granz, M., and Landgraf, T. (2020). When explanations lie: Why many modified bp attributions fail. In *International Conference on Machine Learning*, pages 9046–9057. PMLR.

Sonnenburg, S., Schweikert, G., Philips, P., Behr, J., and Rätsch, G. (2007). Accurate splice site prediction using support vector machines. In *BMC Bioinformatics*, volume 8, pages 1–16. Springer.
Steiner, M. C., Gibson, K. M., and Crandall, K. A. (2020). Drug resistance prediction using deep learning techniques on hiv-1 sequence data. *Viruses*, 12(5):560.

Van Marck, H., Dierynck, I., Kraus, G., Hallenberger, S., Pattery, T., Muyldermans, G., Geeraert, L., Borozdina, L., Bonesteel, R., Aston, C., et al. (2009). The impact of individual human immunodeficiency virus type 1 protease mutations on drug susceptibility is highly influenced by complex interactions with the background protease sequence. *Journal of virology*, 83(18):9512–9520.

Williams, C. and Seeger, M. (2001). Using the nyström method to speed up kernel machines. In *Proceedings of the 14th annual conference on neural information processing systems*, number CONF, pages 682–688.

Yang, W. and Deng, L. (2020). Predba: A heterogeneous ensemble approach for predicting protein-dna binding affinity. *Scientific Reports*, 10(1):1–11.

Zhang, K., Tsang, I. W., and Kwok, J. T. (2008). Improved nyström low-rank approximation and error analysis. In *Proceedings of the 25th international conference on Machine learning*, pages 1232–1239.

Zien, A., Rätsch, G., Mika, S., Schölkopf, B., Lengauer, T., and Müller, K.-R. (2000). Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics*, 16(9):799–807.

Zolopa, A. R., Shafer, R. W., Warford, A., Montoya, J. G., Hsu, P., Katzenstein, D., Merigan, T. C., and Efron, B. (1999). Hiv-1 genotypic resistance patterns predict response to saquinavir–ritonavir therapy in patients in whom previous protease inhibitor therapy had failed. *Annals of internal medicine*, 131(11):813–821.

Zuallaert, J., Godin, F., Kim, M., Soete, A., Saeys, Y., and De Neve, W. (2018). Splicerover: interpretable convolutional neural networks for improved splice site prediction. *Bioinformatics*, 34(24):4180–4188.
A Societal and Environmental Impact

Medical data are notoriously biased against minorities and there are numerous examples for machine learning models that learn this biases and have a severe deterioration in performance with regard to minorities (see e.g., (Morley et al., 2020)). We did not include a statement on the societal impact of our work into the main manuscript due to the used benchmarks. The HIV benchmark consists of viral sequences, which renders the evaluation regarding human metadata unnecessary. On the contrary, improving the treatment of HIV infection and AIDS has a beneficial impact on society. On the other hand, the splice site benchmarks are created using human genes. However, the original authors did not include any necessary metadata into the benchmarks, which renders a evaluation as described above impossible. We will include the evaluation of potential bias against minorities in future work, if the model is used in a real-world application on medical data that is potentially biased. Since this bias normally arises from the data, such considerations are task-specific and not model-specific.

All experiments were conducted using a single NVIDIA GeForce GTX 1080 Ti GPU. All experiments together required a total of 124 hours of computing time. This resulted in total emissions of 13.39 kg CO\(_2\)e, which is equivalent to burning 6.71 kg of coal. To compensate this emissions, 0.22 tree seedlings have to sequester carbon for 10 years. These estimations were calculated using the Machine Learning Impact calculator\(^1\) by Lacoste and colleagues (Alexandre Lacoste et al.: Quantifying the carbon emissions of machine learning. arXiv preprint arXiv:1910.09700 (2019)).

B Setting the Scaling Parameter

Given a sequence \(x\) with length \(|x|\), we found empirically that setting \(\beta = \frac{|x|^2}{|x|^4}\) compensates for the transformation of sequence positions as introduced in the manuscript (see Figure 4).

---

\(^{1}\)https://mlco2.github.io/impact/
C Normalized Position Frequency Matrix

A set of biological sequences of length \( k \) can be easily transformed in to a position frequency matrix (PFM) by counting the occurrences of each nucleotide or amino acid from an alphabet \( A \) at each position and constructing a matrix \( M = (m_{ij})_{i=1,...,|A|; j=1,...,k} \), where \( m_{ij} \) is equal to the number of occurrences of nucleotide or amino acid \( i \) at position \( j \). The PFM can then be easily transformed into a normalized position frequency matrix (nPFM) \( M_{\text{norm}} = (m_{\text{norm},ij})_{i=1,...,|A|; j=1,...,k} \) by dividing each entry of every column \( m_j = \sum_i m_{ij} \) of \( M \) by the \( \ell_2 \)-norm of the respective column, i.e.

\[
m_{\text{norm},ij} = \frac{m_{ij}}{\|m_j\|_2}.
\]

Using this definition of an nPFM, the set of all \( k \)-mers is a subset of the set of all sequence motifs. Considering a \( k \)-mer as a special motifs, each column of the corresponding nPFM has only one element set to one while all other elements are set to zero.

