universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Discovery of novel TLR8 agonists

verfasst von | submitted by

Bertram Fitz BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Magister pharmaciae (Mag. pharm.)

Wien | Vienna, 2024

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 605
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Pharmazie
Betreut von | Supervisor:	Dipl.-Ing. Dr. Gerhard Wolber

Acknowledgements

I would like to thank Prof. Dr. Gerhard Wolber for welcoming me into his research group and providing me with a professional and inspiring environment in which to complete my Master's thesis. The opportunity afforded me the chance to gain a profound understanding of a field of pharmacy that I had previously been relatively unfamiliar with, and to obtain knowledge that has had a long-lasting impact on me.

Furthermore, I would like to express my gratitude to pharmacist Valerij Talagayev, who provided invaluable guidance and support throughout my tenure at the FU Berlin. I am indebted to him for his expertise and commitment, without which I would not have been able to write this thesis. He taught me a plethora of skills that I had previously considered beyond my capabilities. Moreover, the straightforward communication with Valerij was instrumental in enabling my stay in Berlin. Thank you very much.

I would also like to extend my gratitude to the other members of the Molecular Design Lab at the FU Berlin. They also spared no effort to help me when I encountered difficulties. Furthermore, I felt very comfortable in this group and I am grateful for the personal attention I received, not just from a professional standpoint. I express my best wishes to all members of this working group for their future endeavours.

Furthermore, I would like to express my gratitude to Prof. Dr. Michael Wirth, the head of the study programme at the Faculty of Pharmacy at the University of Vienna. The organisation of my external Master's project was facilitated considerably by his accessibility, direct communication, and prompt responses.

Finally, I would like to express my gratitude to my family. Your support not only accompanied me through my studies, but also made my experience in Berlin possible in the first place. It is a time I will never forget.

Abstract

The Toll-like receptor family is one of the main representatives of the so-called pattern recognition receptors and plays a decisive role in human immunity to various pathogens. Toll-like receptor 8 has been the subject of recent research interest due to its special role at the interface between innate and adaptive immunity. This interest has been driven by the recognition that TLR8 is not as inactive in humans as previously assumed. Agonists of Toll-like receptor 8 are promising drug candidates that could be used in various diseases, including various cancers, viral infections, allergies and asthma. Research is also constantly uncovering new, astonishing details about the complex function of the receptor in the human body. Despite its enormous potential, only few agonists for Toll-like receptor 8 are currently available. The objective of this master's thesis was to identify potential drug candidates using computer-aided methods. As part of the thesis, a pharmacophore-based virtual screening was conducted, followed by docking experiments into a crystal structure of the receptor. Additionally, machine learning models were developed to predict the activity of molecules in a database on Toll-like receptor 8.

Zusammenfassung

Die Familie der Toll-like Rezeptoren ist einer der Hauptvertreter der sogenannten „Pattern Recognition Receptors“ und spielt eine entscheidende Rolle bei der Immunität des Menschen gegen verschiedene Krankheitserreger. Lange Zeit wurde die Forschung am Toll-like Rezeptor 8 vernachlässigt, da angenommen wurde, er wäre im Menschen inaktiv. Seine spezielle Rolle an der Schnittstelle zwischen angeborener und adaptiver Immunität wurde in den letzten Jahren jedoch erkannt, was zu einem verstärkten Forschungsinteresse geführt hat. Agonisten am Toll-like Rezeptor 8 stellen vielversprechende Arzneistoffkandidaten dar, die in verschiedenen Krankheitsbildern zum Einsatz kommen könnten, darunter verschiedene Krebserkrankungen, virale Infektionen, Allergien und Asthma. Die Forschung bringt zudem immer neue, erstaunliche Details über die komplexe Funktion des Rezeptors im menschlichen Körper ans Licht. Trotz des enormen Potenzials sind bislang nur wenige Agonisten für den Toll-like Rezeptor 8 verfügbar. Ziel der vorliegenden Masterarbeit war es daher, mithilfe verschiedener computergestützter Methoden Moleküle zu entdecken, die die Grundlage zur Entwicklung neuer Arzneistoffe bieten könnten. Im Rahmen der Masterarbeit wurde ein pharmakophorbasiertes virtuelles Screening mit anschließendem Dockingexperiment in eine Kristallstruktur des Rezeptors durchgeführt. Zudem wurden Modelle entwickelt, die mittels maschinellen Lernens die Aktivität von Molekülen in einer Datenbank am Toll-like Rezeptor 8 voraussagen können.

Contents

1 Introduction

The human body has two main defence mechanisms against invading microbiota, the innate and the adaptive immune system. The innate immune system is characterised by a rapid but non-specific response to infection. The adaptive immune system is able to generate highly specific responses to the pathogen, but with a delay of up to six weeks. Together, they provide comprehensive protection against the constantly evolving and mutating bacteria, viruses and cancer cells that cause disease in humans. Despite their mutable nature, research has shown that there are common structural elements in these microbiota that can be recognised by the immune system. Cells of the innate immune system possess pattern recognition receptors that enable them to recognise these structural elements. Their activation triggers an immediate first immune response as well as the initiation of an adaptive immune response.[1]

1.1 Toll-like receptors

The Toll-gene was first discovered in Drosophilia melanogaster in 1985 by Christiane Nüsslein-Volhard of the Max Planck Institute in Tübingen, Germany. It was found to be involved in the embryogenesis of the fly. A decade later, Hoffmann et al. in Strasbourg linked Toll to the insect's immune defence, in particular its susceptibility to fungal infections. It was the first time that the gene had been associated with immune responses.[2] Human homologues of the gene, as well as its genetic product, the so-called "Toll-like" receptors, were discovered in 1997 by Ruslan Medzhitov and Charles Janeway at Yale University, with their ligands unknown. A first link between lipopolysaccharide recognition by the immune system and TLRs was made in 1998 by Yang et al. and Kirschnig et al..[1] Twelve Toll-like receptors have since been discovered in mice, ten of which are also found in humans.[3]

Toll-like receptors, or TLRs for short, are a crucial part of the human body's immunity to pathogens and cancer. They are a family of germline-encoded, structurally highly conserved pattern recognition receptors (PRRs) capable of recognising different types of viruses, bacteria or malignant cells.[4][5] After initially being associated only with innate immune responses, in recent years more attention has been given to the effects of TLRs in adaptive immune responses, highlighting their role as an interface between the two systems.[6] Activation of TLRs has been linked to the release of proinflammatory cytokines, increased phagocytosis, autophagy and cell death.[4] Their ligands are molecules whose structures are associated with infection, danger or damage, known as pathogen-associated molecular patterns (PAMPs) or damage-associated molecular patterns (DAMPs, sometimes called danger-associated molecular patterns or alarmins).[4][7]

PAMPs are highly conserved structural elements of pathogens that are essential for their survival and can be found on their surface or in their fragments.[8] Lipopolysaccharides from gram-negative bacteria, lipoteichoic acid from gram-positive bacteria, lipoarabinomannan from mycobacteria, peptidoglycan, unmethylated DNA and bacterial lipoproteins are just some examples of PAMPs.[1] DAMPs are constitutively expressed endogenous molecules, most of which have an important physiological function and are only released and exposed to TLRs in response to stress or cell damage. They include cytosolic and nuclear components such as high mobility group box 1 (HGMB1) proteins, RNA, DNA, heat shock proteins, histone proteins, purine metabolites and β-amyloid, as well as extracellular matrix proteins such as proteoglycans, heparan sulphate, fibrinogen and fibronectin.[7] Most cells in the body express some subsets of TLRs at low levels. However, high levels of expression can be observed in cells of the innate immune system, particularly in antigen-presenting sentinel cells such as macrophages or dendritic cells. With some variation, they express virtually all subtypes of TLRs.[9]

1.1.1 The TLR subtypes

The ten subtypes of Toll-like receptors found in the human body are numbered from TLR1 to TLR10. Each receptor subtype has a unique type of ligand that it can recognise, depending on its localisation. While TLR1, TLR2, TLR4, TLR5, TLR6 and TLR10 are located on the cell surface where they recognise extracellular components, TLR3, TLR7, TLR8 and TLR9 are endosomal receptors that usually bind to nucleic acids.[10] A typical Toll-like receptor consists of a leucine-rich N-terminal extracellular or intraendosomal domain for pattern recognition, a single-spanning transmembrane element, and a C-terminal intracellular Toll-interleukin-1 receptor (TIR) domain for signal transduction.[9][10] This cytoplasmic domain shares sequence homology with the interleukin-1 receptor and consists of a conserved region of approximately 200 amino acids.[11][12] (Figure 1)

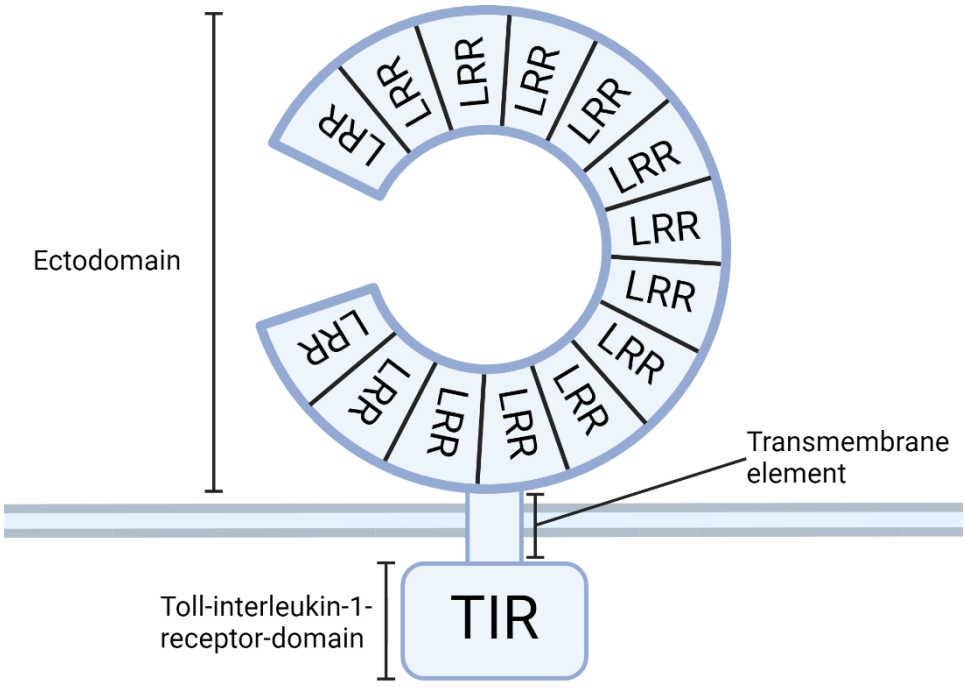


Figure 1: *Schematic structure of Toll-like receptors.*[13]

1.1.1.1 TLR signalling

After binding their ligands, TLRs dimerise, with the exception of TLR8 and TLR9, which are already dimerised even when not activated.[14] Subsequently, they change conformation in order to bind an adapter molecule to their intracellular TIR domain.[9] These adapter proteins usually also contain a TIR domain themselves that can bind to the TIR domain of the receptor. Different subtypes of TLRs use different adapter proteins.[15]

With the exception of TLR3, all TLRs ultimately interact with the adaptor protein Myeloid differentiation primary response 88, better known as MyD88.[9] While TLR2 and TLR4 interact with MyD88 via the adaptor protein MAL/TIRAP, other TLR subtypes can bind MyD88 directly.[16] MyD88 essentially triggers the same signalling cascade as the interleukin-1 receptor.[17] TLR3, however, interacts with TRIF instead of MyD88. It also requires an adapter protein, TICAM-1, to interact with TRIF. TLR4 can interact with both MyD88 and, via the adapter protein TRAM, with TRIF.[16] TLR signalling can therefore generally be divided into two main pathways, MyD88-dependent and MyD88-independent.[15]

The differences in signalling between the receptor subtypes are reflected in different effects. While MyD88-dependent pathways lead to the production of inflammatory cytokines, MyD88-independent pathways are also associated with the release of type I interferon along with inflammatory cytokines. Both lead to the activation of NF-κB and MAP kinases, but the MyD88-independent pathway also activates IRF3.[15] Although the differences may seem small, in many cases these are the details that account for the specific effects. An example is TLR-induced differentiation of naive T-cells, where the subtype of T-cells they differentiate into depends on the particular receptor that has been activated.[16]

Further variation in the effects of TLRs is due to the fact that a single pathogen may contain multiple PAMPs and thus activate multiple subtypes of TLRs. This can happen simultaneously or at different stages of infection. For example, an ssRNA virus could trigger TLR7 and then replicate to produce dsRNA that is recognised by TLR3. A similar situation occurs in an infection with several pathogens at the same time. The effects of activating multiple TLRs have been described to be synergistic in most cases. However, some antagonism has also been reported. This so-called receptor "cross-talk", in which the signalling cascades of several receptors interact and influence each other, results in a sophisticated initial response that is pathogen-specific even before the adaptive response has begun.[18] In conclusion, the exact immune response to TLR activation depends on the receptor subtypes involved and the time at which they are triggered.[16][18]

1.1.1.2 TLRs play a central role in immunity and disease

TLR activation in disease has been described in many publications as a double-edged sword. On the one hand, their regulatory effects and ability to generate immune responses are essential elements of the body's defence strategy against various threats. On the other hand, this important and complex role in the immune system means that dysfunction and imbalance have all the more serious consequences and undesirable outcomes. In particular, their influence on T-cells and dendritic cells has been linked to a variety of diseases, including CNS, pulmonary, gastrointestinal, renal, dermatological, cancer and chronic inflammatory diseases.[19][20]

TLRs appear to play a particularly ambiguous role in cancer. It has been proposed that TLR-induced chronic inflammation contributes to cancer mutagenesis, tumour growth and invasion through the release of immunosuppressive cytokines, growth factors, proangiogenic factors, extracellular matrix-modifying enzymes and reactive oxygen species (ROS).[21] In particular, activation of the NF-κB and MAPK pathways by TLRs has been linked to tumour progression through induction of anti-apoptotic proteins such as Bcl-2 and chronic inflammation via COX2.[22] Indeed, damage-associated molecular patterns (DAMPS), which can activate TLRs, are abundant in the tumour microenvironment due to the large number of dying cells.[9] Ultimately, the effect of TLRs on tumours depends on the tumour phenotype.[10]

1.1.2 Toll-like receptor 8

Scientific interest in Toll-like receptor 8 has grown enormously in recent years. Initially overlooked because of its inactivity in mice, it has been the subject of many studies since its activity in the human immune system was discovered.[3] TLR8 is an endosomal receptor that can recognise single-stranded RNA molecules associated with infection or damaged and dying cells.[23] TLR8 mainly recognises guanosine and uridine (GU)-rich ssRNA, regardless of origin. However, it has also been shown to be activated by other types of ssRNA, suggesting that there may be some patterns that can activate the receptor that have yet to be discovered.[24] It cannot discriminate between pathogen and self ssRNA, rather its selectivity for pathogens results from its location in the endosomal compartment. Since self-RNA is rapidly degraded in the extracellular compartment under physiological conditions, it usually only activates TLR8 when mistakenly taken up by immune cells in autoimmune diseases.[25] Pathogens whose ssRNA fragments activate TLR8 include Coxsackie B, Sendai, Influenza A, HIV type 1, but also many different types of bacteria such as Staphylococcus aureus, group B Streptococcus and Streptococcus pneumoniae.[23][26-28] The genetic information for TLR8 is located on the germline chromosome X, in the more distal region of the Xp22 locus. It encodes a protein of 1041 amino acid residues with a weight of 119.8 kDa.[11] It has been proposed that the location of the TLR8 gene on the X chromosome may contribute to sex-specific susceptibility to certain diseases, such as systemic lupus erythematosus in females or high COVID mortality in males.[29][30] Diseases associated with defects in TLR8 signalling include cancer, susceptibility to infection, autoimmune diseases and allergies.[31-35]

1.1.2.1 TLR8 strucutre

Like the other TLRs, TLR8 consists of three main domains.[9][10] The extracellular domain consists of 26 modules of leucine-rich repeats (LRRs) totalling more than 800 amino acid residues. It forms a ring-like structure and is responsible for ligand binding.[14][36] The transmembrane domain consists of a single helix of 34 mostly uncharged, hydrophobic amino acid residues.[37] The cytoplasmic Toll-interleukin-1 receptor domain consists of a conserved region of about 200 amino acids. It is capable of binding to the adaptor protein MyD88, allowing further signal transduction.[11][12][14][38] Some TLR subtypes require adaptor proteins to bind to MyD88, but TLR8 is thought to be able to bind to MyD88 both directly and through adaptor proteins.[38]

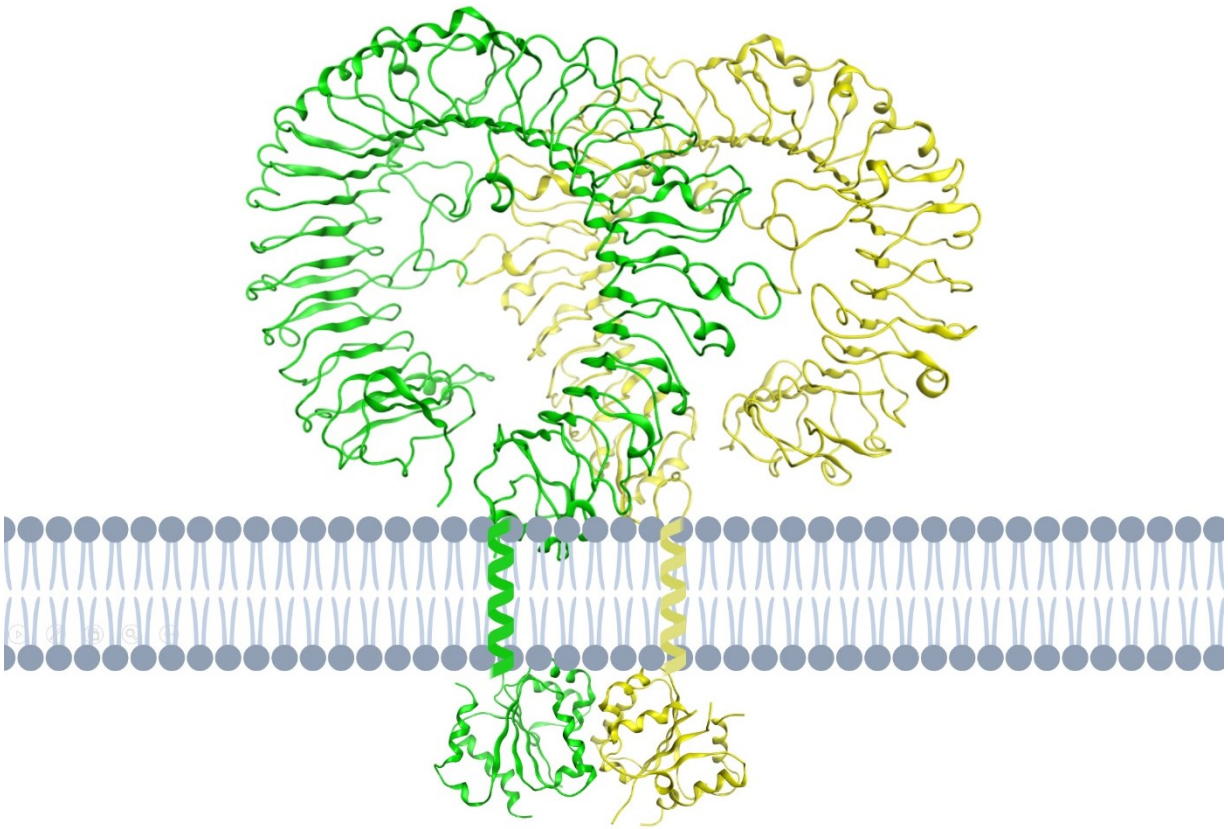


Figure 2: *A schematic overview of the entire TLR8 structure in the endosomal membrane. Adopted from Tanji et al., 2015 (PDB: 4R08) and Khan et al., 2005 (PDB:1T3G)*[13][23][39-41]

