FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

ASSIGNMENT OF BACHELOR'S THESIS

Title:	Antibody sequence analysis using Deep Learning
Student:	Ján Jendrušák
Supervisor:	Ing David Příhoda
Study Programme:	Informatics
Study Branch:	Computer Science
Department:	Department of Theoretical Computer Science
Validity:	Until the end of summer semester 2020/21

Instructions

Antibodies are proteins produced by the immune system to detect and neutralize intruders. Their analysis is crucial for understanding immune processes as well as for designing new antibodies and vaccines. Deep learning is showing significant progress in binding prediction, secondary structure prediction and other sub-tasks in the analysis of the protein universe. Additionally, transfer learning has gained an indispensable role in natural language understanding, which bears similarities to biological sequence analysis. The goal of this thesis will be to:

1) Utilize transfer learning on an antibody classification task using an existing deep learning model pre-trained on a corpus of antibody sequences.
2) Train the model with and without fine-tuning of intermediate layers and report results on a left-out dataset.
3) Apply the model to a study not used in the train set and report biologically relevant results in a case study format.
4) Discuss results and contribution of fine-tuning, if any.

References

Will be provided by the supervisor.

doc. Ing. Jan Janoušek, Ph.D.
Head of Department

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
Dean

Prague January 7, 2020

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Bachelor's thesis

Antibody sequence analysis using Deep Learning

Ján Jendrušák

Department of Computer Science
Supervisor: Ing. David Příhoda

July 30, 2020

Acknowledgements

I would like to express my sincere gratitude to the supervisor Ing. David Příhoda for all the help and insights throughout the development of this thesis. Also, I would like to thank my family for the support and patience.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

In Prague on July 30, 2020 . . . . . .

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Jendrušák, Ján. *Antibody sequence analysis using Deep Learning.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2020.

Abstrakt

Pokroky v sekvenovaní umožňujú vedcom zbierať veľké množstvá DNA sekvenovaných dát a spolu s výpočtovými zdrojmi poskytovanými výpočtovými klastermi umožnňujú využívať výpočtové prístupy napomáhajúce v mnohých biologických procesoch ako vývoj vakcín alebo návrh protilátok, ktoré momentálne využívajú pracné a drahé *in-vitro* techniky.

Spracovanie prirodzeného jazyka (NLP) je odbor zaoberajúci sa slovami a sekvenciami slov. Analógia medzi vetami, teda sekvenciami slov a biologickými sekvenciami, teda sekvenciami aminokyselín viedla k mnohým experimentom, ktoré aplikujú modely na spracovanie prirodzeného jazyka na modelovanie biologických sekvencií.

V tejto práci som natrénoval NLP modely hlbokého učenia, predtrénované na ľudských protilátkach z the Observed Antibody Space database, s cieľom klasifikácie špecificity protilátok voči Hepatitíde typu B. Finálny model je schopný predikovať špecificitu protilátok voči Hepatitíde typu B s nasledovným F1 skóre a ROC AUC proporcioniálnym k veľkosti klonotypov: 0.116, 0.829 pre validačné dáta a 0.333, 0.509 pre testovacie dáta (z inej vakcinačnej štúdie), v danom poradí. Tento model bol prekonaný jednoduchším modelom používajúcim molekulárne odtlačky s nasledovnými výsledkami: 0.155, 0.715 a 0.474, 0.645, v odpovedajúcom poradí.

V práci je navrhnutá postupnosť operácií, ktorá môže byť ďalej vylepšená pomocou iných metód alebo jej rozšírením o ďalšie predspracovanie dát alebo iné reprezentácie protilátok. Napriek tomu, že výkon *in-silico* metód je horší oproti *in-vitro* metódam, poskytujú lacnejšiu, rýchlejšiu a škálovateľnejšiu alternatívu alebo dodatočnú techniku, ktorá môže byť v budúcnosti ďalej vylepšovaná.

Kľúčové slová hlboké učenie, NLP, hepatitída typu B, strojové učenie, klasifikácia, protilátky

Abstract

Advances in sequencing technologies allow scientist to gather large amounts of DNA sequence data and together with computational resources provided by high-performance computing clusters allow the use of computational approaches aiding in many biological processes such as vaccine development or antibody design which currently employ laborious and expensive *in-vitro* techniques.

Natural Language Processing (NLP) is a field dealing with words and sequences of words. The analogy between sentences as sequences of words and biological sequences as sequences of amino acids led to experiments applying NLP models for modelling biological sequences.

In this work, I train deep learning NLP models pre-trained on human antibodies from the Observed Antibody Space database in order to classify Hepatitis B specificity of antibodies. The final model can predict antibodies' Hepatitis B specificity with following F1 scores and ROC AUC proportional to the clonotype sizes: 0.116, 0.829 for validation data and 0.333, 0.509 for the test data (from different vaccination study), in the given order. This model has been outperformed by a simpler model using molecular fingerprints yielding following results: 0.155, 0.715 and 0.474, 0.645, in the same order.

The thesis outlines a pipeline which can be further improved by using different models or extending the pipeline by additional data preprocessing or different representations of antibodies. Even though the performance of the *in-silico* methods is worse than the performance of *in-vitro* methods, they provides a cheaper, faster and more scalable alternative or additional technique that can be further improved.

Keywords deep learning, NLP, hepatitis B, machine learning, classifaction, antibodies

Contents

List of Figures

List of Tables

List of Algorithms

Introduction

Artificial intelligence (AI) has proved to be a field with very broad application domain. One such aspiring field is bioinformatics which focuses on applying computer science and AI techniques to biology.

Development of antibodies and vaccines is crucial, and it is necessary to do so promptly. However, antibody design and vaccine development are very long and expensive processes, often consisting of many "trials and errors". The Ebola outbreak or the most recent COVID-19 pandemic are a case in point. In such times, developing a vaccine in a fast manner is the most important thing to prevent spreading and to "control" the outbreak as much as possible.

Advances in sequencing technologies allow scientist to sequence antibodies from living organisms quickly and in large amounts. Therefore these subsets of present antibodies can be used to represent the immune system, analyse it and perform various tasks, such as analysing subjects' response to a vaccine, by using statistical, data mining and machine learning techniques. These tasks can be performed on a rising number of high-performance computers. analysing and modelling antibody properties can speed up and reduce the costs of antibody design and vaccine development.

The similarities between natural language text and biological sequences and previous efforts of learning the representation of proteins using NLP deep learning architectures inspired me to choose a similar approach with the current state-of-the-art model for multiple NLP tasks. And therefore, the main goal of this thesis is to analyse possible approaches to predict antibody specificity and explore how large amounts of available data can be used to train a pre-trained Natural Language Processing (NLP) model and use it to make classification predictions that can replace or supplement currently used *in-vitro* tests.

Numerous deep learning models, as well as simpler baseline approaches with comparable performance, are the outcomes of this thesis. These models can be used as an *in-silico* alternative or supplementary approach to the current *in-vitro* tests.

Goals

The goal of this thesis is to explore and compare various antibody specificity prediction methods and to design and implement the pipeline of data preprocessing and modelling on a vaccination study and use it on another vaccination study. The main product of this thesis is a deep learning pipeline capable of antibodies' specificity prediction defined as a binary classification task.

The theoretical part covers various concepts in both bioinformatics and machine learning necessary for understanding the methods used in the practical part.

The practical part covers analysis and preprocessing of the data and implementation of the whole modelling and validation pipeline, and finally compares results with other simpler (baseline) approaches. An implementation of the representation part of the NLP model is not an objective of this work and is based on the work of my thesis supervisor David Příhoda (results in progress, unpublished).

Related work

Antibody–antigen binding, also called antibody specificity, is one of the essential *protein–protein interaction (PPI)* in immunology. PPIs are the processes responsible for a wide range of biological processes (cell-to-cell interactions, cell growth...) and are the critical element in understating the protein's function [1]. Numerous computational methods have been proposed in the field of PPIs. Even though numerous advances, PPIs remain a very challenging problem [2]. Predicting PPIs is hard also due to the diversity of proteins, long-range dependencies, and generally the variable nature of chemical space.

From the deep learning perspective, there have been approaches such as LSTM and *convolutional neural network (CNN)* architectures from [3], *Ens-Grad* ensemble of 5 CNNs and single fully connected neural network from [4], *MaSIF* (molecular surface interaction fingerprinting) framework based on the *geometric deep learning* from [5]. Overview of other deep learning methods not only for predicting PPIs can be found at [6]. From a technical perspective, the closest method to the one proposed in this work is the *Tasks Assessing Protein Embeddings* (TAPE) benchmark introduced in [7]. This paper focuses on pre-training and various downstream tasks using *BERT* architecture which is the predecessor architecture of the one used in this thesis. The downstream tasks in the paper include structure prediction tasks, protein engineering tasks and an evolutionary understanding task. However, it does not include the antibody–antigen binding task.

Specifically, in the field of antibody–antigen binding, [8] proposes a method in which authors show how clustering by paratopes (*paratyping*) as well as clonotyping can be used to predict whether an antibody is *Pertussis toxoid*–binding or non-binding.

Outline

The work is split into 5 chapters. In *Chapter 1: Theoretical background* all the theoretical background from both biological and AI point of view, necessary to understand the following chapters is provided. *Chapter 2: Methods* focuses on the data used in this work as well as the methods for antibody classification proposed in this thesis, namely, data preprocessing the representations used for training the models, models themselves and evaluation techniques. *Chapter 3: Results* sums up the performance of the trained models on validation and test data. *Chapter 4: Discussion* provides a discussion about the models' performance and the achieved results. *Chapter 5: Case study* serves as a stand-alone case study which showcases the usage of the models on another study together with a discussion of biological aspects. The *Conclusion* sums up the outcomes of the thesis and discusses ideas for future work.

Chapter 1

Theoretical background

1.1 Antibodies and the immune system

This chapter discusses the biological aspects of the thesis and the essential background necessary to understand the topic at hand.

1.1.1 The immune system

The *immune system* is a defence system of a living being which protects it from various disease-causing germs (also called *pathogens*). The tasks of the immune system are to recognize such harmful germs and substances, neutralize them and to adapt to be more potent against them in the future.

The immune system consists of 2 subsystems:

The innate immune system (also called "non-specific") provides a general way of fighting foreign substances, for example through the skin or digestive system.

The adaptive immune system (also called "specific") provides a tailored way of fighting harmful germs and substances by producing *antibodies* targeting given germs or substances. This is commonly referred to as "acquired", "learned" or specific immune response.

For more details on this topic, see [9].

1.1.2 Antibodies

Responses of the adaptive immune system are carried out by *lymphocytes* which are white blood cells. There are 2 types of lymphocytes, namely: *B-cells* (B-lymphocytes) and *T-cells* (T-lymphocytes). These lymphocyte types are responsible for 2 different adaptive immune responses: *antibody responses* and

cell-mediated immune responses, respectively. B-cells take care of antibody responses by secreting *antibodies* which are then responsible for neutralizing antigens. *Antigens* are structures that cause an *immune response* from the immune system. For instance, proteins on the surfaces of pathogens are antigens. More details can be found at [10].

Antibodies (also known as *immunoglobulins*) are Y-shaped *proteins* consisting of 4 *polypeptide* chains: *"two identical* light (L) chain*s (each containing about 220 amino acids) and* two identical heavy (H) *chains (each usually containing about 440 amino acids)"* [10]. Antibody structure can be seen in figure 1.2. Some regions of antibodies are more important for antigen binding or non-binding. The most variable region *CDR3* has a significant impact on the binding properties of the antibody [11]. *Paratope* is a part of an antibody which binds to its counterpart in antigen called *epitope* (Figure 1.1).

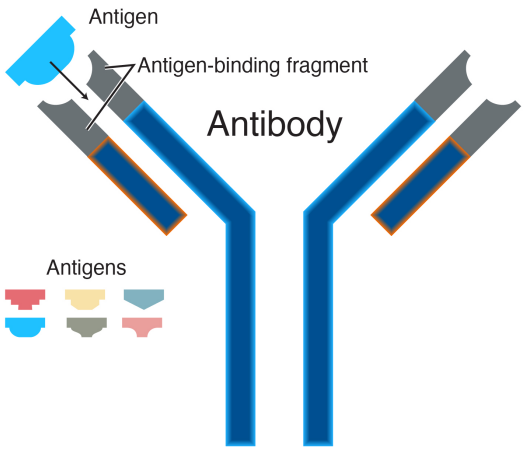


Figure 1.1: Scheme describing the "lock–key" relation between antibodies and antigens [12]

Deoxyribonucleic acid (DNA) is a hereditary material responsible for carrying genetic information of humans and most of the other organisms. The information in DNA is encoded by 4 nucleotides, namely: adenine (A), guanine (G), cytosine (C) and thymine (T).

An antigen induces production of antibodies, and then these antibodies bind to that specific antigen. The high diversity of the adaptive immune system, such that there are specific antibodies for many different antigens, is mainly enabled by a process called *V(D)J recombination*. In this process, the variable region of an antibody is assembled from separate gene segments, namely, *V (variable)*, *J (joining)* and in some cases, *D (diversity)*. The antibodies are then detected by B-cell receptor (BCR) of a B-cell allowing it to act [14].

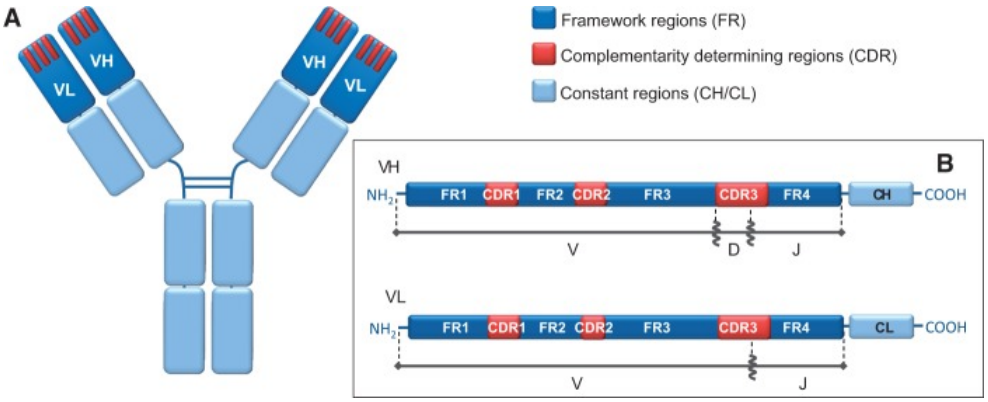


Figure 1.2: Antibody structure: (A) General antibody structure highlighting the position of constant regions, variable heavy chain (V_H) and variable light chain (V_L) domains. CDRs are represented as 3 red bars in each variable region (V_H and V_L). (B) Linearized representation of the V_H and V_L regions. In this work, the focus is put only on V_H and its CDR3 region. [13]

Amino acids, which are made up from triplets of nucleotides, are fundamental building blocks of peptides and proteins. Amino acids can be represented by their corresponding letter codes which allows one to represent proteins (and therefore antibodies) as sequences of characters. In this thesis, antibodies are represented in such a way while using only *natural amino acids* (Table 1.1).

1.1.3 Sequence alignment

Sequence alignment is a common technique for comparing conserved sequences. As stated in [15], sequence alignment is a procedure which attempts to find which positions within a pair of sequences share a common evolutionary history by finding which amino acids on which positions are shared.

Sequence alignment over the whole lengths of the sequences is called *global alignment*. On the other hand, *local alignment* focuses on aligning the query sequence to the best sub-region in the target sequence. Formally, global alignment for sequences s_1 and s_2, while allowing gaps ("-" symbol) can be understood as insertions of gaps ("-" symbols) to either of sequences such that resulting sequences s'_1 and s'_2 of the same length are considered to be the alignment between the original sequences s_1 and s_2 (Figure 1.3). In the case of local alignment same definition applies while arbitrary suffix and prefix sequences can be removed from both s_1 and s_2 prior to the aligning process (inserting gaps).