D Orthogonal Projection onto the Subspace \( \mathcal{E} \)

We denote a motif position pair by \( y = (\omega, p) \). Then, the explicit parametrization derived by Mairal in [Mairal, 2016] Appendix A also holds for our kernel. This can be easily shown by simple calculus using Mairal’s definition:

\[
f_y := \sum_{j=1}^{n} \alpha_j^* \varphi(z_j) \quad \text{with} \quad \alpha^* \in \arg \min_{\alpha \in \mathbb{R}^n} \left\| \varphi(y) - \sum_{j=1}^{n} \alpha_j \varphi(z_j) \right\|_\mathcal{H}^2.
\]

Then the following holds:

\[
\left\| \varphi(y) - \sum_{j=1}^{n} \alpha_j \varphi(z_j) \right\|_\mathcal{H}^2 = \langle \varphi(y), \varphi(y) \rangle_\mathcal{H} - 2 \langle \varphi(y), \sum_{j=1}^{n} \alpha_j \varphi(z_j) \rangle_\mathcal{H} + \sum_{j=1}^{n} \alpha_j \varphi(z_j), \sum_{i=1}^{n} \alpha_i \varphi(z_i) \rangle_\mathcal{H}
\]

\[
= 1 - 2 \sum_{j=1}^{n} \langle \varphi(y), \alpha_j \varphi(z_j) \rangle_\mathcal{H} + \sum_{j=1}^{n} \sum_{i=1}^{n} \langle \alpha_j \varphi(z_j), \alpha_i \varphi(z_i) \rangle_\mathcal{H}
\]

\[
= 1 - 2 \sum_{j=1}^{n} \alpha_j K_0(y, z_j) + \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_j \alpha_i K_0(z_j, z_i)
\]

\[
= 1 - 2 \alpha^T K_{ZZ} \alpha + \alpha^T K_Z \alpha
\]

Here, \( K_{ZZ} = (K_0(z_i, z_j))_{i=1,...,n;j=1,...,n} \) is the Gram matrix formed over the anchor points \( z_1, \ldots, z_n \), \( K_{ZZ}^{-1} \) is the (pseudo)-inverse square root of the Gram matrix, and \( K_Z((\omega, p)) = (K_0(z_1, (\omega, p)), \ldots, K_0(z_n, (\omega, p)))^T \). \( K_0 \) is the kernel function introduced in the main manuscript. From here the argument is similar to Mairal’s argument in the Appendix of [Mairal, 2016].

E Deriving the Position-Aware Motif Kernel from Motif Functions

For a sequence \( x \) over an alphabet \( A \) and a motif \( \chi \in \mathbb{R}^{|A|^k} \), we define the motif function as

\[
\phi_x(\chi, t) = \sum_{p=1}^{\|x\|} \phi_{x,p}(\chi, t),
\]

with

\[
\phi_{x,p}(\chi, t) = \exp \left( -\alpha \|\chi - \omega_p\|_2^2 - \frac{\beta}{2\sigma^2} \|t - \tilde{p}\|_2^2 \right)
\]

and position variable \( t \in \mathbb{R}^2 \).
Consider two sequences \( \mathbf{x} \) and \( \mathbf{x}' \) of same length, i.e. \( |\mathbf{x}| = |\mathbf{x}'| \). The motif kernel between them is given as the inner product of the associated motif functions \( \phi_{\mathbf{x}} \) and \( \phi_{\mathbf{x}'} \):

\[
K_{\text{PAM}}(\mathbf{x}, \mathbf{x}') = \phi_{\mathbf{x}} \cdot \phi_{\mathbf{x}'} = \int \int \phi_{\mathbf{x}}(\chi, t) \cdot \phi_{\mathbf{x}'}(\chi, t) \, d\chi \, dt
\]

\[
= \sum_{p=1}^{\text{dim}} \sum_{q=1}^{\text{dim}} \int \int e^{-\alpha \|\chi - \omega_p\|_2^2 - \frac{\beta}{2\sigma^2} \|\omega_p - \omega_q\|_2^2} e^{-\beta \|\omega_p - \omega_q\|_2^2} \int \int e^{-\beta \|\omega_p - \omega_q\|_2^2} \int \int e^{-\alpha \|\chi - \omega_p\|_2^2} \, d\chi \, dt.
\]