While most other Toll-like receptors dimerise once they bind their ligand, Toll-like receptor 8 is present as a dimer even in the absence of its ligand. It dimerises with other TLR8 molecules, making it a homodimer.[42][43] The N-termini of the chains form the ectodomain of the receptor, while the C-termini form the cytoplasmic TIR-domain.[36]

1.1.2.2 TLR8 expression

TLR8 is expressed mainly in myeloid immune cells such as neutrophils, monocytes, macrophages and myeloid dendritic cells, and in regulatory T-cells.[44] Other cell types with confirmed TLR8 expression include hepatocytes, where it is important in fighting hepatitis C infection, and neurons, where it is involved in neuronal apoptosis.[45][46] High levels of expression can also be found in lung and peripheral blood leukocytes. It has also been detected in placenta, spleen, lymph nodes and bone marrow.[11]

TLR8 monomers are expressed with a specific sequence between leucine-rich repeats 14 and 15 in their extracellular domain, which sterically prevents them from dimerising with other TLR8 monomers. This so-called "Z-loop" of about 30 amino acids is found in TLR7, 8 and 9 and must first be cleaved from the receptor by furin-like proprotein convertase and cathepsins in order for the two chains to associate, rendering the receptor ready for activation.[42] Evidence suggests that TLR8, like other endosomal TLRs, is transported from the endoplasmic reticulum to the endosome via the endoplasmic reticulum-resident membrane protein UNC93B1.[3]

Figure 3: *A TLR8 ectodomain monomer with an uncleaved Z-loop highlighted in orange. This loop prevents the dimerisation of two TLR8 monomers and must be cleaved before activation can occur. Adopted and modified from Tanji et al., 2016 (PDB: 5HDH)*[42][47]

1.1.2.3 TLR8 binding sites

A TLR8 monomer has two binding sites in its ectodomain, one for ssRNA fragments and another for uridine. The receptor can only be activated when both binding sites have their ligand bound simultaneously.[23] Binding of only one ligand alone is not sufficient to activate the receptor.[48] Both the uridine and the ssRNA fragment that activate TLR8 usually originate from GU-rich RNA that has been digested in the endosome by enzymes such as RNase T2.[49] As a homodimer, both binding sites are present twice in the entire TLR8 structure, for a total of 4 binding sites per receptor.[23] While the ssRNA binding sites are located around LRR 10-13 within the ring of each homologous ectodomain chain, the uridine binding sites are located between the two chains and consist of LRR 11-14 of one chain and LRR 16*-18* (asterisks indicate the location on the other chain) of the other chain.[23]

When targeting TLR8 with agonistic drugs, it is usually the uridine binding site that is targeted in the activated state of the receptor. Many known TLR8 agonists, such as resiquimod, induce a conformational change in the receptor to its activated state upon binding.[48] The uridine binding site forms a hydrophobic pocket. Interactions with residues Phe346, Tyr348, Gly376, Val378, Ile403, Phe405, Gly572 and Val573 of this pocket have been identified as important for receptor activation.[14][48] A detailed view of the binding pockets of the receptor is shown in Figure 4.

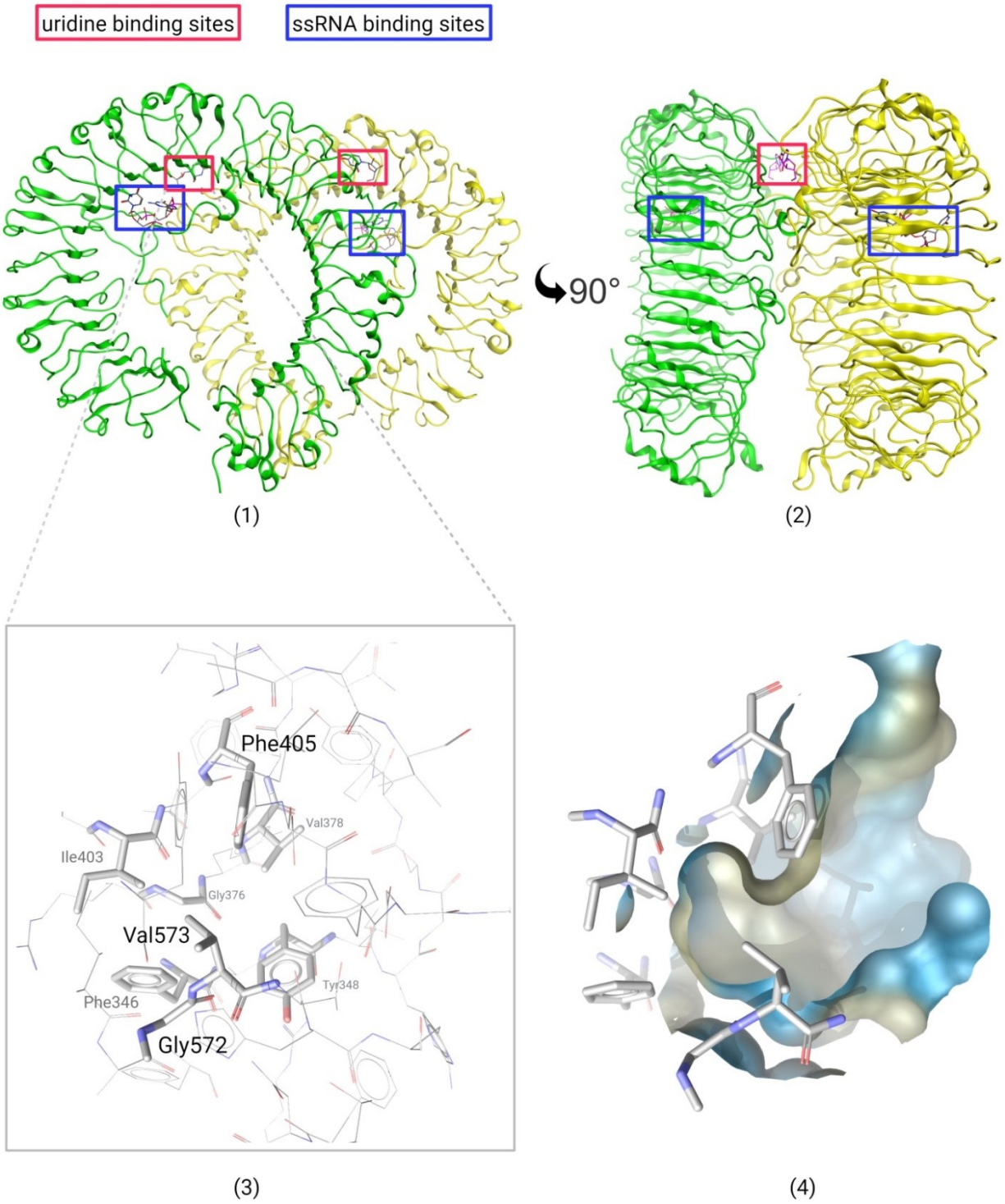


Figure 4: *TLR8 binding sites. (1) While the binding sites for RNA are located within the two ring structures, the binding sites for uridine are between the rings. (2) The same model rotated by 90 degrees. From this angle, it becomes evident that the uridine binding sites are situated between the two chains. In this view, they are positioned one behind the other. (3) Detailed view of a uridine binding site. The amino acid residues that are important for activating the receptor are labelled and rendered as sticks. They form the hydrophobic binding pocket, which is often targeted by drugs. (4) The binding pocket from a different angle and with a visualisation of the hydrophobic and polar surface area. The polar surface areas are shown in blue, the apolar ones in yellow. In this view, the hydrophobic binding pocket is very clearly recognisable. Adapted from Tanji et al., 2013 (PDB: 3W3N)*[14][50]

1.1.2.4 Ligand binding and structural reorganisation

An important aspect of TLR8 activation is the rapid response to pathogens. Colocalisation experiments have shown that viral ssRNA can be found in the endosomes of TLR8-expressing cells about 15 to 20 minutes after infection, where it can potentially trigger TLR8. The exact time depends on the infected cell type.[51] After binding both uridine and an ssRNA fragment, the receptor undergoes a conformational change in which the rings formed by each of the two chains of the receptor rotate slightly in opposite directions. This results in a hinge-like movement that reduces the distance between the two C-terminal cytoplasmic TIR-domains of each chain from about 50 Å to about 30 Å.[14] It is thought that the conformational change in the receptor also induces a conformational change in the TIR-domains, allowing them to dimerise in close proximity and subsequently bind the adapter protein MyD88.[14][42][52] The exact mechanism of the interaction between the TIR-domains and the adapter proteins of TLRs is still under debate.[53]

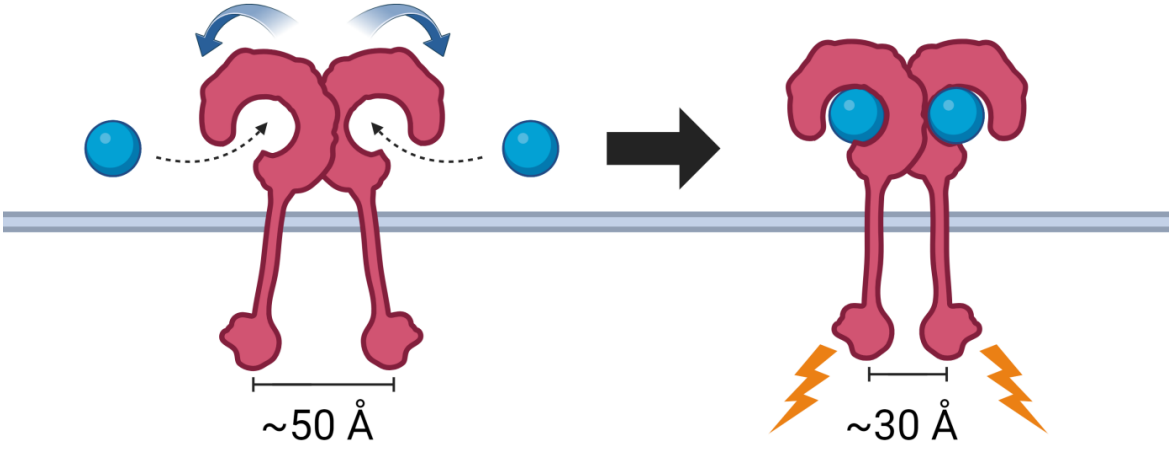


Figure 5: *The hinge-like motion of the TLR8 dimer. The red structures are the receptor subunits, the blue balls represent the ligand. Adapted from Tanji et al., 2013.*[13][14]

1.1.2.5 TLR8 signalling cascade

MyD88 consists of a TIR-domain, similar to the intracellular TIR-domain of TLR8, and a death domain. Its TIR domain binds to the TIR domain of the receptor, allowing its death domain to form a complex with interleukin-1 receptor-associated kinases (IRAK) 1 and 4, forming the so-called "myddosome". (Figure 6: (1)) It essentially acts as an adapter protein for the TAK1-TRAF6 signalling pathway, which is also triggered by inflammatory cytokines such as interleukin 1.[16][54][55] IRAK4 phosphorylates IRAK1, which then recruits and forms a complex with tumour necrosis factor receptor-associated protein-6 (TRAF6). [12][56] (Figure 6: (2)) The IRAK1 and TRAF6 complex dissociates from the cytoplasmic domain of TLR8 and associates with transforming-growth-factor-β-associated-kinase (TAK1) and TAK1-binding-proteins 1 and 2 (TAB1 and TAB2) at the plasma membrane, where TAB2 is predominantly located.[12][57] (Figure 6: (3)) In the process, IRAK1 is degraded.[12] TAB2 acts as a linker between TRAF6 and TAK1. The formation of the complex allows TAK1 to be phosphorylated by TAB1.[57] (Figure 6: (4)) TAB2 is also phosphorylated, but it is not yet known by which enzyme.[57] The phosphorylation causes TAB2 to dissociate from the plasma membrane and the entire remaining complex of TRAF6, TAK1, TAB1 and TAB2 to translocate into the cytosol. Here, ubiquitin conjugating enzyme 13 (UBC13) and ubiquitin conjugating enzyme E2 variant 1 (UEV1A) form a complex with TRAF6, ubiquitylating it.[12][57] (Figure 6: (5)) TRAF6 then activates TAK1 by attaching K63-linked ubiquitin chains to its lysine residues.[12][58] (Figure 6: (6)) Activated TAK1 is a kinase that can activate mitogen-activated protein kinases (MAPK) and the inhibitor of nuclear factor-κB (inhibitor of nuclear factor-κB (short IκB)) kinase complex, triggering both the MAPK and NF-κB pathways.[12] (Figure 6: (7))

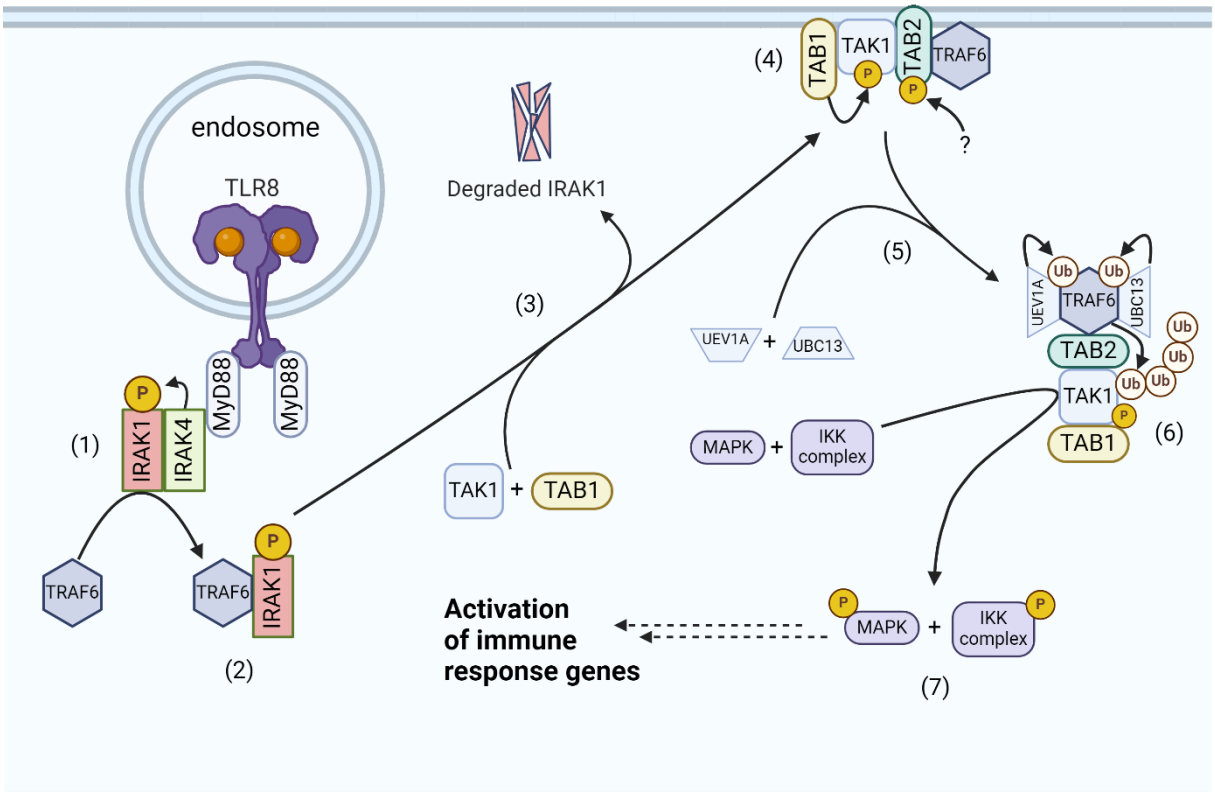


Figure 6: *The signalling pathway of Toll-like receptor 8. The numbers are referenced in the chapter. The kinase that phosphorylates TAB2 is still unknown.*[12][13][16][54-58]

1.1.2.6 Effects of TLR8 activation

TLR8 activation triggers the NF-κB pathway and the release of a plethora of pro-inflammatory cytokines and chemokines, including IFN-α, TNF-α, IL-12, IL-1α/β, IL-6, IL-8, MIP-1αβ, MIP-3αβ and some IFN-γ, although at lower levels than TLR7.[10][59]

An important physiological effect of TLR8 is the differentiation of monocytes into monocyte-derived dendritic cells and the activation of myeloid dendritic cells.[10][59] Dendritic cells are professional antigen-presenting cells and can assist in the activation of helper T-cells, killer T-cells and B-cells. Thus, TLR8 has a profound regulatory effect on the immune system and plays an important role in physiological and pathological processes. Therefore, it has great potential both as a drug target and in pathogenesis.[60]

Another promising effect of TLR8 activation, particularly in cancer research, is the counteraction of the immunosuppressive effect of regulatory T-cells and the induction of apoptosis in myeloid-derived suppressor cells. Both are associated with the failure of anti-tumour immunotherapy.[10]

In addition, the effects of TLR8 have been linked to enhanced antibody-dependent cytotoxicity (ADCC) and stimulation of natural killer cell activity.[10] Antigen-presenting mDCs activate $CD8^+$ cytotoxic T cells, leading to increased cytotoxic activity in tumours.[10][59] TLR8 activation also induces proliferation in naive $CD4^+$ T-cells and Th_1 differentiation.[51]

TLR8 is also involved in receptor cross-talk as described in Chapter 1.1.1.1. One study showed that its activation upregulates TLR2 in human monocytes. Other studies have shown that TLR8 activation inhibits TLR7 and TLR9 activation. In murine DCs, TLR8 deficiency induces TLR7 overexpression, which has been linked to the development of autoimmunity.[3]

1.1.2.7 TLR8 as a drug target

Through its complex regulatory effects on cells of the immune system, TLR8 has been implicated in the pathogenesis of many diseases. Selective TLR8 agonists are being investigated for the treatment of various solid tumours, including low-grade B-cell lymphoma, ovarian epithelial carcinoma, fallopian tube carcinoma, peritoneal cavity cancers, head and neck cancer, breast cancer, colorectal cancer and liver cancer. Ventirx's selective TLR8 agonist VTX-2337 (motolimod) has been shown to favourably synergise with the effects of rituximab, trastuzumab and cetuximab. Dual TLR7/8 agonists are being investigated in solid tumours, CTCL, advanced solid tumours, melanoma and colorectal cancer. They have also been shown to favourably influence the effect of PD1/PDL1 inhibitors. As described at 1.1.1.2, TLRs in general often have ambiguous effects in cancer. Although protumorigenic effects have not yet been reported for TLR8, it cannot be ruled out due to its similar mechanism of action.[10]

Another promising use for TLR8 agonists is as adjuvants in vaccines against viruses.[10][16] Due to the receptor's activating effect on T-helper cells and B-cells, this is not limited to ssRNA viruses.[61] Studies are underway investigating the efficacy of TLR8 agonists against hepatitis B, a DNA virus. GS-9688 (selgantolimod) is a potent and selective oral TLR8 agonist and a promising candidate for the treatment of chronic hepatitis B. It is also being investigated in people with hepatitis B and HIV.[10][62] Interestingly, it has been observed that HBV infection actually decreases TLR8 levels in PBMCs.[51] Moreover, the role of TLR7 and TLR8 polymorphisms in chronic hepatitis C infection has been discussed. Data suggest that decreased IFN-α production in individuals with TLR7 and TLR8 SNPs causes increased susceptibility to chronic hepatitis C.[63]