Various kinds of scoring are being used to decide which alignment is best. The scoring parameters are scores for matches, mismatches, starting a gap and extending a gap. *Substitution scoring matrix* can be used for scoring matches and mismatches. Such matrices contain scores proportional to the probability

Table 1.1: List of all natural amino acids with their abbreviations and letter codes

Abbreviation	Code	Full name
Ala	A	Alanine
Arg	R	Arginine
Asn	N	Asparagine
Asp	D	Aspartic acid
Cys	C	Cysteine
Gln	Q	Glutamine
Glu	E	Glutamic acid
Gly	G	Glycine
His	H	Histidine
Ile	I	Isoleucine
Leu	L	Leucine
Lys	K	Lysine
Met	M	Methionine
Phe	F	Phenylalanine
Pro	P	Proline
Ser	S	Serine
Thr	T	Threonine
Trp	W	Tryptophan
Tyr	Y	Tyrosine
Val	V	Valine

that amino acid i mutates into amino acid j for all amino acid pairs [16]. An example of such a matrix is the *BLOSUM62* (Figure 1.4). These terms are tied to a hypothesis based on observations, called *sequence homology*, which is a similarity that is attributed to a shared ancestry in the evolution of the sequences.

```
CCAGTGTGGCCGATACCCCAGGTTGGCACGCATCGTTGCCTTGGTAAGC
CCAGTGTGGCCGATGCCCGTGCTACGCATCGTTGCCTTGGTAAGC

Alignment:
Ref+: 1 CCAGTGTGGCCGATaCCCcagGTtgGC-ACGCATCGTTGCCTTGGTAAGC 49
        ||||||||||||||| |||   ||  || ||||||||||||||||||||||
Qry+: 1 CCAGTGTGGCCGATgCCC---GT--GCtACGCATCGTTGCCTTGGTAAGC 45
```

Figure 1.3: An example of global alignment of 2 sequences of different lengths (49 and 45). Uppercase letters (and vertical lines) symbolise matches, lowercase letter mismatches and "-" symbolises a gap (example from [16])

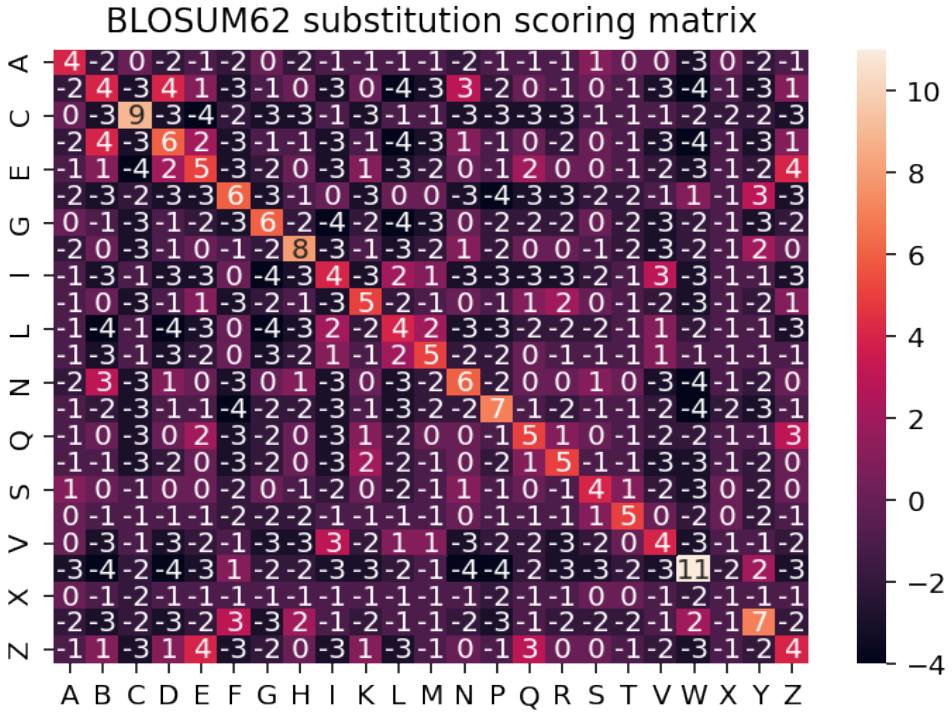


Figure 1.4: Score values of the BLOSUM62 substitution scoring matrix

1.1.4 Clonotyping

Clonotyping is a common approach of clustering antibodies together based on their shared genetic history which often causes antibodies from the same clonotype to bind to the same epitope [8]. This technique ensures error correction and preserving a maximal amount of data while removing artificial diversity [17]. Widely used conditions for 2 antibodies to belong to the same clonotype is to have same V, J regions, CDR3 length and highly similar CDR3 region (measured by *Hamming distance*). Usual CDR3 homology range used in clonotyping is 80–100 %.

Hamming distance between 2 words of the same length can be defined as a number of character positions at which the words differ [18]. Formally, for $x, y \in F^n$ where $x = (x_0, x_1 ... x_n)$, $y = (y_0, y_1 ... y_n)$, F is a finite alphabet, and F^n is a set of words from this alphabet of length n, Hamming distance can be defined as follows:

$$d(x, y) = |\{i \in [0, n] \mid x_i \neq y_i\}| \tag{1.1}$$

1.1.5 Next-generation sequencing

Next-generation sequencing "refers to the deep, high-throughput, in-parallel DNA sequencing technologies developed a few decades after the Sanger DNA sequencing method first emerged in 1977" [19]. The main advantage against its predecessors is that NGS is massively parallel and provides extremely high throughput at a much lower cost.

The growth of the amount of data, which emerged thanks to NGS, has created a lot of opportunities in antibody science, as described in [20].

1.2 Vaccines

In the late 18th century Edward Jenner invented a procedure called *vaccination* which is a process of inoculating a healthy individual with a dead or weakened pathogen in order to provide protection for future encounters with the pathogen. After exposure to vaccination, some of the B and T cells activated by the antigen will differentiate to *memory cells* which are the cells providing long-lasting immunity. During this process, the B-cells undergo random mutations in order to improve antibodies' ability to bind to the new antigen. For more information, see [14].

Hepatitis B is a viral infection attacking the liver and is caused by the hepatitis B virus. The vaccine for Hepatitis B, which is one of the most common vaccines nowadays, is successfully used to prevent infection of Hepatitis B.

1.3 Serological tests

Serological tests are commonly used laboratory tests to find antibodies specific to a selected antigen. Such tests rely on a host's ability to produce an antibody for a specific antigen. The usual approach is to take a *serum* (blood) from the host, mix it with antigen and inspect the following reaction. One of the most widely used serological tests nowadays is the *enzyme-linked immunosorbent assay (ELISA)* test. ELISA test makes use of enzymes linked to antibodies which react with a substrate and produce colour (Figure 1.5). The intensity of the colour is proportional to the concentration of antibodies that are present. For more information about ELISA tests, see [21]. ELISA tests can be used in vaccination studies to study a patient's response to the vaccination (such as Hepatitis B vaccination) by detecting antigen-specific antibodies.

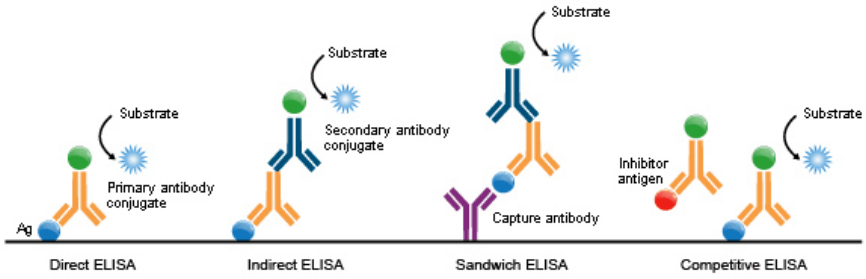


Figure 1.5: A high-level overview of how different types of ELISA tests work [22]

1.4 Machine learning

1.4.1 Supervised learning

Supervised learning is a strategy in machine learning which goal is to harness training data pairs of feature vectors ($x = (x_0, x_1, ...x_n)$ where n is the dimension of the *feature space*) and corresponding targets (y) to learn to predict targets for previously unseen feature vectors.

Classification is one of the basic tasks in supervised learning. The goal is to predict a discrete (i.e. having only a finite number of values) target, also called *label*, based on the given input feature vector. A common type of classification is a *binary classification* problem where the target is one of 2 classes that are usually named as *positive* and *negative* class.

There are many evaluation metrics for classification tasks. Widely used and easily interpretable way of evaluating a binary classifier is to use a confusion matrix. Let's consider y as a true label of a data point x from dataset X and y' as a label that has been predicted by classifier at hand. In the following definition, -1 is a negative label, and $+1$ is a positive label. Then, a confusion matrix for a binary classifier (Table 1.2) is a matrix consisting of 4 terms, namely:

True Positives (TP) :

$$TP = |\{x \mid y = +1 \wedge y' = +1\}| \tag{1.2}$$

True Negatives (TN) :

$$TN = |\{x \mid y = -1 \wedge y' = -1\}| \tag{1.3}$$

False Positives (FP) :

$$FP = |\{x \mid y = -1 \wedge y' = +1\}| \tag{1.4}$$

False Negatives (FN) :

$$FN = |\{x \mid y = +1 \wedge y' = -1\}| \tag{1.5}$$

Table 1.2: A confusion matrix of a binary classification task

		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

Terms from the confusion matrix are used to compose other terms useful for evaluating classifiers, which will be used throughout the thesis:

Precision is defined as a proportion of correct positive predictions. In the case of this work, it is a number of sequences correctly predicted as Hep B–binding divided by a number of all sequences predicted to be positive.

$$Precision = \frac{TP}{TP + FP} \tag{1.6}$$

Recall is defined as a proportion of actual positive data points that were predicted correctly. In the case of this work, it is a number of sequences correctly predicted as Hep B–binding divided by a number of positive sequences.

$$Recall = \frac{TP}{TP + FN} \tag{1.7}$$

False positive rate (FPR) is defined as a proportion of wrongly predicted data points as negative and the total number of negative data points. In the case of this work, it is a number of sequences wrongly predicted as Hep B non-binding divided by a number of negative sequences.

$$FPR = \frac{FP}{FP + TN} \tag{1.8}$$

Accuracy is defined as a ratio of correctly predicted labels over all predictions. The main shortcoming of accuracy is that it provides a distorted view of the performance for an imbalanced dataset. As an example, for a dataset with 90 % of negative data points and 10 % of positive datapoins, a classifier making only negative predictions would achieve accuracy of 0.9 (90 %). That is why accuracy is not a suitable metric for highly imbalanced datasets.

In the case of this work, accuracy is a number of sequences correctly predicted as either Hep B–binding or non-binding divided by a number of all sequences.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.9}$$

F1 score is defined as a harmonic mean of precision and recall.

$$F_1 = \frac{2}{Precision^{-1} + Recall^{-1}} \tag{1.10}$$

There are also curves that are commonly used for evaluation of binary classifiers, such as *Receiver operating characteristic (ROC)* curve which plots recall against the FPR and *Precision-Recall curve (PRC)* which plots precision against the recall. In both cases, the values are plotted for a variable positive classification threshold. The area under such a curve can be reported as a single-value metric and be used for the classifier's evaluation.

1.4.2 Decision tree

Decision tree is one of the most basic models that can be used for classification. The idea of decision trees is to iteratively build up a tree from given training data such that at each iteration the best split is selected. For a feature with discrete values, the data can be split according to each of the values. For a feature with continuous values, a threshold is necessary.

The goodness of the split is measured in multiple ways. The one used in *Classification And Regression Trees (CART)* algorithm is called the *gini index*. Gini index can be seen as a probability of misclassification of a new element. Therefore it reaches its minimum (0) if all data points have the same label and its maximum (0.5) when there is the same ratio of all class labels. For dataset D consisting of k classes and p_j which is a relative frequency of class j in D, the gini index is defined as follows:

$$GI(D) = \sum_{i=0}^{k} p_i(1 - p_i) \tag{1.11}$$

During a decision tree construction, gini index can be used within the *information gain* where the split with the highest information gain is selected. Let's consider dataset D, feature i for which the information gain is being computed, and D_0, D_1 which are the resulting datasets after the split, then information gain is defined as:

$$IG(D, i) = GI(D) - \frac{|D_0|}{|D|}GI(D_0) - \frac{|D_1|}{|D|}GI(D_1) \tag{1.12}$$

For making predictions using the constructed decision tree for a data point, path from the root node to leaf node is followed by taking paths based on the conditions set during the training. The leaf node determines the predicted label by a majority vote of the training data points in the leaf node.

A common extension of decision trees is to use ensembles of decision trees such as *Random Forests* or *Extremely randomized trees (ET)* (Methods Section 2.2.1.3). One of the main advantages of these randomized ensembles is its ability to mitigate overfitting.

1.4.3 Unsupervised learning

In *unsupervised learning*, there are no targets for the data. Unsupervised learning is more about finding patterns, similarities or summary of given data.

Clustering is a typical example of unsupervised learning. The main goal of clustering is to group a set of objects such that similar objects are in the same clusters (groups) and dissimilar objects are separated into different clusters (groups) [23].

Another example of unsupervised learning is a nonlinear dimensionality reduction technique called *t-Distributed Stochastic Neighbor Embedding (t-SNE)* [24]. It is especially intended for visualization of high dimensional data (usually in two or three dimensions).

1.4.4 Self-supervised learning

In the world, there are large amounts of unlabeled data, but much less labeled data. This problem has been addressed by a field called *self-supervised learning* which can be seen as a subclass of unsupervised learning. The goal of self-supervision is to get supervision from the unlabeled data itself. Hence model learns to predict a hidden part of its input using the other parts from the input. This way, it is possible for models to learn the representation of input without specific labels. Self-supervised learning is a technique utilized in the latest state-of-the-art NLP models [25]. A typical task for NLP models is to mask out random words of the input and let the model predict what words should be filled in there (Figure 1.6).

1.4.5 Train-test split

In machine learning it is essential to evaluate trained estimators such that estimator does not perform well only on the data it has been trained on but also on the previously unseen data (estimator "generalizes" well). *Overfitting* is "learning too much" from the training data and therefore capturing patterns and noise in the data which is unlikely to occur generally. On the other hand, *underfitting* is failing to capture underlying trends and patterns in the training data (Figure 1.7).

For this reason, data is commonly split to *train* and *test* datasets. The most common way to do so is a *random split*. The model is then trained on the train set, and its ability to generalize is evaluated on the test set.

Data can also be split into 3 parts, namely *train*, *validation* and *test* set. The validation set is useful for model selection or optimizing *hyperparameters*, which in short are model's parameters that have to be set before training and are not learned from training. Otherwise, if models or hyperparameters would be chosen based on the test set, the model's ability to generalize would be compromised, and it could be overfitting to the test set.

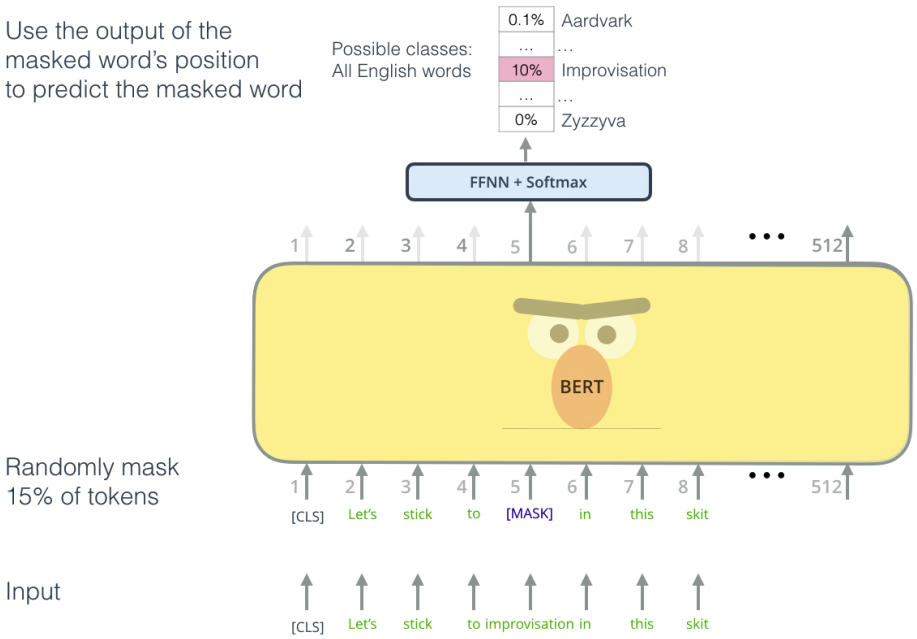


Figure 1.6: "Masking out" self-supervised language modeling task on a BERT model [26]

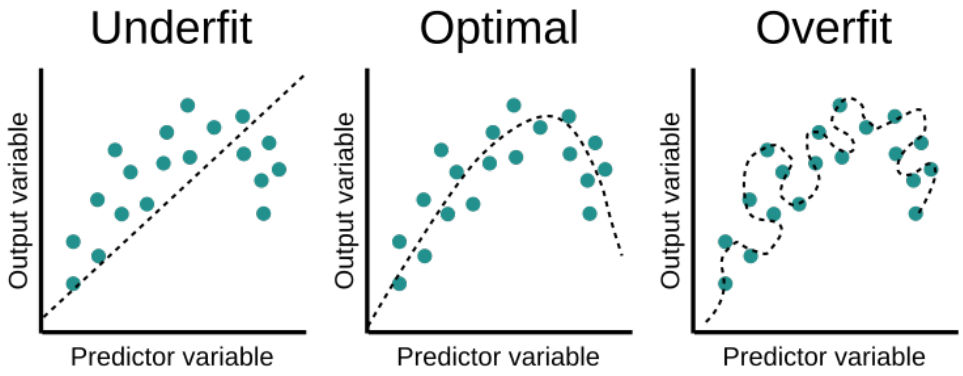


Figure 1.7: Plots showcasing how underfit, overfit, and a well fit predictors behave [27]

1.4.6 Undersampling

Undersampling is a common practice in machine learning which consists of removing data points from data. One of the most popular methods is to use *random undersampling*, in which data points are being removed randomly. This technique is particularly useful to get balanced dataset (dataset with equal distribution of classes) out of an imbalanced one by undersampling the majority class(es). The other reason for undersampling might be to reduce the computational resources needed for training.

1.5 Deep learning

Deep learning is a subfield of machine learning utilizing neural networks that are based on trying to build more complex representations based on the simpler ones [28].

1.5.1 Perceptron

Perceptron is the basic building block of *neural networks*. Let $\mathbf{x} \in \mathbb{R}^n$ be an input vector, $\mathbf{w} \in \mathbb{R}^n$ be a weights vector and $w_0 \in \mathbb{R}$ a *bias*, σ a non-linear activation function then perceptron can be defined as follows:

$$f(\mathbf{x}) = \sigma\left(w_0 + \sum_{i=1}^{n} w_i x_i\right) \tag{1.13}$$

The major shortcoming of the single-layer perceptron is that it is able to represent only linear functions. For example, it cannot represent simple XOR function in 2-dimensional space because it is not linearly separable. However, the XOR function is linearly separable by a hyperplane in the space of dimension 3. Therefore using more layers can solve the problem of representing non-linear functions.

1.5.2 Feedforward neural network

Feedforward neural network or *multilayer perceptron (MLP)* is the basic deep learning model. Name *feedforward* comes from the concept that information in these networks flows only forward, meaning there are no feedback connections intended for feeding the output of the network back to it as an input. The goal of such models is to approximate function f^*. Therefore a feedforward network tries to find the best parameters θ such that f_θ results in the best approximation. As the name "multilayer perceptron" suggests, these architectures consist of multiple layers. The layers between the input and output layer are commonly called *hidden layers*. The layers of MLP can be seen as functions, and then the whole network can be seen as a composition of these functions. For example let's take a 3-layer MLP with f_1, f_2, f_3 layer functions, then $f(\mathbf{x}) = f_3(f_2(f_1(\mathbf{x})))$ is the definition of the whole MLP (Figure 1.8). More information can be found in [29].

In fact, in [31], it has been shown that *universal approximation theorem* holds for MLP with a single hidden layer. This theorem states that any function can be approximated by such MLP with an arbitrary degree of accuracy.