with \( \omega_p, \omega_q \in \mathbb{R}^{|A|} \) and \( \tilde{p}, \tilde{q} \) from the upper half of the unit circle. The above derivation uses the fact that the variables \( \chi \in \mathbb{R}^{|A|} \) and \( t \in \mathbb{R}^2 \) are not restricted and thus (using the substitutions \( y = \chi, \gamma = \alpha, \delta = \omega_p, \epsilon = \omega_q, N = |A| \) and \( y = t, \gamma = \beta, \delta = \tilde{p}, \epsilon = \tilde{q}, N = 2, \) respectively) can be removed via

\[
\int e^{-\gamma (\|y - \delta\|_2^2 + \|y - \epsilon\|_2^2)} \, dy
\]

\[
= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-\gamma (y_1 - \delta_1)^2 + \ldots + (y_N - \delta_N)^2 + (y_1 - \epsilon_1)^2 + \ldots + (y_N - \epsilon_N)^2) \, dy_1 \ldots dy_N
\]

\[
= \int_{-\infty}^{\infty} e^{-\gamma (\|y - \delta\|_2^2 + \|y - \epsilon\|_2^2)} \, dy_1 \ldots \int_{-\infty}^{\infty} e^{-\gamma (\|y - \delta\|_2^2 + \|y - \epsilon\|_2^2)} \, dy_N
\]

\[
= \int_{-\infty}^{\infty} e^{-\gamma (\|y - \delta\|_2^2 + \|y - \epsilon\|_2^2)} \, dy_1 \ldots \int_{-\infty}^{\infty} e^{-\gamma (\|y - \delta\|_2^2 + \|y - \epsilon\|_2^2)} \, dy_N
\]

\[
= \int_{-\infty}^{\infty} e^{-\gamma y_1^2 + \ldots} \, dy_1 \ldots \int_{-\infty}^{\infty} e^{-\gamma y_N^2} \, dy_N
\]

\[
= \sqrt{\frac{\pi}{2\gamma}} e^{\frac{(2\gamma y_1^2 + \ldots)^2}{2\gamma}} \ldots \sqrt{\frac{\pi}{2\gamma}} e^{\frac{(2\gamma y_N^2 + \ldots)^2}{2\gamma}}
\]

\[
= \sqrt{\frac{\pi}{2\gamma}} \left( e^{\frac{(\gamma (\delta_1^2 + \ldots) - y_1^2)}{2\gamma}} \ldots e^{\frac{(\gamma (\delta_N^2 + \ldots) - y_N^2)}{2\gamma}} \right)
\]

\[
= \frac{\pi}{2\gamma} e^{-\frac{\gamma (\delta_1^2 + \ldots)}{2\gamma}} \ldots e^{-\frac{\gamma (\delta_N^2 + \ldots)}{2\gamma}}
\]

\[
= \sqrt{\frac{\pi}{2\gamma}} e^{-\frac{\gamma (\|\delta - \epsilon\|_2^2)^2}{2\gamma}}
\]

using

\[
\int_{-\infty}^{\infty} e^{-ay^2 + by + c} \, dy = \sqrt{\frac{\pi}{a}} e^{\frac{b^2}{4a} + c}
\]

with \( a = 2\gamma, b = 2\gamma (\delta_i + \epsilon_i) \), and \( c = -\gamma (\delta_i^2 + \epsilon_i^2) \).
The last equality in Eq. (11) results from the normalisation of the nPFM as defined in Eq. (7), i.e., the column-wise unit ℓ2-norm demanded earlier in Section 5.1 of the main manuscript. Consider two normalized position frequency matrices (nPFM) \( A \in \mathbb{R}^{|A| \times k} \) and \( B \in \mathbb{R}^{|A| \times k} \), with \(|A|\) being the size of the alphabet over which the motif is created and \(k\) being the length of the motif. One can define two vectors \( a \) and \( b \) given as flattened nPFMs \( A \) and \( B \), i.e., the columns are concatenated to convert the matrices into vectors. Utilizing that each column of a nPFM has unit ℓ2-norm by definition, the following equality is obtained:

\[
\|a-b\|^2 = (a_{11} - b_{11})^2 + \ldots + (a_{|A|1} - b_{|A|1})^2 + \ldots + (a_{1k} - b_{1k})^2 + \ldots + (a_{|A|k} - b_{|A|k})^2 \\
= a_{11}^2 + \ldots + a_{|A|1}^2 + b_{11}^2 + \ldots + b_{|A|1}^2 + \ldots + a_{1k}^2 + \ldots + a_{|A|k}^2 + b_{1k}^2 + \ldots + b_{|A|k}^2 \\
- 2(a_{11}b_{11} + \ldots + a_{|A|1}b_{|A|1} + \ldots + a_{1k}b_{1k} + \ldots + a_{|A|k}b_{|A|k}) \\
= 2k - 2ab \\
\Rightarrow \frac{1}{2}\|a-b\|^2 = ab - k
\]