Antagonists of TLR8 are being investigated for their potential use in chronic inflammatory diseases such as lupus erythematosus or rheumatoid arthritis.[48] Recent research with the specific TLR8 inhibitor TH1027 has shown suppression of TLR8-mediated inflammatory responses observed in human monocyte cell lines, peripheral blood mononuclear cells (PBMCs) and samples from patients with rheumatoid arthritis.[10] IMO-8400 is a first-in-class oligonucleotide TLR7/8/9 antagonist being investigated for the treatment of immune-mediated inflammatory diseases such as psoriasis in individuals with TLR aberrations.[48]

The effects of Toll-like receptors on haematopoietic stem and progenitor cells (HSPCs) are an area of current investigation. Studies have shown that treatment with the TLR7/8 agonist resiquimod (R848) leads to an increase in haematopoietic stem cells (HSCs). However, these cells have a reduced ability to repopulate. The treatment also led to reduced overall HSPC mobilisation. Systemic administration of TLR7/8 agonists appears to have an effect on haematopoiesis, including the expansion of dendritic cells in the bone marrow. This effect holds potential for immunotherapeutic intervention, for example through cancer vaccines or immune checkpoint inhibitors.[64]

Involvement of TLR8, among other TLRs, has been discussed in the pathogenesis of Alzheimer's disease, where it may contribute to phagocytosis of β-amyloid but also lead to neuroinflammation. Other pathogeneses in which TLR8 may play a role include myocarditis and cardiomyopathy through molecular mimicry of TLR8 agonists by cardiac myosin.[65][66] In addition, TLR8 has been implicated in the pathogenesis of asthma, rhinitis and atopic dermatitis.[67] TLR8 agonists are currently under investigation in allergy and asthma therapy.[51]

1.2 Computer-aided drug design

Computer-aided drug design (CADD) is a relatively new area of drug discovery that has grown tremendously in popularity and importance in recent years with the increased accessibility and lower cost of powerful computers, as well as advances in structural biology.[68] It has become an important tool in reducing the costs associated with high throughput biological screening, while increasing the ability to investigate new potentially active compounds and understand their binding modes to a target at the molecular level.[69] CADD is commonly used in early drug discovery, such as hit identification, hit-to-lead optimisation and lead optimisation. It encompasses a variety of different methods and approaches. Methods such as virtual screening, docking and molecular dynamics simulations are based on the study of a digital model of molecules and proteins. These molecular models can be used to explore protein-ligand interactions to establish a relationship between their physicochemical properties and their biological effect. This relationship is known as a quantitative structure-activity relationship. CADD uses the structure-activity relationship (SAR) to predict the affinity and activity of molecules on a given target.[70] Other methods that are more abstract, data-driven and do not primarily involve a molecular model include in-silico ADMET modelling, quantum mechanical calculations, machine learning and artificial intelligence.[71-73]

1.2.1 The two approaches to CADD

In research, information about a particular target or its ligands is often sparse. For example, high-throughput screening may yield hits for a target whose structure is not yet known. Studying the features that these ligands have in common provides information about the molecular features that are important for binding and inducing the desired effect, as well as spatial information about the binding site on the target. This information can be used in the design of new compounds or to find other molecules with similar properties. It is one of two main approaches to drug discovery known as "ligand-based" drug design.[70]

Conversely, there are cases where the target structures are known but no ligands have been discovered. Studying a structure of the target in order to theorise about the properties of a possible ligand for that target is known as "structure-based" drug design. Three-dimensional digital structures of a protein, obtained in X-ray scans, cryo-electron microscopy or NMR, published together with papers and which are freely available on the internet, are commonly used for this purpose.[70][74][75] Incomplete structures can be modelled using a similar structure in the process of homology modelling.[70]

When information on both structure and ligands is available, aspects of ligand-based and structure-based drug design are often combined in hybrid methods or carried out simultaneously or sequentially. Using several different methods or analysing the differences between approaches can lead to further and more precise insights. This is particularly advantageous when the different approaches can be used to mitigate the shortcomings of each other.[76] Ligand-based methods tend to overfit the data on which they were trained, while structure-based methods often rely on scoring functions that are inherently limited in their accuracy and generalisability.[76][77] Combining the methods does, of course, incur computational costs.[76]

1.2.2 Key concepts in CADD

The following chapters outline the key concepts and methods in CADD that are relevant to this work.

1.2.2.1 Pharmacophores

Pharmacophores are a way of digitally representing molecular interaction features, including information about conformation and stereochemistry. They are a powerful tool in computational drug discovery that can be used to find molecules with desired properties in a database, to assess the similarity of a molecule to another molecule or set of molecules, and to design molecules with specific interactions and for specific purposes.[78]

The types of interactions typically found in pharmacophores are those considered to be the most important in protein-ligand interactions, as outlined in Section 1.2.2.5.1.1. These are hydrophobic interactions, hydrogen bond donors and acceptors, positive and negative ionisable interactions and aromatic interactions such as π-stacking.[79][80] Furthermore, 3D-pharmacophores also contain information about the three-dimensional shape of the binding site and the geometrical features of a possible ligand, such as atomic distances, angles and dihedral angles.[69][79] In addition, many pharmacophore concepts include information about the space that is not to be occupied by the ligands because it is already occupied by the target.[79]

The concept has become so important in computational chemistry that in 1998, Camille G. Wermuth et. al. created an official definition for IUPAC, the key points of which are:

> *'A pharmacophore is the ensemble of steric and electronic features that is necessary to ensure the optimal supramolecular interactions with a specific biological target structure and to trigger (or to block) its biological response'.*

> *'A pharmacophore does not represent a real molecule or a real association of functional groups, but a purely abstract concept that accounts for the common molecular interaction capacities of a group of compounds towards their target structure.'*

> *'A pharmacophore can be considered as the largest common denominator shared by a set of active molecules.'*[79]

Depending on whether it is structure-based or ligand-based, a pharmacophore may consist of the molecular interaction features that a molecule must have to bind to a particular target, or the common features of a number of molecules.[74][79] 3D-pharmacophores have been used in this work.

To assess the performance of a pharmacophore in screenings of never-before-seen data, it is usually screened against libraries of known active and inactive structures.[81] Inactive structures can be known inactives, molecules with undesired activity, or decoys.[82] Decoys are molecules that are believed to be inactive on a particular target while also being very similar in structure or molecular properties to compounds that are active, making them difficult to correctly classify as inactive by models used in drug discovery such as pharmacophore-based screening, docking and machine learning.[83] Decoys that are similar in physical properties to known ligands, but are assumed to be inactive due to their different structure, are called "property-matched decoys".[84] They are often used to test such models and assist in the refinement process, where the ratio of true to false positives is analysed.[81][83]

This allows the calculation of various statistical metrics such as sensitivity, specificity, yield of actives, enrichment factor and goodness of hit-list, as well as the widely used receiver-operator curves, or "ROC curves" for short. The goal is to obtain a model that provides as many true positives and as few false positives as possible.[81] Special attention must therefore be paid to the correct abstraction of the included features. Pharmacophores whose interaction features are too general, so that many molecules fulfil them, have low selectivity and therefore poor predictive ability. On the other hand, too many features reduce the generalisability of the model at the expense of the ability to discover new molecules with structures different from those of the known ligands. Therefore, when designing a pharmacophore, the essential features should be identified and only those that are important for the desired effect and have predictive power should be included in the model.[83]

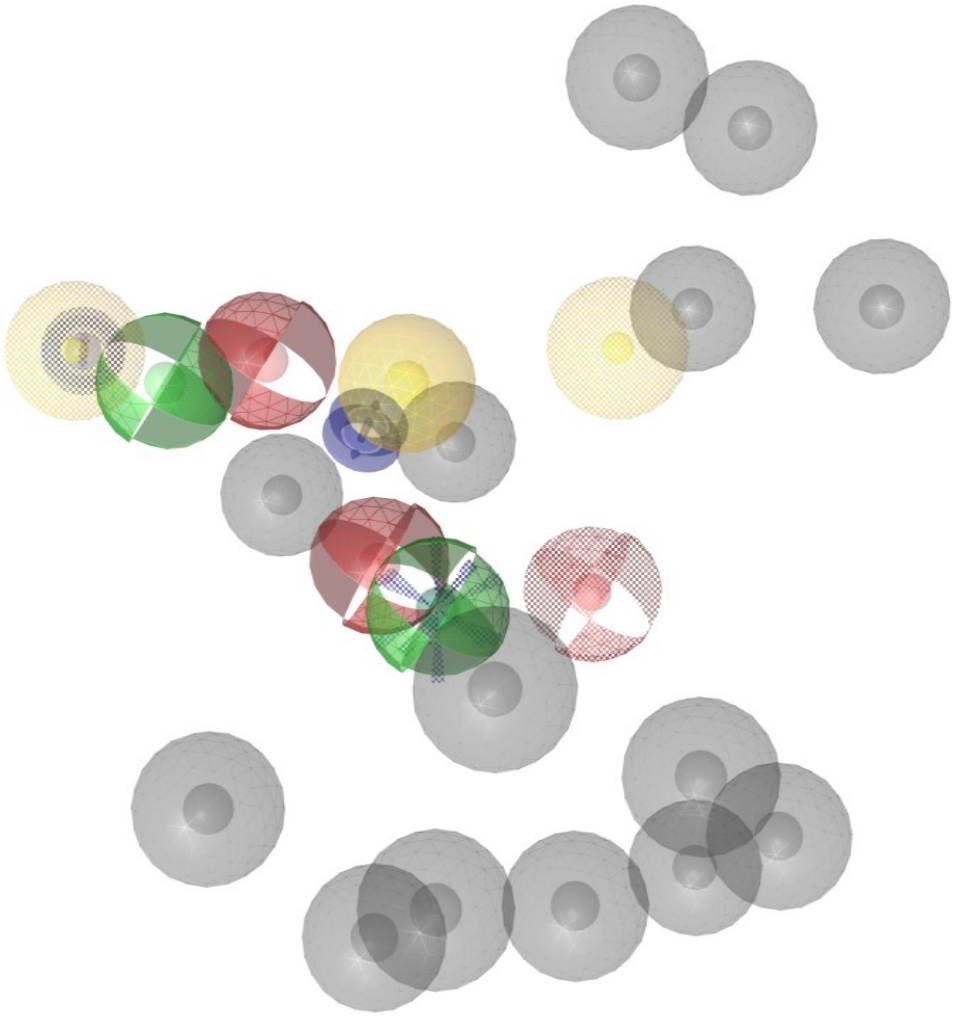


Figure 7: *A 3D-pharmacophore. This is the visual representation of the pharmacophore obtained in Chapter 3.2.2.1, rendered in LigandScout. Please refer to Chapter 2.1.1.1 for a detailed description of the interaction features. In LigandScout, optional features are displayed in a checkerboard pattern.*

1.2.2.1.1 Structure-based pharmacophores

In structure-based pharmacophore generation, the surface of the target is tested for its ability to form the important non-covalent molecular interactions described in Chapter 1.2.2.5.1.1. Centres of gravity for the interaction types are calculated and visualised, indicating favourable positions of complementary interaction features on a potential ligand molecule. These positions are used to calculate a pharmacophore of interaction features that a ligand should have in order to bind to the target of interest.[74]

1.2.2.1.2 Ligand-based pharmacophores

Ligand-based pharmacophore generation involves studying the possible molecular interactions of a large number of compounds known to be active on the desired target. The individual structure of each molecule only allows for a limited number of conformations.[79] These conformations are superimposed and all possible molecular interactions are tested. The vectors for the possible interactions are then clustered, which allows the calculation of the pharmacophoric features.[69] The pharmacophore is the result of solving how the molecules can be superimposed so that they all have similar interactions in similar places.[79] While the pharmacophore itself is trained from most of the available data, a small fraction of it is used as positives for validation.[82] Compared to structure-based pharmacophores, ligand-based pharmacophores tend to be slightly less accurate, as in many cases there is more than one solution to this superposition experiment.[79] On the other hand, this could also be an advantage in the discovery of novel or unusual binding solutions, as a common problem with structure-based pharmacophores is their lack of flexibility.[82]

1.2.2.2 Molecular modelling

Molecular modelling encompasses all methods that involve the calculation, display and manipulation of realistic three-dimensional digital molecular structures and their physicochemical properties.[85] Such models can be obtained from experimental data like NMR or X-ray structures or, in cases where data is incomplete, from homology modelling using a known structure. Various methods and calculations can then be applied to these models to understand the interactions that take place between ligands and their targets.[70] Some of the most important methods are undoubtedly the calculation of molecular geometries and energies as part of the so-called force field methods. They can be seen as an application of the knowledge of atoms and the forces acting on them at the molecular level. Electrostatic repulsion and attraction, ideal bond angles and lengths, hydrophilic and hydrophobic effects or Van-der-Waals interactions, but also quantum chemical models are considered to obtain energetically favourable three-dimensional structures of molecules or complexes of molecules.[85] Ultimately, a theory about the structure-activity relationship is formed, which allows conclusions to be drawn for the design of possible ligands.[70] Common methods in CADD, in which molecular models are an integral part, are virtual screening, molecular docking and molecular dynamics simulations.[86][87]

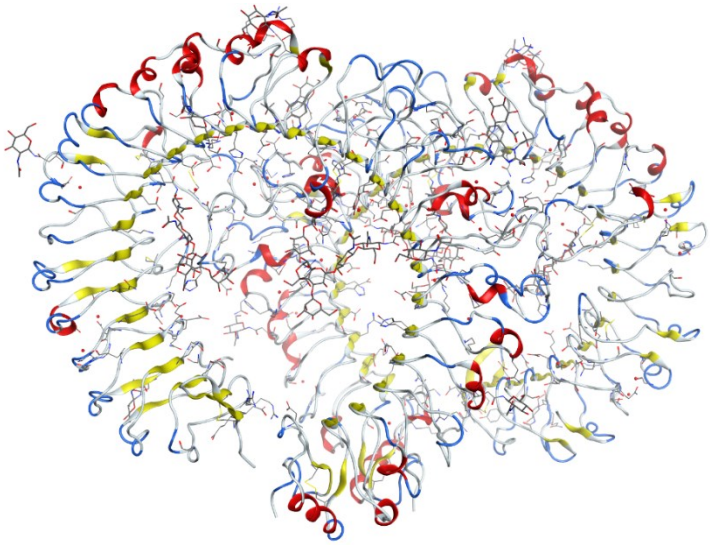


Figure 8: *A molecular model. A PDB structure is an example of a molecular model. This is the unmodified model of TLR8 prepared for docking in Chapter 3.2.4.1, shown in MOE. The programme allows users to perform various manipulations on the structure. Adapted from Tanji et al., 2013 (PDB: 3W3N)*[14][50]

1.2.2.3 Virtual screening

Virtual screening is the automated search of a database of virtual molecular structures for molecules with desired properties.[88] Compared to the de novo design of molecules from fragments, searching existing databases for molecules that match the query has the advantage that the method can be used to examine libraries of compounds with known chemical syntheses or even those of corporate chemical compound suppliers.[69]

Virtual screening can be either structure-based or ligand-based. Structure-based screenings typically involve docking a library of molecules into a structure to generate hits based on a scoring function. Ligand-based screenings focus on the structural and physicochemical properties of the molecules as well as molecular descriptors. For example, the molecules are compared to a predefined set of molecular interactions, often represented by pharmacophores, or the overall shape and volume of the structures to that of a reference molecule.[76]

1.2.2.4 Molecular docking

Molecular docking is the process of digitally inserting a library of molecules into the binding site of a computerised protein structure. Programs generate many different spatially possible conformations of the molecules and subsequently dock them into the binding site. The RMSD values between the position of the ligand atoms and those of the structure are calculated to determine whether the docking was successful. The results are then ranked according to a scoring function. The scoring function determines which features are considered important by the algorithm. Simpler models will only score the docking poses by their complementary shape, more sophisticated models will also consider their binding enthalpy and the most sophisticated will also consider entropy, essentially calculating the free binding enthalpy. The results also depend on the type of docking program used. It determines whether the protein and ligand are rigid or dynamic and whether rotatable bonds are considered.[89]

1.2.2.5 Hit selection

Since only a small number of hits from virtual screenings can actually be developed into drugs, screening is often performed on large molecule libraries to obtain a large number of promising candidates. This leaves researchers with a large number of molecules whose structure-activity relationships must be tested using a variety of methods to find those most likely to be active and safe in vivo. In fact, this process, known as hit selection, can be the most challenging task in a virtual screening pipeline.[90]

1.2.2.5.1 Key considerations in hit selection

Several techniques are usually applied to find evidence of activity and to filter out inactive or unsuitable molecules. Many of them follow the same basic considerations but use different methods to evaluate the molecules with different results. Only the most promising molecules are taken forward for further investigation.[90] As outlined in the following chapters, some important considerations are the type and number of interactions, the rigidity of the molecule, the shape complementarity of the binding pose to the target and the likelihood of it actually occurring, and the overall bioavailability of the molecule, as well as the bioavailability.

1.2.2.5.1.1 Evolution of understanding interactions

An important concept that has stood the test of time was proposed by Emil Fischer in 1894. He proposed that ligands fit into the binding pocket of their target like a key into a lock. This statement still holds true under modern evidence and shape complementarity is therefore still an important concept in CADD today.[91] In 1958, the concept was extended by D.E. Koshland when it was discovered that the binding sites of many proteins actually undergo a small conformational change to accommodate the ligand and form a protein-ligand complex. This is known as "induced-fit" and emphasises the plastic nature of protein-ligand interactions.[92]

Today it is known that, in addition to shape complementarity, specific protein-ligand interactions are responsible for their intrinsic effects. These interactions not only hold the ligand in place, but usually induce conformational changes in a receptor, activating it and allowing it to transduce signals from the cell surface to the cytoplasm.[91] The vast majority of drugs interact with their targets through non-covalent interactions. They are therefore of particular interest in computer-aided drug design.[93] The most important interactions between the functional groups of a ligand and its receptor are hydrogen bond donors, hydrogen bond acceptors, ionic interactions, hydrophobic interactions, metal complexes and π-interactions between aromatic structures, and cation-π interactions.[94] It is preferable for a molecule to interact with its target at at least three different contact points to stabilise the binding position in three-dimensional space. This increases the likelihood that the effect will actually be transduced at the binding site.[79] Consistent with this, it has been observed that larger molecules tend to have better selectivity for their targets due to their number of interactions, while smaller molecules often have more unselective and less pronounced effects.[95] However, large molecules have drawbacks when it comes to bioavailability, as described in Chapter 1.2.2.5.1.4.

1.2.2.5.1.2 Binding affinity

As proposed by Leonor Michaelis and Maud Leonora Menten in 1913, the formation of protein-ligand complexes can be viewed as chemical reactions.[96] According to the laws of thermodynamics, these reactions can only occur spontaneously if the process is energetically favourable. The more energetically favourable, the stronger a ligand binds to its target.[97] Stronger interactions are associated with a greater effect on the target and are therefore often desired in drug discovery. This interaction strength is known as binding affinity.[98]

A good indicator of the affinity of a molecule for its target is the free binding enthalpy ΔG.[99] Negative values indicate an exothermic reaction and are therefore desirable.[97] As can be seen from its formula, ΔG depends on the binding enthalpy ΔH and the entropy ΔS at temperature T.

$$\Delta G = \Delta H - T \cdot \Delta S$$

A high negative binding enthalpy ΔH indicates a release of energy upon binding and therefore a strong bond between the molecule and its target. However, low temperature and low or even negative entropy can still make a binding event unlikely.[97][99] Finding the right balance between ΔH and the term -TΔS is a difficult process in computational drug design, as the two effects cancel each other out. In general, many non-covalent interactions are associated with negative ΔH.