1.5.3 Activation functions

Activation functions are the essential parts of neural networks. By [30], activation functions serve to approximate non-linear functions, because no matter how many layers it has "a linear function of a linear function is linear".

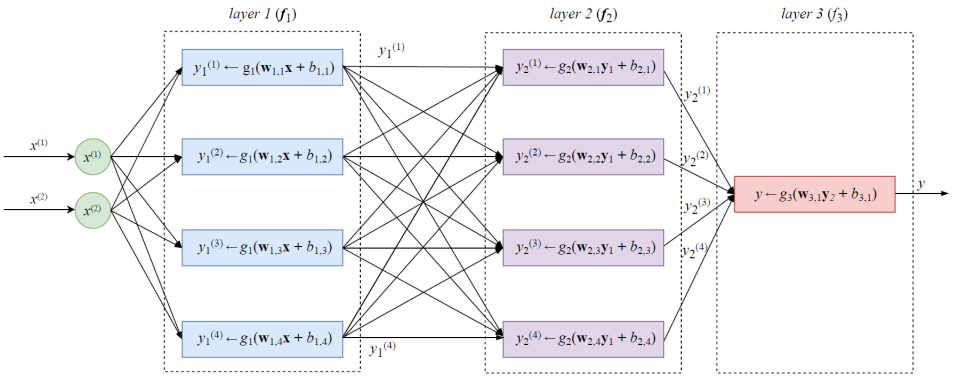


Figure 1.8: Structure of a multilayer perceptron with 2-dimensional input, 2 hidden layers with 4 units and a single unit output layer [30]

Rectified linear unit (ReLU) (Figure 1.9) is an activation function defined as:

$$relu(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{otherwise} \end{cases} \tag{1.14}$$

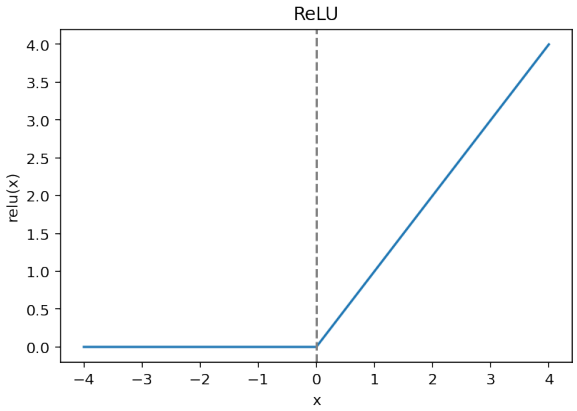


Figure 1.9: Rectified linear unit (ReLU)

Softmax is widely used activation function mainly in the last layer of multi-label classification neural networks. Softmax turns *logits*, which refer to the outputs of the last layer of a neural network prior to applying activation functions, into corresponding probabilities that sum up to 1. Let n be a number of classes, $\mathbf{y} = (y_1, ..., y_n) \in \mathbb{R}^n$ be the vector consisting of logits, then softmax function $\sigma \colon \mathbb{R}^n \mapsto \mathbb{R}^n$ is defined as:

$$\sigma(\mathbf{y})_i = \frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_j}} \tag{1.15}$$

Then the output of softmax function is a probability distribution over all classes. Often used variation of softmax is *LogSoftmax* which is defined as follows:

$$\sigma(\mathbf{y})_i = log\left(\frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_j}}\right) \tag{1.16}$$

In both cases, the position in the output vector with maximum value can be interpreted as a prediction of the corresponding label.

1.5.4 Backpropagation

In order to train neural networks, it is necessary to optimize (minimize or maximize) an objective function of some sort. In the case of neural networks when minimizing the objective function, such function is called a *loss function*, a cost function or an error function. The loss function is calculated for the output after performing a forward pass in the network, and *gradient descent* is used to compute the change necessary to minimize the loss function. In gradient descent technique,a *gradient* of the loss function w.r.t. to the weights and biases is calculated, and step (size of the step is dependent on the *learning rate*) in the direction of the negative gradient is taken. Formally, in the n-dimensional space, for a function $f : \mathbb{R}^n \mapsto \mathbb{R}$, gradient $\nabla f : \mathbb{R}^n \mapsto \mathbb{R}^n$ at point $\mathbf{x} = (x_1, ..., x_n)$ is defined as:

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f}{\partial x_1}(\mathbf{x}) \\ \vdots \\ \frac{\partial f}{\partial x_n}(\mathbf{x}) \end{pmatrix} \tag{1.17}$$

The shortcoming of this optimization technique is that it might get stuck in a local minimum or a saddle point of the loss function. For more information, see [29].

Negative log-likelihood (NLL) is an example of a loss function which is commonly used in conjunction with LogSoftmax for classification tasks and can be found grouped into a single function called *cross-entropy loss*. It is defined as a LogSoftmax value corresponding to the correct class label multiplied by -1. Formally, let n be a number of classes, i the index of a softmax position corresponding to the true class label, $\mathbf{y} = (y_1, ..., y_n) \in \mathbb{R}^n$ the vector consisting of logits, then the conjunction of LogSoftmax and NLL, i.e. the cross-entropy loss can be defined as:

$$loss(\mathbf{y}) = -log\left(\frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_j}}\right) \tag{1.18}$$

1.5.5 Transformers

In *sequence modelling*, especially in *Natural Language Processing*, deep learning models play an essential role. State-of-the-art models moved from *word2vec* models introduced in 2013 [32] to pure *recurrent neural network (RNN)* capable of processing sequential inputs of variable lengths (as opposed to a regular MLP) [33, 34, 35]. Simple RNN suffered from *vanishing gradient*, which prevented them from capturing long-range dependencies (analysed in [36]). To mitigate the vanishing gradient problem and capture long-term dependencies, special RNN architecture called *long short-term memory (LSTM)* has been proposed in [37]. After that, architectures based on *transformers* [38] became the latest state-of-the-art. The main advantage of transformers is that they are not of sequential nature as previous models, and therefore the parallelization is much easier.

The "cornerstone" mechanism of transformers is *attention*. Attention mechanism introduced a way to "pay attention" to the relevant parts of a sequence while "ignoring other parts"(Figure 1.10). An attention function can be perceived as a mapping of a *query* and a set of key-value pairs to an output, where the query, keys, values and output are all vectors. A weighted sum of the values is the output, where the weights assigned to the values are computed using a *compatibility function* of the query with the corresponding key [38]. There are many variants of attention, but the one relevant for this work is the *scaled dot-product attention* (Figure 1.11a).

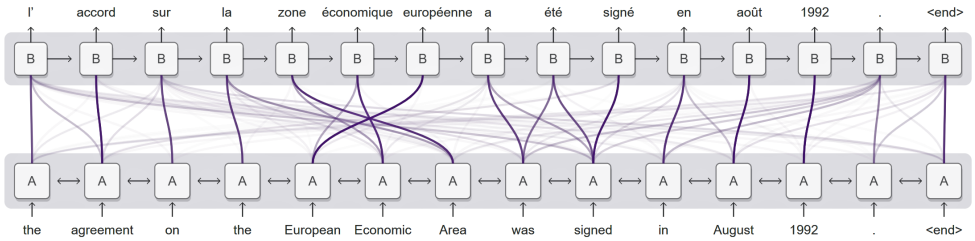
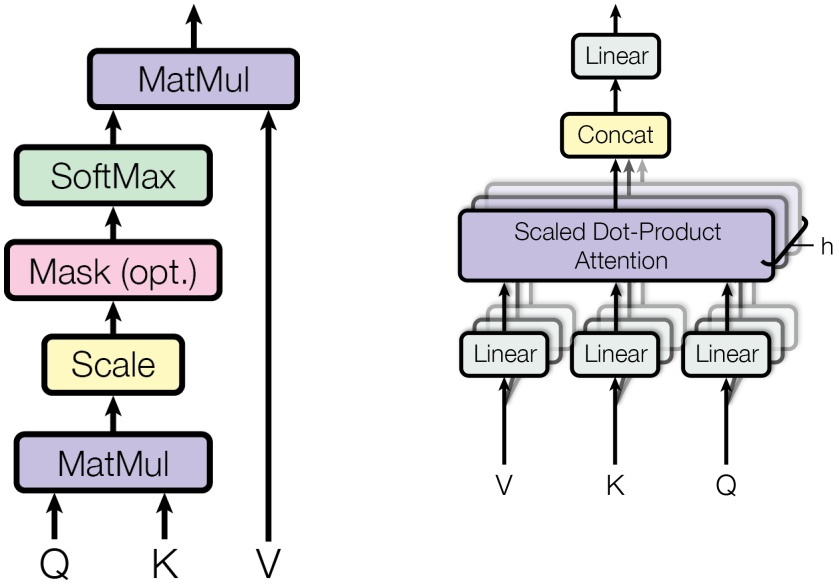


Figure 1.10: Neural machine translation attention visualization for a sentence translation from English to French [39]

Scaled dot-product attention from [38], takes queries and keys of dimension d_k and values of dimension d_v as an input. It computes the dot products of queries and keys and scales them by dividing them by $\sqrt{d_k}$ and applies a softmax function which outputs the weights for the values. This can be viewed as following matrix operations, where Q, K and V are matrices of queries, keys and values:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1.19}$$

Multi-head attention from [38], allows models to attend to information



(a) Scale Dot-Product Attention

(b) Muti-Head Attention

Figure 1.11: (a) Scaled Dot-Product Attention. (b) Multi-Head Attention consisting of multiple attention layers [38]

from different representation subspaces at the same time. It can be seen as applying multiple attention layers. Formally:

$$\begin{aligned} MultiHead(Q,K,V) &= Concat(head_1,...,head_h)W^O \\ \text{where } head_i &= Attention(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{1.20}$$

Where the parameter matrices are as follows:

- $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$
- $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$
- $W_i^V \in \mathbb{R}^{d_{model} \times d_k}$
- $W^O \in \mathbb{R}^{hd_v \times d_{model}}$

As described in [38], the architecture of a transformer has an "encoder–decoder" structure, where *"encoder maps an input sequence of symbols* $(x_1, ..., x_n)$ *to a sequence of continuous representations* $\boldsymbol{z} = (z_1, ..., z_n)$*. Given* $\boldsymbol{z}$*, the decoder then generates an output sequence* $(y_1, ..., y_n)$ *of symbols one element at a time"*. The model is *auto-regressive*, meaning that it uses the previously generated output as its input.

Besides the attention sub-layers in both encoder and decoder, there is a fully connected feedforward network applied to each position separately and identically. This network consists of 2 linear transformations with ReLU in between:

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2 \tag{1.21}$$

Embedding layers in the architecture are used to convert the input and output tokens to vectors of dimension d_{model}. The output of embedding layers is summed up with a *positional encoding* vector of the same length to inject information about the order of the sequence.

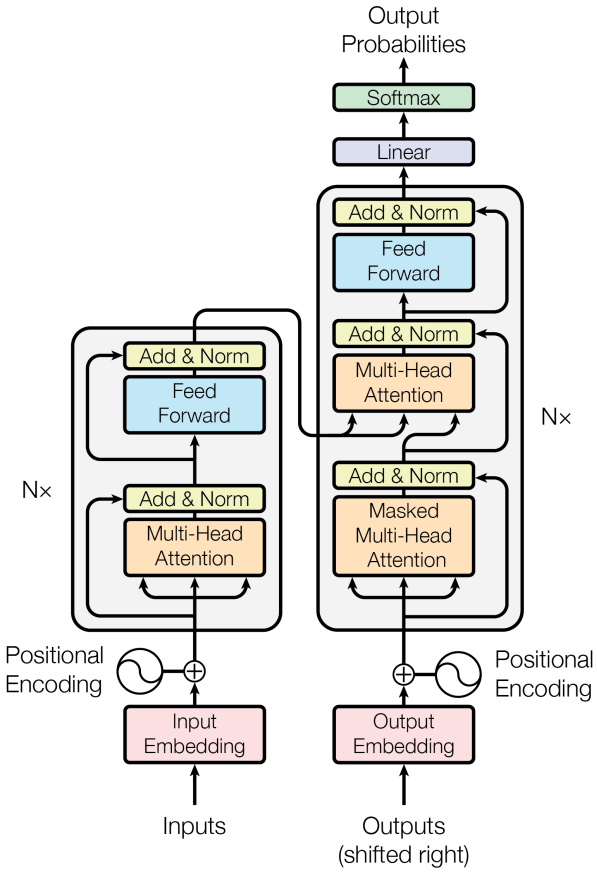


Figure 1.12: The transformer model architecture [38]

1.5.6 Transfer learning

Intuitively, as the name suggests, transfer learning is about transferring knowledge and using this prior knowledge as a starting point to continue learning further. Formally, *transfer learning* can be defined as follows [40]:

> *"Given a learning task T_t based on D_t, and we can get the help from D_s for the learning task T_s. Transfer learning aims to improve the performance of predictive function $f_T(\cdot)$ for learning task T_t by discover and transfer latent knowledge from D_s and T_s, where $D_s \neq D_t$ and/or $T_s \neq T_t$. In addition, in most cases, the size of D_s is much larger than the size of D_t."*

This way of *pre-training* models is especially useful for deep learning models for fields where the amount of specific data is insufficient, such as in computer vision or Natural Language Processing.

Fine-tuning is a process that comes after pre-training the model. After the model has been pre-trained, it can be further improved for task-specific data while re-using the knowledge from pre-training. Fine-tuning can be done in various ways. For example, some of the model's (pre-trained) parameters might be "frozen", and only subset of the parameters may be further changed. This allows to reduce the training time, save computational resources and avoid potential overfitting.

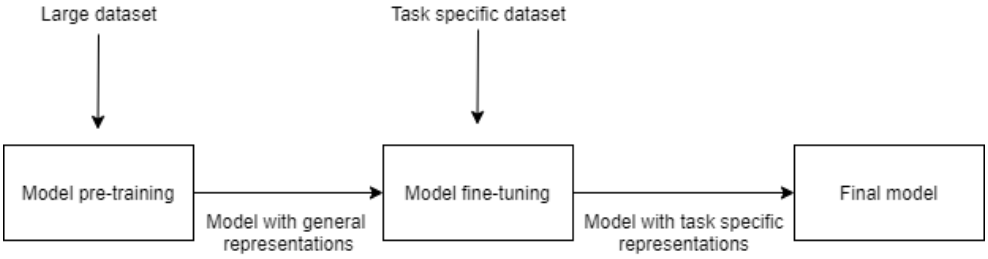


Figure 1.13: High-level overview of model pre-training and fine-tuning

1.5.7 BERT and RoBERTa

Bidirectional Encoder Representations from Transformers (BERT) proposed by [41] is a deep learning architecture constructed from *Transformer Encoders* which were proposed by [38]. This architecture formed the basis for the model architecture used in this work.

In [41], 2 model sizes have been introduced: smaller $\text{BERT}_{\text{BASE}}$ and very large $\text{BERT}_{\text{LARGE}}$ which is the model that achieved the state-of-the-art performance on multiple NLP tasks (GLUE [42], MultiNLI [43]...). These 2 models are stacks of *Transformer Encoders* (Figure 1.14) with a few different hyperparameter values (Table 1.3) from the original Transformer Encoder from [38].

As described in [38], in the pre-training phase, 2 objectives are used:

Masked Language Model (MLM) objective inspired by so-called "Cloze procedure" [44]. In this objective, randomly selected tokens in the input

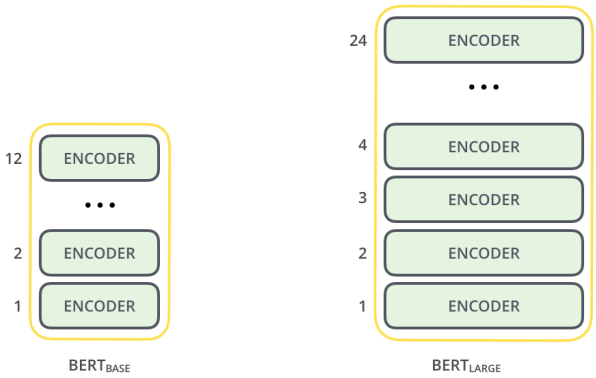


Figure 1.14: High-level architecture of BERT models consisting of stacks of Transformer Encoders [26]

Table 1.3: Values of hyperparameters of the original Transformer Encoder from [38], values used in $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ and values used in $\text{RoBERTa}_{\text{SMALL}}$, the model used in this thesis. Namely the number of encoder layers, attention heads in the Multi-Head attentions and number of hidden units in the feedforward networks

Architetcture	Encoder layers	Attention heads	Hidden units
Original	6	8	512
$\text{BERT}_{\text{BASE}}$	12	12	768
$\text{BERT}_{\text{LARGE}}$	24	16	1024
$\text{RoBERTa}_{\text{SMALL}}$	4	8	256

are masked out, and the goal for the model is to predict the vocabulary id of the masked word based on the other parts (Figure 1.6). In the case of BERT pre-training, random 15 % of words is selected and out of these 80 % is masked out, 10 % are left unchanged and 10 % are replaced with a randomly selected token. In the case of this thesis, amino acid letters are considered to be words, and therefore amino acids are being masked out, and model is trying to predict them.

Next Sentence Prediction (NSP) objective is a binary classification task, where for 2 given segments model should classify whether they follow each other in the original data (text) or not.

Few changes in the architecture of BERT and the pre-training process resulted in *Robustly optimized BERT approach (RoBERTa)*, its better-performing successor proposed by [45]. RoBERTa revealed undertraining of BERT and exhibited performance gains for numerous benchmarks. The main architectural difference is removing the NSP objective. It has been shown [45] that removing the NSP for $\text{BERT}_{\text{BASE}}$ results in matching or slightly improved performance on the downstream tasks. RoBERTa is an architecture especially optimized

for pre-training and self-supervision. Analogically to the BERT there are 2 models: $\text{RoBERTa}_{\text{BASE}}$ and $\text{RoBERTa}_{\text{LARGE}}$ with the same hyperparameter values of the architecture from Table 1.3.

Both BERT and RoBERTA yielded outstanding results for numerous tasks (GLUE benchmark [42], RACE benchmark [46]...) by pre-training on large amounts of data and fine-tuning the models for these downstream tasks.