(14)

F HIVdb

HIVdb is one of the largest, publicly available databases containing resistance information against antiretroviral drugs that are used for the treatment of an HIV infection or an acquired immunodeficiency syndrome (AIDS). We used available information for eight PI drugs, six NRTI drugs, and four NNRTI drugs (data was downloaded on 14 April 2021). The PI dataset contained 819 isolates, each represented by an amino acid sequence of length 99. The NRTI dataset contained 489 isolates, each represented by an amino acid sequence of length 240. Finally, the NNRTI dataset contained 583 isolates, each represented by an amino acid sequence of length 240. For each isolate, available drug resistance information for each drug of the corresponding type was given as a change in fold resistance compared to the wildtype. We used thresholds provided by HIVdb to convert fold resistance values into the discrete classes 'susceptible' and 'resistant'.

HIVdb stores the isolate information as a tab-separated table. Each position is represented either by a '-', if the amino acid is the same as in the wild type, or by the mutation at that position. Furthermore, there are columns for each drug with the fold resistance of the isolate compared to the wild type. We provide a script that translates these tables into a FASTA file that can be used directly as input for the provided DataSet object. HIVdb provides two fold resistance thresholds for each of the drugs within the database. These thresholds are used by our preparation script to assign each isolate to either the 'low resistance', 'medium resistance', or 'high resistance' class. The DataSet object combines 'medium resistance' and 'high resistance' classes into the 'resistant' class on data access.

Table 3 provides detailed information about sample count, class distribution, best performing parameters, and classification performances for each of the tested drugs. The amount of isolates for each of the drugs differs from the total number of isolates, since not all isolates had fold resistance information for each of the drugs. We excluded drugs with less than 100 isolates from all further evaluations. Each of the drugs is indicated using the official three-letter abbreviation. The corresponding complete names and trade names are written below.
Table 3: Overview of the HIVdb datasets. For each drug, basic dataset statistics are displayed as well as the best values for the positional uncertainty parameter $\sigma$ and the number of anchor points. Furthermore, mean values as well as standard deviations for four performance parameters (accuracy, F1 score, area under the receiver operating characteristic curve (auROC), and Matthew’s correlation coefficient (MCC)) achieved on validation sets during a stratified 5-fold cross validation are shown.