However, many interactions also deprive the binding partners of many degrees of freedom, resulting in negative ΔS, which compensates for ΔH. Conversely, positive changes in ΔS are accompanied by positive ΔH due to the disruption of non-covalent interactions.[97]

A common strategy in drug design is to reduce the effects of entropy while preserving as many interactions as possible.[100] One way to achieve this is to limit the number of theoretically possible degrees of freedom that a molecule is deprived of when it binds to its target. A molecule that is very flexible in its structure will have a large negative change in entropy upon binding, resulting in a higher free enthalpy ΔG according to the formula. Compounds with limited flexibility but many interactions have an energetic advantage over those with many interactions and high flexibility.[99]

Another driving principle of binding energy is the solvation of receptor and ligand in the aqueous environment of the body. Proteins and drugs are surrounded by a water envelope that is bound by hydrogen bonds. The water molecules seek an ordered, energetically favourable state. The formation of this shell works best with polar groups because the water molecules can form hydrogen bonds with the polar groups and create order. However, the shell cannot form properly with non-polar groups. The result is a disordered and energetically unfavourable state. In order for a molecule to bind to its target, it must shed its solvation shell and form new bonds with the protein. Whether this represents an energy gain or loss depends on how the solvation shell interacts with the receptor and ligand. For polar groups, this perturbation only represents an energy gain if the total number of hydrogen bonds formed between the ligand and the protein, as well as between the displaced water molecules, is greater than the number of hydrogen bonds between the protein, ligand and their solvation shells before the binding event. In the case of apolar groups, however, the displacement of the polar water shell by another apolar group usually represents an energy gain. In this case, the displaced water molecules can form hydrogen bonds with each other after the binding event and are no longer bound to the target. This increases the total number of hydrogen bonds as well as the freedom of movement of the water molecules, which overall represents a more energetically favourable state.[94] A high number of hydrophobic interactions between a molecule and its receptor is therefore usually associated with good binding affinity. However, a high number of possible hydrophobic interactions is also associated with poor water solubility of the molecule and therefore a compromise in bioavailability.[101]

1.2.2.5.1.3 Geometry and energy optimisation

Yet another important consideration in the study of protein-ligand interactions is the fact that molecules, and especially ligands, are flexible structures. Due to effects such as electrostatic repulsion, stressed or even impossible binding angles and torsions, or stereochemical hindrances, not all theoretically possible conformations are equally as likely to occur in reality. Molecules strive to exist in the conformation associated with the lowest potential energy when unbound. The difference in conformational energies between the bound and unbound states has an effect on binding affinity, as bioactive conformations are not necessarily at the energy minimum of the molecule.[102][103] To form a binding theory it is essential to study models with energetically favourable and realistic conformations. Many algorithms have been developed to calculate the most energetically favourable conformation of molecules. This calculation is known as energy minimisation. The algorithms are often based on force field methods, which are essentially quantum chemical models of molecular mechanics that can be used to calculate the potential energy of atoms.[104][105] These models are only approximations and can give different results. However, they are a very important tool for designing drugs whose bioactive conformation is close to their energetic minimum, which maximises the probability that the molecule will be in the correct conformation at the target site. This in turn increases the likelihood of a binding event occurring. One of the most widely used energy minimisation algorithms is the Merck Molecular Force Field, published in 1994. Better known as MMFF94, it was originally developed for use in molecular dynamics simulations, but has since found widespread use in many types of simulations, particularly for energy minimisation of small molecules.[104][106][107]

1.2.2.5.1.4 Bioavailability

A concept that further emphasises the importance of a good balance between hydrophobicity and hydrophilicity among other physicochemical properties of a molecule, is bioavailability. It depends on the absorption, distribution, metabolism and excretion of the drug.[108] If a molecule is not actively transported by a protein, its absorption is almost exclusively related to its lipophilicity, as it gives the molecule the ability to pass through biolayers.[109] At the same time, polar groups are important for solubility, which is crucial for absorption in the gastrointestinal tract and, if not transported by a protein, also for distribution to the periphery via the bloodstream.[109][110] In addition, very hydrophobic compounds are able to cross the blood-brain barrier, which is undesirable in most cases.[111] Groups that are easily conjugated by enzymes, such as hydroxyl, amine or carboxylic acids, oxidisable groups and predetermined breaking points, such as ester and amide bonds, accelerate the elimination of a molecule.[110] As do molecular sizes above 500 to 600 Daltons, as such molecules are eliminated by the liver.[109] Furthermore, the larger a molecule, the slower it will diffuse across biomembranes, so the average drug should preferably be rather polar, with just enough lipophilicity to be sufficiently bioavailable without crossing the blood-brain barrier, and a sufficient size to avoid being too mobile and too rapidly eliminated.[108][109][111] When all the constraints on drug design described above are taken into account, a relatively clear design space emerges in which most small-molecule drugs lie. The best known rule of thumb for the properties of drug-like molecules is Lipinski's Rule of Five. It was established in 1997 by Lipinski et al. at Pfizer. This empirical rule states that small molecules are likely to have poor absorption and bioavailability if they have more than five hydrogen bond donors, more than 10 hydrogen bond acceptors, a molecular weight greater than 500 daltons and a partition coefficient (log P) greater than 5.[112]

1.2.3 Machine learning

Data can be thought of as a set of recorded facts. Information consists of patterns within data that describe circumstances. Some of these patterns are obvious and easy to decipher. Others are often overlooked or only implied by other patterns. Methods that seek to discover patterns and bring out implicit relationships in data are known as data mining. They explore the underlying structural rules in a dataset to classify data or make predictions about missing or future data. Machine learning is the technological application of data mining in which the machine, a computer, learns the structural patterns within a dataset in order to communicate the information to humans in a form they can better understand.[113] It has been described as the intersection of statistics, computer science and artificial intelligence.[114]

While in the past computer-aided drug design was largely based on theoretical and empirical rules, recent advances in algorithms, data storage capacity and the accessibility of chemical databases have enabled researchers to data mine large chemical datasets, leading to a sharp increase in the application of machine learning methods in chemistry over the last decade.[115] A common application of machine learning models in chemistry is the exploration of their structure-activity relationship, as described in Chapter 1.2.[111][116]

1.2.3.1 Fundamentals of machine learning

Machine learning approaches can be divided into two basic categories. In supervised learning, the algorithm is presented with actual input-output pairs and attempts to apply the rules learned from them to unseen data. The most important methods of this type are classification models, which are used to classify data points, and regression models, which are used to predict continuous variables. Supervised learning is widespread, easy to understand, and primarily used for automated decision making.[114][117] In unsupervised learning, only the input data is known and the algorithm is never presented with output data.[114] It includes types of learning such as association learning, where there is no predefined class and a common feature between the data points is used to associate them, finding a similarity. Another task in unsupervised learning is clustering, where multiple examples that are naturally similar in different ways are grouped together based on property types that are not necessarily shared by all data points. Unsupervised learning is useful in situations where the desired output is more abstract, rather than a discrete value or category.[118]

The best approach to a problem depends not only on the task, but also on the data. Data analysis is a critical step in any machine learning project. Most structured data typically falls into two categories, numeric and categorical. Numerical data can be continuous, such as wind speed or time, or discrete, such as the occurrence of an event. Categorical data, on the other hand, takes on fixed values like labels. A special case of categorical data is binary data, which can only take two values, such as 0 and 1. Understanding data type is essential for choosing appropriate visual displays, data analysis techniques, statistical models and ultimately the right algorithm. Other important aspects are the central tendencies, variability, dispersion and distribution of the data at hand. Variance and standard deviation, although sensitive to outliers, are common measures of variability and indicate whether the data is spread out or tightly clustered. More robust measures include mean absolute deviation, median absolute deviation and percentiles. Common visualisations of the distribution of data include tools such as box plots, frequency tables, histograms and density plots. In addition, analysis of correlations between predictors and the target variable is essential in machine learning projects. Correlation coefficients and matrices are used to quantify the relationships between numerical variables. Scatter plots are used to visualise these relationships.[119]

An important fact to keep in mind when working with machine learning algorithms is that they can only make predictions on unseen data, for which information is implicit in the training data. Data selection and preparation is therefore one of the most important tasks of the project.[114] The availability of quality data is a bottleneck in many machine learning schemes. Information often comes from different sources. Integrating, standardising and cleaning the data is often a lengthy process, but it is essential to ensure that the model is trained with real, validated instances. Missing values, imprecise data and sparse data, i.e. data where most instances are 0 or none, are some of the problems with data sets that need to be dealt with.[120] Different algorithms have different abilities to deal with these problems, with each having specific data requirements and problem contexts where it excels.[114] For example, algorithms that treat numerical data attributes as if they were on a ratio scale work better on data that is normalised to lie in a fixed range, usually between 0 and 1.[120] The problem statement, data set, data representation, and the chosen algorithm need to be aligned before model building.[114]

1.2.3.2 Decision trees

The relationship between data representation and learning algorithms is an important aspect of machine learning. Data representations not only help humans understand the patterns in the data, but they also dictate the methods, and therefore the learning algorithms, that can be used to discover those patterns. The simplest representation of data is a table, a basic tool for discovering underlying relationships. More sophisticated representations are linear models, which can be used when the relationship between the target variable and the input data is assumed to follow a linear function. However, due to the assumption that the data can be accurately described by a hyperplane, these representations are limited in their ability to capture complex and non-linear relationships.[121][122] Representations that accurately capture non-linear relationships include a variety of different methods. One of the most common is decision trees. They classify data into a hierarchy of yes and no questions, resulting in a tree of nodes. Data in these nodes is further tested with yes or no questions until endpoints, called leaves, are reached.[122][123] By their nature, decision trees make no assumptions about the relationship between the target variable and the input data, as linear models do. They learn the relationship from the data itself. Therefore, their classification rules can be very complex to account for all data points. Such models are called non-parametric because the function and its parameters describing the data are not predefined.[124]

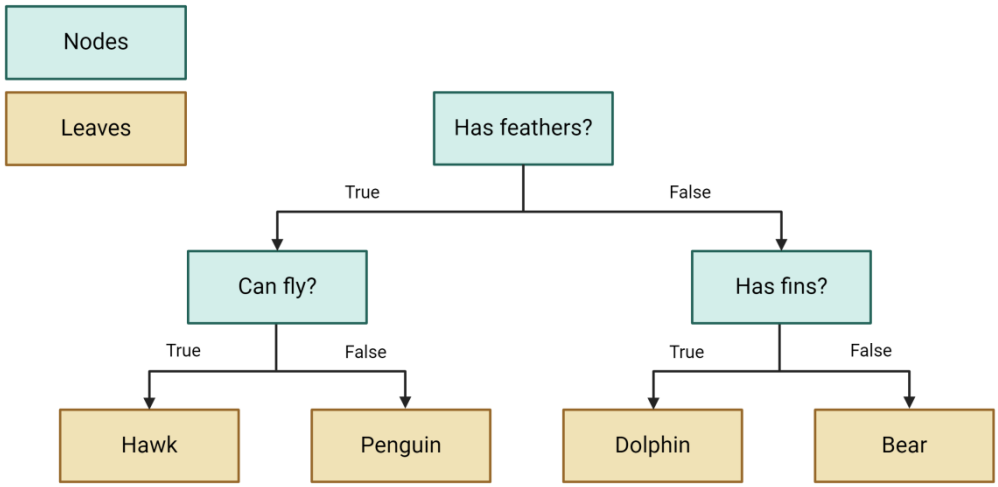


Figure 9: *An example of a decision tree. The nodes (green) are divided until the leaves (orange) are reached. Adapted from Müller and Guido, 2018.*[13][123]

1.2.3.2.1 Decision tree-based learning algorithms

Decision tree-based algorithms can be seen as a form of supervised learning because they learn their classification rules from correctly classified data sets.[118] Despite their nature, they can be built not only for classification but also for regression problems by grouping features with similar numerical values and recursively partitioning them. Decision tree learning schemes have two main advantages over other algorithms. Firstly, they are easy to visualise, smaller trees can even be understood by non-experts. Secondly, they do not require any scaling, such as normalisation or standardisation of features. They generally work well with a mix of different scales and binary and continuous features.[123] The main disadvantage, however, is that as non-linear models, decision trees can be built to any complexity until all the leaves are pure and contain only data points that have been correctly classified by the target variable. This is usually undesirable in machine learning algorithms. Such a model overfits the data and has poor generalisability (see 1.2.3.4). There are two basic strategies to avoid overfitting in decision trees. Pre-pruning involves deliberately stopping the generation of the tree at an early stage. This can be done by limiting the maximum depth of the tree, the maximum number of leaves, or by setting a threshold for the minimum number of data points in a node to split it further. Post-pruning removes or collapses nodes with little information after the tree has been generated. These strategies against overfitting are common to all decision tree-based algorithms. However, despite these measures, individual decision trees often struggle with overfitting and robust generalisation performance. To combat this problem, decision tree ensemble methods have been developed.

Ensemble methods are models that combine several machine learning models to create a more powerful model. Two of the most common ensemble methods based on decision trees are random forests and gradient boosting regression trees. In random forests, multiple trees are built simultaneously. For each tree, about one-third of the data points in the training data are randomly replaced with data points from the other two-thirds of the data, creating a slightly different data set for each tree, but no new data points.[123] For the final result, the classification of all data points is then voted on in classification tasks and averaged in regression tasks.[123][125] In gradient boosting regression trees, the trees are built sequentially rather than simultaneously, with each subsequent tree trying to correct the mistakes of the previous one. Both methods have in common that they introduce another parameter that affects overfitting, the number of trees built.[123]

1.2.3.2.1.1 Gradient Boosted Regression Trees

Despite having regression in their name, gradient boosted regression trees, like all decision trees, can be used for both regression and classification tasks. As mentioned in the previous chapter, unlike random forests, they build trees sequentially, so that each tree tries to correct misclassifications made by the previous tree. Gradient boosted regression trees are usually very shallow, with depths of one to five. The idea of this method is to sequentially combine simple models that make increasingly good predictions about the data.[123] To achieve this, the differences between the predictions and the actual values are quantified by a loss function. Solving this loss function for the trees reveals a gradient, the magnitude and direction of which is then used to guide the construction of the next tree.[126] This introduces another important parameter, the learning rate. It controls how strongly each subsequent tree is allowed to correct previous mistakes.[123] The term "boosting" in ensemble learning techniques means that learning weights are added to the instances where previous trees misclassified, so that they are prioritised in the learning process.[125] Another difference with random forests is that they do not involve randomisation. Instead, the overfitting problem is tackled by rigorous pre-pruning.

Also, in random forests, more trees are usually associated with better results, whereas in gradient boosted regression trees, too many trees lead to overfitting. Therefore, one of the main drawbacks of this algorithm is its sensitivity to hyperparameter tuning. However, if tuned correctly, it can provide more accurate results than random forests. Like other decision tree-based methods, this algorithm usually works well on a mixture of binary and continuous features without prior scaling, although it struggles with high-dimensional sparse data. The most important hyperparameters to set for gradient boosted regression trees are the number of trees to build, the maximum depth of the trees, the minimum samples per leaf, the minimum samples to split a node, and the learning rate.[123]

1.2.3.3 Featurisation

Regardless of the model, a challenge in machine learning is to present the data to the computer in a way that it can understand.[114] When working with molecules, a digital representation must be used so that the algorithms can process it. Many properties of molecules that can be expressed in numbers can be used for this purpose. These range from physicochemical properties to quantum chemical descriptions to digitised structural information. Describing molecules in this way is known as featurisation. Which featurisation is best suited for a machine learning model depends on the information that is required. The most common featurisations are molecular descriptors and fingerprints. Descriptors are usually physicochemical properties of molecules calculated from their structural information. Fingerprints are arrays of ones and zeros, each representing the presence or absence of a particular feature in a molecular structure. Python packages commonly used for the featurisation molecules are RDKit and Mordred.[127][128]

One problem with binary representations of molecular structures is that, given only information about the atoms and bonds in a molecule, the number of ways to label them is equal to the factorial of the number of atoms of the molecule. In order for computers to handle molecular structures in a consistent manner, a unique way of enumerating the atoms in a molecule is required. The use of a standard representation with a reproducible numbering of the atoms according to a set of rules is known as canonicalisation.[129]

1.2.3.3.1 Morgan algorithm in canonicalization

A common algorithm used in canonicalisation is the Morgan Algorithm, named after its inventor and published in 1965.[129][130] The Morgan algorithm works in an iterative fashion, assigning a unique sequence of numbers to the extended connectivity of each atom. To achieve this, the structure is conceived without all hydrogen bonds. The canonicalisation takes place in two phases.

In phase 1, the algorithm calculates how many atoms each atom is bounded to. These numbers are called connectivity values. Next, it assigns to each atom the sum of the connectivity values of all its direct neighbours. This process is repeated several times. Each iteration introduces new numbers for the connectivity values. The algorithm stops iterating over the connectivity values when the number of unique connectivity values in the molecule is the same or less than in the previous iteration.