In the case of classification problem special `[CLS]` token is used on the first position of the input. Corresponding first output token is then used to accumulate information about the whole sentence (in the case of this thesis whole sequence) and is used as an input for the classifier network, fully-connected feedforward network (Figure 1.15).

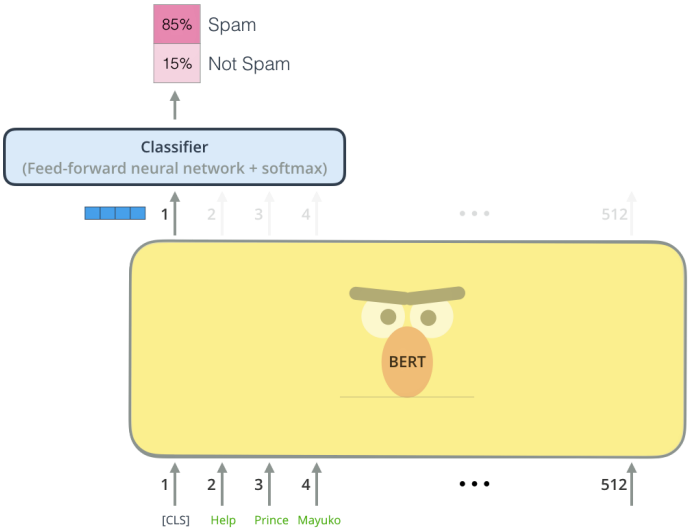


Figure 1.15: Illustration of a binary classification using BERT [26]

Chapter 2

Methods

This chapter describes my approach to solve the task of predicting antibody–antigen binding, including data preprocessing, models that have been used as well as how the models have been evaluated in Chapter 3: Results.

2.1 Data

2.1.1 Observed Antibody Space

Advances in the next-generation sequencing (previously described in Section 1.1.5) allowed gathering large amounts of natural antibody sequences. However, the data is usually scattered, and different formats are being used, which makes working with different data sources and re-applying already developed techniques harder. The *Observed Antibody Space (OAS)* database, introduced in [47], mitigates this problem by collecting and processing antibody variable region sequences to the same format from more than 60 different studies (resulting in over one billion sequences).

OAS database is structured into studies which are further divided into data units. A data unit is a collection of sequences with the same metadata information. Metadata of a data unit consists of:

Chain annotating whether given sequences are heavy or light chain sequences.

Isotype (class) of antibodies which is either provided or determined by the constant region.

Age of the human subject.

Disease (if applicable) of the subject at the time of B-cell extraction.

Vaccine (if applicable) given to the subject prior to B-cell extraction.

B-cell subset from which antibodies originate.

Species of the subject. For example human, mouse, camel...

Author who first published the study together with the date of publication.

Link to the publication with the study.

Size defining the number of non-redundant (unique) sequences.

B-cell source (organ/tissue) from which the B-cells were extracted from.

Subject (identifier) whose B-cells were extracted.

Longitudinal defining the timepoint at which B-cells were extracted (if the study has been conducted over a period of time).

For each sequence from a data unit following data is provided:

Amino acid sequence of the variable region formatted as a string.

V gene annotation according to the *IMGT numbering scheme* [48].

J gene annotation according to the IMGT numbering scheme.

CDR3 region defined by IMGT.

IMGT-numbered sequence amino acid sequence annotated according to the IMGT numbering scheme.

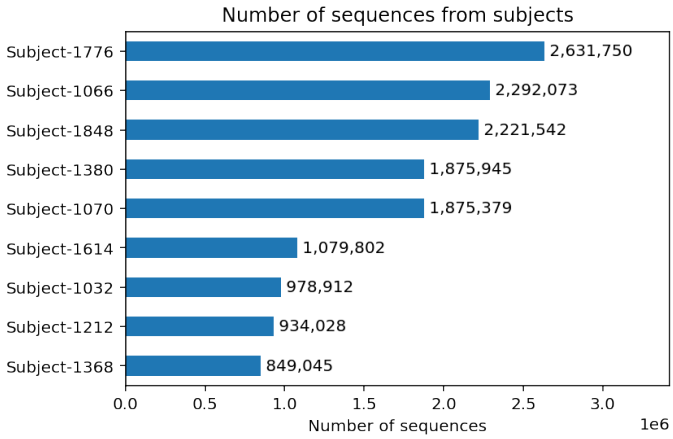
Redundancy defining how often does this sequence appear in the given data unit.

Full documentation of the data and links to download the data can be found at [49].

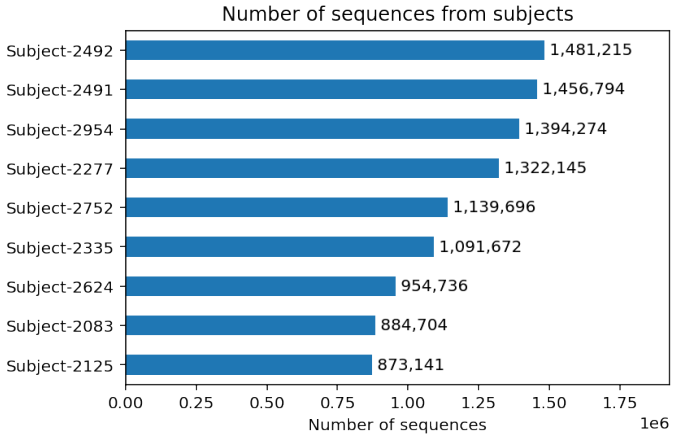
2.1.1.1 Galson 2015 and 2016

Galson 2015 [50] and Galson 2016 [51] are the two Hepatitis B vaccination studies from OAS database included in this thesis. In both studies, Hep B labels have been constructed by ELISA test on blood serum and are further used in this work as a "ground truth".

For each study, there are 9 source subjects from which the sequences originate. Distribution of sequenced sequences among subjects can be seen in Figure 2.1. The distribution of B-cell annotations can be seen in Figure 2.2.



(a) Galson 2015

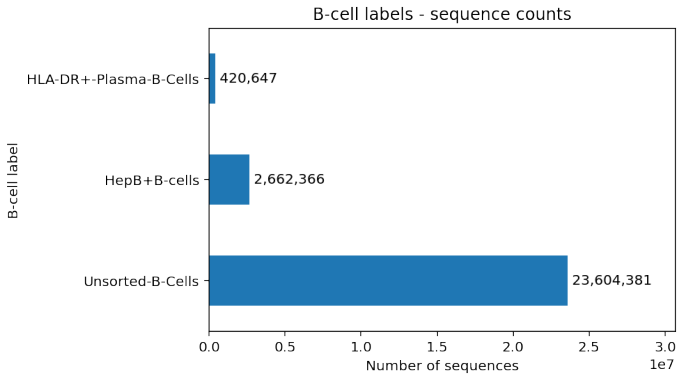


(b) Galson 2016

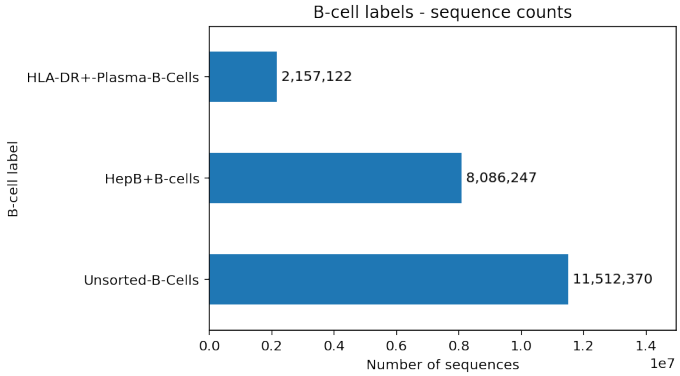
Figure 2.1: Number of sequences from all subjects in the (a) Galson 2015 and (b) Galson 2016 studies

2.1.2 Clonotyping

In the beginning, data size is reduced, and the artificial diversity is removed by *clonotyping*. Based on the clonotyping definition (Chapter 1 Section 1.1.4), the data is initially grouped by V, J regions and CDR3 lengths. Each of such groups is then clustered (in parallel) into clonotypes in a greedy fashion by constructing the largest clonotypes first. Within the group, as it already fulfills pre-requisites of same V, J regions and same CDR3 length, 2 sequences might be in the same clonotype if there is a maximum of 1 mismatch per 12 amino acids [52]. Meaning that sequences of length < 12 might contain 1 mismatch, sequences of length in $[12, 24)$ might contain 2 mismatches...If there are multiple clonotypes with the largest size, the "tightest" one is selected



(a) Galson 2015



(b) Galson 2016

Figure 2.2: Number of all B-cell labels in the (a) Galson 2015 and (b) Galson 2016 studies, where "HepB+B-cells" is the positive label for our task

by taking the one with the minimal number of total mismatches in CDR3 region between the centroid sequence and all other sequences within the cluster (full pseudo code is available in Algorithm 1).

For each of the clonotype sequences with the most common CDR3 region within the clonotype are extracted, and the representative sequence is randomly sampled out of these sequences. Those are then used as an input for the models. This way, the data consists of sequences that make up the immune repertoire, which is still highly heterogeneous but does not contain duplicates and artificial diversity. In this thesis, "clonotypes", "representative sequences" and "sequence clusters" are used interchangeably.

Each of the clonotypes (representative sequences) is marked as positive (binding to Hep B antigen) or negative (non-binding to Hep B antigen) as a "majority vote" of the original sequences in the clonotype, where everything except Hep B antibodies is considered as non-binding (i.e. antibodies with label "HLA-DR+-Plasma-B-Cells" are considered as non-binding). Formally,

$y\colon \mathbb{C} \mapsto \{-1, +1\}$ where $\mathbb{C}$ is a set of all clonotypes, while for $c \in \mathbb{C}$ c_+ is a set of binding sequences from given clonotype and c_- is a set of non-binding sequences, -1 and $+1$ match negative and positive class accordingly, then:

$$\forall c \in \mathbb{C}\colon y(c) = \begin{cases} +1 & \text{if } \frac{|c_+|}{|c_+|+|c_-|} > 0.5 \\ -1 & \text{otherwise} \end{cases} \tag{2.1}$$

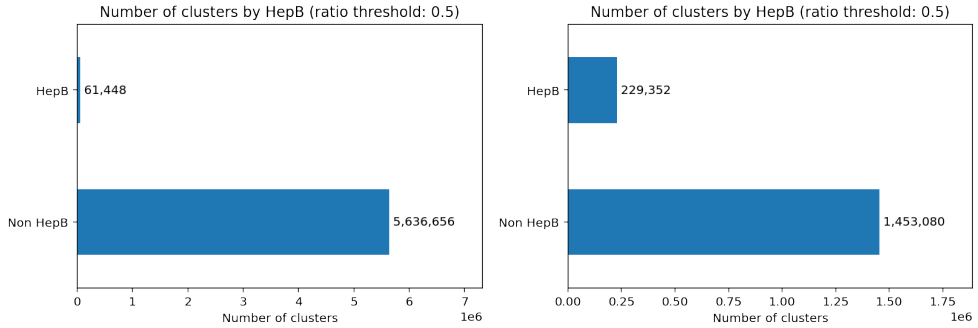
Algorithm 1 High-level overview pseudocode of clonotyping algorithm that has been used. The exact implementation uses *Pandas* library for data grouping and handling and *numpy* for computing CDR3 mismatches and getting the largest clusters.

```
function CLUSTER_CLONOTYPES(sequences)
    grouped_sequences ← sequences grouped by V, J regions and CDR3 length
    clusters ← []
    for each group in grouped_sequences do
        group_clusters ← []
        unclustered_seq_cnt ← len(group)
        while unclustered_seq_cnt > 0 do
            largest_clusters ← all largest clusters from the group ▷ Based on the CDR3 mismatches
            best_cluster ← the "tightest" cluster from the largest_clusters
            unclustered_seq_cnt ← unclustered_seq_cnt − len(best_cluster)
            group_clusters ← group_clusters ∪ best_cluster
        end while
        clusters ← clusters ∪ group_clusters
    end for
    return clusters
end function
```

Final counts of Hep B antibodies and non–Hep B antibodies on Galson 2015 and Galson 2016 are shown in Figure 2.3. Clonotyping reduced the number of data points heavily (Table 2.1). Most of the clonotypes consist of sequences from single subjects (Figure 2.4).

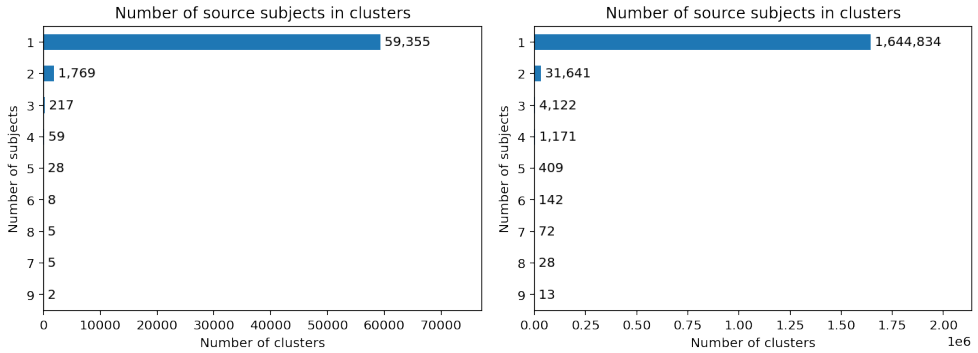
Table 2.1: Size reductions achieved by clonotyping on both Galson 2015 and Galson 2016 datasets

Study	Sequences count	Clusters count
Galson 2015	14,738,476	5,698,104
Galson 2016	10,598,377	1,682,432



(a) Galson 2015 (b) Galson 2016

Figure 2.3: Targets distribution in the (a) Galson 2015 and (b) Galson 2016 studies. Non–Hep B sequences greatly outnumber the Hep B sequences, and that is the reason why negative undersampling is used in this work (Section 2.1.3)



(a) Galson 2015 (b) Galson 2016

Figure 2.4: Number of source subjects for clusters (clonotypes) in the (a) Galson 2015 and (b) Galson 2016 studies. As figures suggest, clusters (clonotypes) originating only from a single subject are the most common

2.1.3 Training and validation data

Sequences from Hep B vaccination study *Galson 2015* [50] have been used for training and validation data.

Following the random split, the *subject split* has been used to split the data for training and validation. Subject split is intended to simulate the real-world scenario, where predictions are performed on subjects different from the training subjects. Based on the Figure 2.4 it is clear that most of the clusters are made up of sequences with a single source subject. Numbers of the clonotypes with single source subjects for each subject range from slightly below $300,000$ up to over 1 million (Figure 2.5).

The single-subject clonotypes of 5 least frequent subjects (Subject-1368, Subject-1070, Subject-1212, Subject-1032 and Subject-1614) are set aside as a validation set. The rest of the data, from 4 subjects, is marked as a training set. This reflects the real-word scenario because the data consists of all the clusters from the remaining 4 subjects including single-subject clonotypes, which represent subject-specific clonotypes, as well as multi-subject clonotypes, which can be seen as a *public antibody repertoire* shared among individuals.

After splitting the data, negative samples from each of the splits are further randomly undersampled to achieve balanced datasets. This random undersampling is weighted by cluster (clonotype) sizes. Therefore in clusters' representative sequences (which make up the training and validation sets at this point), the ones originating from larger clusters are more likely to get sampled than sequences from smaller clusters. Final training and validation set sizes are $99,004$ and $23,892$, respectively.

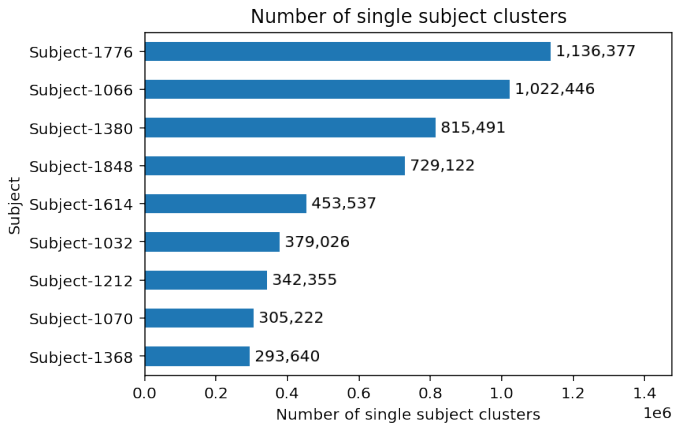


Figure 2.5: Graph showing how many single-subject clusters there are in Galson 2015 per each subject

In general, sequences within a single Hep B cluster (clonotype) seem to mostly share the same or have very similar CDR3 regions (Figure 2.6). In contrast, sequences within non–Hep B clusters are more diverse in terms of CDR3 regions (Figure 2.7).

2.1.4 Test data

Sequences from a separate Hep B vaccination study *Galson 2016* [51] have been used as test data. As opposed to the training and validation data, besides clonotyping and using representative sequences, the test data is not randomly undersampled in any way.

Figure 2.6: Sequence logos, representing the frequency of amino acids at each position, of the top 10 clusters with the most Hep B sequences and Hep B ratio above 0.95

2.2 Models

2.2.1 Baseline models

This section describes the methodology used for constructing various baseline models which served for comparison to the deep learning approach. Different data representations have been used for the baseline models and for the deep learning approach.

2.2.1.1 Fingerprints

Molecular fingerprints have been used to create a chemical feature representation of the protein sequences. The idea of molecular fingerprints lies in generating feature vectors by applying kernels to molecules [53]. The usual procedure consists of extracting features, hashing them and setting bits/positions of output vector based on those features. The fingerprint type used is the *Circular (Morgan)* fingerprint described in [54], with binary representation, final vector length of 512 and a radius of 6. The fingerprints are constructed only from CDR3 regions.