| Type | Drug | Samples | Class Distribution (%) | Best Parameters | Accuracy | F1 Score | auROC | MCC |
|------|------|---------|------------------------|-----------------|----------|----------|-------|-----|
|      |      |         |                        | $\sigma$ | anchors |          |       |     |
|      |      |         | susceptible | resistant |          |          |       |     |
| PI   | ATV  | 477     | 59.3                  | 40.7          | 16       | 50       | 0.918±0.041 | 0.901±0.051 | 0.969±0.011 | 0.832±0.086 |
|      | DRV  | 273     | 87.9                  | 12.1          | 4        | 99       | 0.945±0.033 | 0.779±0.118 | 0.970±0.019 | 0.757±0.132 |
|      | FPV  | 737     | 65.3                  | 34.7          | 8        | 99       | 0.912±0.022 | 0.873±0.032 | 0.965±0.018 | 0.806±0.049 |
|      | IDV  | 771     | 58.5                  | 41.5          | 16       | 75       | 0.930±0.019 | 0.915±0.022 | 0.975±0.008 | 0.856±0.038 |
|      | LPV  | 612     | 67.2                  | 32.8          | 4        | 50       | 0.920±0.012 | 0.881±0.017 | 0.970±0.006 | 0.822±0.026 |
|      | NFV  | 793     | 46.5                  | 53.5          | 16       | 99       | 0.927±0.014 | 0.932±0.013 | 0.971±0.008 | 0.853±0.027 |
|      | SQV  | 775     | 63.4                  | 36.6          | 8        | 99       | 0.939±0.015 | 0.916±0.022 | 0.980±0.009 | 0.869±0.033 |
|      | TPV  | 306     | 76.8                  | 23.2          | 8        | 75       | 0.876±0.044 | 0.725±0.108 | 0.898±0.032 | 0.646±0.135 |
| NRTI | 3TC  | 477     | 45.3                  | 54.7          | 16       | 120      | 0.937±0.023 | 0.944±0.023 | 0.973±0.013 | 0.879±0.049 |
|      | ABC  | 480     | 36.9                  | 63.1          | 16       | 180      | 0.919±0.017 | 0.935±0.014 | 0.962±0.017 | 0.828±0.033 |
|      | AZT  | 477     | 58.5                  | 41.5          | 16       | 120      | 0.899±0.025 | 0.882±0.026 | 0.965±0.016 | 0.799±0.047 |
|      | D4T  | 479     | 57.6                  | 42.4          | 16       | 120      | 0.854±0.045 | 0.826±0.056 | 0.916±0.030 | 0.701±0.094 |
|      | DDI  | 479     | 52.0                  | 48.0          | 16       | 240      | 0.862±0.033 | 0.854±0.038 | 0.914±0.029 | 0.726±0.066 |
|      | TDF  | 395     | 73.4                  | 26.6          | 8        | 240      | 0.843±0.034 | 0.711±0.070 | 0.871±0.050 | 0.609±0.093 |
| NNRTI| EFV  | 511     | 55.4                  | 44.6          | 16       | 180      | 0.924±0.019 | 0.913±0.023 | 0.965±0.015 | 0.846±0.039 |
|      | ETR  | 183     | 66.1                  | 33.9          | 8        | 120      | 0.886±0.043 | 0.829±0.065 | 0.909±0.061 | 0.750±0.093 |
|      | NVP  | 515     | 47.0                  | 53.0          | 16       | 120      | 0.920±0.019 | 0.924±0.019 | 0.966±0.010 | 0.843±0.036 |
|      | RPV  | 83      | 66.3                  | 33.7          |           |          | Removed from analysis due to low sample count |
| Type   | Abbreviation | Name          | Trade Name                   |
|--------|--------------|---------------|------------------------------|
| PI     | ATV          | Atazanavir    | Reyataz, Evotaz, others      |
|        | DRV          | Darunavir     | Prezista, Prezocib, others  |
|        | FPV          | Fosamprenavir | Lexiva, Telzir               |
|        | IDV          | Indinavir     | Crixivan                     |
|        | LPV          | Lopinavir     | -                            |
|        | NFV          | Nelfinavir    | Viracept                     |
|        | SQV          | Saquinavir    | Invirase, Fortovase          |
|        | TPV          | Tipranavir    | Aptivus                      |
| NRTI   | 3TC          | Lamivudine    | Epivir, Epivir-HBV, Zeffix, others |
|        | ABC          | Abacavir      | Ziagen, others              |
|        | AZT          | Azidothymidine| Retrovir, others            |
|        | D4T          | Stavudine     | Zerit                       |
|        | DDI          | Didanosine    | Videx                       |
|        | TDF          | Tenofovir disoproxil | Viread, others |
| NNRTI  | EFV          | Efavirenz     | Sustiva, Stocrin, others    |
|        | ETR          | Etravirine    | Intelence                   |
|        | NVP          | Nevirapine    | Viramune                    |
|        | RPV          | Rilpivirine   | Edurant, Rekambys          |

### G Splice Site Benchmarks

Table 4: Overview of the benchmarks used to evaluate CMKN on the splice site prediction task.

| Benchmark   | Type          | Samples | Class Distribution (%) | Best Parameters |
|-------------|---------------|---------|------------------------|-----------------|
|             |               |         | pos       | neg       | k | σ | anchors |
| NN269       | Acceptor (train) | 5788    | 19.3      | 80.7      | 2 | 16 | 45     |
|             | Acceptor (test)  | 1089    | 19.1      | 80.9      |   |    |        |
|             | Donor (train)   | 5256    | 21.2      | 78.8      | 5 | 4  | 30     |
|             | Donor (test)    | 990     | 21.0      | 79.0      |   |    |        |
| DGSplicer   | Acceptor (train) | 322155  | 0.6       | 99.4      | 5 | 8  | 45     |
|             | Acceptor (test)  | 80539   | 0.6       | 99.4      |   |    |        |
|             | Donor (train)   | 228267  | 0.8       | 99.2      | 5 | 4  | 45     |
|             | Donor (test)    | 57067   | 0.8       | 99.2      |   |    |        |

