Generalizable models of antigen presentation for the characterization of understudied immune-landscapes

Linda Mazánová

4th Year Project Report
Artificial Intelligence and Computer Science
School of Informatics
University of Edinburgh

2022

Abstract

Antigen presentation is an essential part of the adaptive immune system. MHC class I molecules regulate the expression of antigen molecules on the surface of antigen-presenting cells by delivering short peptide sequences to the cell surface [1]. Predicting peptide presentation by MHC class I molecules enables us to design more specific vaccines and immunotherapies. Understanding the MHC class I pathway helps us to learn more about cancer and virus detection [2].

We present a novel application of the DC-Causal interpretability technique grounded in causal theory, which was originally proposed as a technique to interpret the classification of occluded images. We apply the DC-Causal technique to the domain of Immunology, aiming to interpret a BERT-based model (ImmunoBERT) that takes as input a sequence of amino acids and predicts whether a peptide will be presented by an MHC class I molecule. DC-Causal's interpretation of the model in a form of a ranked list of positions enhances our understanding of the model and provides us with insights into which parts and positions of the amino acid sequence have the highest responsibility for the predictions made by the model. Understanding how ImmunoBERT works increases our trust in its predictions and gives us higher confidence to use it for real-life applications. We show that the DC-Causal technique produces sensible explanations which are comparable to explanations from state-of-the-art interpretability techniques such as LIME and SFL. In addition, we show that produced explanations have supporting evidence in biological studies.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Linda Mazánová*)

Acknowledgements

I would like to express my sincere gratitude to my project supervisor Dr. Ajitha Rajan for her continuous support and insightful advice throughout the academic year.

I would like to extend my sincere thanks to Hans-Christof Gasser for his prompt and helpful advice whenever I needed to clarify some aspects of his machine learning model.

My appreciation also goes out to my family and friends for always supporting me and motivating me to do my best.

Table of Contents

Chapter 1

Introduction

1.1 Motivation

State-of-the-art machine learning models, which are often complex Deep Neural Networks (DNNs) that have anywhere from thousands to millions of parameters, are becoming popular in various applications and revolutionizing a wide range of industries. Recent research has seen successful applications of DNNs such as Convolutional Neural Networks [3] and Deep Reinforcement Learning models [4]. In this work we will cover another DNN, a specific language model - Bidirectional Encoder Representations from Transformers (BERT) [5].

The advances in machine learning algorithms, which became more sophisticated over time, enabled us to make breakthroughs and solve challenges that were previously impossible to solve with the tools we had at our disposal. However, in practice, we still often have to divert and use much simpler models that can be easily interpreted such as linear regression or decision trees, when we apply machine learning to real-world problems. We cannot trust and allow complex models to make life-critical decisions without trust, transparency, and a thorough understanding of how they work.

Machine learning has changed everything - from the way we interact on social media, do online shopping, play games, and translate text, to the way we form our opinions based on the information which we are presented when we search on the internet. Especially given the fast progress such as increasing prediction accuracy or decreasing requirements for training a DNN, the space of application opportunities for DNNs is growing exponentially. There are promising approaches to successfully apply machine learning within fields such as medicine, banking, or autonomous cars. Given how much impact machine learning has and will have in the future, we need to make sure we trust this technology before we allow it to make important decisions that shape our lives.

To trust a DNN, we need to understand the processes inside the network - which involves a series of computations and algorithms that capture patterns and relationships from the input data. The challenging aspect is, that there is no established way that would enable us to gain an understanding of these networks. Observing the computations inside the network provides transparency but not necessarily a sufficient understanding

of the processes. There is a need to interpret these models in such a way that humans can understand this interpretation which will increase their trust and confidence in these models.

To address this trust requirement, some techniques try to provide insight into how the algorithms arrive at the solutions. One of the approaches is to produce an explanation that interprets and explains a model in a sufficient and meaningful way that increases our trust and confidence in the model.

1.2 Application domain

This work will address the interpretability of a DNN in the field of Immunology. DNNs have an immense potential to help with tasks such as improving treatments, designing vaccines, and early diagnosis of diseases. To fulfill this ambition of making AI useful, it is necessary that the medical practitioners who are working with these systems, the immunology research scientists, and other stakeholders in the domain trust the decisions made by the DNNs and that can only be achieved through sufficient interpretation. The goal of interpreting a model should be to explain the model such that the experts trust and understand how it works and produces outputs without them having background knowledge about DNNs or machine learning.

Antigen presentation is an important research problem in Immunology [1]. More specifically, this work will explore antigen presentation by major histocompatibility complex (MHC) class I proteins, which is an important process in the human adaptive immune response. The process starts with peptide generation from proteins. The MHC-I molecule can present peptides on the cell surface. In the case of MHC-I molecules, peptide-MHC class I (pMHC) complexes are presented on the surfaces of cells for recognition by the T cells. The interaction between T cell receptors and pMHC complexes triggers T cells to start a cellular immune response [2].

This biological process above can be split into the following parts:

- Antigen (peptide) presentation by MHC-I proteins
- Recognition of the presented antigens by the T cell receptors
- Cellular immune response

We will restrict our work to the first part of this process which is antigen presentation by the MHC-I molecules. In machine learning terms, we translate this to a classification task - a machine learning model (ImmunoBERT) is trained on a dataset of experimentally collected samples and classifies new unseen samples to categories 0 and 1 based on whether a peptide will be presented by MHC-I molecule or not. Classification of peptides into categories by the model which was trained on a large and representative dataset aims to provide insights based on data about antigen presentation and enhance our understanding of the process.

1.3 Aims

By applying the DC-Causal technique grounded in causal theory, which was proposed as a potential interpretability technique in the field of image classification, we aim to explore a novel application to interpret a machine learning model (ImmunoBERT) that classifies amino acid sequences. The analysis of the explanations produced by the technique will allow us to gain an insight into how ImmunoBERT makes classification decisions, what patterns the model captured from the training data and whether there is an agreement between what ImmunoBERT has learnt about peptide presentation by MHC-I molecules and the findings in biological research.

Further, we aim to evaluate the interpretability coverage and efficiency of applying the DC-Causal algorithm and assess its suitability for interpreting ImmunoBERT. We will evaluate the technique by statistical analysis of the explanations produced by it and compare the explanations to explanations produced by other state-of-the-art interpretability techniques.

1.4 Project contributions

1. We implement the DC-Causal interpretability technique from the field of image classification as a novel application for the task of explaining the model's (ImmunoBERT's) classification of amino acid sequences.
2. We conduct experiments to analyse the explanations produced by the technique to understand ImmunoBERT's decision making and evaluate what the model has learnt from the data about the process of antigen presentation.
3. We assess the coverage, quality, and validity of the explanations produced by the DC-Causal technique.
4. We design and conduct experiments to compare the explanations produced by the DC-Causal technique to the explanations obtained from other state-of-the-art interpretability techniques.
5. We use a similarity measure to compare ranked explanations.

Chapter 2

Background

This chapter describes the basics of the biological and machine learning concepts that are relevant to our work. We provide a brief introduction of amino acids, peptides, proteins, antigens, and the MHC-I pathway, which we believe are useful to know for understanding the context of the application of the model and reasoning about the model's interpretation. We will introduce the machine learning model, ImmunoBERT, that will be used to classify amino acid sequences. Finally, we will introduce the concept of causality which is the foundation for the DC-Causal technique.

2.1 Biological concepts

It is useful for the reader to understand some of the basics of the underlying biology. In particular, that strings of amino acids are used to build peptides and proteins. In general, peptides are shorter than proteins, and proteins are built up from hundreds to thousands of amino acids.

Proteins are essential for the function of our body and the immune system. They are complex molecules created by chaining together amino acids that are connected via peptide bonds [6]. Depending on the 3D structure of the molecule, proteins have numerous essential functions. Examples of proteins include antibodies, hormones, enzymes, and many others. Hormones are messenger proteins that coordinate processes between cells, tissues, and organs. Enzymes are proteins that facilitate and speed up chemical reactions in cells and help with the formation of new molecules [7]. Proteins that have a similar function to T cells are antibodies - they are involved in the immune response, they bind to specific foreign substances to protect the body and control which proteins will enter the cells. In particular, they can protect the cells against substances that come from cancer or viruses. The 3D molecular structure of proteins is determined by the way how the sequence of amino acids is arranged [8].

The organic molecules that are used to build proteins are amino acids. They are small organic molecules linked together by peptide bonds to create one or more chains called polypeptides. There are 20 amino acids occurring in nature. Amino acids are coded by combinations of three DNA nucleotides, determined by the sequence of genes. Each

amino acid consists of a central carbon atom linked to an amino group (—NH2), a carboxyl group (—COOH), a hydrogen atom, and a variable side chain (organic R group) which determines the properties of the amino acid [9]. Amino acids are the basic building blocks of proteins, the longest linear sequence of amino acids within any protein is the primary structure of that protein [6].

Knowing what proteins and peptides are will enable us to understand the MHC-I pathway which is an antigen presentation pathway to which we are applying machine learning to gain some insights into the process of peptide presentation by MHC-I proteins. The pathway enables the detection of antigens like cancer cells and viruses by the immune system. It presents parts of antigen proteins (peptides) from inside a cell on its membrane surface, allowing immune cells, one example of such cells being T cells, to detect these peptides and terminate the cell. It is essential for notifying the immune system that there are infected cells present. MHC-I molecules are expressed on the cells' surfaces and present peptide fragments derived from proteins [10].

Further, we would like to clarify that the literature often mentions 'antigen presentation' but we more specifically refer to 'peptide presentation' throughout this work. An antigen is any molecule or substance capable of stimulating an immune response. Each antigen has different surface features and epitopes, which are parts of the antigen molecule that are being recognized by the T cells or antibodies [11]. Antigens can be very small, only containing a few amino acids, and in this case, the whole antigen is in contact with the T cell or antibody - these small chains of amino acids can be referred to as peptides and this is the kind of antigens we will consider in our work. In contrast, other antigens can be bacteria, viruses, or other substances. We are only considering the small peptides which correspond to the epitopes of protein antigens.

MHC-I is a molecule that spans the cell membrane. Its function is to bind fragments of peptides derived from substances like pathogens and code for proteins displayed on the cell surfaces. It helps the immune system to detect viruses, cancer, or macrophages that have ingested infectious microorganisms [12]. The cell can recognize a foreign fragment attached to the MHC-I molecule and binds to it. MHC-I molecules are one of two classes of major histocompatibility complex. The human leukocyte antigen (HLA) complex is a human equivalent of the MHC complex. The HLAs corresponding to MHC class I are HLA-A, HLA-B, and HLA-C [13]. MHC is polygenic as it contains a variety of MHC genes [14]. Every individual possesses a different set of MHC molecules with different ranges of peptide-binding abilities. MHC is also polymorphic so the gene tends to vary in the population [15].

2.2 Machine Learning: ImmunoBERT

Bidirectional Encoder Representations from Transformers (BERT) is a multi-layer transformer encoder. It uses bidirectional pre-training on unlabeled textual data and masked models to improve approaches to solving tasks in the domain of Natural Language Processing (NLP). The attention model which can learn contextual relations between words enables BERT to gain a deeper insight into the dependencies in text. BERT is associated with the novel 'Masked language modeling' technique where a

proportion of words in each sequence are replaced with $< MASK >$ token before training the model. The model predicts the original words based on the other words in the word sequence. It has been shown that fine-tuning a pre-trained BERT model only requires 1 additional output layer to create state-of-the-art DNNs which have been applied to solve NLP challenges such as language inference [16].

The BERT language model outlined above was used as a base model to create ImmunoBERT [17] which is the machine learning model we use. The model takes as input an amino acid sequence of variable lengths which is comprised of 4 parts: peptide, MHC-I protein sequence, N-flank, and C-flank. ImmunoBERT uses a pre-trained Tasks Assessing Protein Embeddings (TAPE) transformer [18] that facilitates transfer learning and allows the model to predict whether a peptide will be presented by MHC-I protein. Each input sequence is represented as an embedded vector and passed through the TAPE encoder with 12 self-attention layers and 12 heads per each attention layer. The model uses a multi-layer perceptron with two fully connected layers with a hidden dimension of 512. It has one output neuron that uses a sigmoid activation function for the presentation probability [17]. The prediction is a continuous value in the interval [0,1] but a threshold is applied at 0.5 which splits the sequences into two classes: class 0 for negative sequences where the peptide is not presented and class 1 for positive sequences where the model predicts that peptide is presented. The threshold enables us to convert the prediction probability to a class representing whether a peptide will be presented. The conversion from prediction to classification task is necessary for applying our explored DC-Causal technique and the other interpretability techniques that we use for comparison.

2.3 Explainability and causality

Explainable AI involves approaches and techniques which attempt to increase our trust in AI models and ease the comprehension of results produced by machine learning algorithms. Explaining a model not only allows us to interpret its decisions but is useful for improving its accuracy, testing the model, and debugging purposes. When we do not understand how a model makes decisions, we cannot ensure representative inclusion of all groups in the training data or detect and resolve bias. One of the common criticisms of explainability is that the techniques under this umbrella term can provide information about how a model makes decisions but those techniques do not answer the question of why. Explainability focuses attention on decision-relevant parts of the algorithms that either contribute to the model's performance on the training set, or the process of decision making when the model encounters new unseen data [19]. In contrast, causality attempts to find causal relationships between the input features within the training data. Techniques grounded in causal theory such as DC-Causal which we explore in this work are becoming an important element of explainable AI. The foundations in causal reasoning can help the methods achieve a higher level of interpretability. Causal approaches are the next step from solving problems of correlations and patterns in data by finding causal relationships in the data. However, the research on this subject suggests that it is difficult to learn causal relationships from observed data without introducing restrictive assumptions about the model and data [20].

More formally, causation is a relation between two events A (the cause) and B (the effect) when A causes B. Counterfactual theories to define causation in terms of a counterfactual relation such that [21]:

An event A causally depends on B if, and only if:

1. If B had occurred, then A would have occurred.
2. If B had not occurred, then A would not have occurred.

DC-Causal uses a framework of actual causality proposed by [22] which extends counterfactual reasoning by considering contingencies which are defined as changes in the current setting. Chockler et al. [23] define an actual cause such that it is based on the concept of causal models, consisting of a set of variables and structural equations describing dependencies between the variables. Actual causes are defined for a given causal model and context (an assignment to the variables of the model), and a propositional logic formula that holds true in the model. The used definition by Halpern [24] states that:

Definition: A subset of variables and their values in a given context is an actual cause of a Boolean formula being True if there exists a change in the values of other values that creates a counterfactual dependency between the values of X and ϕ (if we change the values of variables in X, ϕ would be falsified).

Although there are many existing interpretability techniques, interpretation of models still is a challenging research question. Many techniques increase the transparency of the model's working and improve our understanding of how a model makes decisions or which features are considered most important. However, knowing how a model makes a certain decision and looking at patterns or correlations is not enough to trust the model if we are not sure why a decision was made. This fact has been proven many times in research and practice. An example from the biomedical domain is a DNN which has learnt to detect a metal token placed on the patient when an X-ray image was taken and this feature was incorrectly correlated with disease prevalence, affecting the predictions of the algorithm [25].

Chapter 3

Related work

To provide some context for our approach we introduce some state-of-the-art interpretability techniques. They differ in numerous aspects such as the scope of explanations and computational complexity. Some of the techniques are gradient-based, which use gradients of DNNs to evaluate the contribution of a selected feature to the model's output. Examples are integrated gradients [26] and Grad-CAM [27]. In general, gradient-based interpretability techniques are well-studied and have contributed to explaining DNNs, especially in the field of computer vision. In contrast, there are perturbation-based techniques that modify the input to a DNN and observe the changes in the output which indicate which parts of the input are most important. The latter category of techniques has advantages due to higher reliability and less noise in the explanations [28]. Examples of perturbation-based techniques which we will introduce are LIME, SHAP, SFL, and DC-Causal. Finally, we will briefly introduce the third category of techniques that are used to interpret Natural Language Processing tasks.

3.1 Local interpretable model-agnostic explanations

Local interpretable model-agnostic explanations (LIME) is an approach that uses simpler models to explain an individual prediction of a DNN by local approximation. The model, which is simpler in its complexity, learns from local samples around the provided input instance and produces a local explanation.

LIME perturbs the inputs, feeds them into the DNN, and observes the corresponding predictions. New perturbed instances are created by randomly masking individual features from the original instance. A feature is 1 if it is included and 0 if it has been masked. The perturbed samples are weighted based on their distance to the original instance. The newly generated dataset is used to train a simpler model, such as linear regression or a decision tree, with the desire that this newly trained simpler model will be a good local approximation of the original model. Each data point is then interpreted with the simpler model [29].