Now the connectivity values from the penultimate iteration are used to canonically number the molecule in phase 2. The atom with the highest connectivity value is assigned number 1. Naturally, this will usually be an atom in the centre of the molecule.[129] All neighbours of atom 1 are numbered in ascending order according to their descending value. The atom with the highest connectivity value bonded to atom number 1 is assigned number 2, the atom with the second highest connectivity value is assigned number 3, and so on. When all neighbours of atom number 1 are numbered, atom number 2 is considered. All neighbouring atoms of atom number 2 that do not yet have a number are numbered. When all the neighbours of atom number 2 have been numbered, all the neighbours of atom number 3 that have not yet been numbered are numbered. This process continues until all the atoms in the molecule have been assigned a unique number.[130] If two connectivity values are the same, they are assigned either randomly, based on the Cahn-Ingold-Prelog system or other rules, depending on the specific subtype of the algorithm being used.[129]

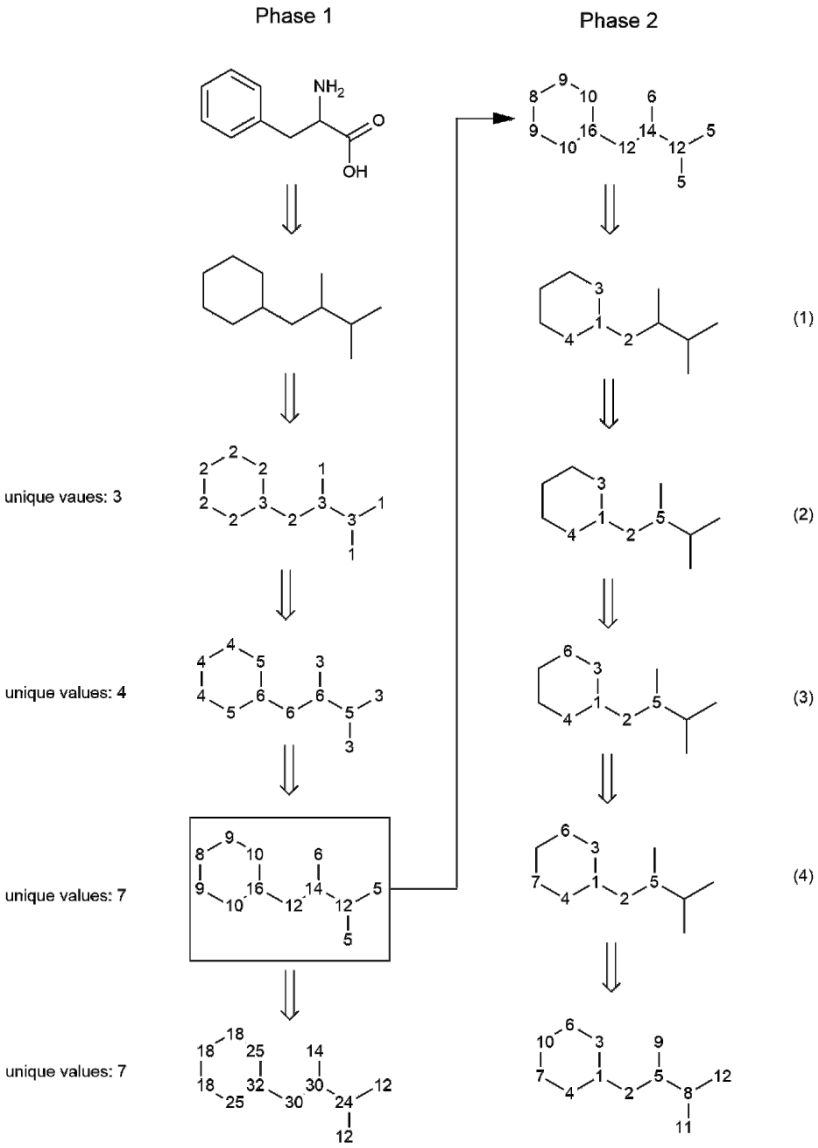


Figure 10: *Canonical numbering of phenylalanine using the Morgan algorithm. (1) The highest value is assigned number 1 and all neighbouring atoms are numbered in descending order. (2) Once all the direct neighbours of atom number 1 have been numbered, the neighbouring atoms of atom number 2 are considered. (3) After atom number 2, atom number 3 is considered. Therefore, atom number 6 is next to atom number 3. (4) The process continues until all atoms have been numbered. Two atoms with the same connectivity value are numbered according to the rules of the specific implementation of the algorithm. Adapted and modified from Engel and Gasteiger, 2018.*[129]

1.2.3.3.2 Molecular fingerprints

Molecular fingerprints are an efficient way of representing chemical structures in numerical terms. They were originally developed to quickly sort out molecules from databases that do not have a particular substructure of interest, without having to use more computationally expensive methods. There are many different types of molecular fingerprints. Many of them encode the structure in an array of binary numbers, where each digit represents the absence or presence of a particular molecular moiety. They can generally be divided into structure-key fingerprints and hash-key fingerprints.

In structure-key fingerprints, also known as knowledge-based fingerprints, a number stands for a whole, human-defined part of a molecule. They are easy to understand and the molecule can be reconstructed from the fingerprint using a dictionary. However, the human decision of which moieties to include introduces a risk of bias, and the predefined moieties make it difficult to work with new scaffolds and unusual structures.

Hash-key fingerprints, also known as information-based fingerprints, attempt to transform the entire graph structure of a molecule into a fingerprint, more like a mathematical graph-vector transformation. This type of fingerprint adapts to all chemistries and can often still be reconstructed using graph optimisation tools such as those used in fragment-based drug design. However, bit collisions can occur during the calculation of the fingerprints, which can lead to unexpected behaviour.[130]

1.2.3.3.2.1 Morgan fingerprints

The Morgan fingerprint is a hash-key fingerprint that encodes information about the chemical neighbourhood of each atom. It is based on the Morgan algorithm, which is also used for canonicalisation as described in Section 1.2.3.3.1. The Morgan fingerprint is a so-called circular fingerprint because it iterates from a central atom to the neighbouring atoms up to a user-defined radius.

Rather than using the number of non-hydrogen atoms to which each atom is bonded, as in canonicalisation, to generate fingerprints, the Morgan algorithm considers a set of chemical invariants that contain information about the atomic neighbourhood of the atoms. This could be, for example, the order of bonds to neighbouring atoms and the atomic number of the neighbour in the periodic table of elements. The chemical invariants considered depend on the specific implementation of Morgan fingerprints.

For each atom, a single unique value of these invariants of all neighbouring atoms is generated in iterations. Each iteration corresponds to a radius of atoms from the atom in question and takes into account the chemical invariants of all atoms in the current radius. This produces a unique number for each iteration, representing the chemical properties found in the current radius. Chemical invariants relating to other atoms are considered between the current radius and the atoms of the previous iteration. When the predefined radius and thus the maximum number of iterations is reached, the numbers of all iterations are hashed into a single key representing a vector that uniquely characterises the particular chemical environment of the atom under consideration. Once these numbers have been hashed for all atoms in a molecule, they are transformed into a fixed-length array of binary values, which is where bit collisions can occur.[130]

1.2.3.4 Model optimisation

When a machine learning model is able to make correct predictions for unseen data based on the rules it learned from the training data, data scientists talk about its ability to generalise. Counter-intuitively, a good ability to generalise does not mean that the rules the model uses to make its predictions can explain all the data it was trained on. If a model explains all the training data perfectly, it usually means that it contains many arbitrary rules that happen to work for the training data, but have no general applicability to the whole type of data, and therefore perform poorly on unseen data. When a model fits too well to the training data, it is called overfitting. Conversely, if the rules the model uses are too simple and have little or no predictive power, data scientists say it is underfitting. There is a sweet spot of model complexity that yields the best generalisation performance. The ideal model complexity also depends on the data set being used. Data with more variation performs well with complex models, while the rules for data sets with little variation need to be chosen very carefully. Model optimisation is therefore a crucial part of programming machine learning models.[131]

1.2.3.4.1 Model evaluation in supervised learning

Evaluating the performance of a supervised machine learning model requires data where the correct output is known and can be compared to the model's predictions. Therefore, the data used to build the model is often split into three subsets with different roles: a larger training dataset, a validation dataset used for model optimisation, and a test dataset used for final model evaluation.[131-133] This is important because the model evaluation itself actually serves two different purposes. One is to adjust the model during the optimisation process to achieve optimal model complexity, and the other is to evaluate the ability of the final model to generalise.[133] If the effects of model optimisation were tested on the training data set, the model would be optimised to predict the data on which it was trained, and would therefore overfit. Similarly, model optimisation cannot be evaluated on the test data, because during the optimisation process, particularly hyperparameter tuning, the model could be optimised to overfit the validation data, which would go unnoticed without a final test set that wasn't involved in the model building process. A test set is supposed to be a simulation of the real data that will be presented to the model once it is finished, and therefore needs to be treated independently to obtain a reliable estimate of the model's performance before it is presented with unseen data.[133][134] For small datasets, splitting the original data into three subsets can be problematic, as data is scarce and more training data is desired, as it is usually associated with a more powerful model. A workaround for this problem is to cross-validate the training set, rather than devoting an entire subset of data to validation alone.[132][134] (Figure 11)

Any evaluation of a machine learning model involves quantifying the errors made in the predictions. This is done using statistical functions known as loss functions. They are able to quantify the extent to which the predictions deviate from their actual values in order to guide the model optimisation process during training. There are many different loss functions, the choice of which depends on the task at hand, such as regression or classification, and the specific behaviour the model is trying to optimise.[135] In fact, minimising loss functions, and thus optimising predictions, is one of the main goals of a machine learning algorithm.[136] Interestingly, many loss functions are similar in nature to the metrics used to evaluate finished machine learning models after training. For example, mean squared error or mean absolute error are both used as loss functions and as evaluation metrics for finished machine learning models.[135][137]

1.2.3.4.1.1 k-fold cross-validation

Cross-validation is a technique that can be used to validate a model without the need for a dedicated dataset. Instead of splitting the data into training and validation sets, the dataset is split multiple times, resulting in multiple training and validation sets. Multiple models are then trained on the training sets, which are then evaluated on the corresponding validation sets so that all data points have been included in both the training and validation sets in different iterations. A common cross-validation method is k-fold cross-validation, where k is the number of splits performed and subsequently the number of models trained. Typically, k is a number between 5 and 10. The data set is divided into k parts, called data folds, of approximately equal size. For a model, one fold is used as the validation set, while the other folds are used as the training set. This process is repeated until each fold has been used once as a validation set. Thus, k models are built. The accuracies of all models are evaluated, providing a comprehensive overview of the model's performance across different data splits. If the averaged cross-validation score of a model is high, it can be assumed that it has the ability to generalise well to unseen data. In addition to saving data, another advantage of cross-validation is that it eliminates the risk that the validation set is unrepresentative of the entire data set by chance, thereby increasing generalisability.[132]

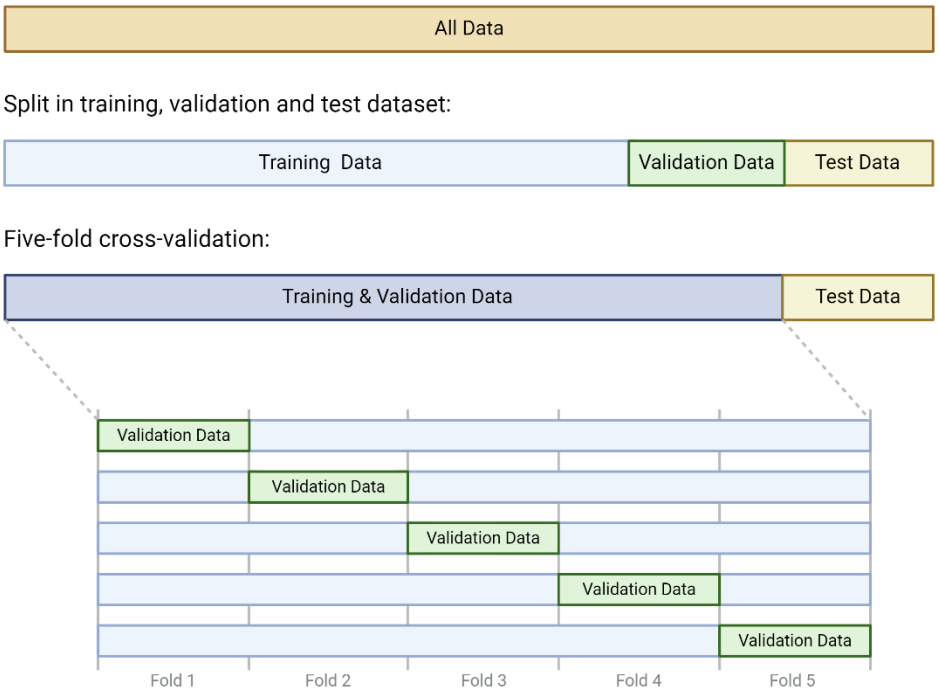


Figure 11: *Common data splitting techniques in machine learning. Cross-validation can be used instead of a validation data set. Adopted from Müller and Guido, 2018*[13][132]

1.2.3.4.1.2 Evaluating numeric prediction models

The fundamental difference in the evaluation of classification and regression models lies in the nature of the predictions. In classification, predictions can be categorised as either correct or incorrect, whereas in regression there is a spectrum of errors of varying magnitude, so it is more a matter of assessing the degree of deviation from the true values. Therefore, different scoring methods need to be applied. Some of the most commonly used metrics in numerical prediction are the mean absolute error, the root mean squared error and the correlation coefficient R^2.[137]

1.2.3.4.1.2.1 Root mean squared error

Mean squared error is the most commonly used error metric. Its square root is often used for better interpretability as it returns the same dimensions as the predicted value. It weighs larger errors more heavily than smaller errors.[137]

$$\text{RMSE} = \sqrt{\frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{n}}$$

p_i = predicted value, a_i = actual value, n = number of instances[137]

1.2.3.4.1.2.2 Mean absolute error

Unlike mean squared error, mean absolute error treats all errors equally and is therefore a useful metric to consider when a disproportionate influence of outliers on the model is not desired. It is simply the averaged size of all errors.[137]

$$\text{MAE} = \frac{|p_1 - a_1| + \ldots + |p_n - a_n|}{n}$$

p_i = predicted value, a_i = actual value, n = number of instances[137]

1.2.3.4.1.2.3 Coefficient of correlation

The statistical correlation coefficient describes the relationship between the two values observed. It can take values between -1 and 1, where the former indicates a perfect negative correlation and the latter a perfect positive correlation, and 0 indicates no correlation. In the context of machine learning, negative correlations should not normally occur, as it is the very purpose of machine learning to correlate positively with the actual values. Unlike the other error metrics discussed, the correlation coefficient is scale independent, meaning that it does not matter if all variables in the equation are multiplied or divided by a constant factor.[137]

$$R^2 = \frac{\frac{\sum_i (p_i - \bar{p})(a_i - \bar{a})}{n-1}}{\frac{\sum_i (p_i - \bar{p})^2}{n-1} \, x \, \frac{\sum_i (a_i - \bar{a})^2}{n-1}}$$

p_i = predicted value, $\bar{p}$ = mean of all predicted values, a_i = actual value, $\bar{a}$ = mean of all actual values, n = number of instances[137]

1.2.3.4.2 Hyperparameters

Machine learning models can learn parameters, such as the correct weights in neural networks or trees in decision tree ensembles, from the data from which they are generated. However, there are some algorithm-specific settings that are critical to successful learning and cannot be learned automatically from the data. These parameters are known as hyperparameters. They control configurations such as the learning rate or the algorithm used to minimise the loss function, and must be tuned for the particular application during model optimisation. This process is an important step in the optimisation of any machine learning model and is referred to as "hyperparameter tuning". Decision tree algorithms in particular have many such parameters and are very sensitive to their settings. Traditionally, hyperparameter tuning is done manually by measuring the effect of the changed settings on the performance of the model using the validation set or cross-validation. However, this approach is increasingly being replaced by automated hyperparameter searches because some algorithms have many parameters to be set, in-depth knowledge of how the algorithm works is required to set these parameters sensibly, the process is generally very time consuming, and finally, many hyperparameter settings are not linearly related to their effects. This last point in particular makes it important to choose a suitable optimisation algorithm, as the search for the globally optimal parameters is complicated by the fact that there are often local performance extremes, which are not easy to identify as such due to the complex interrelationships. Automated hyperparameter optimisation saves time, usually finds better hyperparameter combinations than manual optimisation, and thus improves performance and ensures better reproducibility of the models.[136]

1.2.3.4.2.1 Grid search

The term "decision-theoretic methods" covers some of the most commonly used approaches to automated hyperparameter search, the most well-known of which is probably grid search. In this method, all combinations of hyperparameters from a grid of lists are systematically and exhaustively tested for each adjustable hyperparameter in order to find the optimal combination. In order to cover a large range of possible parameters, an initial grid search with widely spaced values in the lists is usually performed first to discover optima. Since the grid search is tied to the user's predefined lists, it can identify regions in the design space that provide better values, but it cannot explore them further. Therefore, the next step is to perform another grid search with values that are closer to each other, based on the combinations that gave good results in the first search. This process is repeated until the optimal combination of hyperparameters is found. A disadvantage of this method is the high computational cost, since for a set of k parameters with n possible values, n^k combinations have to be tested.[136]

1.3 Aim of the present thesis

TLR8 is central to the human immune system.[10] Novel TLR8 agonists could improve our ability to modulate this receptor, providing new and improved therapeutic options for infectious diseases such as influenza, hepatitis B and HIV.[23][62] In addition, activation of TLR8 may be beneficial in the treatment of various cancers, particularly those that are resistant to conventional treatments.[10] Selective TLR8 agonists may even be able to be administered systemically due to their good safety profile.[138] Therefore, the aim of the present work was to discover new promising molecules that could be developed into agonists for Toll-like receptor 8 using various methods of computer-aided drug design.

2 Methods

2.1 Programs

This chapter covers all the computer programs used for this thesis, all of which were run on the Ubuntu operating system version 22.04.

2.1.1 LigandScout

LigandScout is a program published by Inte:Ligand GmbH that provides many functionalities related to computer-aided drug design. Most importantly for this work, it can be used to calculate and manipulate pharmacophores, search databases using these pharmacophores, and calculate and display molecular interactions between macromolecules and ligands, allowing the investigation of binding modalities. The LigandScout version 4.4 expert was used in this work. Its graphical user interface consists of four different "perspectives" that represent the main functionalities of the program: "structure-based", "ligand-based", "alignment" and "screening".[139] The most relevant functions for this work will be outlined below.

2.1.1.1 espresso

A notable feature of LigandScout is espresso. It can generate pharmacophores, visual representations of structural features of a molecule that can show interactions between macromolecules and ligands, such as hydrogen bond donors and acceptors, hydrophobic regions, ionised atoms, aromatic π-stacking and others which are described in more detail in Chapter 1.2.2.1. These pharmacophores can also be edited in the software to suit the needs of the researcher. An index of possible molecular interaction features and their meaning can be found in Figure 12.