The NN269 benchmark consists of 1324 true targets for both acceptor and donor datasets that were collected from 269 human genes. The acceptor sequences consist of 90 nucleotides with the splice site acceptor dimer AG at positions 69 to 70. The donor sequences consist of 15 nucleotides with the splice site donor dimer GT at positions 8 to 9. Furthermore, the benchmark includes 5553 acceptor decoys and 4922 donor decoys. Each decoy sequence has the splice site dimers (AG for acceptor sites, GT for donor sites) at the same positions as the true targets. The DGSplicer benchmark consists of 2380 true acceptor targets and 2379 true donor targets that were extracted from 462 multi-exon human genes. The acceptor sequences have a length of 36 nucleotides with the dimer AG at positions 26 to 27. The donor sequences have a length of 18 nucleotides with the dimer GT at positions 10 to 11. As with NN269, decoy sequences are included in the benchmark, i.e., 400314 pseudo acceptor sites and 282955 pseudo donor sites. Both benchmarks are split into training and test sets. The detailed statistics about the used benchmarks for the prediction of splice sites can be found in Table 4.

To deal with the huge imbalance between positive and negative samples within the DGSplicer benchmark, we randomly under-sampled the negative samples in the training sets to achieve a class ratio of \( \frac{N_p}{N_{rn}} = 0.25 \), where \( N_p \) is the number of samples in the positive class and \( N_{rn} \) is the number
of samples in the negative class after resampling. Class ratios in validation and test sets were kept unchanged.

H  Hyperparameter Optimization

Experiments were conducted with Python 3.6.9 using imbalanced-learn 0.7.0, numpy 1.18.5, scikit-learn 0.23.2, scipy 1.5.2, and torch 1.6.0.

H.1 HIV Drug Resistance Prediction

H.1.1 CMKN

CMKN models have 5 hyperparameters associated with the motif kernel layer that need to be optimized. These include the k-mer length \( k \), the motif comparison parameter \( \alpha \), the position scaling parameter \( \beta \), the positional uncertainty parameter \( \sigma \), and the number of anchor points.

We used prior (biological) knowledge to fix three of the hyperparameters. This allowed us to reduce computation time and lower the CO\(_2\) footprint of our experiments. We fixed the motif length \( k \) to 1, since considering longer motifs offers no advantage in the case of resistance mutations against antiretroviral drugs. Previously conducted research suggest that a drug resistance is caused by exchanging single amino acids which justifies to set the motif length to 1. Additionally, we fixed \( \alpha \) to 1.0 to ensure that the impact of inexact motif matching is not reduced. This takes the fact into account that some positions can have several different mutations causing resistance against antiretroviral drugs and the exchange of a single amino acid can cause a drug resistance. Therefore, inexact motif matching contains valuable information which justifies the value for \( \alpha \) we used. Lastly, the position scaling parameter \( \beta \) was fixed to \( \frac{\|x\|^2}{10} \), where \( \|x\| \) denotes the length of the input sequences. This value compensates for the transformation of the sequence positions as described in Section B.

The other two hyperparameters, the positional uncertainty \( \sigma \) and the number of anchor points, were optimized using a simple grid search. We used a small but sensible number of values for the grid to keep the CO\(_2\) footprint of our experiments as low as possible. The following values were used for the grid search:

\[
\begin{align*}
\sigma & \in \{1, 2, 4, 8, 16\} \\
\# \text{anchors}_{\text{PI}} & \in \{50, 75, 99\} \\
\# \text{anchors}_{\text{NRTI}} & \in \{120, 180, 240\} \\
\# \text{anchors}_{\text{NNRTI}} & \in \{120, 180, 240\}
\end{align*}
\]

The different choices for the number of anchors for PI and NRTI/NNRTI drugs were chosen to take account of the different sequence lengths. The optimal parameter choice for each of the drugs can be found in Table 3.

All CMKN models were trained using PyTorch’s implementation of the ADAM algorithm and class-balanced loss as introduced in (Cui et al., 2019). Training lasted for 200 epochs and no early-stopping was used. The learning rate started at 0.1 and was dynamically adjusted using the ReduceLROnPlateau method.

The value combination that maximized the most of the four performance measures, accuracy, F1 score, auROC, and MCC, was selected for further analysis.