The advantage of LIME is that it works reliably for images, text, tabular data and provides flexibility when selecting the simpler model to produce the most suitable

explanations [30]. The time requirements to run this technique are low compared to techniques like SHAP or DC-Causal.

The disadvantage is that the technique might produce explanations that are not realistic for a given dataset or model where the explanations of two very close instances can be inconsistent. The random sampling of perturbed data points for the simpler model uses a Gaussian distribution which assumes independence of features and can produce instances that would never occur in the data. Further, [31] shows that LIME can be used to generate models whose explanations can be controlled and manipulated to mask biases. It concludes that LIME is not sufficient for ascertaining discriminatory behavior in sensitive applications which makes it more difficult to trust explanations produced by this technique.

3.2 Shapley Additive exPlanations

Shapley Additive exPlanations (SHAP) is a local interpretability technique, which uses Shapley values to distribute scores in a fair manner among all features of an input instance by calculating the contribution of each feature to the overall model's prediction.

The Shapley values, which are well-known in the field of game theory, determine how to distribute a prediction amongst features in the most optimal way by assuming that each feature value is a player in a coalition and the prediction of the model is the payout [30]. SHAP weighs the instances according to the weight the coalition would get in the Shapley estimation. If a coalition consists of a single feature or a coalition consists of all features except for one, we can isolate the effect of that particular feature and its contribution to the model's prediction.

SHAP considers the scenario of making an individual prediction to be a game where it explains the gain by taking the prediction of an instance which is being explained and subtracting the average predictions for all other instances. The feature values which are not masked are splitting the gain (prediction value) - this is repeated for all possible subsets of the original feature space and the Shapley value for each feature is calculated as the average marginal contribution of that feature across all coalitions [32].

Computing all possible coalitions for models like ImmunoBERT in which we consider larger feature and feature-value spaces is computationally infeasible since the solution increases exponentially as a function of the features and feature values. There are approximations available or an option to use Monte-Carlo sampling [33]. In addition to SHAP, Lundberg et al. [34] proposed KernelSHAP which is a kernel-based approach inspired by local surrogate models. It guarantees fair distribution of the difference between the prediction and average prediction for the features. Similar to our explored approach of compositional responsibility, it produces contrastive explanations and has a grounding in theory.

Disadvantages of this approach include the need to approximate a solution due to the computational complexity of evaluating 2^k coalitions. Using an approximation decreases the precision of Shapley values because it involves sampling random instances which increases the variance of the estimates. This method also assumes independence of

all features because when it marginalizes a feature, it samples a feature from its value distribution and it might produce instances that are impossible in the real-life context of a problem. This method is worse than our explored compositional responsibility or LIME in the sense that we require access to all data to calculate the Shapley values for the new instance, rather than just using the model and its prediction. It is also questionable whether the Shapley values improve the model's interpretability because the outputs of the techniques themselves often tend to be complex.

3.3 Statistical Fault Localisation

Statistical Fault Localisation (SFL) is an interpretability technique taken from the domain of software testing, proposed by the creators of the Deep Cover tool [35], as a novel application to the field of Explainable AI. SFL is an effective method for localizing the causes of failures in code. It ranks different parts of code according to a score calculated by various well-known fault localisation measures such as Tarantula and Zoltar.

Let us define a sub-part to be the smallest unit of an instance, such as a pixel of an image, a letter in a word, or a single amino acid in a peptide sequence. Given an input instance X and its output prediction y, the method produces a set T(x) of random mutants X'. Each mutant masks a random subset of the input and these mutants are passed into the DNN which predicts the output y' for each mutant. Using this set of mutants, the individual parts of the input are ranked according to a selected SFL measure and for every sub-part of the input instance, the technique computes vector $\langle a, b, c, d \rangle$ where a is the number of mutants in the set of mutants labeled with the same label as the original input where the considered sub-part is not masked, b is the number of mutants in the set of mutants labeled with the opposite label where the considered sub-part is not masked, c is the number of mutants in the set of mutants labeled with the same label as the original input where the considered sub-part is masked, d is the number of mutants in the set of mutants labeled with the opposite label where the considered sub-part is masked. The SFL measure is applied to a set of vectors, where each one represents a sub-part of the original instance that is being explained, to determine and rank the importance of each sub-part. Following the algorithm in Sen et al. [35], an explanation is created by first masking the original input completely and then iteratively adding sub-parts to the explanation, starting with the sub-parts with the highest importance and continuing until a point is reached when the set of sub-parts becomes sufficient to classify an instance.

The advantages of this method include efficient computation of a good approximation of an explanation. The computational complexity is lower than the complexity of SHAP and DC-Causal. This approach has been shown, with the support of benchmark experiments, to have a higher explanation accuracy than other state-of-the-art techniques such as LIME, SHAP, and Grad-CAM for the task of constructing an explanation of the classification using the VGG16 model [36] which received as input randomly sampled 1000 images. SFL is model-agnostic so it can produce explanations for any DNN, although it needs to be tested on more models and various kinds of tasks to support a wider acceptance.

A disadvantage of SFL is that it has not been extensively tested for explaining the decisions of a DNN when the input data are not images. The quality of the explanations depends on the quality of the generated mutants. To mitigate the risk of generating an unbalanced set of mutants, the DC SFL paper proposes an algorithm that ensures the balance between the classified and misclassified mutants. It is important to note that this technique, similarly to SHAP and LIME, can produce mutants of the original input which would otherwise not be possible in the real data.

3.4 Approaches taken from NLP

This section covers techniques from the field of Natural Language Processing (NLP) to put our explored DC-Causal in the context of a wider family of interpretability techniques. We mention approaches from NLP because they attempt to address the view that amino acid sequences form structures in which the individual amino acids are not necessarily independent of each other. A single change in an amino acid sequence can entirely change the protein's structure and function [37] so assuming independence of amino acids in a sequence could potentially become problematic. This constraint is similar to NLP problems because words in sentences are not independent. This group of methods includes sample-based and semantic explanation methods which create short explanations rather than using the entire length of input data. The reason why we do not pursue the exploration of these techniques is that we would require a language model trained on large amounts of annotated data which we do not have at our disposal. These techniques tend to be computationally expensive and elaborate as they require extra feature engineering or training of an additional model.

Language modeling is an approach that has been shown to solve challenges in NLP such as predicting relationships between words and sentences by analysis of the context of words. Occlusion and Language Models (OLM) is an interpretability technique that takes advantage of language models and occlusion to sample valid and syntactically sensible replacements with high likelihood, given that the context of the original input is known. The focus is on the syntactic understanding of the language represented as a discrete distribution. This approach assumes that local regions and neighborhoods do not have to be representative of the model's behavior and the model's predictions at points with zero probability are not contributing to the model's decision-making process [38]. The advantages include that this method was shown to perform well on sentence classification (as a sequence of words) and was tested on the most popular NLP models. The disadvantages include that this method was designed to consider words in a sentence as language features whereas for our task we would need to change language features to individual amino acids - this change of features might affect the accuracy of the technique negatively.

The minimal Contrastive Editing (MICE) technique is an NLP-applied approach for generating explanations by producing contrastive explanations of the model's decisions. The aim is to change instances such that they change model outputs to the contrast case. The technique can be split into 2 parts: First, an editor model is trained to map word edits with labels of the model. It is followed by the process of masking the most important parts of the text for the given label and training the editor model to reconstruct

these parts of text given the masked text and target label as inputs. In the second part of this technique, contrastive edits are generated by masking different proportions of the input instance and giving masked inputs together with contrast labels to the editor model. The results have shown that MICE is capable of producing edits that are not only contrastive but also minimal and in agreement with human interpretations [39]. Initially, this approach was applied to solve NLP problems such as sentiment and topic classification. The advantage of MICE is that it can make contrastive changes for any discriminatory predictor and since it has achieved an accuracy 95.9% on the IMDB dataset, it might be a promising approach to apply to amino acid sequences and our classification task of interest. MICE would be capable of creating explanations of model predictions in the form of targeted changes that cause the ImmunoBERT model to change its original prediction to the contrast prediction. MICE has been applied to sentences where it had to produce insertions and deletions that would produce contrast cases. This technique shares some similarities with DC-Causal as it also tries to change the model outputs to the contrastive case by masking and occlusion. It might be worthy of an argument that sentences are closer to sequences than images are, but we do not have a language model and a sufficiently large dataset required for this approach.

3.5 DC-Causal technique

The DeepCover Causal (DC-Causal) technique proposed by Chockler et al. [23] is a method that aims to provide causal explanations and is based on causal theory, which was introduced in section 2.3. The technique was designed as a tool for interpreting the classification of occluded images and was motivated by the ideas of compositionality and responsibility. It is a perturbation-based approach that tries to address the current limitations of other perturbation-based techniques by using the concepts of causal reasoning. Actual causality, apart from interpreting the model and its decisions, considers counterfactual reasoning and contingencies. In a sense, it is a novel approach to produce explanations as there is an element of causality involved, with strong theoretical foundations in causal theory.

The method uses the theory of actual causality and defines responsibility that quantifies causality and expresses the degree of responsibility for any actual cause defined as the minimal change required to create a counterfactual dependence. The responsibility is defined as 1/(1+k) where k is the size of the smallest contingency. The responsibility measure for an actual cause can take any value between 0 and 1, where 0 means no causal dependency and 1 means very strong causal dependency. We will cover this technique and its implementation in more detail in the following methodology section.

Chapter 4

Methodology

This chapter starts with describing data selection and encoding of input sequences. We present the DC-Causal technique with detailed explanations of key steps of the algorithms and justify the adaptions we made to interpret ImmunoBERT.

4.1 Data

4.1.1 Data sources

The training data used for training ImmunoBERT plays a significant role in the ImmonoBERT's decision-making process as everything that the model has learnt is conditioned on this data. Obtaining a well-representative and inclusive dataset of sufficient size with required samples is challenging. ImmunoBERT's training data is comprised of two distinct data sources, a collection of peptides from EL assays mapped to the GRCh38 Homo sapiens reference genome combined with proteins within the Ensembl v94 database [40] and the second source being the HLA Ligand Atlas [41]. The data is pre-processed such that it only includes sequences with HLA proteins and where a peptide of specified length of 7-15 amino-acids is observed. Each sample in the data was obtained in an experiment which measured the presented peptides from a cell-line or an individual. The experiments only provided observations which are positive samples where a peptide was presented. To achieve the best results, the dataset had to be balanced so negative samples with no peptide presented needed to be artificially created. For each sequence associated with a peptide presented on the MHC-I protein, there was one decoy (negative sample) generated by randomly selecting a position within the protein of the positive sample as the starting point of the negative sample's peptide. This procedure resulted in having 50/50 class split in the data and is considered to be a standard approach proposed by J. O'Donnell et al. [42].

For interpretation, the goal was to match the distribution of the training data, maintain a balanced dataset and produce datasets which would allow for a fair comparison which other interpretability techniques. We follow a similar sampling procedure to the method proposed by Gasser et al. [17] and restrict the data to 9-mer peptide observations and decoys which results in sequences of length 73 if we ensure that n-flank and c-flank

parts of the sequences are present. We sample the test dataset to observe the ability of ImmunoBERT to classify unseen sequences. We created 12 experimental datasets, each having 1000 sequences with 50/50 split of positive and negative class. Each dataset corresponds to a particular MHC allele:

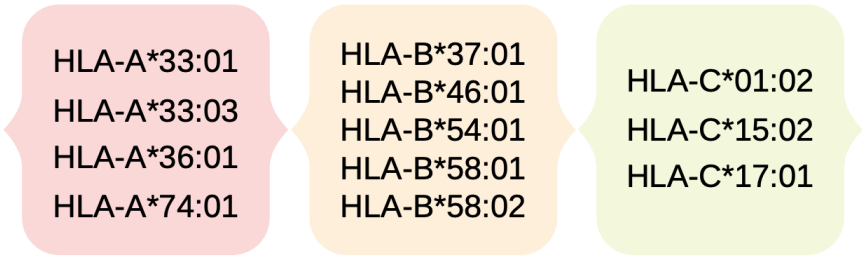


Figure 4.1: HLA alleles - data split

4.1.2 Encoding of input sequences

The inputs to the ImmunoBERT model are uniquely identified embedded vectors represented as embedding matrix of tensors. For easier interpretation of the input sequences, similarly to the approach taken by [17], we converted each input sequence into an array of integers of length 73 representing each input as [n-flank, peptide, c-flank, MHC]. The first 15 positions of the array represent n-flank, following 9 positions encode the peptide sequence, following 15 positions encode the c-flank and the rest encodes the MHC sequence as shown on figure 4.2. Each index represents a specific position within the sequence array and the value in the sequence array represents unique ID of the amino acid present at that position. Each amino acid is encoded as a specific integer value according to TAPE vocabulary [18]. The justification for this encoded representation of sequences is that it contains equivalent information to the original embedding matrix with respect to what is required for the interpretation but allows for direct interpretation of the sequences and enables us to produce ranked lists of positions.

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,
39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56,
57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72]

legend: n-flank | peptide | c-flank | mhc

Figure 4.2: Input sequence encoding

4.1.3 Masking justification

The DC-Causal method involves the process of mutating sequences in which a part of the input sequence needs to be masked. Our initial approach involved the use of BLOSUM62 substitution matrix [43] which assigns similarity scores to all pairs of amino acids. We explored substitution of an amino acid with the most distant and most similar amino acid. We also explored random replacement in which we replaced an amino acid which we wanted to mask with an amino acid randomly drawn from a prior distribution of amino acids. In addition, we tried sampling an amino acid from a posterior distribution of amino acids conditioned on the position in the sequence. None of these techniques has shown to improve the performance of DC-Causal or contribute to producing more informative explanations. The best performance according to our specified metrics that measured the technique's coverage and quality of explanations was achieved when we used encoded $< MASK >$ token. Therefore, we used this masking approach in all of our experiments. The reason why using the $< MASK >$ token in mutants contributed towards better performance of the technique is supported by our observation that ImmunoBERT is not very sensitive to small changes in individual amino acid values and the model was trained with the use of the $< MASK >$ token as a part of its known vocabulary of amino acids. We noted that variation achieved by replacing an individual amino acid X for some other amino acid X' does not change ImmunoBERT's classification for small number of replacements. The original DC-Causal paper [23] validates our approach of selecting the $< MASK >$ token for masking by empirically showing that the choice of masking had very little importance on the performance of the algorithm so our approach for the evaluation was to select the masking technique which produced the most informative results for our application of the technique.

4.2 DC Causal algorithm

We present the implementation of the DC-Causal algorithm proposed by Chockler et al. [23] for the task of explaining classification of amino acid sequences. We provide more detail about the algorithm steps which follow the original implementation. We highlight the differences and adaptations that differentiate our algorithm from the original version and adapt it to achieve better performance when interpreting ImmunoBERT.

4.2.1 Definitions and overview

Definitions. We define the cause for the ImmunoBERT's classification of an input sequence X to be a group of amino acids, this group being a subset **c** of X, if and only if there exists a witness subset S comprised of amino acid groups (subsets) of X such that the following statements hold true:

1. S does not contain **c** (a subset **c**, that we are currently considering as a potential actual cause for classification, is not contained in the witness subset S).
2. Masking any amino acid (or equivalently unmasking any already masked amino acid) or group of amino acids in S does not change the classification of the

sequence (where unmasking means insertion of an amino acid a to a position p such that sequence[p]=a occurs in the original input sequence).

3. Masking the subset **c** which we are currently considering as a potential actual cause, masking all unmasked amino acids and unmasking all masked amino acids in the witness S changes the classification of the sequence X.

If 1, 2 and 3 hold, it follows that the subset of amino acid groups S is the witness to the fact that an amino acid group **c** is the actual cause for the classification of the sequence X.

Assumptions. Similarly to [23], we made the following assumptions which enabled us to apply the algorithm to the problem of explaining ImmunoBERT's classification decisions:

- We only consider singleton actual causes and assume that individual amino acids in a sequence are independent. The reason is, we cannot assume any prior dependence between the positions or individual amino acid values because there are no known dependencies in the data between the amino acids within an input sequence.
- The amino acids with the highest responsibility are located within the groups of amino acids with the highest responsibility. The reasoning is that if a group of amino acids contains one or several amino acids with a high importance contribution towards the DNN's decision, it follows that the entire group will obtain a non-zero responsibility.