Figure 2.7: Sequence logos, representing the frequency of amino acids at each position, of the top 10 largest clusters, where 9 out of clusters would be considered as non–Hep B based on the 0.5 ratio threshold (the one Hep B cluster is marked by bold title)

2.2.1.2 k-mers

The second data representation for baseline models is based on *k-mer* frequency. K-mers are defined as subsequences of length k. K-mers can be seen as an analogy of *n-grams* [55] used in NLP. K-mer frequency can provide a fixed-size vectors for variable-length sequences. However, the issue with k-mers (and n-grams) is that they cannot incorporate long-range relations within the sequence.

Actual feature vectors built from 3-mers contain (unnormalized) number of occurrences of given 3-mer in the given CDR3 sequence. Given 20 natural amino acids, final vectors are of length 8000, since there are 20^3 possible 3-mers.

2.2.1.3 Extremely randomized trees

Extremely randomized trees (ET), introduced by [56], is an ensemble model based on decision trees which randomizes both attribute and cut-point choice when splitting the tree. The construction of trees is similar to the algorithm

used for decision trees (Theoretical background Section 1.4.2), but the best node split is evaluated only for a subset of randomly chosen k features and for each of such feature x, the threshold for split is sampled uniformly at random from the interval $[min(x), max(x)]$. Afterwards, the best split from them is selected using a criterion measuring the goodness of the split (Theoretical background Section 1.4.2). The ET model serves as a baseline model for comparison to the suggested deep learning approach.

2.2.2 RoBERTa

In the deep learning approach, raw amino acid sequences are used as an input for *Robustly optimized BERT approach (RoBERTa)* which is responsible for both the sequence representations of antibodies and their classification.

The architecture used in this work is a reduced RoBERTa architecture which will be referred to as $RoBERTa_{SMALL}$. While the original BERT architectures $\text{BERT}_{\text{LARGE}}$ and $_{\text{BASE}}$ contain 110 million and 340 million parameters respectively, the $\text{RoBERTa}_{\text{SMALL}}$ architecture used int this work slightly less than $600,000$ parameters. This architecture has following differences from the original $\text{RoBERTa}_{\text{BASE}}$ architecture (1.3):

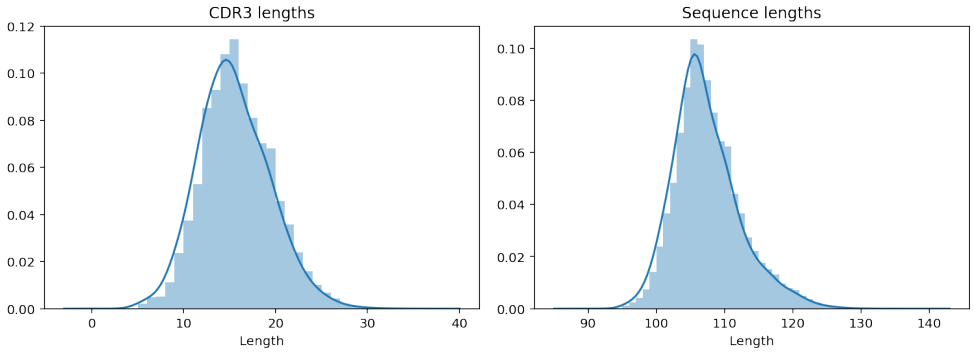
- 4 encoder layers
- 128 hidden size
- 256 inner hidden size in feedforward networks
- 8 attention heads

Based on the pre-training data, the maximum input sequence lengths of the architectures have been selected as 32 and 144 for *CDR3 model* and *heavy sequence model*, respectively.

Based on the sequence length distributions from Galson 2015 (Figure 2.8), all the full variable heavy chain sequences fit in this range. For CDR3 sequences there are some sequences which are longer and are therefore skipped during training. However, predictions on longer CDR3 sequences are made by truncating the input sequence (trailing region of the sequence is cut off).

2.2.2.1 Pre-training

RoBERTa model has been pre-trained by thesis supervisor David Příhoda (results in progress, unpublished) on a large corpus of human antibodies available from the OAS database. There are 2 models, one pre-trained on whole variable regions of the heavy chains and the other one pre-trained on CDR3 regions of the heavy chains. Temporal split has been used for the data, where 2011–2017 studies have been used as training data, 2018 studies as validation data and 2019 studies as test data. These splits have been further randomly



(a) CDR3 sequence lengths in Galson 2015 (b) Heavy sequence lengths in Galson 2015

Figure 2.8: Distribution of lengths of sequences in the Galson 2015 study for (a) CDR3 sequences and (b) full heavy sequences

down-sampled to 20 million sequences each while keeping the ratios of original source data units.

2.2.2.2 Training

Pre-trained architectures have been further trained on the Hepatitis B antibody specificity classification task in 2 ways.

First training approach took place on the *whole architecture*, fine-tuning both parts of the architecture, *encoder* responsible for the modelling of antibodies and *classification head* responsible for predicting the Hep B–binding property.

In the second approach, *only the classification head* (single fully connected layer) has been trained while keeping the parameters of the encoder "frozen".

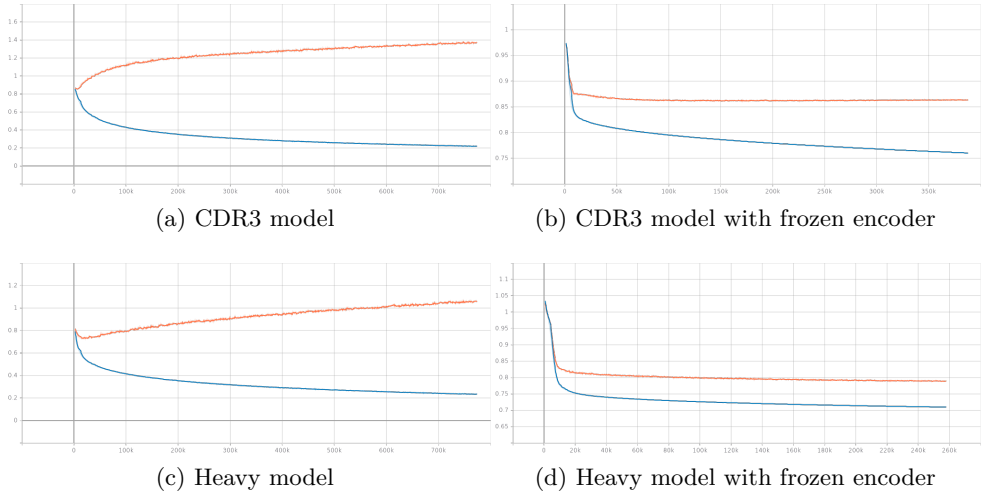
In both cases, the deep learning models have been trained on balanced training data (49,502 positive, 49,502 negative sequences) and balanced validation (11,946 positive, 11,946 negative sequences) data from the subject split. The validation data is used to control overfitting of the models. Numerous hyperparameter setups have been tried, including increased learning rates (such as $1e{-}4$ and $1e{-}3$ instead of $1e{-}5$), more training epochs (ranging from 2000 to 6000), more regularization by rising dropout probabilities (from 0.1 for both regular dropout and attention dropout to 0.25, 0.3 and 0.5), adding class weights by using a weighted loss function (such as 1:2, 1:3, 1:99 for negative to positive class weight ratio). Eventually all the final models have been trained with the following set of hyperparameters:

- 2000 epochs
- 256 batch size
- adam optimizer

- $1e{-}5$ target learning rate with inverse square root learning rate scheduler and $10,000$ warmup updates
- update frequency set to 1 (updating after every batch)
- 0.1 dropout and 0.1 attention dropout
- 144 max positions in positional embeddings for full variable heavy chain sequences and 32 for CDR3 sequences
- validate interval set to 1 (each epoch)

The number of epochs (2000) proved to be long enough to start overfitting or plateauing. Following the training, the best checkpoints have been taken based on the Negative log-likelihood (NLL) loss on balanced validation data, while LogSoftmax is being used as an activation function in the final layer.

Afterwards, models have been re-trained on the complete Galson 2015 dataset by combining the balanced training and validation splits while keeping the number of epochs set to the epoch from which the best checkpoint came from. These models were then considered to be final models applied to Galson 2016 test data (Results section 3.4).



(a) CDR3 model

(b) CDR3 model with frozen encoder

(c) Heavy model

(d) Heavy model with frozen encoder

Figure 2.9: Training (blue) and validation (orange) loss progress throughout the 2000 training epochs for RoBERTa models with encoder fine-tuning (a, b) and for RoBERTa models with frozen encoder. The curves have been smoothened by exponential moving average with a coefficient equal to 0.6

2.3 Evaluation

Evaluation of models in this work is generally performed in 2 ways. The *non-weighted evaluation* which treats all the representative clonotype sequences as equal with regard to evaluation. The second method is the *weighted evaluation* in which weights are used for all the metrics and other evaluation techniques proportionally to the size of the clonotype cluster from which given representative sequence originates (Methods Section 2.1.2). The weighted evaluation is intended to reflect real-world usage when larger clusters would occur more often, and therefore their correct prediction is more important.

For evaluation on the validation dataset, predictions are made on the both balanced (11,946 positive, 11,946 negative sequences), and whole data (11,946 positive, 1,761,834 negative sequences) and the performances are reported.

Evaluation on the test dataset proved to be a hard task, and therefore the performance is reported in regards to the similarity to the training data. The best matches of all Galson 2016 positive test sequences are found in both positive and negative Galson 2015 training sequences by aligning the sequences and getting the sequences with the best scores. The global alignment function from module `pairwise2` of `Biopython` library [57] has been used with *BLOSUM62* substitution matrix, open gap penalization set to -5 and extending gap penalization set to -2.

Afterwards, *identity* is used as a number of matching positions within the alignment divided by the alignment's length, where a pair of gaps is considered to be a mismatch. Then the positive data points are split based on the *identity* percentage of their CDR3 regions (value is multiplied by 100) into 6 bins consisting of sequences with 0–50 %, 50–60 %, 60–70 %, 70–80 %, 80–90 % and 90–100 % identity. For each of the classifiers, positive classification threshold is found such that all the models yield as close to the given FPR (it is either an exact match or below given FPR value). Then all classifiers, having roughly the same FPR, are used to make predictions, and recalls are reported for all the bins.

2.4 Implementation

On the implementation side of the thesis, Python programming language has been used for the majority of the tasks. GNU Make has been used for Makefile targets using simple bash commands or Python scripts to perform more complex tasks and ensure reproducibility together with the conda package manager [58]. Most of the data analysis and exploratory work has been done in jupyterlab environment, which was used to interact with the jupyter notebooks [59] together with papermill [60] to parametrize the notebooks.

`Pandas` [61] library has been used for data handling and together with `numpy` [62] for computations on the data, such as in Clonotyping. Circular

fingerprints (Section 2.2.1.1) have been constructed via `RDkit` [63], `Biopython` library [57] has been used for sequence alignments and scoring (Section 2.3), `scikit-learn` [64] for the random split, k-mers (Section 2.2.1.2) generation and baseline models. The RoBERTa models have been trained using `fairseq` library [65] and further used as a `PyTorch` [66] model for making the predictions. For the plots and various visualizations `matplotlib` [67], `seaborn` [68], `logomaker` [69], `tensorboard` [70] and `MulticoreTSNE` [71] have been used.

Because of the dataset sizes, all the data has been stored, analysed and processed on a high-performance computing (HPC) cluster. Most of the tasks have been first split into smaller parts, submitted and in the end merged. This provided the first level of parallelization (and saving of time), while also `multiprocessing` module has been used as another level of parallelization within the Python source code. All of the deep learning models have been trained on GPUs on this cluster.

Chapter 3

Results

The goal of the thesis was to implement a pipeline capable of binary classification of antibodies to either Hep B–binding or non-binding class. The data from 2 vaccination studies, Galson 2015 [50] and Galson 2016 [51] was first preprocessed by performing clonotyping and taking a single representative sequence of each clonotype (Section 2.1.2). The sequences from Galson 2015 have been split into training and validation set via the subject split method and negative sequences of the validation set have been undersampled to achieve a balanced validation set (Section 2.1.3).

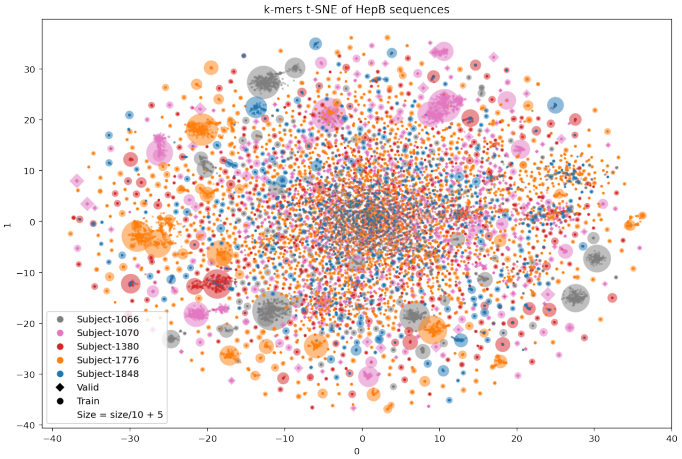
Models have been evaluated on representative sequences of single-subject clusters from 5 subjects (validation data) in Section 3.3 as well as its balanced variation (balanced validation data) in Section 3.2 and on Galson 2016 study in Section 3.4. The baselines models and deep learning models mentioned in this chapter have been trained on $49,502$ positive and $49,502$ negative single and multi-subject representative clonotype sequences from 4 subjects (Methods Section 2.1.3).

RoBERTa pre-trained on a large corpus of human full variable heavy chain sequences which has been further fine-tuned is used as a model for the classification of antibodies to be either Hep B–binding or non-binding. This final model has been selected based on the performance on the balanced validation dataset (Results Chapter 3), and then used to make predictions on representative sequences of clonotypes from study Galson 2016. The model has been trained using `fairseq` library [65] and further used as a `PyTorch` [66] model. All the methods are described in details in Chapter 2.

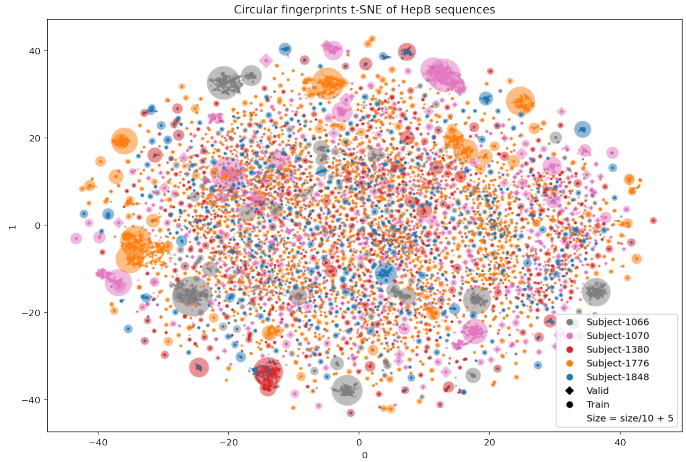
3.1 Data exploration

One of the outcomes of the thesis was also an analysis of the Galson 2015 and Galson 2016 vaccination studies. Various visualizations of the sequences are available throughout the Chapter 2: Methods.

For different representations, t-SNE has been used to generate visualizations of positive Galson 2015 sequences as well as balanced subset including both negative and positive sequences from Galson 2015 (aggregation of balanced training and validation sequences) using k-mers (Figure 3.1a and Figure 3.3a), Circular fingerprints (Figure 3.1b and Figure 3.3b) and final layer embeddings from fully fine-tuned RoBERTa$_{\text{SMALL}}$ models using both CDR3 (Figure 3.2a and Figure 3.4a) and full variable heavy chain sequences (Figure 3.2b and Figure 3.4b).