H.1.2 Models used for Comparison

We used standard parameters for the random forest classifiers trained on our HIVdb datasets that matched the parameters used in (Raposo et al., 2020). The number of trees was set to 500 while the number of features to look at when searching for the best split, was set to the square root of the total number of features. For all other parameters the default values of the sklearn.ensemble.RandomForestClassifier class were used. This is coherent with the training procedure used in (Raposo et al., 2020). The input sequences were ordinal encoded before passing them to the classifier, i.e., each letter was replaced by an integer reflecting the position of the letter in the alphabet.
For support vector machines (SVMs) with polynomial kernel, we performed a grid search to optimize the degree of the polynomial kernel and the regularization parameter. The grid consisted of the following values:

$$C \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$$

$$\text{degree} \in \{1, 2, 3, 4, 5\}$$

Similar to the CMKN models, the parameter combination that maximized the most of the four performance measures, accuracy, F1 score, auROC, and MCC, was selected for further analysis. We used the same ordinal encoding that was used in the RF experiment to encode the input for the SVMs. For the model selection of SVMs with oligo kernel, we used the input encoding as described in (Meinicke et al., 2004). The hyperparameters of this method are the length of $k$-mers denoted by $k$, the position uncertainty parameter $\sigma$, and the regularization parameter $C$. Similar to CMKNs, $k$ was fixed to 1 due to biological reasons. The other two hyperparameter were optimized with the following grid:

$$C \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$$

$$\sigma \in \{1, 2, 4, 8, 16\}$$

The optimal parameters for the SVM models were:

| Drug | SVM\text{poly} | SVM\text{oligo} |
|------|----------------|----------------|
|      | $C$ | degree | $C$ | $\sigma$ |
| ATV  | 1.0 | 3     | 0.1 | 4     |
| DRV  | 0.001 | 1    | 0.01 | 1    |
| FPV  | 0.1 | 5     | 1.0 | 1     |
| IDV  | 10.0 | 2    | 1.0 | 1     |
| LPV  | 1.0 | 3     | 1.0 | 1     |
| NFV  | 10.0 | 2    | 1.0 | 1     |
| SQV  | 1.0 | 3     | 1.0 | 1     |
| TPV  | 0.001 | 2    | 0.1 | 1     |
| 3TC  | 0.1 | 4     | 0.1 | 1     |
| ABC  | 0.1 | 4     | 0.1 | 1     |
| AZT  | 0.1 | 5     | 1.0 | 1     |
| D4T  | 0.1 | 3     | 0.1 | 1     |
| DDI  | 0.1 | 3     | 0.1 | 1     |
| TDF  | 0.01 | 5    | 1.0 | 1     |
| EFV  | 0.01 | 5    | 0.1 | 2     |
| ETR  | 0.1 | 4     | 10.0 | 4    |
| NVP  | 10.0 | 1    | 0.1 | 4     |
| RPV  | Removed from analysis due to low sample count |

The CNN architecture used for DRM coverage comparison in section 9 was first published in (Steiner et al., 2020). First, we tried to train CNN models using the R script provided by the authors on our datasets but, unfortunately, the script exited with errors and was not usable. Instead of debugging the provided script, which would already pose the risk of changing the training environment compared to the original publication, we decided to implement the same architecture using PyTorch. This course of action provided the additional benefit that the training of both neural network architectures, CNN and CMKN, used the same framework and followed the same standards which improves comparability of the methods. The first layer was an embedding layer that took the nominal encoded sequences and learned a three-dimensional embedding. Afterwards two one-dimensional convolutional layers with ReLu activation were used. Both convolutional layers had a kernel size of 9 with 32 filters. A one-dimensional MaxPooling layer with a pooling size of 5 was used between the two convolutional layers. The flattened output of the second convolutional layer was fed into fully-connected layers for the classification. CNN models were trained using PyTorch’s implementation of the ADAM algorithm and class-balanced loss as introduced in (Cui et al., 2019). Training lasted for 200 epochs and no early-stopping was used. The learning rate started at 0.1 and was dynamically adjusted using the
Post hoc methods were needed to calculate feature importances for some models. For convolutional neural network (CNN) classifiers, we applied a permutation feature importance analysis and for the random forest (RF) classifiers, we calculated impurity-based feature importances. Position importances were directly assessable for CMKNs and oligo kernel SVMs (SVM\textsubscript{oligo}). CMKNs performed similar to CNNs and RFs. All three significantly outperformed SVM\textsubscript{oligo}. The center line of each box indicates the median. The height of the boxes represents the inter quartile range (IQR) with the upper and lower whiskers set to 1.5 times the IQR. Outliers are depicted by circles.

ReduceLROnPlateau method. We did not include the performance of the CNN architecture by Steiner et al. into our main manuscript due to the non-competitive performance on our datasets (i.e., they reached a mean MCC of 0.1). However, we included the description here since Steiner’s models are used in section 9. The CNN results shown in the main manuscript were achieved with a network resulting from a simple ablation test, i.e., we replaced the kernel layer of an CMKN model with a standard convolutional layer with the same parameters.