High-level overview. The algorithm produces smaller explanations by only including parts with non-zero responsibility and sorting sequence positions according to their responsibilities. It calculates responsibility of a group of amino acids (a subset of the original sequence instance) and recursively distributes this responsibility further to all amino acids within that group. The process can be broken down into the following parts:

Algorithm 1: Responsibility - Takes a partitioned sequence or partitioned subset of a sequence (where partitioned means split into s non-overlapping parts) and calculates the responsibility of each part. Let one of the parts be p. The responsibility of p is calculated by selecting a mutated sequence from the space of mutated sequences which satisfies the following:

1. p is not masked in the mutated sequence
2. the mutated sequence has the same classification as the original sequence
3. masking p changes the classification of the mutated sequence
4. a mutated sequence with the minimum number of masked amino acids satisfying conditions 1,2,3 is selected

Algorithm 2: Compositional Responsibility - Given the result from the responsibility algorithm 1, refine amino acid groups with non-zero responsibility by applying

the Responsibility algorithm again on a more refined granular level.

Algorithm 3: Compositional Explanation - Start the Compositional Responsibility algorithm 2 again with the same input sequence partitioned in a different way. Repeat N times (where N is the number of iterations of the algorithm), merge the results from all iterations and finally sort the amino acids in the descending order of responsibility. The reason for trying different initial partitions of the sequence is that the initial partition influences the final responsibility values given to each amino acid due to the random nature of the partitioning process which needs to be repeated multiple times to improve the accuracy of the responsibility assignment. From this ranked list of amino acids, an explanation can be iteratively constructed by masking the entire sequence and adding the amino-acids with the highest responsibility one-by-one until the masked sequence has the same classification as the original sequence.

4.2.2 Description of the algorithm

Responsibility

Inputs: amino-acid sequence, partition of positions, masking method
Outputs: responsibility map mapping each amino acid group to a responsibility value

Key steps:

1. Given a partitioned sequence as an input, create a mutant space for the partitioned input by masking the input instance using the provided masking method - all possible mutation combinations = 2^s - 1 where s is the number of randomly chosen groups into which we split the sequence.
2. Calculate the degree of responsibility of each amino acid group in partition by finding the minimum difference k between the mutant sequence and original sequence such that the part which we are considering as actual cause candidate is not masked, the class of the mutated sequence is equal to the class of the original sequence and masking the part which we are considering as actual cause candidate will result in a change of classification. This minimum difference which represents the number of masked amino acids in the selected mutant will be assigned to k.
3. The final responsibility r for a given amino acid group is calculated as 1/(1+k).
4. After computing responsibility for each amino acid group, return the responsibility map which maps each group of amino acids to its r value.

Compositional Responsibility

Inputs: amino acid sequence, partition of positions, number of parts s, responsibility map, masking method
Outputs: updated responsibility map mapping amino acids to respective responsibility values

Key steps:

1. Recursively apply the responsibility algorithm (1) to refine the amino acid groups with non-zero responsibilities until the point of termination is reached. Select positions which occurred in amino acid groups with non-zero responsibilities, partition them again and apply the responsibility algorithm to this new shortlisted (smaller) subset of partitioned positions.
2. Iteratively repeat on a smaller subset of amino acids in every iteration and after every iteration update the responsibility map.
3. Refinement of shortlisted positions finishes when a termination condition is reached. The termination condition is reached when the length of the shortlisted positions (which need to be considered for further refinement) is less than the number of parts s. This occurs when each part contains at most 1 singleton position.
4. Termination can also occur if all parts have equal responsibility (this includes the common case in which all parts have responsibility value 0).

Compositional explanation

Inputs: amino acid sequence, number of tried random partitions N , number of parts s, masking method
Outputs: an explanation (minimum subset of responsible positions)

Key steps:

1. Repeat application of compositional responsibility (algorithm 2) to a different random partition split in every iteration (N specifies the number of iterations).
2. Average the degree of responsibility for each position over the set of all introduced partitions.
3. After finishing N iterations of applying compositional responsibility algorithm, rank the positions according to their degree of responsibility in descending order.
4. Mask all 73 positions of a sequence. Iteratively add one-by-one amino-acids at the most important positions (with the highest degree of responsibility) in the sequence until the classification of the explanation is not equivalent to the classification of the original sequence.
5. Return the explanation as the minimal explanation with respect to the definition introduced in the DC-Causal paper.

4.2.3 Justifications of adaptations for ImmunoBERT's interpretation

We made several design decisions to make our algorithm more suited for explaining classification of amino acid sequences. The decisions about value assignment to the parameters were made after analysing the initial performance of the technique. The following changes differentiate our algorithm from the original DC-Causal algorithm:

Increased the number of splits - parameter s: The original algorithm split the input images into 4 parts. We consider sequences of length 73 and this setting resulted in individual parts being large in size. The algorithm would terminate immediately in $>$ 90% cases. We varied the number of splits from 2 to 7 and found that the most optimal number is 5 to minimize early termination of the algorithm that occurs when all parts have responsibility 0. For $s < 5$, we frequently observed early termination without finding an explanation for a sequence (even for larger N) because no such mutant which would satisfy the responsibility criteria was found. In contrast, increasing s to values greater than 5 would significantly increase time complexity of the algorithm and would not contribute towards higher quality of explanations or higher coverage of explained sequences.

Greedy first partition selection: The randomness of the process of splitting the sequence into s parts implies that there are many partitioning possibilities and a reasonably high probability that the partitioned sequence does not produce mutant space in which any member satisfies the responsibility criteria - given how sparse these mutants which contain an actual cause are. As a consequence, each part of the sequence is assigned the importance score of 0 and the algorithm terminates immediately. We avoided early termination by trying new random initial partitions 30 times until we let the algorithm to terminate. This has increased the average proportion of sequences which were possible to explain by the algorithm to 63.6%, which is the highest achieved coverage by the technique for 10 random partitions of sequence (N=10).

Combinatorial refinement: After the first 1-2 refinement iterations, the algorithm would terminate because the amino acid groups were too small to make any difference to the ImmunoBERT's prediction. This is a consequence of the responsibility requirement which is to find a mutant with the same classification as the original sequence but a different classification after masking a single part of the sequence (the actual cause). To avoid this problem, when the partitions were sufficiently small, we combined multiple partitions with non-zero responsibility together to form a pool of shortlisted amino acids which would be partitioned instead of further refining a single partition. This adjustment has increased the interpretability coverage of the algorithm and also the average length of an explanation. In addition, it contributed towards producing more refined and informative explanations.

Data-parallel approach: The time complexity of the original DC-Causal algorithm is $O(2^s nN)$ where s is the number of parts into which we split the sequence, n is the number of amino acids in each sequence and N is the number of random partitions. In our case s=5, n=73 and N=10. To decrease running time we run the algorithm in parallel such that we split the set of sequences into 5 batches and run the same algorithm in parallel on multiple workers in Google Colab, ensuring that each worker has approximately the same workload. This approach increases the speed of DC-Causal 5 times (with the use of 5 workers) which was better performance improvement compared to parallelization by splitting the sequence into s parts and working on each of those parts separately in parallel.

Chapter 5

Experiments

This chapter presents the main research questions which we aim to address. We provide details of design of the experiments and explain specifics of our selected metrics.

5.1 Research questions

RQ1 Do the explanations produced by DC-Causal have solid grounding in Biology?
Explores whether there are any insights provided by the DC-Causal explanations that agree with the findings in biological experiments and studies.

RQ2 What is the coverage and quality of explanations produced by DC-Causal when interpreting ImmunoBERT?
Analyses the suitability of DC-Causal technique for interpreting ImmunoBERT by considering how many amino acid sequences are interpreted by the technique and assessing the quality of provided explanations.

RQ3 How do the explanations produced by DC-Causal compare to other state-of-the-art interpretability techniques?
Aims to compare explanations provided by DC-Causal technique to two other state-of-the-art techniques.

5.2 Design of experiments

We designed a set of experiments to explore ImmunoBERT's classification decisions and evaluate the suitability of application of DC-Causal technique to explain ImmunoBERT. We run each experiment on 12 different HLA datasets, each dataset consisting of randomly sampled 500 positive samples and 500 decoys (negative samples). We describe data preparation in section 4.1.1.

Experiment 1: Peptide analysis. We apply the technique exclusively to the 9 positions of the peptide and fix the other parts of the amino acid sequence (MHC and flanks). In this experiment, only the peptide positions can be split and masked which implies that only the peptide positions can be considered as actual cause of classification of the

sequence. The aim is to observe how the algorithm performs on simpler sequences. The reason why we chose exploring peptide as our base experiment is that our hypothesis, supported by the findings by Gasser et al. [17], is that the peptide sequence is the most important discriminator between the classes so the peptide positions should have the highest responsibility.

Experiment 2: Peptide with MHC analysis. We apply the technique to peptide and MHC sequence without considering flanks. The explanations produced by this experiment rank at most 43 positions of the peptide and MHC. Experiment 2 considers more positions than experiment 1 but avoids the noise of having large amount of positions which could randomly end up in the group with high responsibility and be falsely up-ranked in the explanation if they occur in groups with most important positions. The objective is to determine whether there are differences between the responsibilities of peptide positions and MHC positions. We explore whether the MHC positions which are ranked higher relative to other MHC positions are in the A,B and F pockets in the peptide-binding groove which are the pockets determined by biological studies as areas of great relevance for peptide binding [44].

Experiment 3: Full sequence analysis. We include all 73 amino acid positions in the analysis to evaluate how the technique performs when it is required to explain the entire input sequence. We explore the relative responsibilities of distinct parts of the sequence and the quality of produced explanations.

Experiment 4: State-of-the-art comparison. We run DC-Causal technique and LIME interpretability technique on the benchmark dataset which was sampled and used to evaluate the interpretability of SFL technique. This experiment explores the performance and effectiveness of the 3 techniques and allows to compare produced ranked lists. Sequences used in comparison include peptide and MHC without surrounding flanks.

Experiment 5: Impact of parameter values. Our hypothesis is that the quality of the individual explanations will vary depending on the number of random partitions of sequence we try during every iteration. We will vary the value assigned to N (the number of random partitions of a sequence) such that N=[5,10,20,50,70,100]. We will only include sampled 1/10 of each dataset in this experiment due to increasing time requirements to run the technique for $N > 5$.

5.3 Selection of parameters and justification

The number of parts into which we split the sequence, the number of random partitions the algorithm performs during every iteration and the number of initial attempts which the algorithm tries when it encounters a sequence which it cannot interpret before it moves to interpret another sequence in the dataset were determined empirically in the initial experiments. We made the following decisions, considering the trade-off between refinement of the explanations and the time requirements to run the technique.

For experiments 1-4 we set the number of tried partitions during each iteration (N) to 10 because we have 12 HLA sets, each having 1000 sequences. The empirically

measured time requirement to run the technique on 1000 sequences is 2-10 hours on a cloud-based service Google Colab. The time varies based on the available GPUs, the type of experiment which we are running and whether we split the sequences into batches. Interpreting a sequence of length 9 (experiment 1) requires to split the input to 3 parts which takes less time than the other experiments which consider lengths 43 and 73 which split the input into 5 parts.

In experiment 5 we increase the number of random partitions (N) and try higher values which significantly increases the time it takes to produce 1 explanation. This approach should improve the explanations by further refinement and the produced ranked lists of positions should be of a higher quality. Interpreting 12 datasets of 1000 sequences by trying the large values for N would not be within a reasonable time scope for this project so we only consider a 1/10 subset of each dataset which is randomly sampled from the original dataset but kept constant throughout the experiment (such that all N values are tried on the same subset of sequences) to ensure fair comparison of the quality of explanations produced with different values of N.

5.4 Metrics measuring quality of interpretation

We designed the following metrics that will track the success of the interpretability technique during evaluation.

1. **Length of explanation** - The length of ranked list containing amino acids with non-zero responsibility. Measures how many positions from an individual sequence were occurring at least once in a group of amino acids that was determined by the DC-Causal technique as an actual cause for the classification of that sequence.
2. **Interpretability coverage** - The % proportion of sequences from given dataset for which the technique produced a ranked list of positions.
3. **Class split** - The % split of class 1 and class 0 sequences for which the technique produced a ranked list.
4. **Hits per explanation** - Given that the technique produced a ranked list for a particular instance, we report the % of partitions (out of N partitions) of that sequence that resulted in non-zero responsibility ranked list.
5. **Distinct ranks** - The number of distinct responsibility values (importance scores) produced in a ranked list. Measures the depth of refinement of an explanation and is a proxy of quality of the produced explanation.
6. **RBO score** - A measure of similarity of the explanations (provided as ranked lists) produced by the DC-Causal technique with other state-of-the art interpretability techniques.
7. **Runtime** - The average time (in seconds) it takes to explain one amino acid sequence.

Chapter 6

Evaluation

The evaluation section presents the analysis of the results produced by the DC-Causal technique and assesses the explanations with respect to our research questions.

We present what patterns from the data are captured and learnt by ImmunoBERT by analysing position responsibilities within sequences. By ranking the positions according to their responsibilities we determine the strength of their contribution to the classification of sequences and show which positions the model has learnt are most important. We validate our results by comparing them with two other interpretability techniques, namely LIME and SFL, and we show how biological studies support our findings to answer the question of whether the results have any supportive evidence grounded in Biology. Finally, we test our method and evaluate the coverage and quality of explanations produced by the DC-Causal technique.

We split evaluation to the following parts:

- Peptide analysis
- Peptide with MHC analysis
- Full sequence analysis (peptide, MHC, n-flank, c-flank)
- Quality of produced explanations
- Performance of the DC-Causal technique
- State-of-the-art comparison
- Limitations
- Threats to validity

6.1 Interpretation of ImmunoBERT - RQ1

6.1.1 Peptide analysis

We display the aggregated summary analysis of responsibility rankings of peptide positions from experiment 1 across HLA-A, HLA-B and HLA-C alleles. We show 9

example visualisations, each displaying results for 1 dataset of 1000 input sequences for each allele. The rest of the visualisations can be found in the appendix.

The results show that there are differences in responsibility profiles across the alleles. The technique assigns responsibility to all 9 positions but identifies positions 9,3,2 and 1 as more responsible and therefore important relative to other peptide positions. Some responsibility profiles, example being HLA-A*36:01, clearly distinguish positions and it is possible to rank the positions. In contrast, responsibility profile for HLA-B*54:01 has positions with uniform responsibility scores and assigns a particularly high responsibility to position 2. In general, for HLA-A we observe a strong dominance of position 9 whereas for HLA-B and C, there is larger variance in the responsibility profiles - in some cases position 1, 2 or 3 dominates and is assigned the highest responsibility.