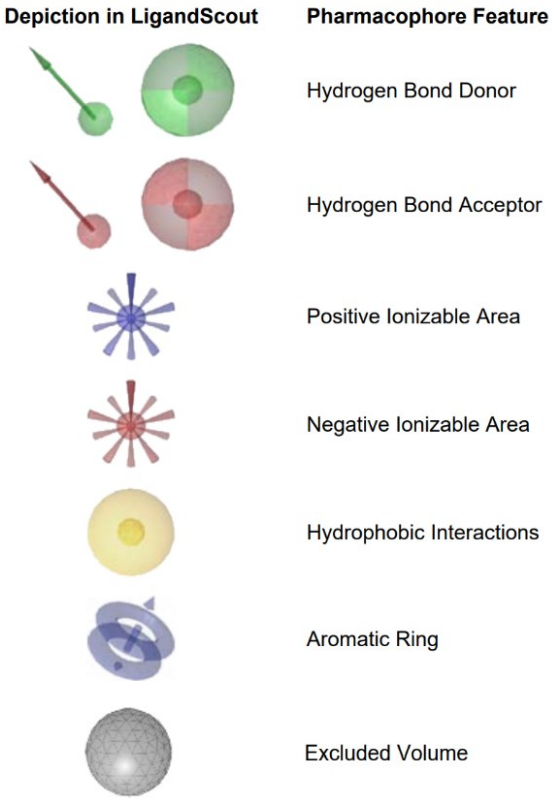


Figure 12: *Most common pharmacophoric interaction features rendered by LigandScout. Optional features are rendered in a checkquerboard pattern. Adapted and modified from LigandScout user manual.*[140]

2.1.1.2 iscreen and idbgen

iscreen is the integrated virtual screening capability of LigandScout. It can be used to screen multiconformational libraries of molecules to obtain hits based on whether they fit the pharmacophore. It can also calculate various statistical metrics to evaluate screening models. These multiconformational libraries can be generated from uniconformational libraries using the idbgen function.[139]

2.1.1.3 Alignment

The alignment perspective can be used to overlay and centre molecules and pharmacophores. This is particularly useful when investigating the structural differences or shared properties of molecules, their fit into a pharmacophore, or in the design process of a pharmacophore.[140]

2.1.2 MOE

Molecular Operating Environment (MOE) is a program developed by the Chemical Computing Group in Quebec, Canada. It can perform various tasks related to computer-aided drug design and is widely used in computational chemistry research. Notable features of MOE are those related to molecular modelling and simulation, molecular docking and molecular dynamics. MOE accepts digital protein structures, for example PDB-files. It can be used to perform various manipulations, homology modelling and calculations on these structures. The most relevant features of MOE for this work are described in the following chapters.[141]

2.1.2.1 PLIF

Protein-ligand-interaction fingerprints, or PLIF for short, is a feature in MOE that can be used to investigate the specific interactions between a ligand and protein residues. PLIF automatically analyses the protein structure loaded into MOE for either surface contacts between a library of ligands and the protein, or potential-based interactions between the ligands and the protein. It can display these interactions in a fingerprint-like graph and also allows the user to select molecules from the ligand database that have specific desired interactions.[141][142]

2.1.2.2 Wash

MOE's wash function allows various operations to be performed on the atoms of molecules in a database, including renaming, removing salts and heavy atoms, changing tautomers and protomers and, most importantly for this work, adding or removing hydrogen atoms according to pH.[141][143]

2.1.3 GOLD

GOLD, an acronym for Genetic Optimisation for Ligand Docking, is software that uses genetic algorithms to dock flexible ligands to protein binding sites. It is published by the Cambridge Crystallographic Data Centre (CDCC). In GOLD, various constraints can be applied to the docking process, including distance, hydrogen bond, hydrophobic region, pharmacophore, similarity, scaffold match and interaction motif constraints. It supports protein files in PDB-file format and molecules in mol, mol2 or ent-file format.[144-146]

2.1.4 ROCS

ROCS is a program from the scientific software company OpenEye that can perform similarity screening of molecular poses by matching surface shapes as well as chemical features.[147] For shape description, ROCS uses the sum of atom-centred Gaussian functions.[148] For chemical features, it uses the features proposed by Mills and Dean in their 1996 paper "*Three-dimensional hydrogen-bond geometry and probability information from a crystal survey*".[148][149] It compares the molecules to be screened against a reference structure by rigidly superimposing them. Subsequently, it ranks them based on the Tanimoto association coefficient, which measures the number of features that both molecules have in common.[148][150][151] The Tanimoto index, also known as the Jaccard index, is a statistical index that measures the similarity of two sample sets. It is the most commonly used metric for this purpose in chemistry.[152] ROCS considers Tanimoto shape for shape comparison, Tanimoto colour for feature comparison, and the combination of these two coefficients, Tanimoto combo.[148][150][151] In screening, ROCS is typically used with pre-calculated multi-conformational libraries of the molecules to be screened. However, it can also be used to verify docking poses by comparing them with the binding pose of a molecule known to be active on the target.[148][149][153]

2.1.5 Corina

Corina is an automated three-dimensional structure generation program that generates the putative conformations of molecules. Its name stands for "COoRdINAtes". It calculates the poses based on a set of predefined rules that take into account numerical values such as bond lengths, angles and geometries, as well as experimentally confirmed data. By default, it generates a low-energy conformation that is likely to be a pose that the molecule will naturally adopt in the gas phase.[154][155]

2.1.6 OpenBabel

OpenBabel is an open-source conversion program for over 110 digital formats representing chemical structures. It supports almost all file formats commonly used in computational chemistry. Remarkably, the software has many built-in models for calculating the data needed to convert one file format to another. For example, some PDB-files do not contain bond information, which OpenBabel automatically determines from the atoms present. Another notable feature of many is the automatic conversion of two-dimensional to three-dimensional structures and vice versa.[156] OpenBabel was one of the first chemical data conversion programs, released around 1992, and has since become an important tool in computer-aided drug design, allowing the use of many programs on a single dataset that use different file formats.[157]

2.2 Services

The following chapters will present the most significant services pertinent to the present work.

2.2.1 DUDE-Z

DUDE-Z is a directory of decoy molecules used in computer-aided drug design, provided by the Shoichet and Irwin Laboratories in the Department of Pharmaceutical Chemistry at the University of California, San Francisco. It is the successor of DUDE and DUD. The database comprises property-matched decoys for 43 targets and can also generate decoys for a library based on typical molecular substructures that will render a molecule inactive.[158]

2.2.2 ChEMBL

ChEMBL is an open-access, large-scale online bioactivity database that contains primarily information about bioactive small molecules that was manually extracted from primary literature. The database is maintained by the European Bioinformatics Institute of the European Molecular Biology Laboratory. ChEMBL was designed with the specific purpose of facilitating drug discovery research, and as a result, it can be searched in an efficient manner for data pertinent to drug discovery problems. The search terms may include targets such as receptors or proteins, as well as compounds, tissues, and other relevant items. Further restrictions may be applied to the results in order to enhance the quality of the data. This enables the database to be searched for agonists of Toll-like receptor 8 that have an EC_{50} value published, as was done in the present work. Another significant attribute of ChEMBL is the capacity to download search results in file formats commonly employed in drug discovery, thus facilitating subsequent data processing and research.[159]

2.2.3 Enamine

Enamine is a chemical services provider headquartered in Kyiv, Ukraine. It synthesises fragments, building blocks for other molecules and small molecule compounds for scientific research. The company provides digital libraries of molecules that it can synthesize and that can be purchased from them. This enables scientists to search through their databases and identify potential compounds that may be active on a desired target, which can then be ordered for further testing.[162]

2.2.4 Protein Data Bank

The Protein Data Bank (PDB) is a freely available database of three-dimensional peptide, protein and nucleic acid crystal structures. These structures have been obtained mainly through X-ray diffraction or in NMR experiments, as well as through electron microscopy. The Protein Data Bank was originally established in 1971 at the Brookhaven National Laboratory. The project was continued by the Research Collaboration for Structural Bioinformatics in 1999 and subsequently merged with similar databases from Europe and Japan in 2003 to create the Worldwide Protein Databank. The PDB can be searched by several categories, including proteins, organisms, enzyme classes, molecules associated with a specific macromolecule, and identifiers from the PDB, as well as identifiers from other websites commonly used in computer-aided drug design.[160][161] The PDB-file format was specifically created to efficiently handle macromolecular data within and from the PDB. For further details, please refer to Chapter 2.3.2.3.

2.2.5 ChatGPT

ChatGPT is a chatbot developed by OpenAI that uses artificial intelligence techniques to process linguistic data, generating responses that are both helpful and natural sounding to the user. The term "GPT" stands for generative pre-trained transformer. Among its numerous applications, it can be an effective tool for writing simple texts, summarising information on various topics, engaging in simple chats for entertainment purposes, and for computer science. However, it is important to note that the data provided by ChatGPT must be verified thoroughly, as it is prone to making mistakes or generating nonsensical responses. One area in which ChatGPT has proven to be particularly useful is in programming. It can rapidly generate computer scripts for tasks that can be entered in human languages, thereby facilitating the programming process.[163-165] The majority of Python 3 scripts for various tasks that were used in the work for the present thesis were written with the assistance of ChatGPT.

2.2.6 BioRender

BioRender is an online platform designed for professionals engaged in scientific research. It provides a range of tools for the creation of scientific illustrations, charts, and other visual representations. The platform offers a plethora of templates for users to choose from, while also allowing for the creation of custom designs. The platform provides a selection of pre-designed icons and graphics that are particularly suited to scientific diagrams and illustrations.[166] Many illustrations in this thesis were created using BioRender.

2.2.7 DeepL

DeepL is a neural network-based language and text enhancement tool provided by DeepL SE. It is trained on a vast corpus of text data and employs deep learning techniques to generate translations and texts that are nuanced and natural sounding.[167] This tool was used for some translations as well as for general text enhancement throughout this work.

2.2.8 ChemDraw

ChemDraw is software from Revvity Signals that has become the standard for creating chemical structures and reaction equations. It is used to produce high quality images for publications and the like. ChemDraw is available in several versions and was used in this thesis under a student licence for various chemical formulae.[168]

2.3 Other

2.3.1 Python 3

Python 3 is a programming language that has become a de facto standard for programming applications in the field of science. This can be drawn to several advantages that come with it, making it especially suitable for scientific purposes. It is distributed under an open-source license, runs on many platforms, has a relatively straightforward syntax for non-computer scientists, excellent compatibility with other software, is easily extensible by a vast number of open-source packages that are available on the internet, and has a very active community, constantly extending the language and allowing for exchange between users. In the field of science, it is most commonly used in conjunction with NumPy, a package that extends the fundamental Python language with a multitude of mathematical operations.[169] Other important packages used in the present thesis are delineated in Chapter 2.3.2.

2.3.2 Python packages

This chapter outlines the most important scientific Python software packages and functions for this project.

2.3.2.1 RDKit

RDKit is a free software package for Python that enables the handling and processing of chemical data of various kinds. Amongst its many features, it is capable of featurising and manipulating chemical structures as well as calculating and performing a variety of operations on the molecules' physico-chemical properties and structural descriptors. It is frequently employed to implement chemical functionalities in Python-based programs and databases, as well as to curate chemical data for subsequent processing.[127][170][171]

2.3.2.1.1 RMSD in RDKit

The root mean squared deviation, or RMSD, is a function within the RDKit software suite that enables the comparison of three-dimensional molecular structures based on their shape. The algorithm superimposes the two structures in such a way that the total root mean squared distance between the atoms of the molecules is minimised. The output is the minimum value of RMSD that could be found for the overlay. A lower RMSD value indicates a greater degree of fit between the two shapes. A value of zero indicates a perfect match between the examined structure and the reference.[172] In terms of shape comparison of a structure to a reference, the function is somewhat analogous to ROCS screenings described in Chapter 2.1.4. Nevertheless, due to the differing underlying algorithms, the results may differ from those obtained using ROCS. They may, therefore, provide some insight into the shape similarity of molecules that are deemed to be similar by ROCS but not by RDKit's RMSD function, and vice versa.[147][172]

2.3.2.1.2 Maximum common substructure

The "findMCS" function in RDKit is capable of identifying the largest possible common substructure within a set of molecules. It permits the specification of certain behaviours. By default, two atoms are considered to be matching if they share the same element, and two bonds are considered to be matching if they possess the same bond type. Furthermore, in the standard settings, carbon rings can be matched with carbon chains.[173]

2.3.2.2 Scikit-learn

Scikit-learn is a free, open-source Python package that encompasses a multitude of functions pertinent to machine learning tasks. The package offers a range of machine learning algorithms, including both supervised and unsupervised learning, as well as techniques for model validation, hyperparameter tuning and statistical analysis of the models. The distinguishing feature of Scikit-learn is that it is entirely dependent on the Python programming language and is therefore accessible to non-computer scientists. Its user-friendliness makes it straightforward to use.[174]

2.3.2.2.1 GridSearchCV

GridSearchCV represents the implemented grid search function within the scikit-learn framework. The algorithm systematically explores all possible combinations of hyperparameters from a predefined grid. In order to identify the optimal parameters, the algorithm automatically performs cross-validation on the performance metrics. Unless otherwise specified, the combination with the highest value for the coefficient of determination R^2 is retained.[174]

2.3.2.2.2 Train_test_split

The scikit-learn train_test_split function enables the splitting of datasets into training and test sets. By default, the data is randomly shuffled prior to splitting it. In order to ensure reproducibility, the argument "random_state" can be specified, which controls the random shuffling. Another crucial parameter to be specified is test_size, which defines the proportion by which the data is to be split.[174]

2.3.2.2.3 GradientBoostingRegressor

GradientBoostingRegressor is the implementation of gradient boosting regression models in scikit-learn. Its hyperparameters are n_estimators (number of trees to be used), learning_rate (contribution of each tree to the final prediction), max_depth (maximum depth of each tree), min_samples_split (minimum number of samples to split a node) and min_samples_leaf (minimum number of samples in a leaf).[174] Gradient boosting regression models are described in detail in Chapter 1.2.3.2.1.1.

2.3.2.3 Matplotlib

Matplotlib is a python package developed by John D. Hunter. It contains a variety of graphical data visualisations that can be used to plot two dimensional mathematical and statistical diagrams. It frequently makes use of NumPy. The objective of Matplotlib was to streamline the plotting process, which can otherwise be a laborious and complex task.[175]

2.3.2.4 Pandas

Pandas is an open-source Python package that encompasses a multitude of functionalities related to the manipulation and processing of structured data. The creation of data frames, indexing and merging datasets, as well as data analysis are some of the most common tasks involving pandas. It is widely used across many scientific fields.[176]

2.3.2.5 NumPy

NumPy, an acronym for "Numerical Python", is a Python package that extends the language with various numerical computing operations. One of its principal characteristics is the multidimensional array object, which can be used and manipulated by the package's functions. The package includes a number of mathematical and logical operations, as well as shape manipulation, sorting and selecting, Fourier transformations, basic linear algebra, basic statistical operations, and random simulation.[177]

2.3.3 Common data formats

2.3.3.1 SMILES

The Simplified Molecular Input Line Entry System, or SMILES, is one of the standard data formats for chemical structure information. The concept was first proposed by David Weininger in 1986. SMILES is one of the most prevalent representations of molecules in computer applications, largely due to its simplicity and high information content. SMILES-strings are to be read from left to right and adhere to six fundamental rules, as elucidated in the book *Cheminformatics – Basic Concepts and Methods* by Johann Gasteiger and Thomas Engel:

1. *"Atoms are represented by their atomic symbols. Ambiguous two-letter symbols (e.g., Nb is not NB) have to be written in square brackets. Otherwise, no further letters are used.*
2. *Hydrogen atoms are omitted because they automatically saturate free valences.*
3. *Neighbouring atoms stand next to each other*
4. *Only double and triple bonds are indicated by the symbols "=" and "#". Single "-" and aromatic (no conjugated) bonds ":" should be omitted. Aromatic substructures are indicated by writing all atoms involved in lowercase letters. Individual parts of a compound are separated by a period. The period indicates that there is no connection between atoms or parts of the molecule. The arrangement is arbitrary.*
5. *Branches are represented by parentheses and may be nested.*
6. *Rings are described by closure panels (digits) to the involved atoms."*[178]

Stereochemistry is represented by the symbol @. A single @ symbol represents the clockwise arrangement (S), while two @ symbols ("@@") indicate the anticlockwise arrangement (R) of the three atoms following the symbol. The designation of E/Z-isomerism at double bonds is indicated by the use of the slash symbol "/" and the backslash symbol "\". The dashes indicate the relative position of the substituents. These can be conceived as being connected, with the highest-ranking substitutes according to the Cahn-Ingold-Prelog convention indicated by the use of dashes. F/C=C/Br is equivalent to F\C=C\Br, as the flat structure viewed from above reveals that fluorine and brome occupy the top-right and bottom-left corners, respectively, or vice versa. Therefore, they are E-isomers. Consequently, the symbol F\C=C/Br represents Z-isomers. If the two dashes are thought of being connected, they form a V-shaped structure where both substitutes are on the same side of the double bond of the flat molecule structure.[178]

2.3.3.2 SD-files

Structural information files, or SD-files, are a type of data that can contain structural information for any number of compounds. They are a format that is accepted by a wide range of computer programs, which is why they are such a common file type used in computational drug design. Essentially, they are a combination of a so-called "Molfile" and other properties of the molecules. Molfiles contain a wealth of information, including the user's name, the program used to create the file, time and date stamps, comments, whether the data in the molfile describes a two-dimensional or three-dimensional structure, the format standard used to write the information, and most importantly, a so-called "connection table". The latter specifies the type of atoms, the bonds, the three-dimensional structures where applicable, charges and stereochemistry in a specific format of blocks and tables. SD-files combine molfiles with non-structural related information such as molecular weight, various molecular descriptors, biological activity or program-related data.[179]

2.3.3.3 PDB-files

The PDB-file format was developed for the World Wide Protein Data Bank (wwPDB), which was launched in 2003. Since then, it has become the standard format for the storage and handling of digital protein information. PDB-files contain the coordinates of each atom of the represented molecules, as well as information pertaining to the secondary and tertiary structures of these molecules. The majority of structures included in PDB-files were derived from X-ray analyses or NMR experiments on authentic proteins.[180] However, they can also be modelled in part or in their entirety digitally.[70] PDB-files commonly feature proteins, their ligands, solvent and other associated molecules. The file contains information about the atoms, bonds, secondary and tertiary structures, disulfide bonds and protein chains.[181]

3 Design process

Figure 13 provides an overview of the methods used in the design of the novel TLR8 agonists, as presented in this thesis. Each step is further detailed in the corresponding prompt.

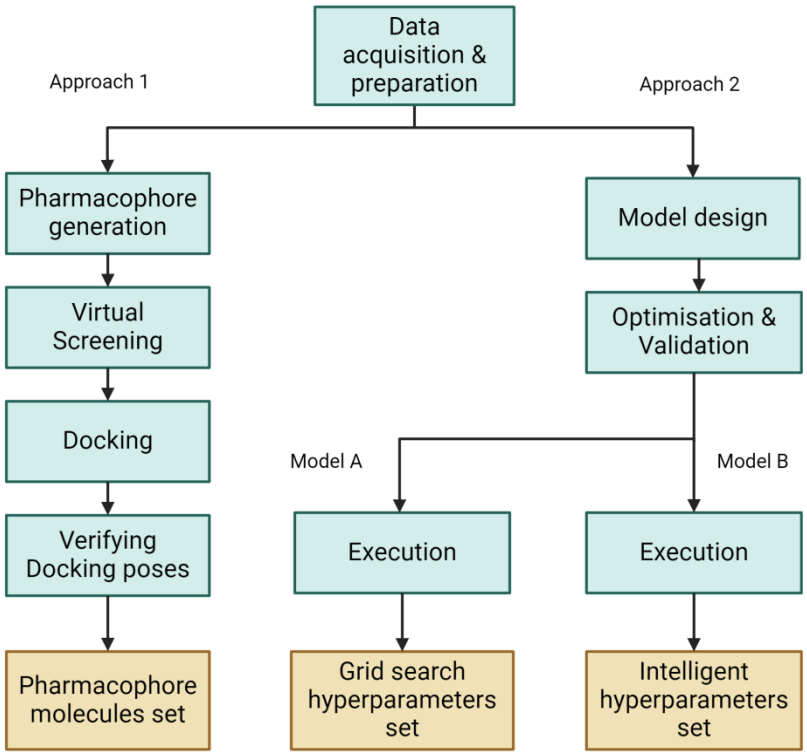


Figure 13: *Workflow for the design process. Data from the same original data set was used for all approaches. During the model optimisation of the machine learning, it was decided to train two machine learning models with different hyperparameters.*[13]

3.1 Data preparation

3.1.1. Extracting activity data from ChEMBL

In the first step, relevant data was acquired from ChEMBL, where the term "human Toll-like receptor 8" was searched. In ChEMBL, all hyperlinks to pertinent information on a target are stored on a respective page named a "target report card". The target report card for the human Toll-like receptor 8 has the ChEMBL identifier CHEMBL5805. On this page, under the section "Associated Bioactivities", an interactive pie chart on the activity types for Toll-like receptor 8 was found. It featured the segment "EC_{50}". When clicked, it redirected to a page that listed compounds with an EC_{50} value for their activity on the human TLR8. This list of compounds comprised the raw activity data that was used in the design process. The data was downloaded directly from the page as a CSV-file, which contained the information in a tabular format. The pie chart also contained a segment labelled as "Active". This data was also assessed, however, it was deemed to be of inferior quality over the "EC_{50}" data, since it included molecules that showed any kind of activity on Toll-like receptor 8, regardless of the concentration. There were some molecules that have been found in both segments.

3.1.2. Pre-processing raw activity data from ChEMBL

Upon inspection, it was discovered that the downloaded CSV-file contained compounds that were agonistically active as well as inactive compounds. As expected, no antagonists were present. Since the inactive compounds were important to later steps in the process, the raw data was split into two files, one for active molecules and one for inactive molecules:

- List 1: EC_{50} Agonist Activity List (312 molecules):
  This list was compiled from the original CSV-file downloaded from ChEMBL. All data points that were ambiguous, explicitly referred to inactive compounds or had an EC_{50} value indicating that the EC_{50} value was not precisely tested (for example, "> 10.0000 nM") were removed.