H.2 Splice Site Prediction

H.2.1 CMKN

In contrast to the HIV experiments, we only fixed two hyperparameters in the CMKN models used for splice site prediction: \( \alpha \) was fixed to 1 and \( \beta \) was fixed as described in Section B. The other hyperparameters were optimized with a grid search using the following values:

\[
\begin{align*}
    k & \in \{2, 3, 4, 5\} \\
    \sigma & \in \{1, 2, 4, 8, 16\} \\
    \#\text{anchors} & \in \{15, 30, 45\}
\end{align*}
\]

The optimal parameter choice for each splice site dataset can be found in Table 4.

All CMKN models were trained using PyTorch’s implementation of the ADAM algorithm and class-balanced loss as introduced in (Cui et al., 2019). Training lasted for 200 epochs and no early-stopping was used. The learning rate started at 0.1 and was dynamically adjusted using the ReduceLROnPlateau method.

I Comparing the InterpretationCapabilities of CMKNs

Comparing the quality of interpretation capabilities of different methods is a difficult problem, due to the ethical and philosophical aspects of interpretation which are hard to express mathematically (Lipton, 2018; Morley et al., 2020). Nevertheless, we performed some quantitative comparisons of the
Figure 6: Interpretation of the support vector machines utilizing the oligo kernel (SVM_{oligo}). Details about the calculation of these matrices can be found in the original manuscript [Meinicke et al., 2004]. SVM_{oligo} had difficulties identifying drug resistance mutation positions (DRMs) and, due to its limitation to $k$-mers, focused on single amino acids for each DRM.

global interpretation between the used inherently interpretable methods (i.e., CMKNs and oligo kernel SVMs) as well as published post-hoc interpretations of non-interpretable methods (impurity-based feature importance for RFs [Raposo et al., 2020] and permutation feature importance analysis for CNNs [Steiner et al., 2020]) on the HIV prediction task. For all methods, the 20 most important positions were identified and the percental coverage of drug resistance mutation positions (DRMs) using these 20 positions was calculated. The results can be found in Figure 5. We found that CMKNs performed similar to post-hoc methods applied on CNNs and RFs when used to identify DRMs. All three methods significantly outperformed oligo kernel SVMs. However, the post-hoc methods were limited to identifying positions without compositional information of the mutations. That lead to an incomplete picture of the biological process and therefore decreases the utility of the interpretation provided by post-hoc methods. Only CMKNs and oligo kernel SVMs were able to provide the complete picture with positional and compositional information of the drug resistance mutations. We used the procedure described by Meinicke and colleagues in the original manuscript to visualize the learned oligo functions from the trained SVMs [Meinicke et al., 2004]. The result is presented in Figure 6. Due to the fact that the oligo kernel is limited to discrete $k$-mers, the compositional variability of DRMs is not identified by oligo kernel SVMs. For identified DRMs, the visualization focuses on single amino acids.

Additionally to global interpretations, CMKN models can also provide local interpretations. Comparing local interpretations cannot be achieved without a user study, since their utility is closely coupled with the intended user. Performing a user study with medical practitioners is a time consuming task, which is out of the scope of this manuscript. Nevertheless, evaluating local interpretations with the intended users is an important research question and will be investigated in future work.
Figure 7: The learned sequence motifs of a CMKN model trained on the NN269 splice site prediction benchmark. The left motif was learned on the donor dataset. The right motif was learned on the acceptor dataset. The splice site is indicated by a red box.

J CMKN Model Interpretability for Splice Site Prediction

We also analyzed how well CMKN models can retrieve the relevant motifs from the splice site prediction task. For this purpose, we retrained a model on the two NN269 training sets (donor and acceptor). We selected motif lengths that are able to capture the compositional variability that contributes to real splice sites in contrast to the decoy sequences. Therefore, the model trained on the donor dataset used a motif length of 8 and the model trained on the acceptor dataset used a motif length of 17. Since there is no positional variability in the splice site benchmarks, we only investigated the learned motifs. Figure 7 shows the motifs learned on the donor (left) and acceptor (right) dataset. Both splice site dimers (GT for donor sites and AG for acceptor sites) are correctly learned by CMKN models. Note that a high compositional variability in the learned motifs for these two positions is expected since they are exactly the same in real splice site sequences and decoy sequences. Therefore, the model does not put high distinction into these two positions resulting in less clear motifs. For the donor sequences, CMKN is able to identify the AG dimer that appears directly before the splice site and is highly predictive for a real splice site sequence. For the acceptor sequences, CMKN correctly learns the poly pyrimidine tract that precedes a real splice site together with the fact that a guanine directly after the splice site dimer is highly predictive for a real splice site. The learned motifs show that CMKN models are able to recover biologically meaningful sequence motifs from datasets.