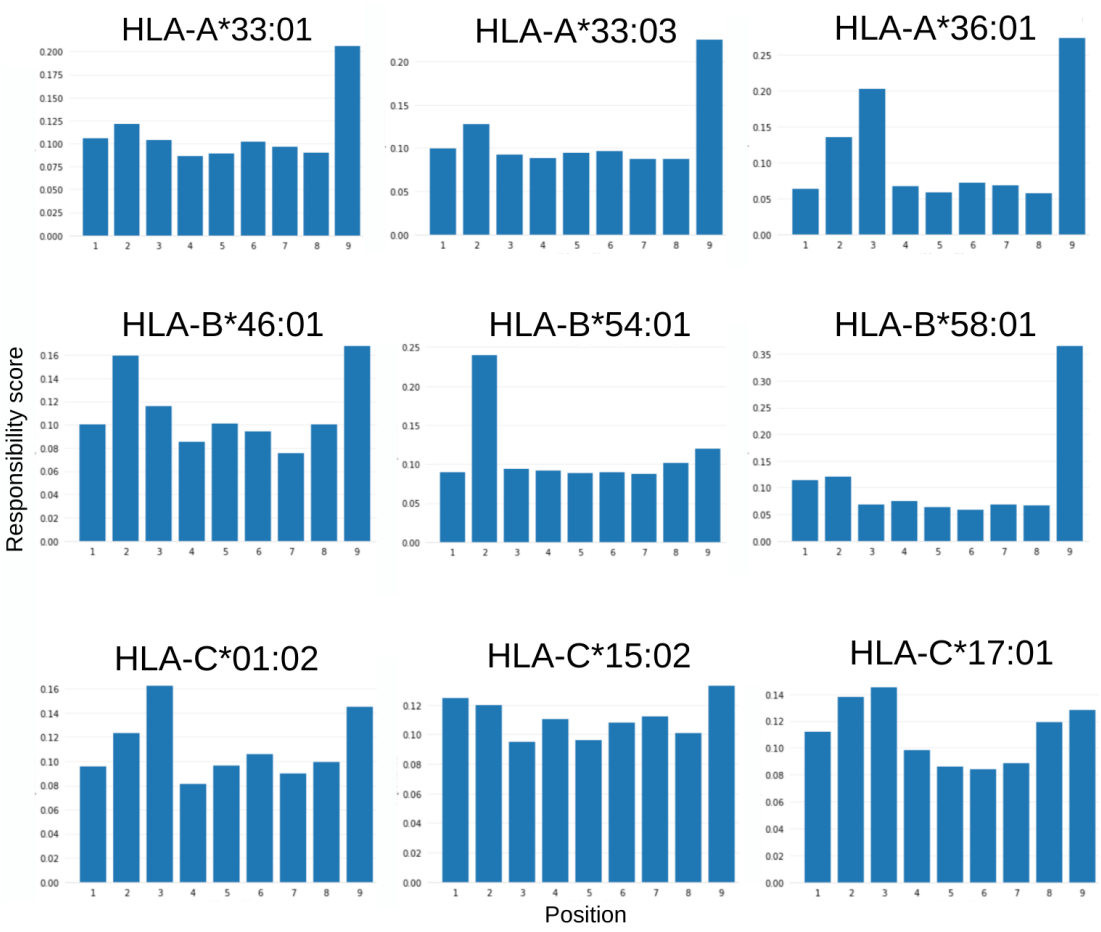


Figure 6.1: Peptide position analysis

We averaged and normalised the responsibility scores per peptide position across all datasets to devise an overall ranking of the peptide positions. Table 6.1 displays normalised responsibility of each peptide position relative to other peptide positions and median rank which represents the average rank of a particular position in an explanation (a ranked list sorted by descending responsibility score). We report the median which in this case better represents the true average rank of a position as it excludes outliers. The results combining all datasets support our earlier findings that position 9 followed

by 1,3 and 2 are most important for ImmunoBERT's classification. These particular peptide positions with highest responsibility scores determined by our method can be found close to the termini of the peptide and are in agreement with the most important positions determined by LIME interpretability technique [17].

position	responsibility	median rank
1	0.105	3
2	0.133	2
3	0.101	3
4	0.080	6
5	0.088	7
6	0.075	7
7	0.090	8
8	0.092	7
9	0.236	1

Table 6.1: Normalised peptide position responsibility scores

6.1.2 Peptide with MHC analysis

We present a summary of experiment 2 results. We focus on MHC positions to explore which positions have higher responsibility relative to other MHC positions shown in Figure 6.2. Firstly, we note that in most cases, MHC positions have lower responsibility scores than peptide positions. Secondly, the responsibility scores of MHC positions are more uniformly distributed and there are fewer differences and spikes in responsibility scores compared to peptide positions.

We compare our results to biological studies. Van Deutekom HW et al. [44] used in silico method which predicted peptide-binding of HLA molecules and measured the effect of how much single substitutions change peptide binding. They found that MHC positions which are situated in the A, B and F pockets are most relevant for peptide binding and peptide presentation. Our results show that in most cases, MHC positions which have responsibility spikes shown on the figures are situated in A, B and F pockets, which is in agreement with this study.

HLA A: HLA-A*33:01 has spikes at MHC position 7 (corresponding to position 63 in the referenced study) that is situated in A pocket and position 12 (ref. pos 73). HLA-A*36:01 has spikes at positions 15 (ref. pos 77) in F pocket and 29 (ref. pos 156) in A pocket.

HLA B: HLA-B*37 has spikes at positions 2 (ref. pos 9), 4 (ref. pos 45), 9 (ref. pos 67) which are all situated in B pocket. HLA-B*46 shows an increased contribution of MHC positions 23 (ref. pos 116) in F pocket, 28 (ref. pos 152) and 29 (ref. pos 156) in A pocket.

HLA C: HLA-C*01:02 allele data show spikes at position 19 (ref. pos 95) in F pocket, 21 (ref. pos 99) in A pocket and 29 (ref. pos 156). HLA-C*15:02 has spikes at positions 10 (ref. pos 69), 20 (ref. pos 97) and 28 (ref. pos 152).

As we have shown, the majority of spikes in MHC correspond to the positions which reside in the A, B and F pocket areas. This supports the theory that binding of peptides

to MHC-I proteins is to a significant extent determined by pockets in the binding groove. In particular, whether a peptide is presented by MHC-I molecule depends on the structure of the MHC protein and the specific features of the binding groove. The spikes correspond to the highlighted pockets which should be responsible for peptide binding and antigen presentation. It is reasonable to expect that ImmunoBERT would try to use these important positions as discriminators between the two classes and when making decisions assign more importance to those particular positions as determinants of peptide presentation. We found that MHC positions in other 6 datasets which we include in appendix follow a similar pattern of spikes at the MHC positions located in A, B and F pockets. In addition, our results are in strong agreement with LIME interpretations of MHC positions in [17].

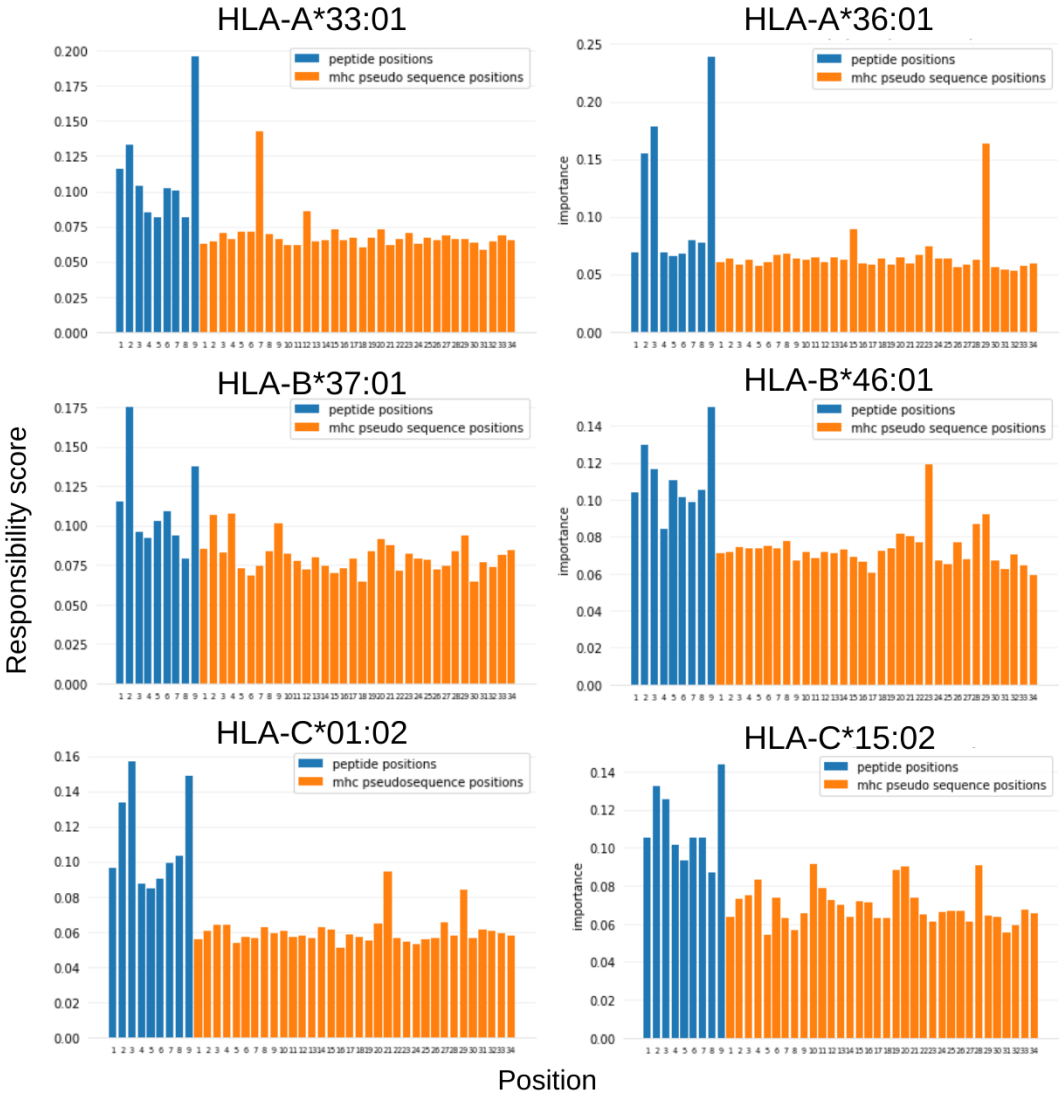


Figure 6.2: Peptide and MHC position analysis

6.1.3 Full sequence analysis

The analysis in this section shows results from experiment 3 which considered all 73 positions within input sequences. There is a clear distinction between positions of the individual parts of input sequences and their respective contributions towards an explanation measured by responsibility score. The peptide positions dominate and are followed by the MHC positions. The n-flank and c-flank have the lowest contribution towards an explanation and their positions are ranked the lowest in the explanations. This importance pattern is present throughout all HLA alleles which were included in our experiments. We observe that there are minimal spikes in the flanks and their importance values are always below 0.05 so the flank positions contribute towards an explanation by small %.

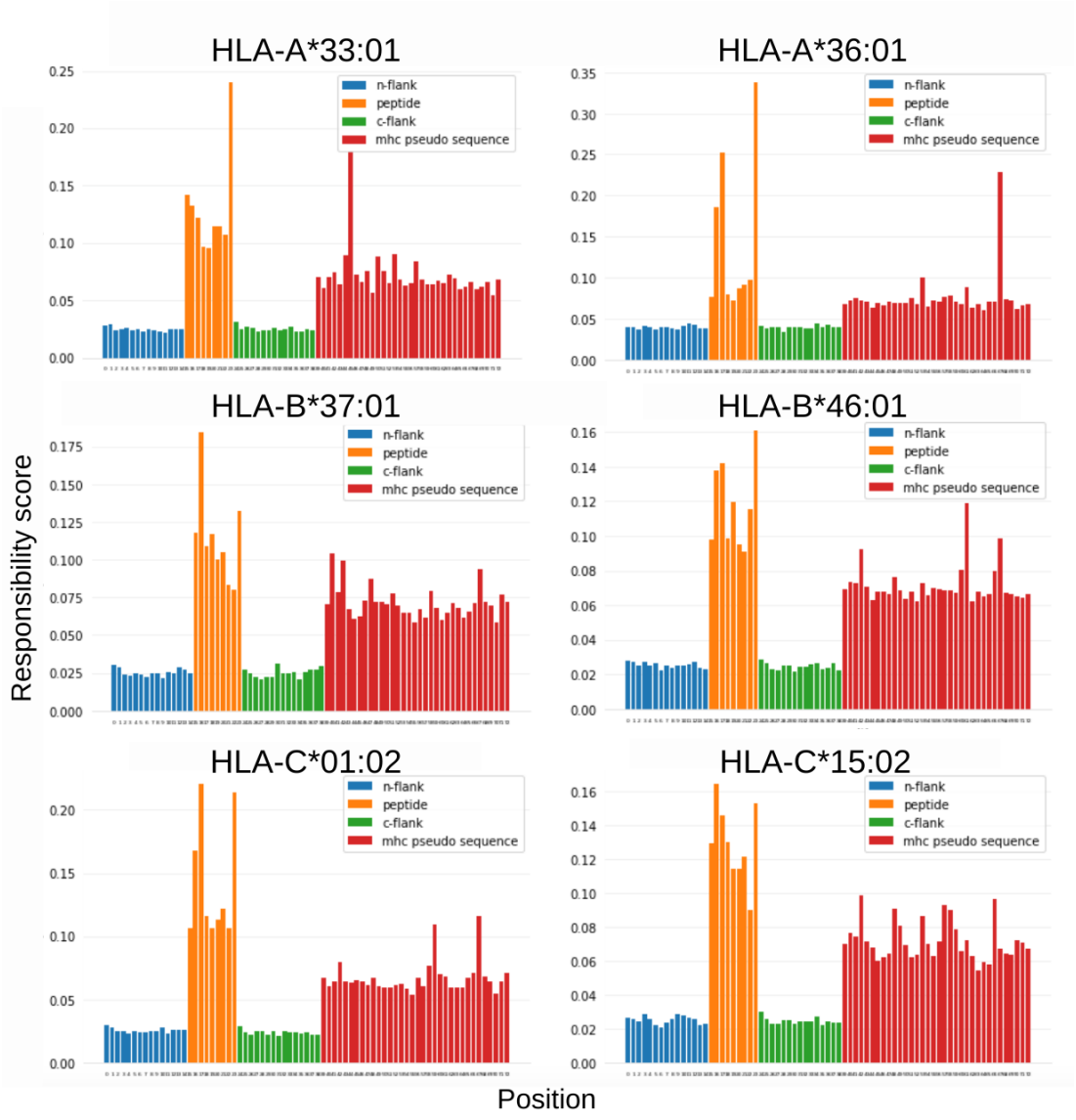


Figure 6.3: Full sequence analysis

Besides ranking the individual positions, it is useful to understand what sequence components have the most significant contributions towards the explanations. We observed that combined contribution towards producing an explanation of the 9 peptide positions is approximately 40% for all alleles which confirms that ImmunoBERT has learnt from the training data that the peptide part is the most important factor of determining whether a peptide will be presented by MHC-I molecule. The second highest importance can be assigned to the MHC sequence which tends to be in range 20-35 % across all HLA alleles. The least significant contributors are the surrounding regions of the peptide, c-flank and n-flank, that have an equal contribution of approximately 15 % towards the ImmunoBERT's decision. The order of contributions of different parts are supported by LIME interpreted results [17]. The lowest importance assignment to flanks is in agreement with the findings of J. O'Donnell et al. [42] who used a different model, MHCflurry 2.0, to predict peptide presentation on MHC-I molecules and their published results show that adding flanks only provides a small improvement in the model's performance.

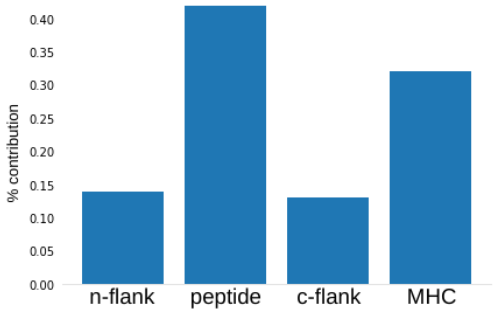


Figure 6.4: Sequence components

6.2 DC Causal explanations coverage and quality - RQ2

6.2.1 Analysis of ranked lists used as explanations

We observed a significant proportion of sequences for which the algorithm did not find any explanation and returned a responsibility map assigning 0 to each position. The interpretability coverage varies with the number of iterations of the algorithm. The more we increase the number of iterations N, the higher the coverage of the technique. The reason for this behaviour is that the mutants created in the algorithm for a particular input sequence do not meet the required conditions outlined in the definition of responsibility (4.2.1). More specifically, for most of the random partitions of a sequence, the algorithm was not able to find a mutated sequence of the same class as the original sequence which after masking a group that is being considered as the 'actual cause' would result in a change of classification of the mutated sequence (we show an example in A.5).

Table 6.2 shows that the coverage of the technique can be improved with increasing N. Our experiments run the algorithm with 10 random partitions where we achieve coverage 63.6% whereas for 100 partitions the coverage increases to 86.0%. Similarly, the average length of ranked lists increases and the explanations have more distinctly

ranked positions with further refinement. For N=100, the length of a ranked list is 42.7 so the algorithm is able to assign responsibility value to all amino acids in the sequence (of full length 43). Out of those positions on average 40.5 are assigned unique responsibility such that the list is perfectly ranked with no positions having overlapping ranks. In contrast, for N=10 the average length of ranked lists is only 29.0 and most of those positions do not have a unique responsibility value because the algorithm could not achieve sufficient refinement for such small N.

We measured the average runtime to interpret 1 sequence. If we do 5 iterations of the algorithm, it takes approximately 1 minute to explain a sequence (without parallelization). In contrast, for larger N it becomes infeasible to use the technique without further parallelization if we require to interpret a larger dataset of sequences.

N	% coverage	avg. length	avg. distinct ranks	avg. runtime (s)
5	53.2	25.2	3.9	61
10	63.6	29.0	5.8	118
20	72.1	38.3	18.1	252
50	75.7	40.6	29.8	398
70	81.4	41.5	31.2	648
100	86.0	42.7	40.5	814

Table 6.2: Varying number of iterations

Table 6.3 shows that there is an imbalance in the ability to produce ranked lists for positive and negative sequences. 73.4% of the sequences for which the technique was able to produce a ranked list of positions, taken from HLA-B dataset, had label class 1. In contrast, only 32.9% of sequences taken from HLA-C dataset had label class 1. We were expecting the split to be balanced for each allele due to a perfectly balanced dataset. Further, if we require the technique not only to produce a ranked list of positions but also minimal explanation (defined in algorithm 3), by masking the entire sequence and flipping its classification, we find that it cannot produce a minimal explanation for a negative input sequence.