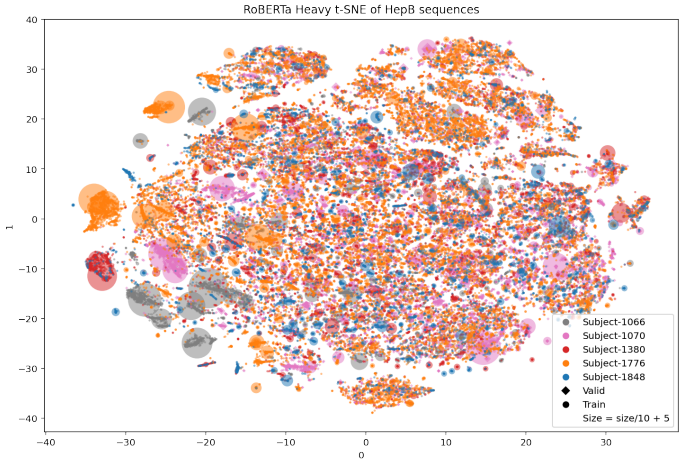
(a) k-mers



(b) Circular fingerprints

Figure 3.1: t-SNE of Hep B–specific (positive) Galson 2015 sequences for (a) k-mers and (b) Circular fingerprints representation

(a) CDR3 sequences

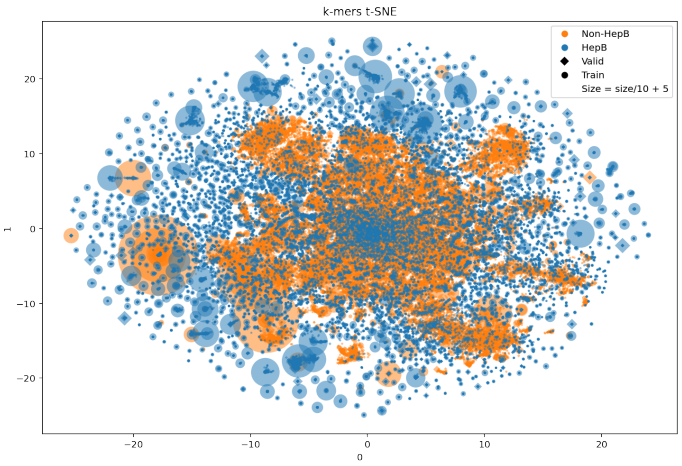


(b) Full variable heavy chain sequences

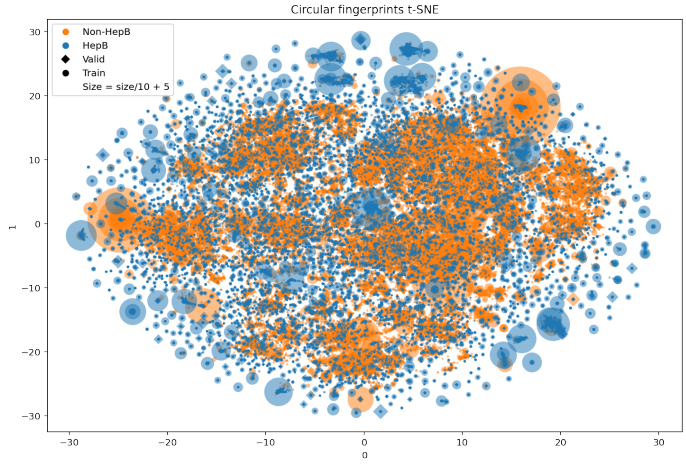
Figure 3.2: t-SNE of Hep B–specific (positive) Galson 2015 sequences for final layer embeddings from RoBERTa using (a) CDR3 and (b) full variable heavy chain sequences

3.2 Balanced validation

The balanced validation dataset (11, 946 positive, 11, 946 negative sequences) is the result of undersampling negatives in all validation sequences while using cluster sizes as weights for the random sampling. This includes results of Extremely randomized trees baselines using k-mers and Circular fingerprints and pre-trained RoBERTa models with and without frozen encoder using CDR3 regions and whole heavy sequences (49, 502 positive, 49, 502 negative sequences). Both weighted and non-weighted metrics are listed in Table 3.1 and discussed in Discussion. The best performing model is $\text{RoBERTa}_{\text{SMALL}}$ with fine-tuned encoder, using full variable heavy chain sequences with 0.476



(a) k-mers



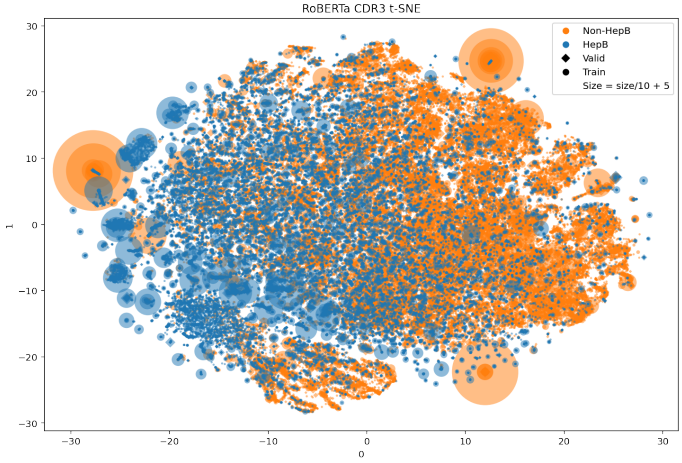
(b) Circular fingerprints

Figure 3.3: t-SNE of Galson 2015 sequences for (a) k-mers and (b) Circular fingerprints representation

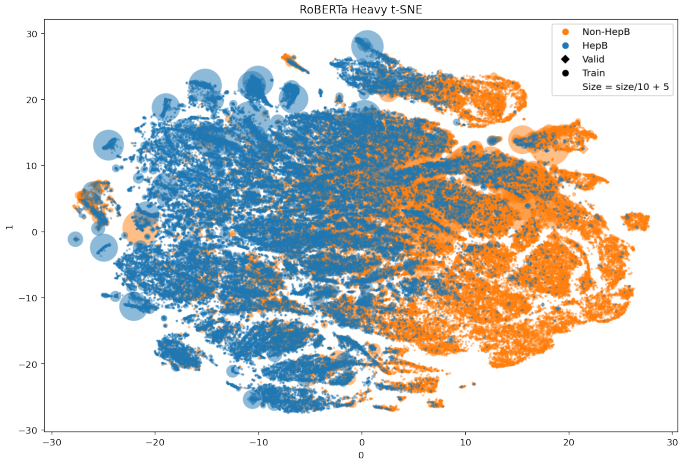
and 0.765 weighted and non-weighted F1 score and 0.740 and 0.845 weighted and non-weighted ROC AUC.

3.3 Full validation

The full validation dataset (11,946 positive, 1,761,834 negative sequences) is a superset of previously used balanced validation dataset (3.2). The full validation dataset corresponds to a real-world scenario of the non-binding antibodies being present in several orders of magnitude higher number than binding antibodies. This includes results of Extremely randomized trees baselines using k-mers and Circular fingerprints and pre-trained RoBERTa mod-



(a) CDR3 sequences



(b) Full variable heavy chain sequences

Figure 3.4: t-SNE of Galson 2015 sequences for final layer embeddings from RoBERTa using (a) CDR3 and (b) full variable heavy chain sequences

els with and without frozen encoder using CDR3 regions and whole heavy sequences (49, 502 positive, 49, 502 negative sequences). Both weighted and non-weighted metrics are listed in Table 3.2 and discussed in Discussion. Extremely randomized trees (ET) model using Circular fingerprints yielded the best F1 score results: 0.155 weighted and 0.0.083 non-weighted. The fully fine-tuned $\text{RoBERTa}_{\text{SMALL}}$ using full variable heavy chain sequences achieved the best ROC AUC: 0.829 weighted and 0.875 non-weighted.

Table 3.1: Galson 2015 balanced validation results including weighted and non-weighted F1 score, area under the precision-recall curve (AP) and area under ROC curve for baseline and deep learning models trained on the training clusters

Model	Weighted valid			Non-weighted valid		
	F1	AP	ROC AUC	F1	AP	ROC AUC
ET (Circular fingerprints)	0.387	0.470	0.667	0.542	0.798	0.777
ET (k-mers)	0.369	**0.473**	0.681	0.594	0.810	0.792
RoBERTa$_{\text{SMALL}}$ (Heavy)	**0.476**	0.457	**0.740**	**0.765**	**0.825**	**0.845**
RoBERTa$_{\text{SMALL}}$ (CDR3)	0.396	0.383	0.647	0.700	0.766	0.764
RoBERTa$_{\text{SMALL}}$ + frozen enc. (Heavy)	0.412	0.377	0.693	0.759	0.768	0.805
RoBERTa$_{\text{SMALL}}$ + frozen enc. (CDR3)	0.417	0.361	0.668	0.697	0.752	0.752

Table 3.2: Galson 2015 full validation results including weighted and non-weighted F1 score, area under the precision-recall curve (AP) and area under ROC curve for baseline and deep learning models trained on the training clusters

Model	Weighted valid			Non-weighted valid		
	F1	AP	ROC AUC	F1	AP	ROC AUC
ET (Circular fingerprints)	**0.155**	**0.214**	0.715	**0.083**	0.185	0.790
ET (k-mers)	0.132	0.213	0.741	0.075	**0.187**	0.812
RoBERTa$_{\text{SMALL}}$ (Heavy)	0.116	0.112	**0.829**	0.049	0.054	**0.875**
RoBERTa$_{\text{SMALL}}$ (CDR3)	0.083	0.074	0.723	0.034	0.039	0.788
RoBERTa$_{\text{SMALL}}$ + frozen enc. (Heavy)	0.088	0.080	0.800	0.037	0.032	0.847
RoBERTa$_{\text{SMALL}}$ + frozen enc. (CDR3)	0.080	0.063	0.727	0.030	0.035	0.773

3.4 Test results

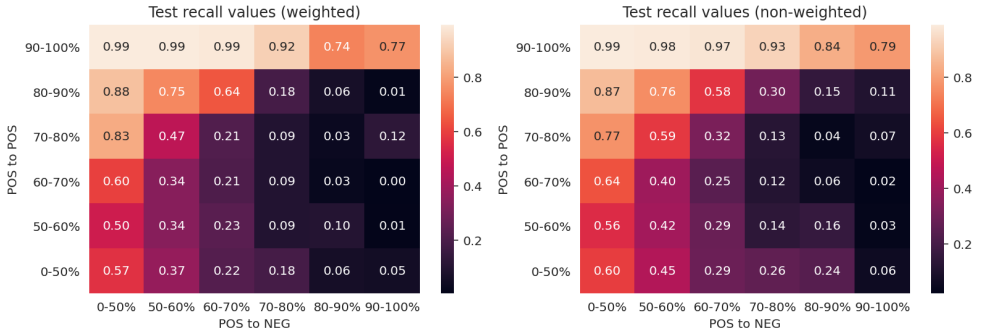
This section lists models' performance on 1,682,432 sequences from different study (Galson 2016 [51]). This includes results of k-mers and Circular fingerprints Extremely randomized trees baselines and pre-trained RoBERTa models with and without frozen encoder using CDR3 regions and whole heavy sequences. All of the models have been re-trained on the balanced training set together with balanced validation set from Galson 2015 study [50] (122,896 sequences). Both weighted and non-weighted metrics are listed in the Table 3.3. The best performing model is Extremely randomized trees (ET) model using Circular fingerprints with 0.474 and 0.261 weighted and non-weighted F1 score and 0.645 and 0.597 weighted and non-weighted ROC AUC.

Table 3.3: Galson 2016 test results including Weighted and non-weighted F1 score, area under the precision-recall curve (AP) and area under ROC curve for baseline and deep learning models trained on both, the training and validation clusters

Model	Weighted test			Non-weighted test		
	F1	AP	ROC AUC	F1	AP	ROC AUC
ET (Circular fingerprints)	**0.474**	**0.575**	**0.645**	**0.261**	**0.292**	**0.597**
ET (k-mers)	0.469	0.570	0.640	**0.261**	0.284	0.594
RoBERTa$_{\text{SMALL}}$ (Heavy)	0.333	0.315	0.509	0.162	0.121	0.454
RoBERTa$_{\text{SMALL}}$ (CDR3)	0.393	0.320	0.499	0.195	0.124	0.453
RoBERTa$_{\text{SMALL}}$ + frozen enc. (Heavy)	0.305	0.315	0.507	0.161	0.125	0.466
RoBERTa$_{\text{SMALL}}$ + frozen enc. (CDR3)	0.386	0.318	0.504	0.191	0.125	0.461

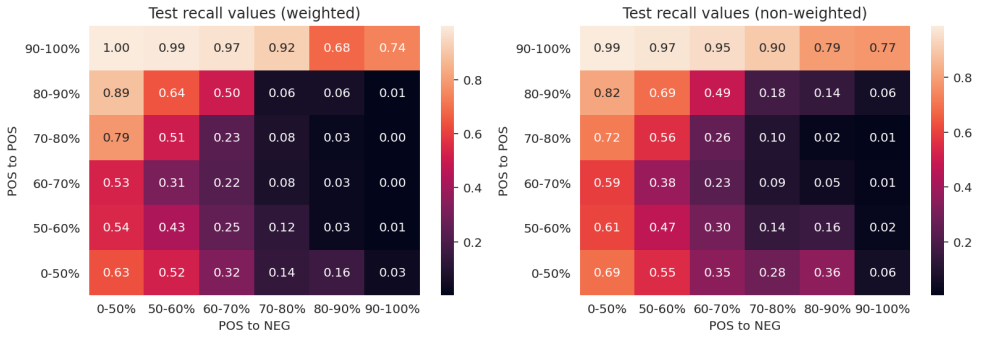
Since it proved to be hard for the models to make predictions on datasets with a large volume of negative sequences, models' evaluations on positive test sequences are also available separately as heat maps divided into bins by their *identity* to negative and positive training sequences. This way, it can be inspected how the model's ability to detect the binding antibodies is affected by their similarity to the positive and negative training sequences. Intuitively, it is expected to observe better performance for sequences highly similar to the positive training sequences and very dissimilar from the negative training sequences and vice versa. A detailed description of this evaluation method is available in Methods Section 2.3. Performance on the data bins split by identity are listed in this section (Figure 3.5, Figure 3.6 and Figure 3.7). The

positive classification thresholds have been set such that the target FPR of all the models is as close as possible to the 0.3 while $FPR \leq 0.3$. The achieved results are further discussed in Chapter 4: Discussion.



(a) k-mers ET (weighted)

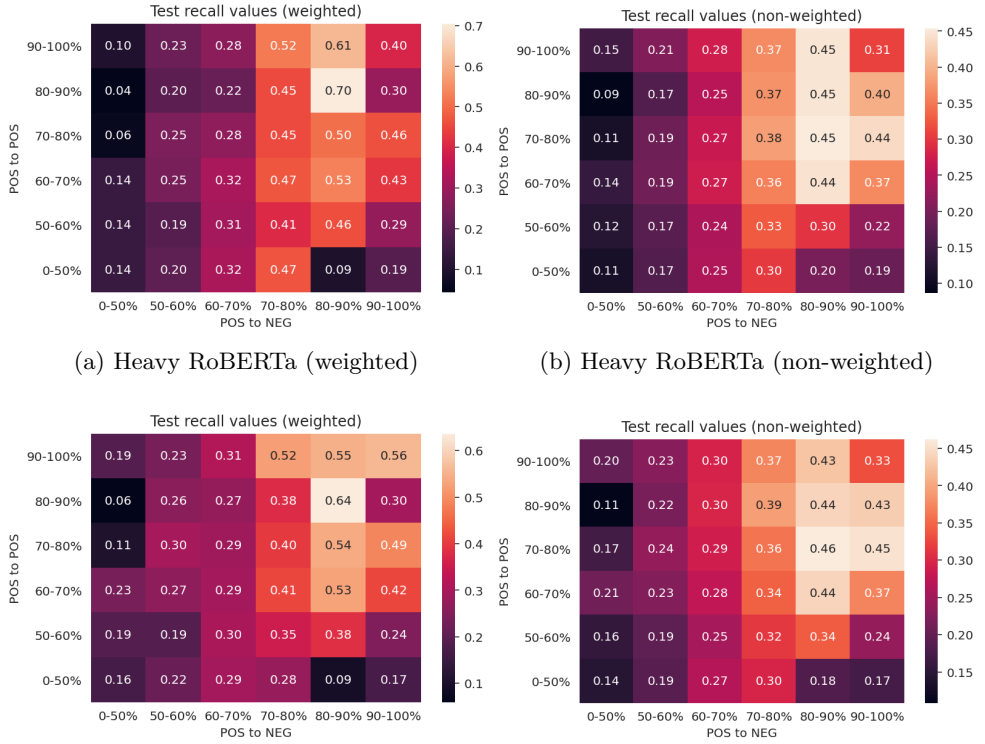
(b) k-mers ET (non-weighted)



(c) Circular fingerprints ET (weighted)

(d) Circular fingerprints ET (non-weighted)

Figure 3.5: Weighted and non-weighted recall values of baseline models on the positive test sequences split into bins by *identity* percentage to training positive (vertical axis) and negative sequences (horizontal axis), while $FPR \leq 0.3$ (Methods Section 2.3). The models perform best for the sequences most similar to the positive sequences and worse for sequences being more similar to the negative sequences



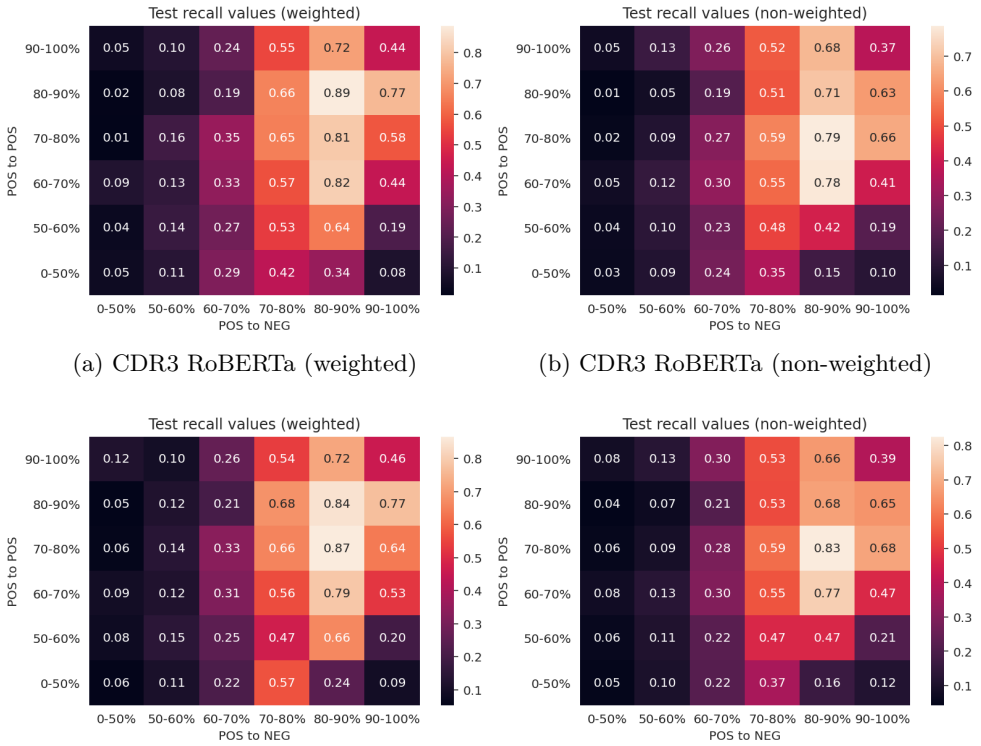
(a) Heavy RoBERTa (weighted)

(b) Heavy RoBERTa (non-weighted)

(c) Heavy frozen encoder RoBERTa (weighted)

(d) Heavy frozen encoder RoBERTa (non-weighted)

Figure 3.6: Weighted and non-weighted evaluation of RoBERTa models, using full variable heavy chain sequences, on the positive test sequences split into bins by *identity* percentage to positive and negative training sequences, while $FPR \leq 0.3$ (Methods Section 2.3). The models seem to achieve higher recall values with rising identity to both positive (vertical axis) and negative training sequences (horizontal axis), with the exception of 90-100 % positive-to-negative bins. Intriguingly, this does not hold for low positive-to-negative identity and rising positive-to-positive identity



(a) CDR3 RoBERTa (weighted)

(b) CDR3 RoBERTa (non-weighted)

(c) CDR3 frozen encoder RoBERTa (weighted)

(d) CDR3 frozen encoder RoBERTa (non-weighted)

Figure 3.7: Weighted and non-weighted evaluation of RoBERTa models, using CDR3 sequences, on the positive test sequences split into bins by *identity* percentage to positive (vertical axis) and negative training sequences (horizontal axis), while $FPR \leq 0.3$ (Methods Section 2.3). The models seem to achieve higher recall values with rising identity to both positive and negative training sequences, with exception of 90-100 % identity bin (upper right corner). Intriguingly, this does not hold for low positive-to-negative identity and rising positive-to-positive identity

Chapter 4

Discussion

Based on the performance on the balanced validation results of binary classification of sequences to either Hep B–binding or non-binding (Results Section 3.2), pre-trained, fully fine-tuned RoBERTa using full variable heavy chain sequences as input would be selected as the best model. Performance on the full validation set of Hep B–specific and non-specific antibodies from Galson 2015 (Section 3.3) showed that the size of the negative set (11,946 negative sequences in the balanced set and 1,761,834 negative sequences in the full set) can affect the performance of the models. For example, weighted and non-weighted F1 score of the best performing model on the balanced validation dataset, RoBERTa using full variable heavy chain sequences, decreased from 0.476 to 0.116 and from 0.765 to 0.049, respectively. The models have also been evaluated on Galson 2016 study (Section 3.4) again containing mostly negative sequences (229,352 positive sequences, 1,453,080 negative sequences) and performance similar to the performance on the full validation set.