- List 2: EC_{50} No Activity List (46 molecules):
  This list comprised only those data points from the original CSV-file downloaded from ChEMBL that were unambiguous in indicating that they refer to inactive compounds.

Subsequently, the following steps were performed for both lists:

- Deletion of multiple entries:
  Multiple data points for the same compound were excluded, with only one retained per compound. For List 1, the data point with the lowest EC_{50} value was always retained.

- Verifying activity in primary literature:
  The activity and positioning of all compounds in the four lists were verified by reading all the annotated primary literature and finding unambiguous references to it. This led to the exclusion or repositioning of numerous data points.

- Standardisation of assays to induced NF-κB response:
  In order to obtain consistent data, all datapoints that did not reference an assay that had the induced change in NF-κB concentration as the measured variable were discarded.

- Deletion of RNA:
  Some of the lists included data points for RNA. The objective of this project is to identify novel small molecule compounds. These datapoints were therefore discarded.

- Deletion of unrelated columns:
  All columns except those containing SMILES-strings, the ChEMBL molecule identifier, EC_{50} values and units for EC_{50} values were removed.

With regards to further steps in the design process, three more CSV-files were created from the processed EC_{50} Agonist Activity List (List 1):

- List 3: EC_{50} Agonist Activity over 500 nM List (169 molecules):
  This list included only datapoints of compounds that had an EC_{50} value of over 500 nM.

- List 4: EC_{50} Agonist Activity under 500 nM List (143 molecules):
  This list included only datapoints of compounds that had an EC_{50} value of 500 nM or less.

- List 5: EC_{50} Agonist Activity under 50 nM List (96 molecules):
  This list included only datapoints of compounds that had an EC_{50} value of 50 nM or less.

As the data from List 5 was to be used to generate the pharmacophores for the screening and docking approach, property-matched decoys were generated from this data set using DUDE-Z for pharmacophore validation. The SMILES-strings generated by DUDE-Z were then converted to a CSV-file.

- List 6: EC_{50} Agonist Activity under 50 nM decoys List (200 molecules):
  This list contained property-matched decoys that were created from the SMILES-strings in List 5 using DUDE-Z.

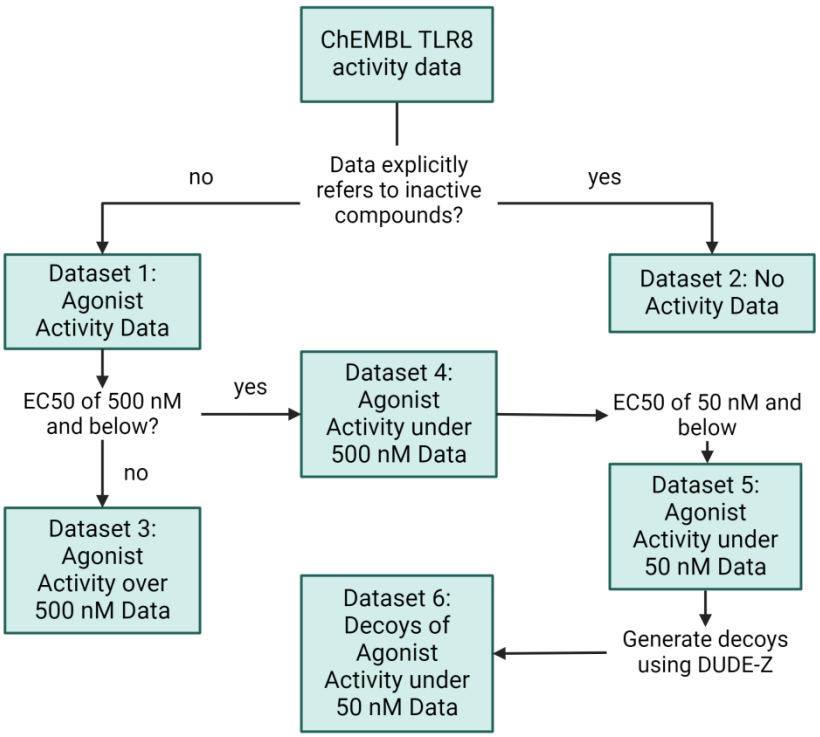


Figure 14: *Splitting of the pre-processed data set downloaded from ChEMBL. The source data set was split into different smaller data sets, which later served different purposes.*[13]

3.2 Approach 1: Pharmacophore-based screening and docking

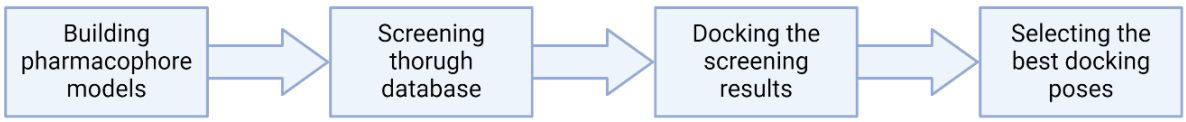


Figure 15: *Workflow for approach 1. These are the general steps that were carried out during the approach.*[13]

3.2.1 Data preparation for pharmacophore generation

3.2.1.1 Creating SD-files with the correct protonation and conformation

In order to study their behaviour in the environment of the human body, the molecules were researched with the correct overall charge that they are assumed to have under these conditions. The protonation step was also important for the correct calculation of the pharmacophore. Since the compounds would most likely be transported throughout the body in the bloodstream, where pH is around 7.4, the first step in creating molecule libraries was to calculate the correct charge that the compounds would have at that pH.[182]

The following steps were carried out consistently for all pre-processed lists. The lists, which were still in the CSV-file format, were imported into the database viewer of MOE. Next, the "Wash" function was selected, where the compounds were set to protonate as if they were in an environment with a pH of 7.4. The protonation was performed and visually verified in the database viewer. Once complete, the correctly protonated compounds were saved in the SD-file format. This allowed the charges and three-dimensional conformations to be saved along with the molecule structures.

The next step involved processing all SD-files with Corina to obtain the correct three-dimensional conformations. This was done in order to obtain reasonable initial conformations, as otherwise the generation of multi-conformational libraries can lead to the generation of relatively unlikely, twisted or stressed conformations.[183]

Corina generated the conformation of each molecule that is the most energetically favourable. The outcome of this pre-processing of the activity data was the generation of seven SD-files, each containing molecules with the correct protonation and three-dimensional conformation at the energy minimum.

3.2.1.2 Generation of multi-conformational molecule libraries

The generation of pharmacophores and the subsequent virtual screenings are typically conducted on multi-conformational molecule libraries. This approach is employed to ensure comprehensive coverage of the conformational space. Given the inherent limitations of sampling, it is not possible to exhaustively cover all possible conformations. Consequently, several different conformations of the molecules are typically considered.[184] This is crucial because, in the context of the ligand-based approach, the exact binding poses of each molecule remain unknown at this stage. In ligand-based pharmacophore generation, only the features that were found the most often are included in the pharmacophore, thus automatically selecting the correct pose for each molecule. Multi-conformational molecule libraries of the ldb-file format were created for all seven SD-files using the idbgen function of LigandScout in the Ubuntu command line.

The command was fitted to process all SD-files in the "icon-fast" setting, which is the standard configuration for high-throughput screenings and generates 25 conformations per molecule. This resulted in the generation of six multi-conformational ldb-files. In accordance with the original lists from which they were derived, they will henceforth be referred to by numbers.

- Library 1: EC_{50} Agonist Activity Library
- Library 2: EC_{50} No Activity Library
- Library 3: EC_{50} Agonist Activity over 500 nM Library
- Library 4: EC_{50} Agonist Activity under 500 nM Library
- Library 5: EC_{50} Agonist Activity under 50 nM Library
- Library 6: EC_{50} Agonist Activity under 50 nM decoys Library

3.2.2 Pharmacophore modelling

3.2.2.1 Creation of a ligand-based merged feature pharmacophore

The pharmacophores were generated using the pharmacophore generation function in LigandScout. The objective of this step is to create pharmacophores that are able to identify highly agonistically active compounds. Therefore, the pharmacophores were trained from library 5, which contains only compounds with an EC_{50} value of 50 nM or lower.

Library 5 was loaded into the ligand-based view of LigandScout. In the "Calculate Pharmacophore" dialogue, the algorithm to be employed in the generation of pharmacophores had to be specified. The available options were either shared or merged pharmacophores. In LigandScout, the principal distinction between a shared and a merged pharmacophore is that the algorithm for the former calculates the average of the molecular interactions based on a predefined set of rules, whereas the latter has fewer constraints, allowing researchers to prioritise certain interactions manually based on their theory and expertise.[140] The merged pharmacophore generation proved to generate better results for this dataset, thus it was decided to use this algorithm.

Choosing a merged pharmacophore requires the user to also define a number of omitted features. This refers to the number of features that are permitted to deviate from a perfect match with all molecules.[140] A number of ten omitted features were identified as the most effective in this case. All other settings were left on the default setting and the merged pharmacophore was calculated.

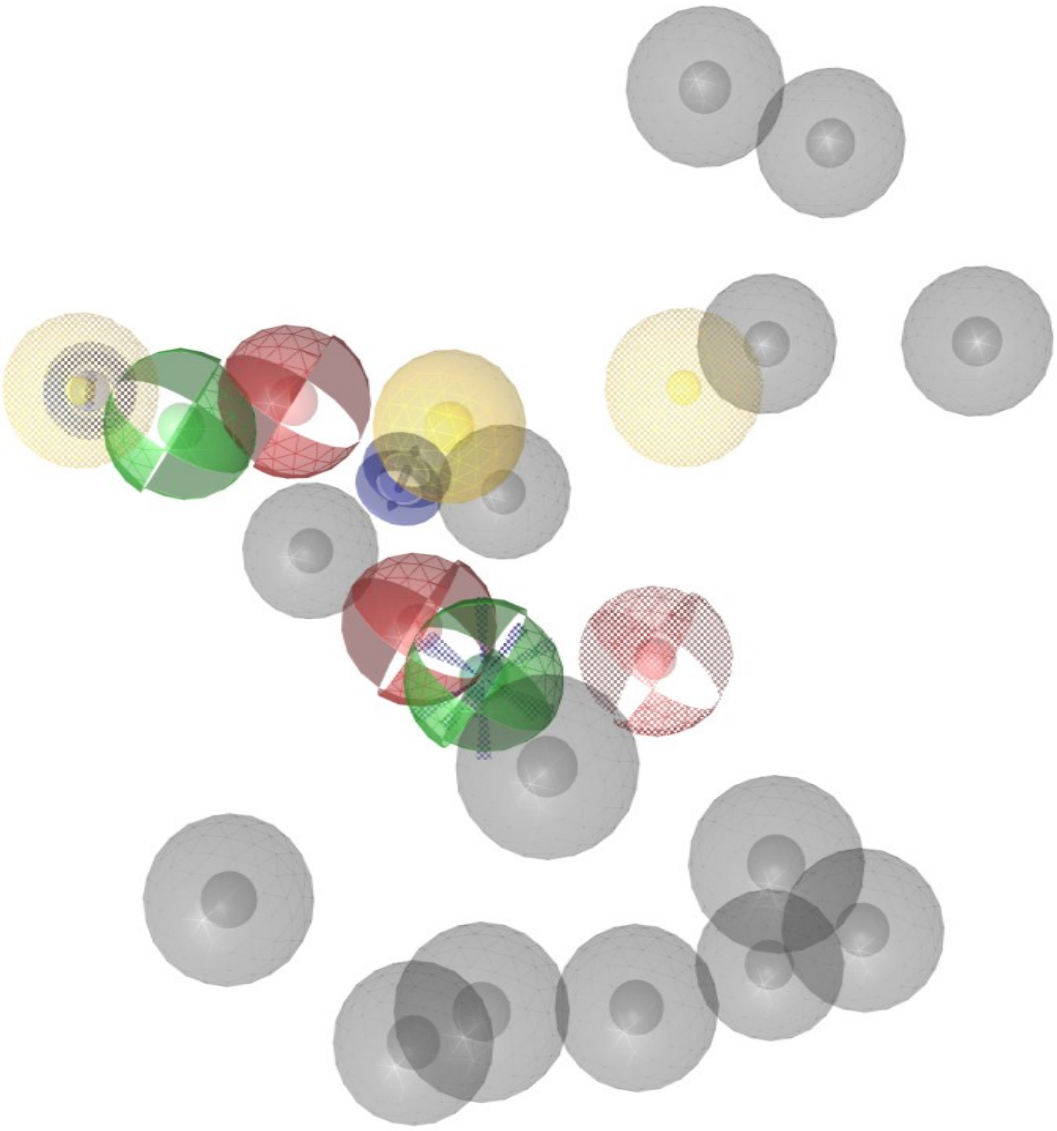


Figure 16: *The merged feature pharmacophore that was obtained, rendered by LigandScout. All other pharmacophores were derived from this model. For a legend of the interaction features, please refer to Chapter 2.1.1.1. In LigandScout, optional features are displayed in a checkerboard pattern.*

3.2.2.2 Refining the pharmacophore model

The performance of all pharmacophores was evaluated based on the ratio of true to false positives and the receiver operator characteristic curve (ROC curve). These curves are widely used in computational drug design and provide a visual representation of the relationship between the true positive rate (sensitivity) on the Y-axis and the false positive rate (1 – specificity) on the X-axis.[81][185] The dotted line indicates the border of insignificance, meaning that a curve below this line indicates that the model does not perform better than randomly choosing molecules from the dataset. A curve above the dotted line indicates a statistically significant discrimination between active and inactive compounds, with the degree of separation proportional to the area under the curve (AUC).[185][186] The ideal ROC curve passes through the top left corner of the plot, maximising the AUC and thus the predictive capability of the model.[187] In LigandScout, in the bottom right corner of the plots for the ROC curves, the AUC and the enrichment factor are noted.

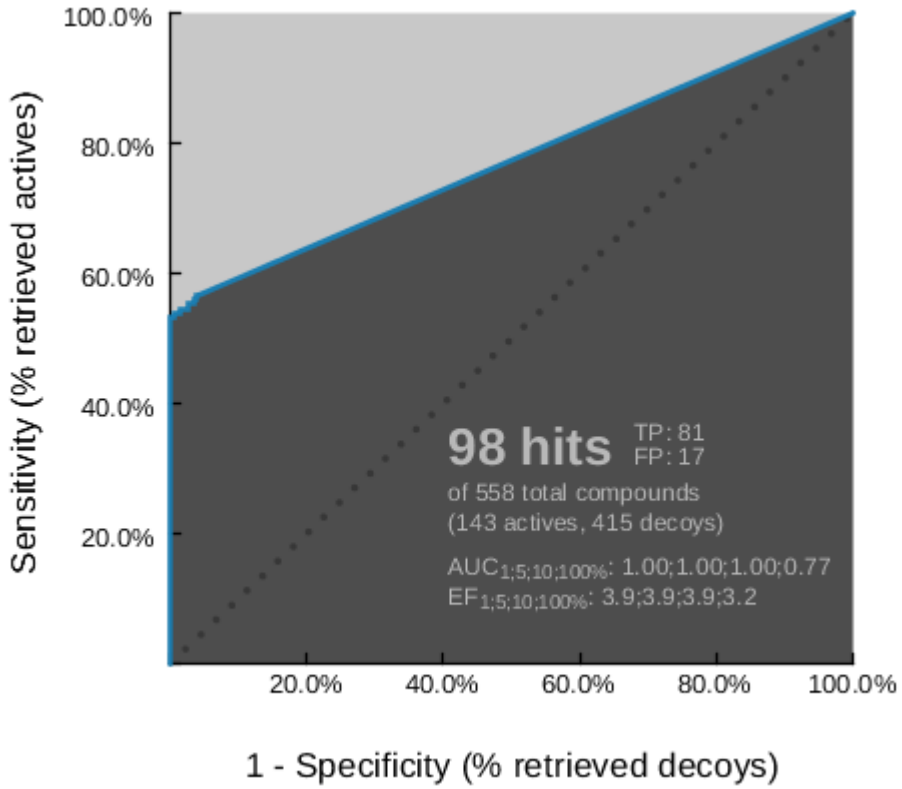


Figure 17: *ROC curve of the merged feature pharmacophore obtained in Chapter 3.2.2.1.*

The merged pharmacophore of library 5 had to be further refined for use in virtual screenings as it had too many features. The model overfitted the training data, which limited its generalisability to unseen data. A good pharmacophore should have as much predictive capability with as few features as possible as described in Chapter 1.2.2.1. Consequently, it was necessary to remove features from the pharmacophore and then validate the model to obtain an efficient model. This was achieved through a process of trial and error, with the performance of the model evaluated after each manual removal or change of a feature in the validation screenings. In multiple iterations, pharmacophore features were removed or modified, performance was evaluated, and each refined pharmacophore version was saved until satisfactory validation metrics were achieved.

For the validation screenings, library 4 (EC_{50} of 500 nM or less) was selected as the molecules that should be identified as active. The inactives were comprised of library 2 (no activity), library 3 (EC_{50} of over 500 nM) and library 6 (decoys of library 5). Library 3 was included in the dataset of inactive compounds in order to optimise the model to not only discern between active and inactive compounds but also favour compounds with good EC_{50} values. Table 1 provides an overview of the datasets used in pharmacophore validation:

Active	**Inactive**
Library 4, EC_{50} of 500 nM or less	Library 2, No Activity
	Library 3, EC_{50} of over 500 nM
	Library 6, Decoys of library 5 (molecules with an EC_{50} of 50nM or less)

Table 1: *Datasets used in pharmacophore validation*

Considerations that were made during the design process include:

- More lipophilic interactions improve affinity but reduce solubility
- The molecules should interact in at least three points with the receptor in order to stabilise the binding pose
- A good balance between directed and undirected interactions was desired
- Optional features (the spheres with the chequerboard pattern in Figure 16) are computationally expensive but have little impact on the performance.[188]
- Streamlined pharmacophores should have only as many interactions as necessary

Exclusion volumes were not modified as they usually indicate space already occupied by the receptor in concordance with Chapter 1.2.2.1. As property-matched decoys are only assumed to be inactive, there is still a chance they could be active.[84] Consequently, they were used with caution. In order to ensure the generation of diverse screening results, three refined pharmacophores were obtained in total during this step. Each of them had certain advantages, which are outlined in the following chapter.

3.2.2.3 Final pharmacophore selection for virtual screening

The three best-performing versions of the refined pharmacophore were selected for virtual screening:

Pharmacophore 1 ("TP78-FP0"):

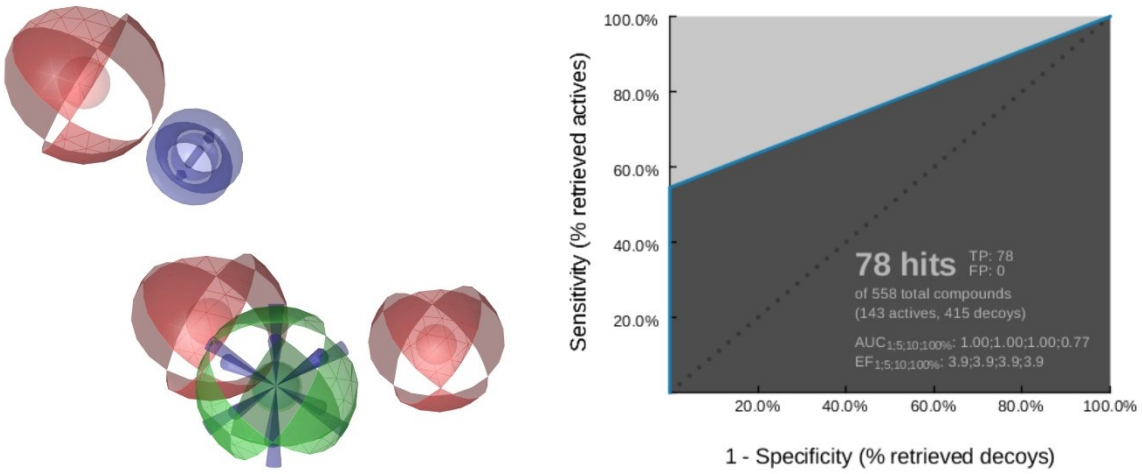


Figure 18: *Pharmacophore 1 and its ROC curve, rendered in LigandScout. The pharmacophore is presented without the exclusion volumes. For a comprehensive illustration, please refer* **to appendix Ia.**

This pharmacophore is rather restrictive, having identified only 78 of the 143 molecules in library 4 as active. However, it also has no false positives. One notable aspect of the validation screening of this pharmacophore is that only two molecule scaffolds remained in the results. This was not a reason to discard it, however, as there may be other molecule scaffolds that fit this pharmacophore but were not included in the validation screening data.