One of the possible reasons for imbalance could be that the negative samples are closely similar to the positive samples and the model is unable to distinguish partially masked sequences because there are only few discriminators - if we mask these discriminators then the mutated sequence can never change classification with further masking. There might be limitations in ImmonoBERT to generalise on unseen HLA-B and HLA-C alleles because it mostly fit on the HLA-A allele data during training and that is why we see a balance of classes for HLA-A data but not for the other alleles.

The average length of explanation shows how many positions out of 43 were assigned non-zero responsibility and were included in the explanation. This metric is useful to evaluate the usability of DC-Causal across different alleles or to compare produced explanations to the explanations generated by other interpretability techniques. On average, the explanations are of length 29 out of the maximum length of 43 which is 67% amino acid positions that were awarded a non-zero responsibility.

HLA allele	% positive	avg. length
HLA A	51.3	29.4
HLA B	73.4	30.9
HLA C	32.9	26.7
AVG	52.5	29.0

Table 6.3: Analysis of produced explanations

6.2.2 Performance of DC-Causal algorithm

We show the coverage of explained instances for every HLA allele and the average number of partitions per explained instance that resulted in a non-zero assignment to at least 1 group of amino acids.

In contrast to the imbalance of class 1 and class 0 explained sequences that we observed (6.2.1), other defined performance metrics such as the average number of hits per explanation or the proportion of all sequences explained are constant throughout all alleles. In particular, we observe that 63.6% of amino acid sequences, when interpreted by DC-Causal, resulted in a ranked list of positions - these ranked lists include both class 0 and class 1 sequences. The other sequences were not successfully interpreted and resulted in an empty list because the technique assigned responsibility 0 to all positions. The average % of hits measures the proportion of successful partitions for explained sequences. For a successful application of this interpretability technique the average success % of 22.7 is not what we hoped for because it means that the algorithm succeeds to find a non-zero responsibility map only in 22.7% of computations.

HLA allele	% coverage	avg. % of hits per explanation
HLA A	63.5	23.1
HLA B	63.8	23.3
HLA C	63.6	21.8
AVG	63.6	22.7

Table 6.4: Performance analysis

6.2.3 Individual explanations

The DC-Causal method produces explanations of varied quality and length based on the number of iterations before termination. An example of an individual instance explanation from the HLA-A-33:01 dataset is shown below. This explanation was produced in an experiment which ranked the peptide and MHC-I protein positions where we set N=10.

[(16, 0.58333), (17, 0.58333), (19, 0.58333), (22, 0.58333), (23, 0.58333), (40, 0.58333), (42, 0.58333), (43, 0.58333), (44, 0.58333), (45, 0.58333), (47, 0.58333), (48, 0.58333), (51, 0.58333), (53, 0.58333), (56, 0.58333), (59, 0.58333), (60, 0.58333), (61, 0.58333), (63, 0.58333), (64, 0.58333), (66, 0.58333), (69, 0.58333), (71, 0.58333), (72, 0.58333), (18, 0.25), (21, 0.25), (41, 0.25), (49, 0.25), (54, 0.25), (58, 0.25), (62, 0.25), (70, 0.25)]

legend: peptide | mhc

Figure 6.5: HLA-A example individual explanation

Each position is represented as a tuple $(position, score)$ and the color-coding specifies the part of the input sequence to which the position belongs. The first 5 positions which all have the same score correspond to the peptide positions 2,3,5,8,9. This particular explanation only awards 2 distinct rank scores 0.58 and 0.25. The algorithm terminated after 2 iterations since there are only 2 distinct scores and the rest of positions are awarded importance 0. This showcases some of the problems we will cover in the limitations section. In particular, we notice that there is no clear distinction between the first 24 positions and we cannot determine which of those positions has the largest importance. This is problematic if we require a method which sufficiently explains individual instances.

As we have shown, the individual explanations produced by the technique when N=10 do not suffice to fulfill the criteria of producing a useful explanation for every input sequence which would be a desirable requirement for a local interpretation technique. If we lower our expectations from the technique and do not require a local explanation for every instance, we can gain an insight by creating a combined explanation which considers position responsibilities across all 1000 data points per each set of input sequences and creates an aggregated summary. This summary sums up responsibility values for each position. We present an example summary for HLA-A33:01 dataset in which we normalised the importance scores.

[(23, 0.19613585185794963), (45, 0.1428693378250874), (16, 0.13315907465584031), (15, 0.11569639028000883), (17, 0.10427961482812763), (20, 0.10221562712268589), (21, 0.10008447385192835), (50, 0.08599492009602092), (18, 0.08540755590382042), (19, 0.08184655533277711), (22, 0.0811748561668619), (58, 0.07324010563345264), (53, 0.0730944793165141), (44, 0.07110852811332168), (43, 0.0709480675725691), (41, 0.07005364111172527), (61, 0.07000287622944798), (46, 0.0698310516071913), (71, 0.06889445765949011), (65, 0.06875734896458513), (57, 0.0668599969852187), (63, 0.0668266045708693), (55, 0.06655478763477847), (47, 0.0663321908781185), (60, 0.06600888022108344), (67, 0.0659776344360417), (42, 0.0658145014868436), (66, 0.06580383723550236), (54, 0.06554716999070831), (64, 0.06547502946692922), (52, 0.06509674406845078), (72, 0.0647945951151993), (51, 0.06472736790681204), (70, 0.06443168059786182), (40, 0.06413856054152961), (68, 0.0638820419653194), (39, 0.06267706858927011), (62, 0.06233489515240594), (59, 0.06203361282821619), (48, 0.061768844958635875), (49, 0.061700642339296306), (56, 0.06019797281538376), (69, 0.058621665203665095)]

legend: peptide | mhc

Figure 6.6: HLA-A example summary explanation

The aggregated summary can be done in multiple ways - considering all data, considering a subset of randomly sampled instances or as shown in Figure 6.6 we consider one HLA allele set of 1000 instances.

In contrast to the individual explanations, a summary explanation does not have ties when ranking positions - ranks are clearly distinguished and we can determine the position responsibilities from this rank. Another advantage is that all positions which were considered by the DC-Causal algorithm have some non-zero responsibility and therefore all appear in this summary explanation.

6.3 Comparison to other techniques - RQ3

We compare the explanations produced by the DC-Causal technique with the explanations from two other techniques that were covered in the background section, namely LIME and SFL. Our approach and the other two techniques produce ranked lists of positions that we can compare by using a similarity measure.

The challenging aspect of list comparison is that the ranked lists have various lengths and they might only have some positions in common. We need a comparison approach that has the ability to work with non-overlapping lists and weighs the higher-ranked positions more than the low-ranked positions. For our purposes, we selected rank-biased overlap (RBO) similarity measure that calculates a score by weighing each position until some specified depth d, using weights from convergent series. The measure has parameter d which specifies the number of positions that the RBO measure considers, starting from the top-ranked position, which we compare and parameter p which specifies the degree of top-weightings of the resulting score. The measure is bounded by interval [0,1] where 0 means disjoint lists and 1 means identical lists [45].

6.3.1 RBO score comparison - peptide comparison for all alleles

In the earlier parts of the evaluation (6.1.3) we found that peptide has the highest overall responsibility score and contributes the most to ImmunoBERT's predictions. We compared the peptide ranking produced by DC-Causal method with the peptide rankings produced by SFL and LIME. Across all allele datasets we found that the ranking of the peptide positions produced by DC-Causal algorithm has similar degree of overlap with the other 2 techniques if we only consider the 9 peptide positions (setting parameters d=9 and p=1 to consider all 9 peptide positions). All techniques produce explanations which highly rank the peptide positions, especially positions 9, 3, 2 and 1. The reason why the RBO score is not higher is that the order of the top positions is rearranged and varies across the techniques - for example, LIME ranks the top 3 positions as [9,2,3] and DC-Causal as [9,3,2] - this has an effect on the RBO similarity measure.

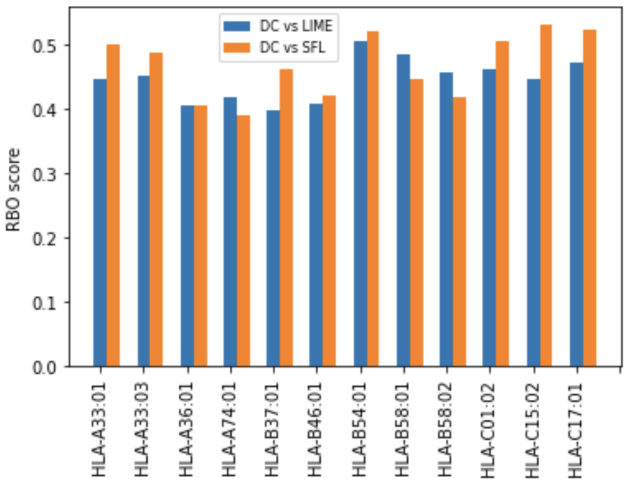


Figure 6.7: Similarity of ranked peptide positions

DC-Causal and SFL peptide ranks have almost the same degree of similarity as DC-Causal and LIME peptide ranks. The mean and median RBO scores are similar for both compared pairs of lists. DC-Causal and SFL pair has a higher RBO score suggesting that their ranked peptide positions tend to agree more although there is lack of significance in the results to conclude this. We find that the distribution of RBO scores for DC-Causal/SFL pair is more variable than the distribution for DC/LIME. The DC/SFL pair explanations match more for certain HLA types and less for the others - there is a higher inconsistency in the RBO scores.

method	mean	median	std	min	max
DC/LIME	0.447	0.449	0.032	0.399	0.505
DC/SFL	0.468	0.475	0.048	0.390	0.532

Table 6.5: Peptide ranked lists RBO score statistics

6.3.2 RBO score comparison of peptide and MHC rankings

The results in this section show the three explanation techniques and their pair-wise RBO scores when we included the peptide sequence with the MHC sequence so the maximum possible length of the ranked lists is 43. We tried multiple values of parameter d which specifies the length of the lists we compare.

We varied parameter p to show that for $p < 0.9$, the explanations produced by the techniques are not very similar. When p is small, the measure weighs the top positions very heavily. We used the formula introduced by Webber et al. [45] to calculate the overall percentage that the d positions contribute towards the total RBO score. If we set p=0.1, the top 3 positions contribute towards 99.9% of the RBO score. In contrast, if we set p=0.95, the top 3 positions contribute towards 34.9% of the RBO score. We observe that as we increase p, we are not putting as extreme weights on the top positions but include the positions from the entire sequence of length d, which increases the strength of similarity.

By setting d=3 we limit the size of the lists to 3 and only include the top 3 most highly ranked positions in the comparison. The top 3 positions achieve RBO score in range 0.27-0.38 when comparing DC-Causal technique and SFL - these 2 techniques are the most similar. In contrast, DC Causal algorithm and LIME have the lowest RBO similarity score. All three pairwise similarity scores are bounded by range 0.24-0.40 (when p=1) which implies that there is some overlap of the TOP 3 positions across the techniques and different HLA alleles. The RBO score increases with increasing p which suggests that all 3 top positions and their ordering contribute positively towards the overall RBO score.

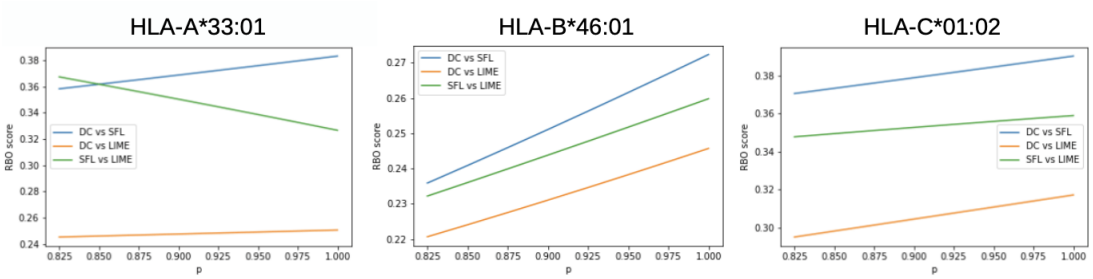


Figure 6.8: RBO similarity measure - top 3 positions

Explanations of length 20 include approximately the upper half of the ranked positions. Compared to d=3, there is overall increase in the similarity of the ranked lists. The curves are closer to each other and strictly increasing with increasing values of p so even the lower-ranked positions still contribute towards increasing the similarity measure. DC Causal/SFL pair continues to show the highest similarity of ranked lists. There are signs of convergence for very high values of p which means that the techniques produce explanations that are similar or dissimilar to each other to the same extent.

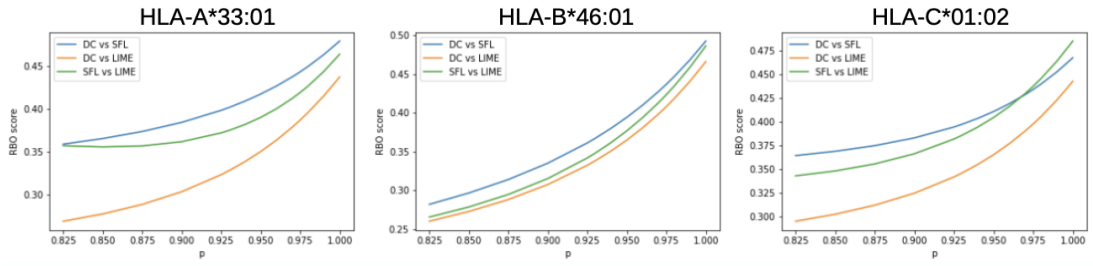


Figure 6.9: RBO similarity measure - top 20 positions

We included the entire length of the ranked lists to show that when the RBO measure considers all 42 positions, we achieve the maximum similarity scores that tend to be around 0.7 for both paired list comparisons DC-Causal/SFL and DC-Causal/LIME, signaling strong positive correlation of the position orderings. Achieving similarity score of 0.7 means that the ranked lists exhibit signs of significant similarity but they are not necessarily co-joint. The disadvantage of considering the entire length of ranked lists and weighing them equally (when p=1) means that we put the same weight on the

lower-importance positions as we put on the highest-importance positions within the ranked lists. A lower RBO score is achieved for the lower values of p in which we put more weight on the highly ranked positions - the RBO graphs are not consistent for all allele types because for HLA A, the SFL/LIME ranked lists are most similar and for HLA C type data the DC-Causal/SFL ranked lists are most similar.

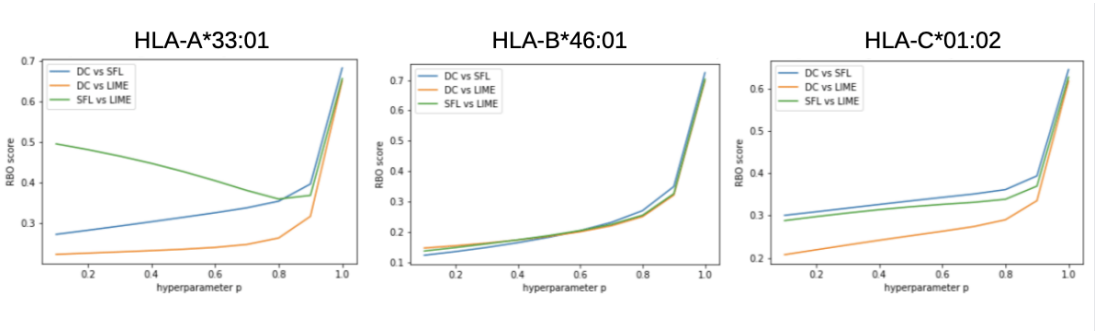


Figure 6.10: RBO similarity measure - full sequence

6.3.3 Limitations

The limitations of applying DC-Causal technique to explain ImmunoBERT can be summarized into the following categories:

Low coverage: In an ideal case, a local interpretability technique would explain 100% of valid inputs. DC-Causal technique explained on average 63.5% input sequences (for N=10) in our experiments, which is not sufficient, but can be improved with increasing the number of partitions during every iteration. The fact that not all instances can be explained implies that the explainability of ImmunoBERT provided by DC-Causal technique is not sufficient to provide full trust and transparency because large proportion of sequences do not have any explanation. The reason for this limitation is that the algorithm requires finding mutants of the same classification as the original sequence which change classification with further masking of a specific group of positions that we consider as 'actual cause candidate'. The required mutants are sparsely distributed in the mutant space but with the correctly partitioned sequence (which can be found by trying a lot of random partitions), they can be found.