Generally, it can be seen that in the case of CDR3 models, overfitting or plateauing has been reached quicker than for the models using full-length heavy sequences. Whereas between the model using the same sequences, the models with encoder fine-tuning seem to train or overfit much quicker than the models with frozen encoder (Figure 2.9). Furthermore, the models with encoder fine-tuning seem to provide better performance than the ones without encoder fine-tuning.

Modest performance on the imbalanced data with a prevalent number of negative sequences can be justified by high diversity in the data. This is especially problematic for the models using whole heavy sequences. On the other hand, models using only CDR3 regions have to deal with lower diversity because of being shorter in comparison to the full variable heavy chain sequences. Furthermore, for some CDR3 sequences, there are both Hep B and non–Hep B antibodies sharing this region and differing only in other parts of the antibody and thus possibly causing problems for a classifier.

Another reason could be the fact that the ELISA test, used to separate the Hep B–specific antibodies (being the ground truth in this thesis), does not provide 100 % sensitivity, leading to some Hep B–specific sequences being present in the negative set.

Moreover, the antibodies might be liable to so-called "activity-cliff", the term common in medicinal chemistry, which means that a compound's activity might change significantly given only a small change in its structure. For more details at this topic, see [72].

ET baseline models using k-mers and Circular fingerprints are the models that exhibit very rational behaviour in regards to the performance on the test data. The higher the positive to positive identity is the higher are the recall values and the lower the positive to negative identity is the lower are the recall values (Figure 3.5).

However, in the case of RoBERTa models, the reasoning behind the results becomes less clear. Models using whole heavy sequences generally performs the best on bins with higher positive-to-positive identity and positive-to-negative identity. However, the performance does not rise with rising positive-to-positive identity with low positive-to-negative identity. In the case of heavy sequences, this could be caused by identity being computed only on CDR3 regions, and therefore the sequence could differ very much in the other parts of the full variable heavy chain sequence. Nevertheless, this trend holds even for the models using only CDR3 regions. This could be partially explained by a cardinality of the sets. In general, there is much fewer sequences in the high identity bins than in the ones with lower identity. Also, the identity has been computed as the best match to the single sequence in the training data, and due to the size of the training dataset ($122,896$ sequences consisting of previous training and validation data), the single match might not be sufficient for a model.

Although the RoBERTa deep learning approaches slightly overperformed baselines in the case of balanced validation dataset (Table 3.1), for the data with prevailing negative sequences, such as in full validation data (Table 3.2) and test data (Table 3.3), Extremely randomized trees (ET) baseline models using Circular fingerprints and k-mers outperform the deep learning models.

The performance decrease between balanced validation and full validation dataset is notable for both the deep learning and the baseline models. One of the reasons could be the diversity of the immune repertoire as such. Antibodies, in general, are very diverse and antibodies for a certain antigen make up only a tiny subset of the whole repertoire where the negative sequences make up the rest. Therefore even though the small positive subset (~200 thousand sequences) can be predicted very well, there might be a lot of "outlier" sequences present in the larger negative dataset (~1.4 million sequences) which are more similar to the positive than negative sequences, and the classifier misclassifies them and thus the False positive rate raises.

Even though the modest results are most likely caused by the negative

sequences, another plausible reason for the performance on Galson 2016 could be the difference in experimental conditions. For example, Galson 2015 [50] patients were previously vaccinated ("booster vaccine") and Galson 2016 [51] patients were not ("vaccine-naive patients").

Chapter 5

Case study

5.1 Introduction

This chapter serves as a stand-alone part which sums up the methods proposed in this work, and showcases use of a deep learning model on a separate Hepatitis B vaccination study (Galson 2016 from [51]). The results are presented and evaluated in a fashion suitable for this task, and potential biological outcomes are discussed.

5.2 Methods

In high-level overview, the data from 2 vaccination studies, Galson 2015 [50] and Galson 2016 [51] is first preprocessed by performing clonotyping and taking a single representative sequence of each clonotype (Section 2.1.2). The sequences from Galson 2015 have been split into training and validation split via the subject split method and negative sequences of the validation set have been undersampled to achieve a balanced validation set (Section 2.1.3).

RoBERTa pre-trained on a large corpus of human full variable heavy chain sequences which has been further fine-tuned is used as a model for the classification of antibodies to be either Hep B–binding or non-binding as annotated by an ELISA assay. This final model has been selected based on the performance on the balanced validation dataset (Results Chapter 3), and then used to make predictions on representative sequences of clonotypes from study Galson 2016. The model has been trained using `fairseq` library [65] and further used as a `PyTorch` [66] model.

Two different ways of evaluation have been introduced in Chapter 2: Methods, namely weighted and non-weighted. In this chapter, only results weighted by clonotype sizes are reported, since giving higher importance to the antibodies from larger clonotypes should result in the better real-word performance. All the methods are described in details in Chapter 2: Methods.

5.3 Results

The RoBERTa$_{\text{SMALL}}$ using full variable heavy chain sequences is a model that would be selected among other classifiers (Table 3.1) based on its performance on the balanced validation dataset with the 0.476 weighted and 0.765 non-weighted F1 score and 0.740 weighted and 0.845 non-weighted ROC AUC (Figure 5.1, Figure 5.2). But on the full validation, it has been outperformed by Extremely randomized trees (ET) models using Circular fingerprints and k-mers (Methods Section 2.2.1.1 and Section 2.2.1.2) as its input.

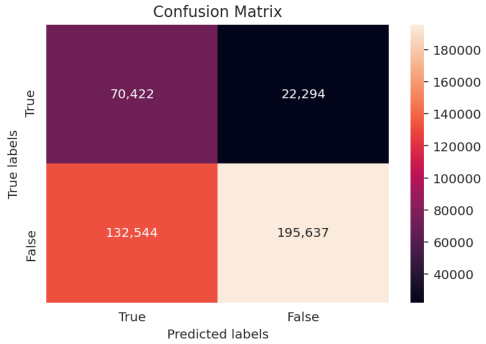


Figure 5.1: Weighted confusion matrix of predictions of RoBERTa$_{\text{SMALL}}$ using full variable heavy chain sequences on the balanced validation data

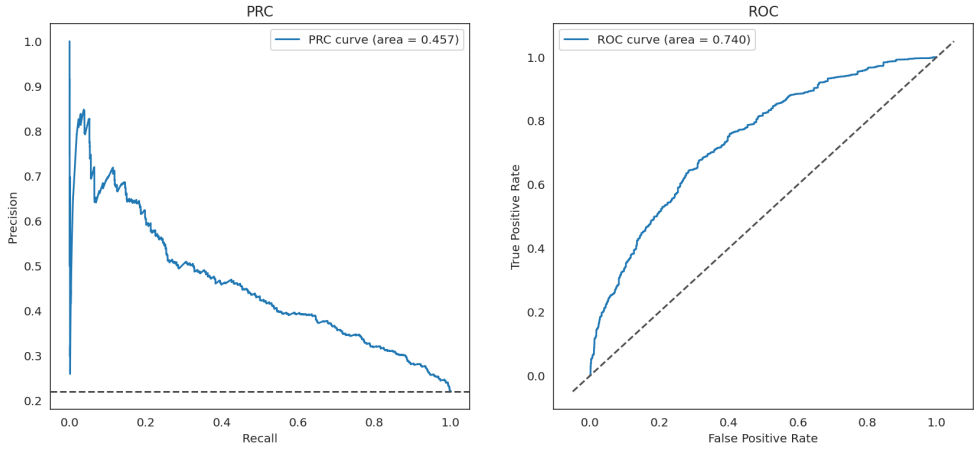


Figure 5.2: Weighted PRC and ROC curves RoBERTa$_{\text{SMALL}}$ using full variable heavy chain sequences on the balanced validation data

The model applied on a study different from the one used for training, Galson 2016 [51], yields modest results (Figure 5.3, Figure 5.4) mainly due to high imbalance in the data (229, 352 positive sequences, 1, 453, 080 negative sequences). The model achieved 0.333 and 0.162 weighted and non-weighted F1 score and 0.509 and 0.454 weighted and non-weighted ROC AUC.

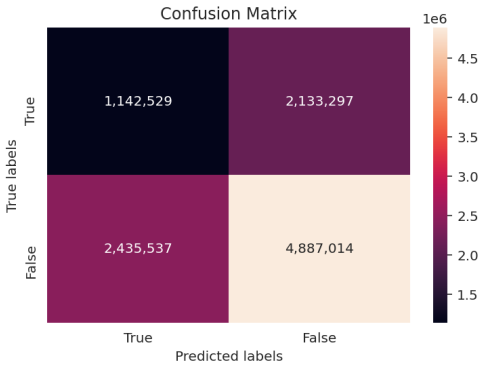


Figure 5.3: Weighted confusion matrix of predictions of RoBERTa$_{\text{SMALL}}$ using full variable heavy chain sequences on the test data

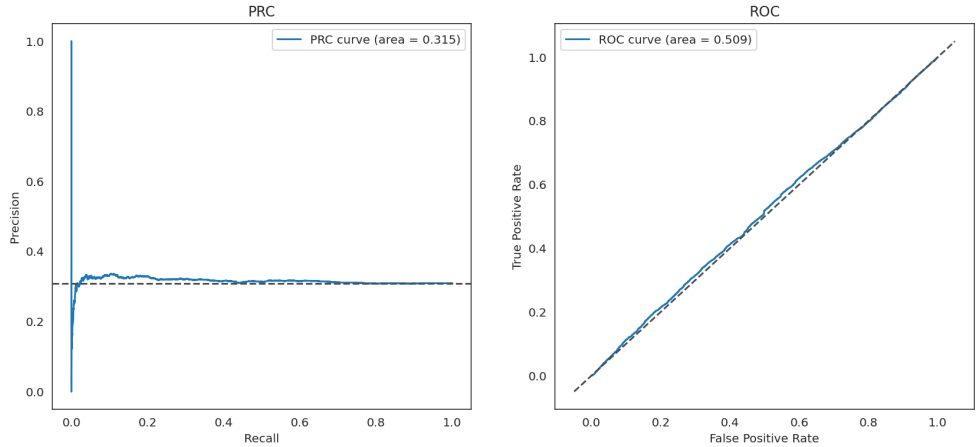


Figure 5.4: Weighted PRC and ROC curves RoBERTa$_{\text{SMALL}}$ using full variable heavy chain sequences on the test data

The final model has also been used to generate feature representations of the positive representative sequences from Galson 2016 and t-SNE has been used to reduce its dimensionality and plot its visualization showing that even the positive representative sequences of clonotypes seem to further cluster into groups (Figure 5.5).

5.4 Discussion

The results show that the deep learning approach applied to classify specificity of antibodies from the Galson 2016 study exhibits modest performance. The ROC AUC values are roughly 0.5 which would be the AUC of a random classifier.

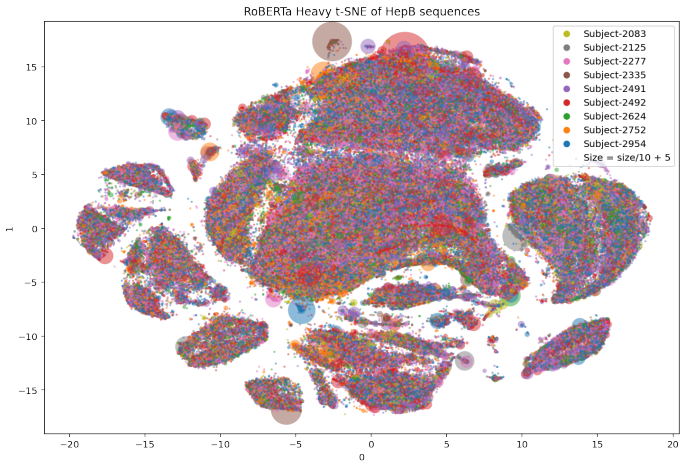


Figure 5.5: t-SNE visualization of the feature representation of positive representative sequences from the Galson 2016 study, generated by $\text{RoBERTa}_{\text{SMALL}}$ model using full variable heavy chain sequences

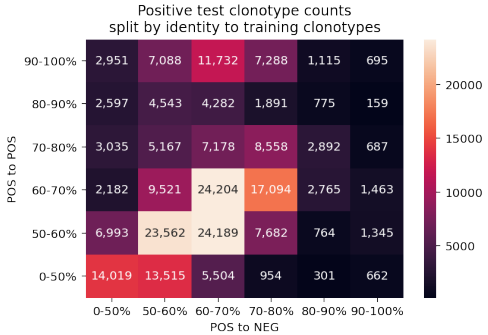


Figure 5.6: Positive clonotype counts in Galson 2016 split by positive-to-positive and positive-to-negative identity to training clonotypes from Galson 2015

It might be partially explained by a small overlap between different studies. It can be seen that the representative sequences of the positive clonotypes from Galson 2016 have mostly same similarity to both positive and negative clonotype representative sequences from training set from Galson 2015 (diagonal line in Figure 5.6).

Even though the performance is modest at the time, new techniques from various subfields of AI (especially NLP) can be added, and the whole process can be reapplied, possibly providing improved results. This would allow models to be applied as a supplementary *in-silico* technique to the current *in-vitro* techniques such as the ELISA tests where it can be used to affirm or deny its results since even the ELISA tests are not entirely accurate.

Conclusion

The goal of the thesis was to analyse the Galson 2015 vaccination study and build a pipeline capable of classifying Hepatitis B specificity of antibody sequences and apply the final model on the sequences from the Galson 2016 vaccination study.

In this work, the data has been analysed and numerous preprocessing techniques, based on the biological properties of antibodies, have been applied. Several sequence representations have been tested with baseline Extremely randomized trees (ET) models, as well as a state-of-the-art NLP model called RoBERTa previously pre-trained on a large corpus of human antibodies.

Also, more sophisticated evaluation metrics based on the similarity between test and training sequences have been used to investigate models' results further. The results have been discussed, and possible causes of the classification errors have proposed in Chapter 4: Discussion.

In general, the deep learning approach seems to yield modest result but has a large potential in further improvements and provides a cheaper alternative or supplementary technique to the *in-vitro* techniques such as the ELISA tests.

Further research

The modest performance of the models does not necessarily mean that the antibody antigen binding cannot be predicted. Considering the "free lunch theorem" [73], it might just suggest a mismatch between the task and the model.

Other reason could be that the 3D structure of antibodies is required to make good predictions about the binding properties of antibodies. However, nowadays, there is a very few resources providing also the structural information data, and its modelling is a challenging problem with ongoing research (for example [74]).

Additionally, the large amounts of negative sequences proved to be problematic. Other data preprocessing techniques, besides the already applied clonotyping, might be helpful. One such example is alternative or additional data clustering via *paratyping* [8], a method grouping antibodies with the same paratope together, where the paratope is usually inferred by using other predictive models. Yet another option which could possibly reduce FPR and thus improve the performance on the negative data is a technique coming from computer vision, mainly used for the object detection task, called *hard negative mining* [75]. In this setup, the positive sequences could be seen as the objects that should be detected and the negative sequences as the background data which prevails.

In conclusion, to make reliable antibody-antigen binding predictions the further research is necessary. Antibody-antigen binding, as well as general protein binding interactions as such, remain an ongoing effort.

Bibliography

1. RAO, V. Srinivasa; SRINIVAS, K.; SUJINI, G. N.; KUMAR, G. N. Sunand. Protein-Protein Interaction Detection: Methods and Analysis. *International Journal of Proteomics*. 2014, vol. 2014, pp. 1–12. Available from DOI: 10.1155/2014/147648.

2. BAKAIL, May; OCHSENBEIN, Francoise. Targeting protein–protein interactions, a wide open field for drug design. *Comptes Rendus Chimie*. 2016, vol. 19, no. 1-2, pp. 19–27. Available from DOI: 10.1016/j.crci.2015.12.004.

3. MASON, Derek M; FRIEDENSOHN, Simon; WEBER, Cédric R; JORDI, Christian; WAGNER, Bastian; MENG, Simon; GAINZA, Pablo; CORREIA, Bruno E; REDDY, Sai T. Deep learning enables therapeutic antibody optimization in mammalian cells by deciphering high-dimensional protein sequence space. *bioRxiv*. [preprint]. 2019. Available from DOI: 10.1101/617860.

4. LIU, Ge et al. Antibody complementarity determining region design using high-capacity machine learning. *Bioinformatics*. 2019, vol. 36, no. 7, pp. 2126–2133. ISSN 1367-4803. Available from DOI: 10.1093/bioinformatics/btz895.

5. GAINZA, P.; SVERRISSON, F.; MONTI, F.; RODOLÀ, E.; BOSCAINI, D.; BRONSTEIN, M. M.; CORREIA, B. E. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. *Nature Methods*. 2019, vol. 17, pp. 184–192. Available from DOI: 10.1038/s41592-019-0666-6.