Early termination: A challenge for DC-Causal is that ImmunoBERT is not sensitive to small alterations in the input sequence. Masking a small proportion of the sequence or changing values of few amino acids does not change classification of the sequence. This insensitivity to small changes might be caused by the nature of the problem but it could also be the curse of having small or not inclusive training dataset when training the model. DC-Causal relies on these small changes that flip classification. This has consequences in further refinement when the technique does not find any suitable mutants that satisfy the responsibility condition, hence the algorithm terminates early after the first or second refinement.

Problems related to binary classification: ImmunoBERT classifies sequences into 2 distinct classes as opposed to multi-class classification into 10+ classes of image

classifiers which were used in the original DC-Causal paper [23]. This imposes a challenge for the model to flip between classes which is a requirement for the technique to work. This brings up a question whether DC-Causal is suitable for explaining DNNs that only classify inputs into few classes. Although the technique can produce ranked lists for positive and negative sequences, it can produce a minimal explanation with respect to the definition in Compositional explanation algorithm (section 4.2.1) only for the positive sequences. We cannot produce a minimal explanation for the negative sequences because the explanation starts with all positions masked - this sequence with all positions masked is already negative so the addition of amino acids that are responsible for negative classification of a sequence should not flip it to become a positive class sequence. The consequences of not being able to produce minimal explanation for negative sequences means that we cannot use the explanations produced by Compositional explanation (algorithm 3) because we would only be able to analyse explanations for positive sequences. We instead do analysis of the ranked lists of positions. There are advantages of this decision: Firstly, we do not limit our evaluation to only class 1 sequences which would remove out 50% of the sequences from the analysis. Secondly, we want to ensure a fair comparison with LIME interpretability technique which produces ranked lists as explanations and not a minimal explanation. Thirdly, there is no guarantee that the explanation found by DC-Causal is minimal because a lot of positions share the same responsibility ranking (for N=10) so whenever there is a tie, the algorithm randomly selects the next position and adds it to the minimal explanation.

Inefficiency: The application of DC-Causal technique to interpret ImmunoBERT is not efficient. On average only 22.7% of partitions of the sequences or subsets of sequences result in further refinement. Although the technique is parallelizable which decreases the running time of interpreting 1000 sequences, most of the computations do not provide any contribution towards explaining a sequence. Most of the partitions and iterations result in a zero-responsibility map. This problem is specific to the interpretation of ImmunoBERT and is a result of the sparse amount of mutants satisfying the responsibility criteria caused by the problem that there are only 2 classes and the fact that ImmunoBERT is not sensitive to small changes in sequences.

Imbalance of classes: Although we provided a perfectly balanced dataset, there is an imbalance of the classes in the distribution of explanations. In particular, there were significantly more positive sequences explained in HLA B data and more negative sequences in HLA C data. Ideally, we would expect an equal split of positive-class and negative-class sequences explained by the interpretability technique. There are multiple factors related to the data which likely caused this limitation which are addressed in the next section 6.3.4.

6.3.4 Threats to validity

Lack of ground truth: There is no ground truth reference that would validate our approach, results and produced explanations. Although antigen presentation on MHC-I proteins and peptide binding have been studied extensively [10], there are unknown aspects of the processes within the MHC-I pathway that require further theoretical studies and experiments. For example, it is not yet known why certain MHC alleles are more susceptible to peptide editing than others [46].

Assumptions: In the process of experimental data collection and generation of negative samples, an assumption was made that if a peptide sequence was not observed during experiments then it will not be presented on the MHC-I protein [17]. This is an assumption that other studies made [42] in their work, otherwise it would be difficult to obtain required data. The distribution of observed peptides might not be a true representation of the distribution of peptides which are presented by MHC-I proteins. Similarly, the negative sequences which were randomly generated from the proteins of the observed samples might not be representative of the distribution of peptides that are not presented.

Representativeness of data: The sequences used for training and explaining the model do not represent the world's population because the observations from experiments (positive sequences) were collected from a group of individuals who do not represent a diverse range of demographics. The data does not include all possible MHC-I proteins and does not take into account the variation of MHC-I proteins across the human population.

Threshold selection: ImmunoBERT is a prediction model that outputs a value in range [0,1] representing the likelihood that a peptide will be presented by the associated MHC-I protein. Interpreting an output which is not discretized is not possible with techniques such as DC-Causal, SFL or LIME. Similarly to [17], we converted the prediction problem to a classification task by the application of a threshold 0.5 so we assume that if ImmunoBERT predicts that the probability of peptide presentation by the associated MHC protein sequence is greater than 0.5 then it is class 1 peptide and will be presented by the MHC-I protein. This value needs to be tested to determine whether it is the most suitable threshold value.

Reliability of predictions: Our produced results are conditioned on the assumption that ImmunoBERT's predictions are correct. It is not well-studied how robust ImmunoBERT is because there are no comparative studies which would compare ImmunoBERT's predictions to some other model's predictions on the same dataset. We cannot state the uncertainty of the predictions. We do not know the distribution of ImmunoBERT's errors so we cannot reliably calculate the estimate of the standard error of ImmunoBERT's predictions.

Chapter 7

Conclusions

7.1 Contributions

The contribution of this project can be split into 3 parts:

1. We explored the application of DC-Causal to interpret ImmunoBERT and evaluated the interpretation coverage, quality and validity of explanations.
2. We have demonstrated that DC-Causal can provide explanations for amino acid sequences which are a new type of input data for the technique.
3. We have shown that the interpretation of ImmunoBERT by DC-Causal is in agreement with the interpretations produced by other techniques and supported by findings from biological studies.

7.2 Summary of results

In this paper we explored the application of DC-Causal for interpreting ImmunoBERT. We have demonstrated that a technique with strong foundation in causal theory that was originally created to explain image classification can be applied to new domains such as Immunology and achieve comparable performance. Our experiments demonstrated that the results are in agreement with other state-of-the-art interpretability techniques.

Our goal was to provide insights into how the model works by creating explanations which improved our understanding of the model's decisions. The technique was able to identify peptide as the most important part of the sequence, especially positions close to the termini of the peptide. In addition, some MHC positions in A, B and F pockets were shown to be of significant importance which is supported by biological studies. The flank positions were not ranked high because of their low contribution towards ImmunoBERT's predictions. We concluded that DC-Causal ranked lists are closer to SFL than to LIME but there are only small differences. The technique has limitations which we discussed. The coverage and quality of explanations could be further improved by increasing the number of iterations.

7.3 Future work

We answered research questions and achieved the main objectives of the project. There are many interesting approaches in the field of explainable AI which are worthy of further exploration and plenty of exciting applications of DNNs which need to be better understood. We will outline some suggestions for further exploration that are relevant to our work although this is not an exhaustive list of ideas.

Taking the project further could be done in various directions. We could address the threats to validity of our project by training a different model and comparing it to ImmunoBERT's performance. We could do further analysis of the data that were used for training ImmunoBERT and seek alternatives for decoy generation approach or how to make the training dataset more representative of a winder range of population demographics. We could increase the training dataset by using Generative Adversarial Networks (GANs) which are machine learning techniques that learn from the training set to generate synthetic data with the same distribution as the training set. Training ImmunoBERT on this expanded dataset could improve the performance and robustness of the model and allow it to generalise better to unseen sequences. We could investigate what would be the best threshold to convert ImmunoBERT's prediction to a classification label and whether it is reasonable to use the threshold of 0.5.

7.3.1 Extension to T cell binding

The idea behind creating ImmunoBERT was to capture biologically meaningful information about peptide presentation. Another step towards achieving the goal of designing better immunotherapy treatments would be to consider next step in the MHC pathway. An extension, which is not considered by ImmunoBERT, but needs to be addressed in order to achieve the objective of helping with the design of specialized vaccines is the second part of the process which is binding of presented antigens to T cells. T cells are circulating cells, specifically designed to bind to antigens [2]. They circulate in our body until they receive an antigen signal and a secondary signal along with instructions in the form of cytokines, that will activate the T cell and start the process of binding to presented antigen on the MHC-I complex [10]. A suggestion for future work would be to train a classification model that explores which presented antigens in the MHC-I complex successfully trigger T cell activation and binding.

7.3.2 Further interpretation of ImmunoBERT

There are 4 interpretability techniques, namely LIME, SHAP, SFL and DC-Causal that were used to interpret ImmunoBERT and provide explanations. All of these methods are perturbation-based techniques. We could compare the explanations constructed using these techniques with a gradient-based approach, such as Grad-CAM [27] which uses gradients computed at individual inputs to produce explanations about the model's decisions. It would be interesting to compare the performance of Grad-CAM with DC-Causal because both techniques originated in the field of image classification and were designed to interpret Convolutional Neural Networks. For amino acid sequences, similarly to our approach, we could obtain a representation of a ranked list as an

output from Grad-CAM and explore whether it aligns with DC-Causal and other techniques. To make this comparison complete we could explore a well-performing NLP interpretability technique using language models, such as MICE, which we introduced in 3.4. Comparing a variety of approaches would give us an understanding of which approach is suited the best to tackle amino acid sequences, which are specific to the biomedical domain and according to our knowledge none of the interpretability techniques was designed specifically to work on amino acid inputs.

7.3.3 Improving DC-Causal technique

We have shown limitations of DC-Causal technique, the greatest obstacle being the time requirement to run the algorithm on many sequence inputs in our experiments and achieving sufficient refinement. We have shown that for obtaining a high quality explanation we require 20+ iterations of the algorithm. The technique runs slower compared to other local interpretability techniques such as LIME and SFL that only run for a few seconds per sequence. This could be improved by further parallelization of the algorithm. To complement our data-parallel approach, we could parallelize the algorithm such that it would start by splitting the sequence into N random partitions of positions and then work on explaining all N partitions in parallel, obtaining a responsibility map for each partition and combining those N responsibility maps into a single map. This would significantly speed up the process, especially for larger N, as we would not work on each random partition sequentially but would run the algorithm on all N partitions in parallel. We could combine this approach with our data-parallel approach and attempt to achieve hybrid-parallelization of the algorithm.

7.3.4 Exploring DC-Causal in Reinforcement Learning

Competing local interpretability techniques such as LIME and SHAP have been tested on inputs such as images, text, tabular data and audio sources. They can provide interpretations for a wide range of models. In contrast, our approach was not tested on a wide range of different inputs. We could not find any research paper which would explore application of DC-Causal technique to other problems than image classification.

Similarly to ranking amino acid positions, we could use DC-Causal to rank policy decisions in Reinforcement Learning (RL) by the means of ranking states of a RL environment according to the importance of the decisions made in those states. A similar technique which originated in the field of software testing, based on spectrum-based fault localization (SBFL), was recently used to interpret policies trained with the use of RL algorithms. Pouget et al. [47] have shown that a ranked list of states can help explain and understand the policies and also simplify complex trained policies while optimizing for the expected reward provided by the environment. This would be an interesting area of application for the DC-Causal technique because SBFL considered correlation of the importance in the ranked list with the relative importance of state for the performance of the policy. DC-Causal algorithm could take this further and explore whether there is causality, not only correlation, which could contribute towards increasing our understanding of how the policy works.

Bibliography

[1] Jacques Neefjes, Marlieke L. M. Jongsma, Petra Paul, and Oddmundn Bakke. Towards a systems understanding of MHC class I and MHC class II antigen presentation, 2011.

[2] Kenneth L Rock, Eric Reits, and Jacques Neefjes. Present yourself! By MHC Class I and MHC Class II Molecules, 2016.

[3] Polina Mamoshina, Armando Vieira, Evgeny Putin, and Alex Zhavoronkov. Applications of deep learning in biomedicine. *Molecular Pharmaceutics*, 13(5):1445–1454, 2016. PMID: 27007977.

[4] Mufti Mahmud, Mohammed Shamim Kaiser, Amir Hussain, and Stefano Vassanelli. Applications of deep learning and reinforcement learning to biological data. *IEEE Transactions on Neural Networks and Learning Systems*, 29(6):2063–2079, 2018.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[6] Polypeptides and Proteins. Accessed: 2022-01-18.

[7] What are proteins and what do they do? Medlineplus genetics. Accessed: 2021-03-22.

[8] Carl Ivar Branden and John Tooze. Introduction to protein structure, Dec 1998.

[9] MJ Lopez and SS Mohiuddin. In *Biochemistry, essential amino acids*. U.S. National Library of Medicine.

[10] Nebojsa Jojic, Manuel Reyes-Gomez, David Heckerman, Carl Kadie, and Ora Schueler-Furman. Learning MHC I—peptide binding. *Bioinformatics*, 22(14):e227–e235, 07 2006.

[11] Gertrudis Rojas. What is the difference between a peptide, an epitope, an antigen and a neoantigen? Accessed: 2022-03-12.

[12] Lewis J et al. Molecular Biology of the Cell. 4th edition. New York: Garland Science; 2002. T Cells Alberts B, Johnson A and MHC Proteins.

[13] MHC class I. Wikipedia. Wikimedia Foundation. Accessed: 2022-02-03.

[14] Charles A. Janeway, Paul Travers, Mark Walport, and Mark Shlomchik. *Immunobiology: The Immune System in Health and Disease. 5th edition. The major histocompatibility complex and its functions.* New York: Garland Science, 2001.

[15] Major histocompatibility complex. Encyclopedia Britannica. Accessed: 2021-10-15.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[17] Hans-Christof Gasser, Georges Bedran, Bo Ren, David Goodlett, Javier Alfaro, and Ajitha Rajan. Interpreting BERT architecture predictions for peptide presentation by MHC class I proteins, 2021.

[18] Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Xi Chen, John Canny, Pieter Abbeel, and Yun S. Song. Evaluating protein transfer learning with tape. *bioRxiv*, 2019.

[19] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable AI: A Review of Machine Learning Interpretability Methods. *Entropy*, 23(1), 2021.

[20] Amir-Hossein Karimi, Bernhard Schölkopf, and Isabel Valera. Algorithmic recourse: from counterfactual explanations to interventions, 2020.

[21] Peter Menzies and Helen Beebee. Counterfactual Theories of Causation. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2020 edition, 2020.

[22] Joseph Y. Halpern and Judea Pearl. Causes and explanations: A structural-model approach. part ii: Explanations. *The British Journal for the Philosophy of Science*, 56(4):889–911, 2005.

[23] Hana Chockler, Daniel Kroening, and Youcheng Sun. Compositional explanations for image classifiers. *CoRR*, abs/2103.03622, 2021.

[24] Joseph Y. Halpern. A modification of the halpern-pearl definition of causality. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, page 3022–3033. AAAI Press, 2015.

[25] John R. Zech, Marcus A. Badgeley, Manway Liu, Anthony B. Costa, Joseph J. Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLOS Medicine*, 15:1–17, 11 2018.

[26] Anmol Nayak and Hariprasad Timmapathini. Using integrated gradients to explain linguistic acceptability learnt by BERT. *CoRR*, abs/2106.07349, 2021.

[27] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *CoRR*, abs/1610.02391, 2016.

[28] Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[29] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. Why Should I Trust You?: Explaining the Predictions of Any Classifier. *CoRR*, abs/1602.04938, 2016.

[30] Christoph Molnar. Interpretable machine learning. Accessed: 2021-12-30.

[31] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. How can we fool LIME and SHAP? Adversarial Attacks on Post hoc Explanation Methods. *CoRR*, abs/1911.02508, 2019.

[32] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874, 2017.

[33] Erik Strumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41:647–665, 2013.

[34] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[35] Youcheng Sun, Hana Chockler, Xiaowei Huang, and Daniel Kroening. Explaining deep neural networks using spectrum-based fault localization. *CoRR*, abs/1908.02374, 2019.

[36] Xiangyu Zhang, Jianhua Zou, Kaiming He, and Jian Sun. Accelerating very deep convolutional networks for classification and detection, 2015.

[37] Christine A Orengo, Annabel E Todd, and Janet M Thornton. From protein structure to function. *Current Opinion in Structural Biology*, 9(3):374–382, 1999.

[38] David Harbecke and Christoph Alt. Considering likelihood in NLP classification explanations with occlusion and language modeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 111–117, Online, July 2020. Association for Computational Linguistics.

[39] Alexis Ross, Ana Marasovic, and Matthew E. Peters. Explaining NLP models via minimal contrastive editing (mice). *CoRR*, abs/2012.13985, 2020.

[40] Yasset Perez-Riverol, Attila Csordas, Jingwen Bai, Manuel Bernal-Llinares, Suresh Hewapathirana, Deepti J Kundu, Avinash Inuganti, Johannes Griss, Gerhard Mayer, Martin Eisenacher, Enrique Pérez, Julian Uszkoreit, Julianus Pfeuffer, Timo Sachsenberg, Şule Yılmaz, Shivani Tiwary, Jürgen Cox, Enrique Audain,

Mathias Walzer, Andrew F Jarnuczak, Tobias Ternent, Alvis Brazma, and Juan Antonio Vizcaíno. The PRIDE database and related tools and resources in 2019: improving support for quantification data. *Nucleic Acids Research*, 47(D1):D442–D450, 11 2018.

[41] Ana Marcu, Leon Bichmann, Leon Kuchenbecker, Daniel Johannes Kowalewski, Lena Katharina Freudenmann, Linus Backert, Lena Mühlenbruch, András Szolek, Maren Lübke, Philipp Wagner, Tobias Engler, Sabine Matovina, Jian Wang, Mathias Hauri-Hohl, Roland Martin, Konstantina Kapolou, Juliane Sarah Walz, Julia Velz, Holger Moch, Luca Regli, Manuela Silginer, Michael Weller, Markus W. Löffler, Florian Erhard, Andreas Schlosser, Oliver Kohlbacher, Stefan Stevanović, Hans-Georg Rammensee, and Marian Christoph Neidert. The hla ligand atlas - a resource of natural hla ligands presented on benign tissues. *bioRxiv*, 2020.

[42] Timothy J. O'Donnell, Alex Rubinsteyn, and Uri Laserson. Mhcflurry 2.0: Improved pan-allele prediction of mhc class i-presented peptides by incorporating antigen processing. *Cell Systems*, 11(1):42–48.e7, 2020.

[43] Blosum62 scoring matrix for amino acid substitutions. Accessed: 2022-02-10.

[44] Hanneke W. M. van Deutekom and Can Keşmir. Zooming into the binding groove of HLA molecules: which positions and which substitutions change peptide binding most? *Immunogenetics*, 67(8):425–436, August 2015.

[45] William Webber, Alistair Moffat, and Justin Zobel. A similarity measure for indefinite rankings. *ACM Trans. Inf. Syst.*, 28(4), nov 2010.

[46] Marek Wieczorek, Esam T. Abualrous, Jana Sticht, Miguel Álvaro Benito, Sebastian Stolzenberg, Frank Noé, and Christian Freund. Major histocompatibility complex (mhc) class i and mhc class ii proteins: Conformational plasticity in antigen presentation. *Frontiers in Immunology*, 8, 2017.

[47] Hadrien Pouget, Hana Chockler, Youcheng Sun, and Daniel Kroening. Ranking policy decisions. *CoRR*, abs/2008.13607, 2020.

Appendix A

First appendix

A.1 Sequence refinement - example

We sampled one HLA-A*33:01 sequence and show its responsibility map produced by DC-Causal technique for varying values of parameter N. Each position is represented as $(position, score)$ tuple and the lists are ranked in descending order of responsibility scores. The refinement of sequence improves with increasing the number of random partitions N.

```
[(23, 1.66666), (68, 1.5833300000000001), (50, 1.16666), (53, 1.16666), (66,
1.16666), (17, 1.0833300000000001), (22, 1.0833300000000001), (59,
1.0833300000000001), (16, 0.33333), (18, 0.33333), (21, 0.33333), (39,
0.33333), (40, 0.33333), (41, 0.33333), (42, 0.33333), (44, 0.33333), (45,
0.33333), (47, 0.33333), (48, 0.33333), (51, 0.33333), (57, 0.33333), (62,
0.33333), (63, 0.33333), (65, 0.33333), (70, 0.33333), (71, 0.33333), (20,
0.25), (46, 0.25), (52, 0.25), (54, 0.25), (64, 0.25), (72, 0.25)]
```

Figure A.1: N=10

```
[(23, 12.249970000000001), (53, 10.416640000000001), (50, 9.499970000000001),
(22, 7.583310000000001), (55, 6.0833200000000005), (17, 5.833310000000001),
(68, 5.083310000000001), (46, 4.5833200000000005), (43, 4.416650000000001),
(16, 4.16666), (48, 4.166650000000001), (20, 3.9166500000000006), (66,
3.8333100000000004), (61, 3.6666600000000003), (15, 3.41665), (69,
3.2499900000000004), (59, 3.1666600000000003), (18, 3.0833200000000005), (70,
3.0833100000000004), (49, 2.9999900000000004), (52, 2.9166600000000003), (60,
2.9166500000000006), (47, 2.91665), (56, 2.6666600000000003), (21,
2.6666500000000006), (40, 2.5833200000000005), (67, 2.4999900000000004), (71,
2.4999900000000004), (58, 2.3333200000000005), (57, 2.2499800000000003), (39,
2.16665), (45, 2.16665), (51, 1.9166500000000002), (65, 1.9166500000000002),
(42, 1.83332), (63, 1.83332), (64, 1.7499900000000004), (62,
1.6666500000000002), (54, 1.49999), (19, 1.24999), (41, 1.24999), (44,
1.16666), (72, 1.0833300000000001)]
```

Figure A.2: N=20

```
[(23, 20.66662), (50, 13.226150000000002), (53, 11.976160000000002), (22,
9.809490000000002), (55, 7.0833200000000005), (17, 6.976170000000001), (58,
6.392830000000001), (68, 6.392830000000001), (16, 6.30951), (20,
6.309500000000001), (46, 5.976170000000001), (43, 5.9523600000000005), (48,
5.476170000000001), (69, 5.05951), (66, 5.0595), (61, 5.03571), (15,
4.97617), (47, 4.89283), (63, 4.8095), (60, 4.726170000000001), (52,
4.70237), (71, 4.55951), (21, 4.559500000000001), (49, 4.476170000000001),
(70, 4.476160000000001), (59, 4.22618), (40, 4.226170000000001), (67,
4.166650000000001), (18, 4.1428400000000005), (51, 3.9523600000000005), (65,
3.8095000000000008), (45, 3.7499800000000003), (56, 3.7261800000000003), (39,
3.7261700000000006), (57, 3.5595000000000008), (62, 3.4761700000000006), (64,
3.4166500000000006), (42, 3.16665), (54, 3.0595100000000004), (72, 2.78571),
(19, 2.5595100000000004), (44, 2.4761800000000003)]
```

Figure A.3: N=50

```
[(23, 49.58321999999997), (50, 32.7261), (53, 29.309450000000005), (22,
27.809440000000006), (16, 17.97614), (58, 17.726110000000002), (17,
16.55946), (20, 16.14279), (60, 14.976110000000002), (51,
14.202300000000003), (67, 13.666600000000003), (21, 12.892800000000001), (63,
12.726130000000001), (56, 12.309470000000001), (64, 12.249960000000002), (55,
12.166620000000002), (46, 12.142800000000001), (39, 11.976130000000001), (61,
11.952340000000001), (48, 11.726140000000001), (40, 11.476130000000001), (59,
11.30948), (52, 10.702340000000001), (68, 10.64281), (70,
10.559450000000002), (43, 10.202330000000002), (62, 10.059470000000001), (15,
9.892800000000001), (65, 9.309460000000001), (71, 9.14282), (66, 9.14281),
(47, 9.059470000000001), (54, 8.55948), (69, 8.14282), (57,
8.142800000000001), (44, 8.05948), (18, 7.559490000000001), (45,
7.499950000000001), (49, 7.309490000000001), (41, 7.1190000000000015), (72,
7.035680000000001), (19, 6.559490000000001), (42, 5.999970000000001)]
```

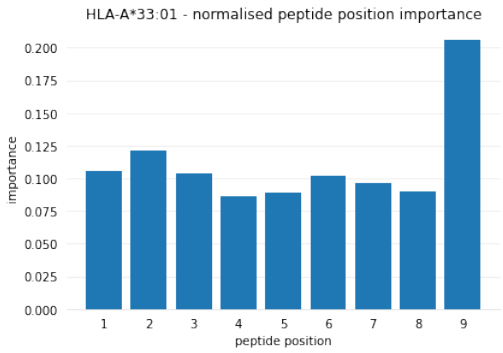
Figure A.4: N=70

```
[(23, 202.3328599999995), (22, 111.61871000000009), (50, 103.17822000000004),
(16, 89.82115999999996), (53, 87.52112999999999), (58, 60.487819999999864),
(61, 58.10690999999991), (55, 55.511709999999916), (17, 54.35453999999989),
(52, 53.85692999999991), (20, 53.27358999999991), (54, 52.104539999999915),
(60, 51.9378399999999), (39, 49.35692999999992), (67, 48.9759399999999), (45,
47.702149999999946), (66, 46.687879999999936), (48, 46.461669999999906), (63,
46.37833999999994), (51, 46.023589999999935), (41, 45.91405999999993), (68,
45.72597999999994), (46, 45.32119999999993), (64, 45.29744999999994), (65,
45.17359999999995), (62, 44.69025999999993), (21, 44.618849999999945), (56,
44.12841999999998), (47, 43.98786999999995), (69, 43.98552999999994), (59,
43.76170999999995), (40, 43.618839999999956), (70, 43.33312999999994), (43,
41.473609999999944), (15, 41.368839999999956), (18, 40.321249999999964), (71,
39.40219999999995), (72, 39.16650999999995), (49, 38.428389999999986), (42,
38.35457999999998), (44, 37.92601999999995), (57, 35.23789999999999), (19,
31.511770000000002)]
```

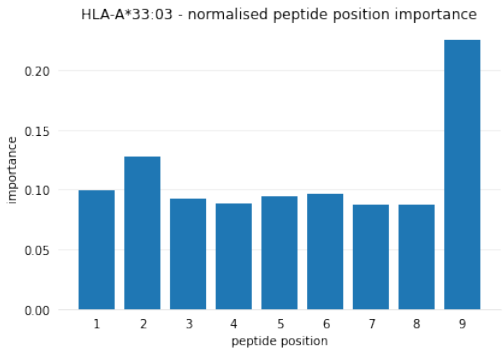
Figure A.5: N=100

A.2 Peptide full results

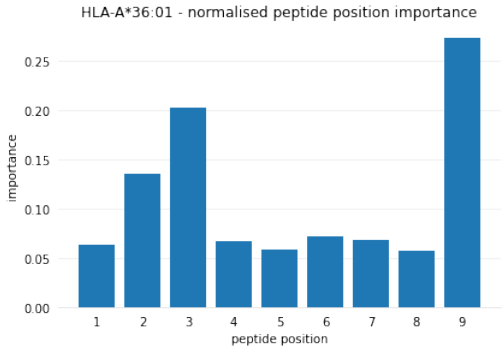
We present full results for 12 HLA allele datasets which relate to the analysis in section 6.1.1. We plotted these figures from experiment 1 explanations. We run the DC-Causal interpretability technique on peptides of length 9. We set parameters N=10, s=3.



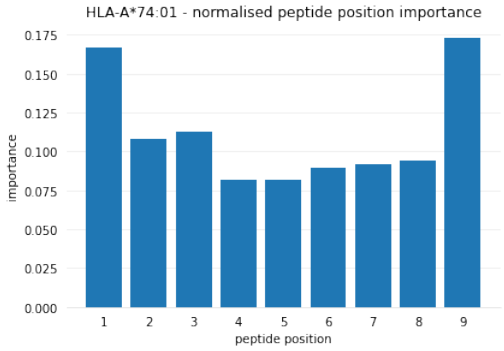
(a) HLA-A*33:01



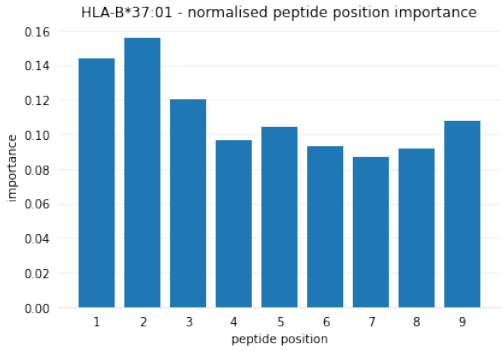
(b) HLA-A*33:03



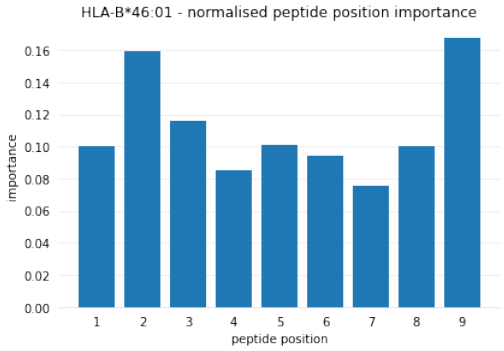
(c) HLA-A*36:01



(d) HLA-A*74:01

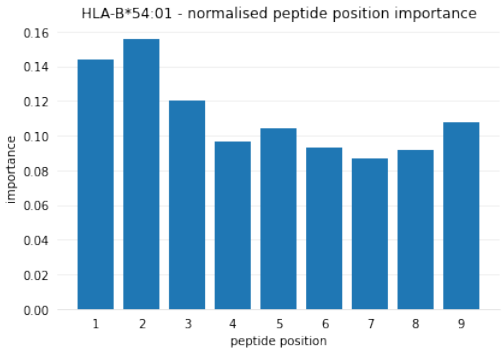


(e) HLA-B*37:01

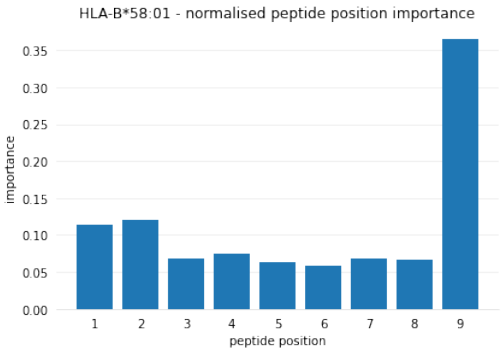


(f) HLA-B*46:01

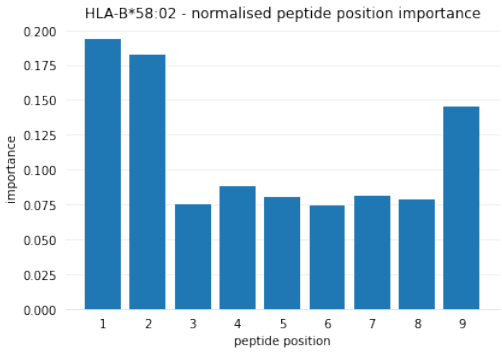
Figure A.6: Peptide positions - part 1



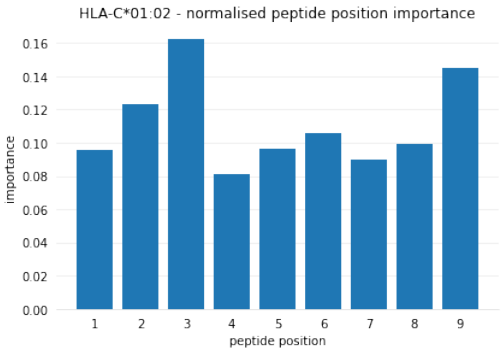
(a) HLA-B*54:01



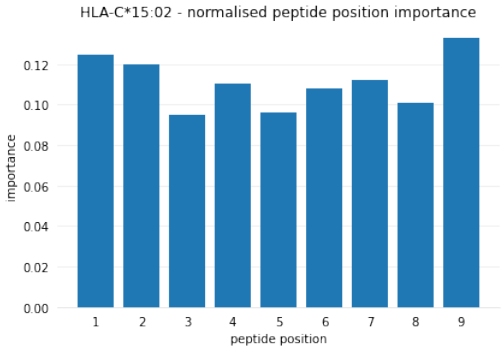
(b) HLA-B*58:01



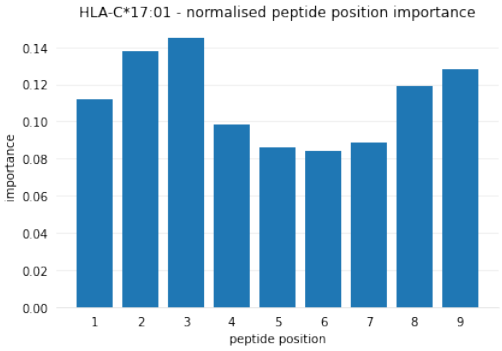
(c) HLA-B*58:02



(d) HLA-C*01:02



(e) HLA-C*15:02

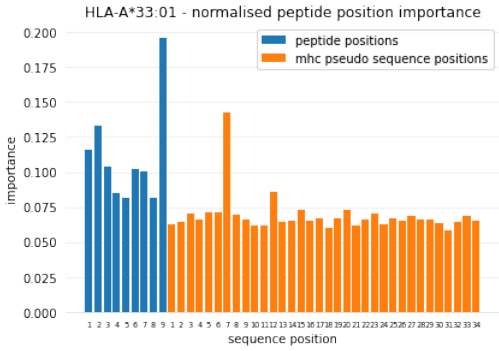


(f) HLA-C*17:01

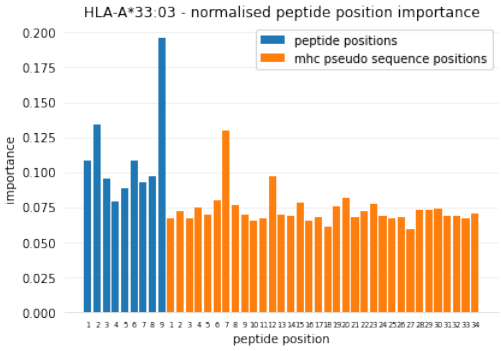
Figure A.7: Peptide positions - part 2

A.3 Peptide with MHC-I full results

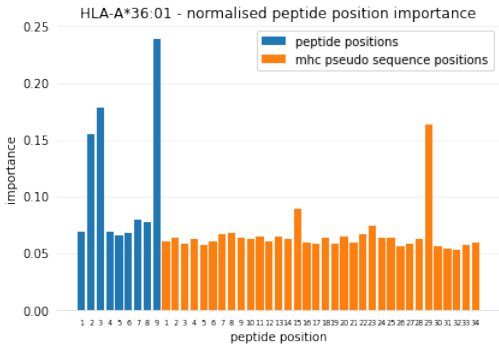
We present full results for 12 HLA allele datasets which relate to the analysis in section 6.1.2. We plotted these figures from experiment 2 explanations. We run the DC-Causal interpretability technique on peptide and MHC-I positions. We set parameters N=10, s=5.