6. GRAVES, Jordan; BYERLY, Jacob; PRIEGO, Eduardo; MAKKAPATI, Naren; PARISH, S. Vince; MEDELLIN, Brenda; BERRONDO, Monica. A Review of Deep Learning Methods for Antibodies. *Antibodies*. 2020, vol. 9, no. 2, pp. 12. ISSN 2073-4468. Available from DOI: 10.3390/antib9020012.

7. RAO, Roshan; BHATTACHARYA, Nicholas; THOMAS, Neil; DUAN, Yan; CHEN, Xi; CANNY, John; ABBEEL, Pieter; SONG, Yun S. Evaluating Protein Transfer Learning with TAPE. *bioRxiv*. [preprint]. 2019. Available from DOI: 10.1101/676825.

8. RICHARDSON, Eve; GALSON, Jacob D.; KELLAM, Paul; KELLY, Dominic F.; SMITH, Sarah E.; PALSER, Anne; WATSON, Simon; DEANE, Charlotte M. A computational method for immune repertoire mining that identifies novel binders from different clonotypes, demonstrated by identifying anti-Pertussis toxoid antibodies. *bioRxiv*. [preprint]. 2020. Available from DOI: 10.1101/2020.06.02.121129.

9. INSTITUTE FOR QUALITY AND EFFICIENCY IN HEALTH CARE (IQWIG). How does the immune system work? In: *National Center for Biotechnology Information* [online]. 2020 [visited on 2020-07-06]. Available from: https://www.ncbi.nlm.nih.gov/books/NBK279364.

10. ALBERTS, Bruce; JOHNSON, Alexander; LEWIS, Julian; RAFF, Martin; ROBERTS, Keith; WALTER, Peter. The Adaptive Immune System. In: *Molecular biology of the cell*. 4th ed. New York, NY: Garland Science, 2002. ISBN 0815332181.

11. XU, John L; DAVIS, Mark M. Diversity in the CDR3 Region of VH Is Sufficient for Most Antibody Specificities. *Immunity*. 2000, vol. 13, no. 1, pp. 37–45. Available from DOI: 10.1016/s1074-7613(00)00006-6.

12. GRAHAM, Bettie J. *Antibody* [online] [visited on 2020-07-25]. Available from: https://www.genome.gov/genetics-glossary/Antibody.

13. MIRSKY, Alexander; KAZANDJIAN, Linda; ANISIMOVA, Maria. Antibody-Specific Model of Amino Acid Substitution for Immunological Inferences from Alignments of Antibody Sequences. *Molecular Biology and Evolution*. 2014, vol. 32, no. 3, pp. 806–819. ISSN 0737-4038. Available from DOI: 10.1093/molbev/msu340.

14. MURPHY, Kenneth; WEAVER, Casey. *Janeway's Immunobiology*. 9th ed. New York, NY, USA: Garland Science/Taylor & Francis Group, LLC, 2017. ISBN 9780815345510.

15. ROSENBERG, Michael S. Sequence Alignment. In: *Sequence alignment: methods, models, concepts, and strategies*. Berkeley, CA, USA: University of California Press, 2009. ISBN 9780520256972.

16. LI, Heng. *On the definition of sequence identity* [online]. 2018 [visited on 2020-07-18]. Available from: https://lh3.github.io/2018/11/25/on-the-definition-of-sequence-identity.

17. GREIFF, Victor; MIHO, Enkelejda; MENZEL, Ulrike; REDDY, Sai T. Bioinformatic and Statistical Analysis of Adaptive Immune Repertoires. *Trends in Immunology*. 2015, vol. 36, no. 11, pp. 738–749. Available from DOI: 10.1016/j.it.2015.09.006.

18. ROTH, Ron M. Introduction. In: *Introduction to coding theory*. New York, NY, USA: Cambridge University Press, 2007. ISBN 9780521845045.

19. KULSKI, Jerzy K. Next-generation sequencing—an overview of the history, tools, and "Omic" applications. *Next Generation Sequencing - Advances, Applications and Challenges*. 2016, pp. 3–60. Available from DOI: 10.5772/61964.

20. MARKS, Claire; DEANE, Charlotte M. How repertoire data is changing antibody science. *Journal of Biological Chemistry*. 2020, vol. 295, pp. 9823–9837. Available from DOI: 10.1074/jbc.rev120.010181.

21. BERKOWITZ, Frank E.; JERRIS, Robert C. Microbiology laboratory methods. In: *Practical medical microbiology for clinicians*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2016. ISBN 9781119066743.

22. ABNOVA CORPORATION. *ELISA* [online] [visited on 2020-07-09]. Available from: http://www.abnova.com/images/content/support/ELISA.gif.

23. SHALEV-SHWARTZ, Shai; BEN-DAVID, Shai. Clustering. In: *Understanding machine learning: from theory to algorithms*. New York, NY, USA: Cambridge University Press, 2014. ISBN 9781107057135.

24. MAATEN, Laurens van der; HINTON, Geoffrey. Viualizing data using t-SNE. *Journal of Machine Learning Research*. 2008, vol. 9, pp. 2579–2605.

25. BROWN, Tom B. et al. Language Models are Few-Shot Learners. *arXiv e-prints*. [online]. 2020, pp. arXiv:2005.14165. Available from arXiv: 2005.14165.

26. ALAMMAR, Jay. *The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)* [online]. 2018 [visited on 2020-07-10]. Available from: https://jalammar.github.io/illustrated-bert/.

27. *Overfitting and underfitting* [online] [visited on 2020-07-10]. Available from: https://www.educative.io/edpresso/overfitting-and-underfitting.

28. LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep learning. *Nature*. 2015, vol. 521, no. 7553, pp. 436–444. Available from DOI: 10.1038/nature14539.

29. GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. ISBN 9780262035613. Available also from: http://www.deeplearningbook.org.

30. BURKOV, Andriy. Chapter 6. In: *The hundred-page machine learning book*. Andriy Burkov, 2019. ISBN 9781999579500.

31. HORNIK, Kurt; STINCHCOMBE, Maxwell; WHITE, Halbert. Multilayer feedforward networks are universal approximators. *Neural Networks*. 1989, vol. 2, no. 5, pp. 359–366. Available from DOI: 10.1016/0893-6080(89)90020-8.

32. MIKOLOV, Tomas; CHEN, Kai; CORRADO, Greg; DEAN, Jeffrey. Efficient Estimation of Word Representations in Vector Space. *arXiv e-prints*. [online]. 2013, pp. arXiv:1301.3781. Available from arXiv: 1301.3781.

33. RUMELHART, David E.; HINTON, Geoffrey E.; WILLIAMS, Ronald J. Learning representations by back-propagating errors. *Nature*. 1986, vol. 323, no. 6088, pp. 533–536. Available from DOI: 10.1038/323533a0.

34. WERBOS, Paul J. Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*. 1988, vol. 1, no. 4, pp. 339–356. Available from DOI: 10.1016/0893-6080(88)90007-x.

35. ELMAN, Jeffrey L. Finding Structure in Time. *Cognitive Science*. 1990, vol. 14, no. 2, pp. 179–211. Available from DOI: 10.1207/s15516709cog1402_1.

36. PASCANU, Razvan; MIKOLOV, Tomas; BENGIO, Yoshua. On the difficulty of training Recurrent Neural Networks. *arXiv e-prints*. [online]. 2012, pp. arXiv:1211.5063. Available from arXiv: 1211.5063.

37. HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long Short-Term Memory. *Neural Computation*. 1997, vol. 9, no. 8, pp. 1735–1780. Available from DOI: 10.1162/neco.1997.9.8.1735.

38. BAHDANAU, Dzmitry; CHO, Kyunghyun; BENGIO, Yoshua. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv e-prints*. [online]. 2014, pp. arXiv:1409.0473. Available from arXiv: 1409.0473.

39. OLAH, Chris; CARTER, Shan. Attention and Augmented Recurrent Neural Networks [online]. *Distill*. 2016. Available from DOI: 10.23915/distill.00001.

40. TAN, Chuanqi; SUN, Fuchun; KONG, Tao; ZHANG, Wenchang; YANG, Chao; LIU, Chunfang. A Survey on Deep Transfer Learning. *arXiv e-prints*. [online]. 2018, pp. arXiv:1808.01974. Available from arXiv: 1808.01974.

41. DEVLIN, Jacob; CHANG, Ming-Wei; LEE, Kenton; TOUTANOVA, Kristina. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv e-prints*. [online]. 2018, pp. arXiv:1810.04805. Available from arXiv: 1810.04805.

42. WANG, Alex et al. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 353–355. Available from DOI: 10.18653/v1/W18-5446.

43. WILLIAMS, Adina; NANGIA, Nikita; BOWMAN, Samuel. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 1112–1122. Available from DOI: 10.18653/v1/N18-1101.

44. TAYLOR, Wilson L. "Cloze Procedure": A New Tool for Measuring Readability. *Journalism Quarterly*. 1953, vol. 30, no. 4, pp. 415–433. Available from DOI: 10.1177/107769905303000401.

45. LIU, Yinhan et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv e-prints*. [online]. 2019, pp. arXiv:1907.11692. Available from arXiv: 1907.11692.

46. LAI, Guokun; XIE, Qizhe; LIU, Hanxiao; YANG, Yiming; HOVY, Eduard. RACE: Large-scale ReAding Comprehension Dataset From Examinations. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, 2017, pp. 785–794. Available from DOI: 10.18653/v1/D17-1082.

47. KOVALTSUK, Aleksandr; LEEM, Jinwoo; KELM, Sebastian; SNOWDEN, James; DEANE, Charlotte M.; KRAWCZYK, Konrad. Observed Antibody Space: A Resource for Data Mining Next-Generation Sequencing of Antibody Repertoires. *The Journal of Immunology*. 2018, vol. 201, no. 8, pp. 2502–2509. ISSN 0022-1767. Available from DOI: 10.4049/jimmunol.1800708.

48. LEFRANC, Marie-Paule; POMMIÉ, Christelle; RUIZ, Manuel; GIUDICELLI, Véronique; FOULQUIER, Elodie; TRUONG, Lisa; THOUVENIN-CONTET, Valérie; LEFRANC, Gérard. IMGT unique numbering for immunoglobulin and T cell receptor variable domains and Ig superfamily V-like domains. *Developmental & Comparative Immunology*. 2003, vol. 27, no. 1, pp. 55–77. Available from DOI: 10.1016/s0145-305x(02)00039-3.

49. *OAS Documentation* [online] [visited on 2020-07-10]. Available from: http://opig.stats.ox.ac.uk/webapps/oas/documentation.

50. GALSON, Jacob D. et al. Analysis of B Cell Repertoire Dynamics Following Hepatitis B Vaccination in Humans, and Enrichment of Vaccine-specific Antibody Sequences. *EBioMedicine*. 2015, vol. 2, no. 12, pp. 2070–2079. Available from DOI: 10.1016/j.ebiom.2015.11.034.

51. GALSON, Jacob D.; TRÜCK, Johannes; CLUTTERBUCK, Elizabeth A.; FOWLER, Anna; CERUNDOLO, Vincenzo; POLLARD, Andrew J.; LUNTER, Gerton; KELLY, Dominic F. B-cell repertoire dynamics after sequential hepatitis B vaccination and evidence for cross-reactive B-cell activation. *Genome Medicine*. 2016, vol. 8, no. 1. Available from DOI: 10.1186/s13073-016-0322-z.

52. GALSON, Jacob D et al. BCR repertoire sequencing: different patterns of B-cell activation after two Meningococcal vaccines. *Immunology & Cell Biology*. 2015, vol. 93, no. 10, pp. 885–895. Available from DOI: 10.1038/icb.2015.57.

53. LANDRUMY, Gregory. *Fingerprints in the RDKit* [online]. 2012 [visited on 2020-07-10]. Available from: https://www.rdkit.org/UGM/2012/Landrum_RDKit_UGM.Fingerprints.Final.pptx.pdf.

54. ROGERS, David; HAHN, Mathew. Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*. 2010, vol. 50, no. 5, pp. 742–754. Available from DOI: 10.1021/ci100050t. PMID: 20426451.

55. JURAFSKY, Dan; MARTIN, James H. N-grams. In: *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. 2nd ed. Prentice Hall, 2008. ISBN 9780131873216.

56. GEURTS, Pierre; ERNST, Damien; WEHENKEL, Louis. Extremely randomized trees. *Machine Learning*. 2006, vol. 63, no. 1, pp. 3–42. Available from DOI: 10.1007/s10994-006-6226-1.

57. COCK, P. J. A. et al. Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*. 2009, vol. 25, no. 11, pp. 1422–1423. Available from DOI: 10.1093/bioinformatics/btp163.

58. *Conda* [online]. Anaconda Inc., 2020. Vers. 4.8.3 [visited on 2020-07-28]. Available from: https://docs.conda.io.

59. KLUYVER, Thomas et al. Jupyter Notebooks – a publishing format for reproducible computational workflows. In: LOIZIDES, F.; SCHMIDT, B. (eds.). *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. 2016, pp. 87–90.

60. *Papermill* [online]. nteract, 2020. Vers. 2.1.2 [visited on 2020-07-28]. Available from: https://papermill.readthedocs.io.

61. REBACK, Jeff et al. *pandas-dev/pandas: Pandas 1.1.0rc0* [online]. Zenodo, 2020. V1.1.0rc0 [visited on 2020-07-26]. Available from DOI: 10.5281/zenodo.3950442.

62. VAN DER WALT, S.; COLBERT, S. C.; VAROQUAUX, G. The NumPy Array: A Structure for Efficient Numerical Computation. *Computing in Science Engineering*. 2011, vol. 13, no. 2, pp. 22–30. Available from DOI: 10.1109/MCSE.2011.37.

63. LANDRUM, Greg et al. *rdkit/rdkit: 2020_03_4 (Q1 2020) Release*. Zenodo, 2020. Release_2020_03_4. Available from DOI: 10.5281/zenodo.3929204.

64. PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011, vol. 12, pp. 2825–2830.

65. OTT, Myle; EDUNOV, Sergey; BAEVSKI, Alexei; FAN, Angela; GROSS, Sam; NG, Nathan; GRANGIER, David; AULI, Michael. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. *arXiv e-prints*. [online]. 2019, pp. arXiv:1904.01038. Available from arXiv: 1904.01038.

66. PASZKE, Adam et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: WALLACH, H.; LAROCHELLE, H.; BEYGELZIMER, A.; D'ALCHÉ-BUC, F.; FOX, E.; GARNETT, R. (eds.). *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8026–8037. Available also from: http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

67. HUNTER, J. D. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*. 2007, vol. 9, no. 3, pp. 90–95. Available from DOI: 10.1109/MCSE.2007.55.

68. WASKOM, Michael et al. *mwaskom/seaborn: v0.10.1 (April 2020)* [online]. Zenodo, 2020. Version v0.10.1 [visited on 2020-07-26]. Available from DOI: 10.5281/zenodo.3767070.

69. TAREEN, Ammar; KINNEY, Justin B. Logomaker: beautiful sequence logos in Python. *Bioinformatics*. 2019, vol. 36, no. 7, pp. 2272–2274. ISSN 1367-4803. Available from DOI: 10.1093/bioinformatics/btz921.

70. ABADI, Martín et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv e-prints*. [online]. 2016, pp. arXiv:1603.04467. Available from arXiv: 1603.04467.

71. ULYANOV, Dmitry. *Multicore-TSNE* [online]. GitHub, 2016 [visited on 2020-07-26]. Available from: https://github.com/DmitryUlyanov/Multicore-TSNE.

72. STUMPFE, Dagmar; HU, Ye; DIMOVA, Dilyana; BAJORATH, Jürgen. Recent Progress in Understanding Activity Cliffs and Their Utility in Medicinal Chemistry. *Journal of Medicinal Chemistry*. 2014, vol. 57, no. 1, pp. 18–28. Available from DOI: 10.1021/jm401120g.

73. WOLPERT, D. H.; MACREADY, W. G. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*. 1997, vol. 1, no. 1, pp. 67–82. Available from DOI: 10.1109/4235.585893.

74. SENIOR, Andrew W. et al. Improved protein structure prediction using potentials from deep learning. *Nature*. 2020, vol. 577, no. 7792, pp. 706–710. Available from DOI: 10.1038/s41586-019-1923-7.

75. DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. 2005, vol. 1, pp. 886–893. Available from DOI: 10.1109/CVPR.2005.177.

Appendix A

Acronyms

AI Artificial intelligence.

AUC area under the curve.

BCR B-cell receptor.

BERT Bidirectional Encoder Representations from Transformers.

CART Classification And Regression Trees.

CDR Complementarity-determining region.

CDR3 Complementarity-determining region 3.

CNN convolutional neural network.

DNA Deoxyribonucleic acid.

ELISA enzyme-linked immunosorbent assay.

ET Extremely randomized trees.

FN False Negatives.

FP False Positives.

FPR False positive rate.

GPU graphics processing unit.

Hep B Hepatitis B.

HPC high-performance computing.

IMGT The international ImMunoGeneTics information system.

LSTM long short-term memory.

MLM Masked Language Model.

MLP multilayer perceptron.

NGS Next-generation sequencing.

NLL Negative log-likelihood.

NLP Natural Language Processing.

NSP Next Sentence Prediction.

OAS Observed Antibody Space.

PPI protein–protein interaction.

PRC Precision-Recall curve.

ReLU Rectified linear unit.

RNN recurrent neural network.

RoBERTa Robustly optimized BERT approach.

ROC Receiver operating characteristic.

t-SNE t-Distributed Stochastic Neighbor Embedding.

TN True Negatives.

TP True Positives.

Appendix B

Contents of enclosed SD card

- `README.md` ..... the file with SD card's content description
- `setup.py` ..... the Python package installation file
- `environment.yml` ..... the conda environment file
- `Makefile` ..... GNU Makefile for submitting the jobs on the HPC cluster
- `notebooks` ..... the directory with jupyter notebooks
- `models` ..... the directory with trained weights of models
- `bin` ..... the directory with Python source codes
  - `fairseq_plugins` ..... the directory with custom fairseq extensions
- `text` ..... the thesis text directory
  - `source` ..... the thesis source files
  - `thesis.pdf` ..... the thesis text in PDF format