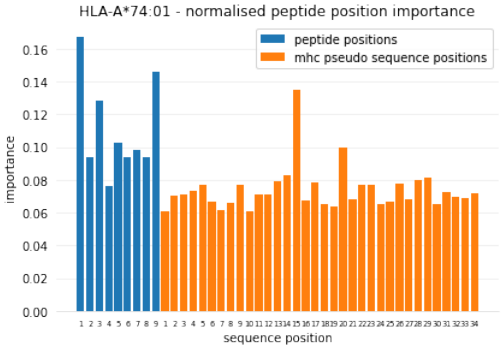
(a) HLA-A*33:01



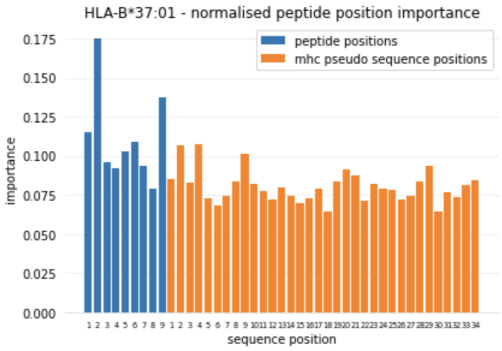
(b) HLA-A*33:03



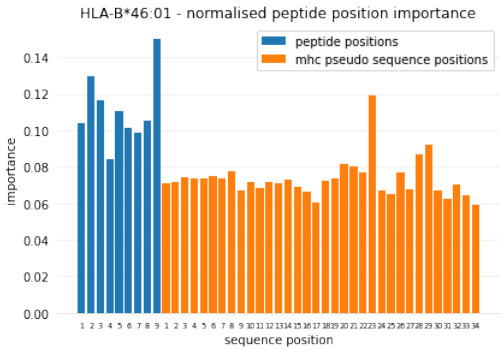
(c) HLA-A*36:01



(d) HLA-A*74:01

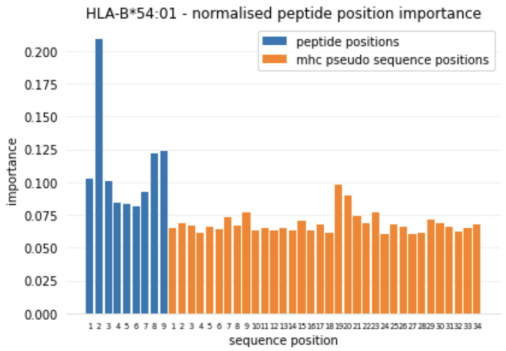


(e) HLA-B*37:01

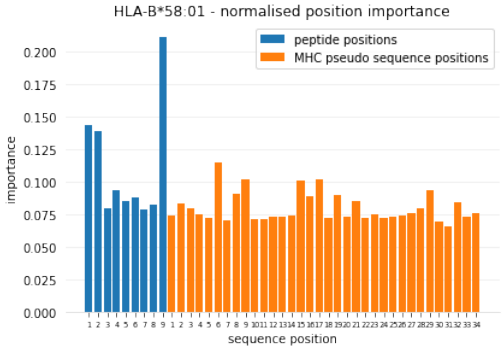


(f) HLA-B*46:01

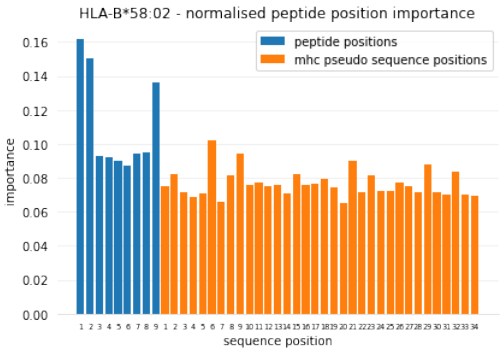
Figure A.8: Peptide and MHC-I positions - part 1



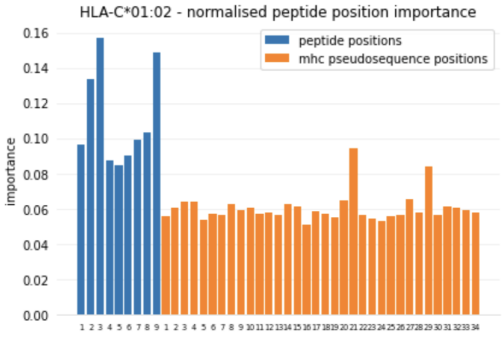
(a) HLA-B*54:01



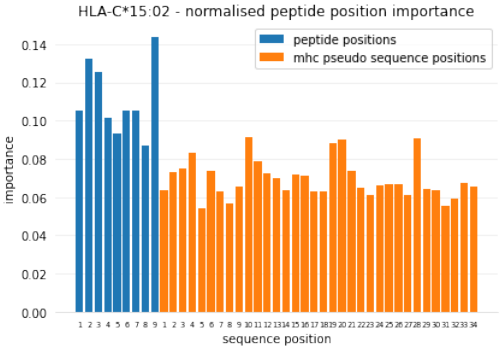
(b) HLA-B*58:01



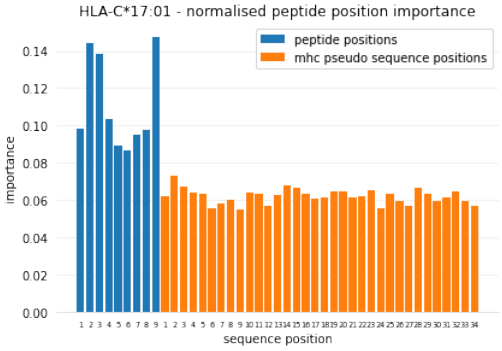
(c) HLA-B*58:02



(d) HLA-C*01:02



(e) HLA-C*15:02

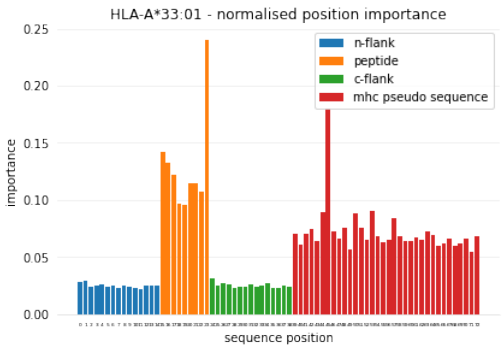


(f) HLA-C*17:01

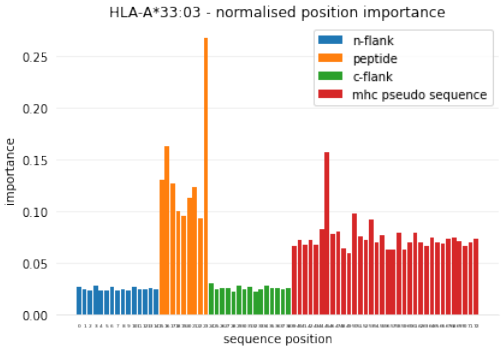
Figure A.9: Peptide and MHC-I positions - part 2

A.4 Peptide, MHC-I and flanks full results

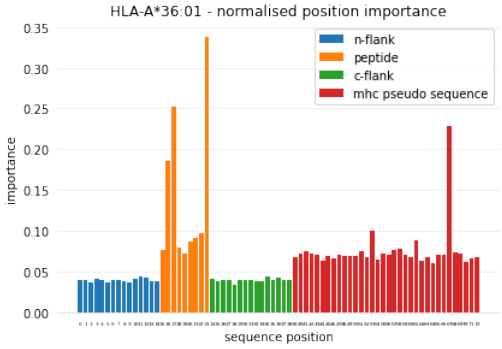
We present full results for 12 HLA allele datasets which relate to the analysis in section 6.1.3. We plotted these figures from experiment 3 explanations. We run the DC-Causal interpretability technique on all 73 sequence positions. We set parameters N=10, s=5.



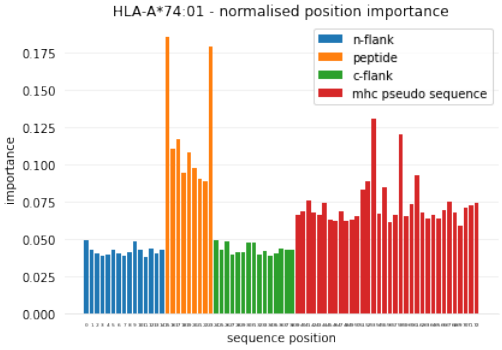
(a) HLA-A*33:01



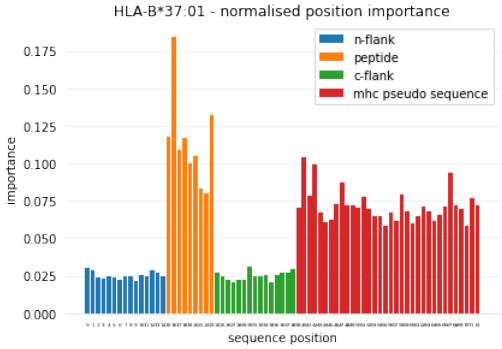
(b) HLA-A*33:03



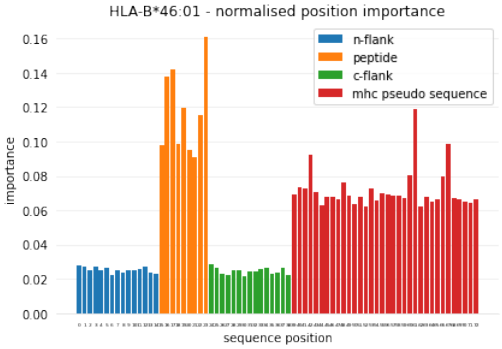
(c) HLA-A*36:01



(d) HLA-A*74:01

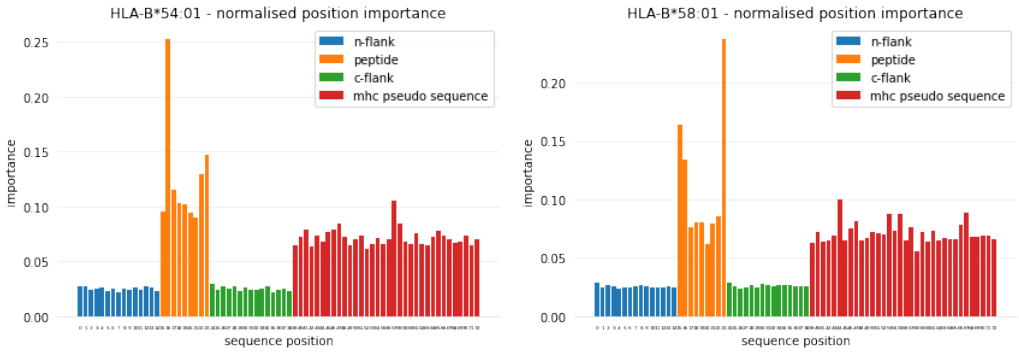


(e) HLA-B*37:01



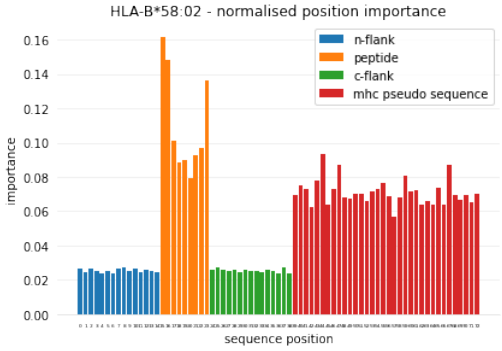
(f) HLA-B*46:01

Figure A.10: Peptide, MHC-I and flank positions - part 1

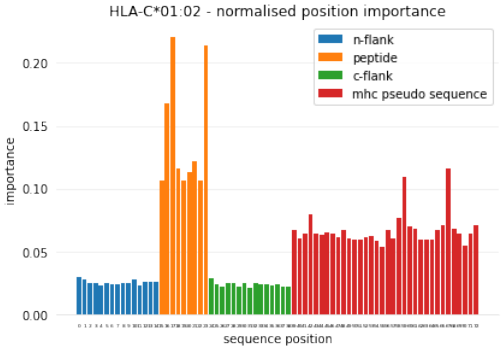


(a) HLA-B*54:01

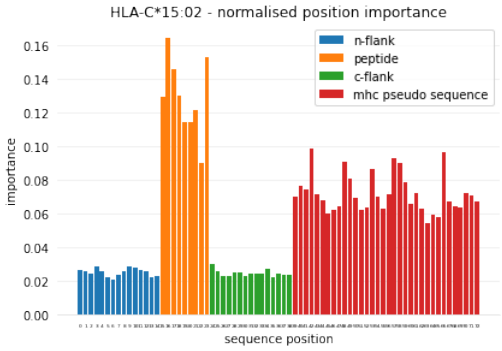
(b) HLA-B*58:01



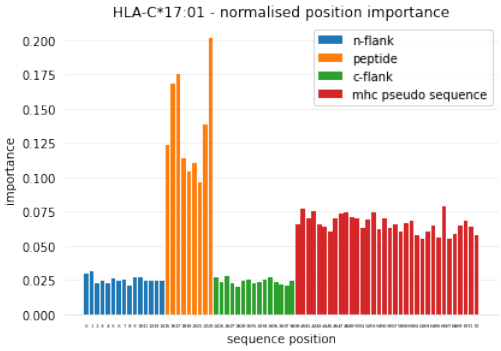
(c) HLA-B*58:02



(d) HLA-C*01:02



(e) HLA-C*15:02



(f) HLA-C*17:01

Figure A.11: Peptide, MHC-I and flank positions - part 2

A.5 Responsibility algorithm - example

In this section we show how we perform one random partition of a sequence. How to create a mutant space. How to find a mutant which satisfies the responsibility criteria described in section 4.2.1.

For simplicity we only show the responsibility algorithm on the peptide of length 9. The peptide we consider is VVMTPPRNR.

Random partition: We split the sequence into 3 parts arbitrarily.

1. VNR
2. VTR
3. MPP

Given the split we create a space of mutants which the responsibility algorithm considers. Each part can either be masked or not masked in the mutant space so we have 7 mutants (not including fully masked sequence). In this example we mask using X token.

VVMTPPRNR no parts masked

VXMTPPRXX part 1 masked

XVMXPPXNR part 2 masked

VVXTXXRNR part 3 masked

XXMXPPXXX parts 1 & 2 masked

VXXTXXRXX parts 1 & 3 masked

XVXXXXXNR parts 2 & 3 masked

Figure A.12: Mutant space

STEP *:
We start with considering part 1 - **VNR** as actual cause for classification.

A. Mutants where part 1 is not masked: VVMTPPRNR, XVMXPPXNR, VVXTXXRNR, XVXXXXXNR

B. Mutants from A with same classification as original sequence: VVMTPPRNR, XVMXPPXNR, VVXTXXRNR

C. Mutants from B which change classification after we mask part 1: XVMXPPXNR

D. Minimum difference mutant from C: XVMXPPXNR $=>$ k=3, r=1/(1+3)

We computed responsibility assigned to amino acids in part 1 **VNR** to be 0.25. Now we repeat **STEP *** but consider part 2 as actual cause, then repeat considering part 3 as actual cause.