Pharmacophore 2 ("TP81-FP6"):

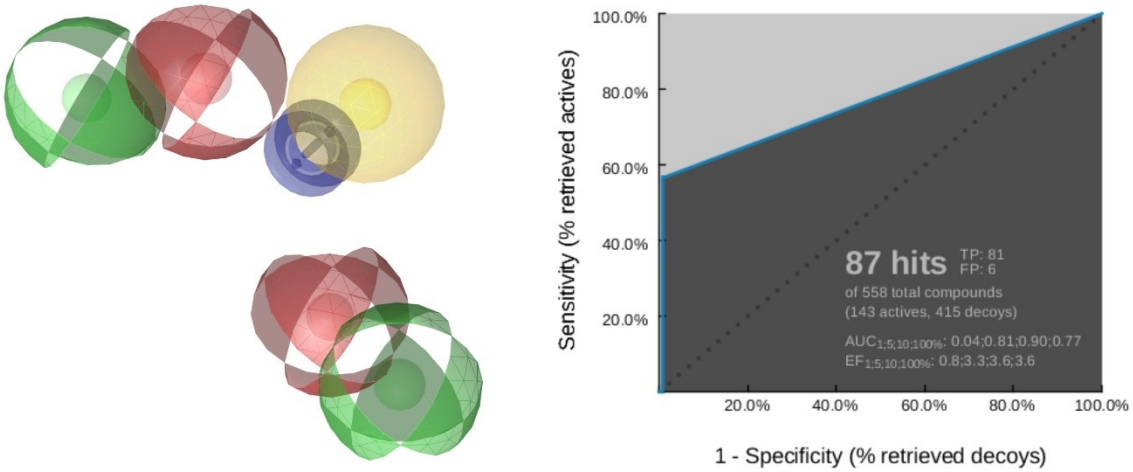


Figure 19: *Pharmacophore 2 and its ROC curve, rendered in LigandScout. The pharmacophore is presented without the exclusion volumes. For a comprehensive illustration, please refer* to appendix Ia.

This pharmacophore was less restrictive, allowing for some false positives. It may be considered to have a better balance between correctly identifying true positives and allowing for unconventional binding modes than pharmacophore 1. This pharmacophore contains all the interactions of the pharmacophore obtained in Chapter 3.2.2.1 but without all the optional features.

Pharmacophore 3 ("TP125-FP45"):

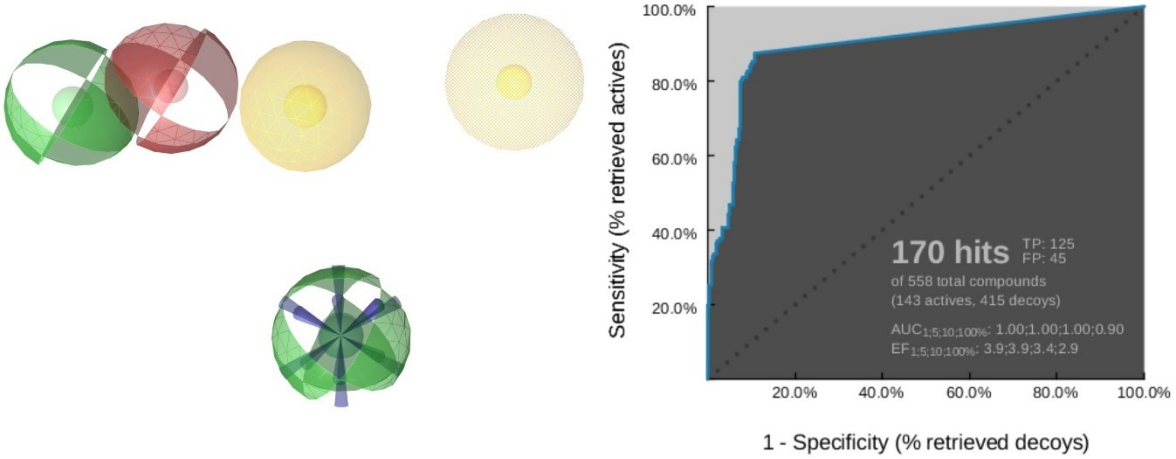


Figure 20: *Pharmacophore 3 and its ROC curve, rendered in LigandScout. The pharmacophore is presented without the exclusion volumes. For a comprehensive illustration, please refer* to appendix Ia.

This pharmacophore is the least restrictive of the three, with a high number of false positives but also the highest number of true positives. It was assumed that it could potentially find more diverse binding solutions in the virtual screening.

3.2.3 Screening through Enamine's screening collection database

A virtual screening was conducted through the Enamine Screening Collection, version of May 2022. This database of around four million small molecules was already available as an ldb-format library with 25 conformations per molecule. The decision was made to select this particular library from chemical service provider Enamine, as it would allow for the final hit selection to be ordered for subsequent in vitro testing. Each pharmacophore was screened against the library separately.

As with the validation screenings for the pharmacophores, this screening was conducted using LigandScout's iscreen function with default settings in command line. In this setting, the scoring function is set to "pharmacophore fit". This means that only the chemical feature overlap is used to assess whether there is a hit. In addition, the number of omitted features is set to 0, which means that the interactions of the conformation of a molecule under consideration must perfectly match the pharmacophore. Omitted features are explained in Chapter 3.2.2.1. In addition, in this setting the program will include the pharmacophore exclusion volumes in the screening and will not consider further conformations of the same molecule after the first matching conformation of a molecule has been found.[140] The results were saved in SD-files and yielded 6,070 hits for pharmacophore 1, 13,786 hits for pharmacophore 2 and 34,387 hits for pharmacophore 3. The screening hits represent molecules from Enamine's database that exhibit similar molecular interaction features to those of the pharmacophores used in the respective screening.

3.2.4 Docking the screening hits

To further investigate the activity of the screening hits, the SD-files obtained in the previous step were virtually docked into a crystal structure of Toll-like receptor 8. This process yielded SD-files with docking poses, which were subsequently reviewed using several methods in order to identify the most active molecules.

3.2.4.1 Preparation of the protein structure

The Protein Data Bank was searched for a structure with high resolution in order to ensure the accuracy of the subsequent docking and calculations. The crystal structure with the tag "3W3N" was selected and downloaded as the basis for the protein model as it had a high resolution of 2.1 Å.[14][50] However, in addition to the receptor, the file also contained several sugar molecules, amino acids and the known dual Toll-like receptor 7/8 agonist resiquimod in its binding pocket. Resiquimod is one of the more well-known actives of TLR8 and was also included in the pharmacophore training data. Like most agonists, it binds the receptor in its uridine binding site.[48] Since the docking program used can automatically detect the correct binding pocket from ligands present in the PDB-file, all molecules were deleted from the file except the receptor itself and resiquimod. This was achieved by loading the structure into MOE and using its sequence editor to remove any unwanted molecules.

Crystal structures, such as PDB-files, frequently contain proteins with missing loops. There are several reasons for this, but often it is simply due to measurement inaccuracies. However, in many cases these loops are important for the correct function of the protein. In order to examine the proteins in their entirety and in their functional state, as well as for reasons of comprehensibility, such loops should, if possible, be calculated by homology modelling and included in the structure.[189][190] Hence, the structure was examined where some missing loops were identified and modelled using MOE's "Loop Modeler" function, accessible within the sequence editor.

The algorithm automatically searched the Protein Data Bank for other crystal structures of Toll-like receptor 8 and presented several solutions for consideration. The missing loops were created according to the Protein Data Bank structure 5N6S.[191][192] Furthermore, the structure was examined for residues of the Z-loop, which prevents TLR8 dimerisation as described in Chapter 1.1.2.2. The remaining residues of this sequence were removed using MOE's sequence editor. Once all unwanted structures had been removed from the PDB-file, MOE's QuickPrep function was employed. This application automatically corrects structural issues using MOE's structure preparation tool, adds explicit hydrogens and assigns protonation states to the structure using the Protonate 3D function, adds tethers to the heavy atoms within the receptor, fixes distant atoms and performs an energy minimisation. The protonation step was carried out at a pH of 7.4 and a temperature of 310 Kelvin, which closely resembles the conditions found in the human body.[182][193] All other settings were left at their default values. Once this step was complete, the structure was ready for docking and was saved as a PDB-file.

3.2.4.2 Docking configuration

The docking process was conducted using GOLD (Genetic Optimization for Ligand Docking), a software tool released by the Cambridge Crystallographic Data Centre (CCDC).[146] The software is described in more detail in Chapter 2.1.3. While the dockings were subsequently executed in the Ubuntu command line, the graphical user interface was used for the configuration file setup.

The binding site was defined as that of the ligand, resiquimod, in the PDB-file. The size of the binding pocket was left on the default of 10 Å. A ligand file was chosen and the number of genetic algorithm runs ("GA runs") set to 25. This was done to obtain multiple conformations for each molecule, ensuring that the correct position is among them, as the algorithm involves an element of randomness.[194] As a scoring function, the default chemplp score was used.[195] It was decided that early termination would not be permitted, meaning that the program would have to calculate 25 conformations for each molecule, even if a good fit was already found. Furthermore, the "generate diverse solutions" option was activated. This option allows the generation of more diverse binding solutions and orientations.[194] The latter two options were chosen, as the computational resources were available and it was preferred to filter for promising docking poses with more accurate methods. Finally, all the docking solutions were saved into a single output file. The docking was initiated for the screening results of pharmacophore 1, pharmacophore 2 and pharmacophore 3 separately.

3.2.4.3 Execution of the dockings

Three dockings were conducted, one for each SD-file obtained in Chapter 3.2.3. Once the process was complete, three SD-files were generated, each containing the compiled docking results for a specific screening result file. The files were named after the pharmacophore used in the screening:

- Set 1: Pharmacophore 1 docking results (60,690 docking poses)
- Set 2: Pharmacophore 2 docking results (13,7860 docking poses)
- Set 3: Pharmacophore 3 docking results (34,3840 docking poses)

3.2.4.4 Filtering docking results

Once the docking solutions for all screening hits had been obtained, the docking poses had to be carefully searched for the most plausible binding modes. This process involved eliminating all but 11 molecules in total across all three sets. The objective was to apply constraints that could be used to identify molecules that were likely to be active in vivo. The three sets of docking poses, each corresponding to a different pharmacophore used in the virtual screenings, were subjected to the same filtering methods. However, all three files were handled separately to ensure that binding poses of all three pharmacophores were included in the final hit selection for diverse binding solutions.

3.2.4.4.1 Filter script for PAINS

Pan-assay interference compounds, or PAINS, are compounds that have substructures that are known to cause false positive results in many biological assays involved in drug discovery.[196] As a result, they are usually removed from the process in order to avoid a waste of resources. A script to remove these structures from an SD-file was readily available at the computational drug design lab of FU Berlin. First, the SMILES-strings of the PAINS in the docking results were ascertained by using the script on all screening results. Subsequently, all entries with the concerned SMILES from all docking results were removed. The script used is provided in appendix Ib. After filtering out PAINS, there were 59,490 docking poses in set 1, 12,1280 docking poses in set 2 and 33,3920 docking poses in set 3.

3.2.4.4.2 Filtering by interaction features in LigandScout

The next step involved researching the binding modes of known agonists in order to gain a better understanding of the structure-activity relationship for the activation of Toll-like receptor 8. Selected crystal structures of TLR8 with known agonists found in the Protein Data Bank were examined. Their PDB-files were opened in LigandScout, and pharmacophores of their interactions with the receptor were created. The accompanying primary literature was reviewed. It was established that the majority of agonists for Toll-like receptor 8 exhibited at least two hydrophobic interactions and one hydrogen bond donor interaction with the receptor.[14][197-200] Therefore, these restrictions were applied to all sets of the docking results.

The prepared protein was opened in LigandScout and the SD-file containing the docking results was inserted using the insert function. The "Calculate Interacting Feature Counts" function under the "Molecule" menu was used to calculate the interacting features. This enabled the selection and removal of entries that did not meet the aforementioned criteria. Subsequently, the energetically most favourable position in the receptor for each pose was calculated using the MMFF94 energy minimisation function in LigandScout. The interacting features were calculated once more, and all entries were deleted that did not show at least two hydrophobic interactions and one hydrogen bond donor interaction with the receptor. Furthermore, all entries for molecules with more than ten rotatable bonds were deleted, as a high number results in an entropy loss and therefore renders the binding pose rather unlikely to occur in reality in concordance to the theory of Chapter 1.2.2.5.1.2.

Following the alteration of the original docking poses during energy minimisation, they had to be restored for further filter steps. Therefore, the SD-files from before and after this step were compared. All entries from the SD-file before energy minimisation were removed that were not also present in the file that had undergone energy minimisation with subsequent removal of entries. This yielded SD-files that contained the original poses but without all entries that did not show the required interactions before or after energy minimisation. After completing this step, there were 14,925 docking poses left in set 1, 40,968 docking poses in set 2 and 90,126 docking poses in set 3.

3.2.4.4.3 Alignment to the original pharmacophore

In an effort to reduce the number of docking poses, it was decided to align each set of docking poses to the pharmacophore that was used to create the screening results that had been used in its docking. This was achieved by utilising the "Align and Calculate Score to Shown Pharmacophore" function within LigandScout's structure-based view. Each pharmacophore was opened, the corresponding set of docking poses inserted, and the score calculated. All docking poses that showed an alignment score of 0 or less, indicating that they could not be aligned to the pharmacophore at all, were discarded. This method yielded SD-files with 13,378 docking poses for set 1, 38,790 for set 2 and 77,329 for set 3.

3.2.4.4.4 Alignment to a pharmacophore of known agonists

The principle behind this step was to filter the docking poses using a pharmacophore with features of selected, known and established TLR8 agonists. For this purpose, the PDB-files that were reviewed for Chapter 3.2.4.4.2 were opened and all of the agonist poses were copied to LigandScout's alignment perspective. Once all the molecules had been aligned, several ligand-based pharmacophores were created.

The shared pharmacophore of all agonists present in PDB-files 3W3L, 3W3M and 3W3N was selected as it produced the least restrictive and most reasonable results. The files contain multiple binding poses of the most common Toll-like receptor 8 agonist, resiquimod, and stem from the same publication.[14][50][201][202] As in Chapter 3.2.4.4.3, the files with the remaining docking poses were aligned to these pharmacophores, their alignment score calculated and those entries discarded that could not be aligned at all, having an alignment score of 0 or less. As the objective was to identify compounds with similar interactions to resiquimod but potentially different structures, exclusion volumes were not included in the pharmacophore. Afterwards, 1,641 docking poses remained in set 1, 5,777 in set 2 and 10,511 in set 3.

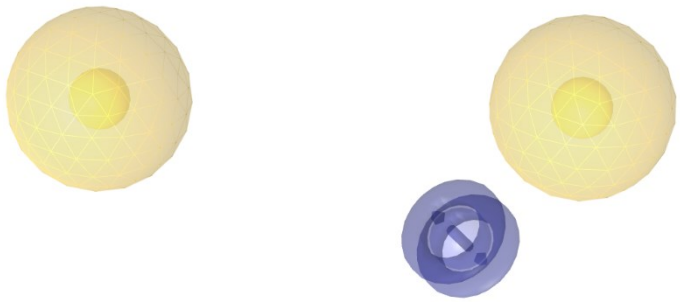


Figure 21: *The shared feature pharmacophore of Resiquimod, rendered in LigandScout.*

3.2.4.4.5 Filtering using PLIF in MOE

In the last two chapters, the three-dimensional coordinates of the docking poses in the SD-files were altered. As the docking poses were going to be aligned to the crystal structure of the protein in this step, the original positions had to be restored in a manner similar to that described in Chapter 3.2.4.4.2. This was achieved by comparing the entries of the files obtained from Chapter 3.2.4.4.4 to the files originally obtained in the dockings and deleting all entries from the original docking files that were not also present in the processed SD-files.

PLIF stands for "protein-ligand interaction fingerprint" and enables the display of interactions between compounds and specific amino acids in proteins, as well as the filtering of molecules that interact with specific amino acids of interest. During the research process, it was discovered that hydrophobic interactions with the protein in a specific hydrophobic pocket are crucial for the activation of Toll-like receptor 8. This hydrophobic pocket is formed by the amino acids Phe346, Tyr348, Gly376, Val378, Ile403, Phe405, Gly572 and Val573.[14] In order to avoid imposing undue restrictions, it was decided to filter for molecules that showed at least one interaction with either of these amino acid residues.

The prepared protein structure, which was also used in the docking process, was opened in MOE. Each SD-file containing the remaining docking poses was then opened in the Database Viewer in MOE separately. Under the dropdown menu labelled "calculate" in the Database Viewer, the PLIF function was summoned by clicking the "Generate..." button. In this instance, the protein was set to be taken from MOE and the "Prepare" button was clicked. Once prepared, under the "Protein" tab, the "Surface" option was activated and the model calculated by clicking the "Generate" button. The results could then be browsed. For all three sets, the only amino acids of the hydrophobic pocket that PLIF was able to find interactions for were Phe405, Val378 and Tyr348. It was decided to remove all entries from the sets of docking poses that did not show at least one hydrophobic interaction with one of these amino acids. Only one hydrophobic interaction was chosen as a criterion because the interactions shown were not entirely analogous to those seen in LigandScout. This is not surprising, however, as the programs have different algorithms and therefore show different results. With this filter applied, there were 829 docking poses left in set 1, 3,715 in set 2 and 4,739 in set 3.

3.2.4.4.5.1 Second run of PLIF after energy minimization

As a further restriction, it was decided to repeat the PLIF interaction screening with the calculated energetically most favourable position. The prepared protein structure, which was previously used in the docking process, was opened in LigandScout’s structure-based view. The SD-files containing the remaining docking poses were imported into the software using the "Insert" function under the "Molecule" tab. The conformational energy was then minimised using the MMFF94 algorithm, and the changes were saved. This was carried out for each SD-file, with the remaining docking poses treated separately. As previously outlined in Chapter 3.2.4.4.5, the prepared crystal structure of the protein used in docking was opened in MOE and PLIF executed. Once more, only the docking poses that demonstrated at least one hydrophobic interaction with either Phe405, Val378 or Tyr348 were retained, while all others were discarded. Again, the three-dimensional coordinates of the SD-files containing the remaining docking poses were restored by creating SD-files containing only the entries from the remaining docking poses from the original docking result data, as described in Chapter 3.2.4.4.5. This constraint left 404 docking poses for set 1, 2,168 for set 2 and 2,171 for set 3.

3.2.4.4.6 ROCS

As the number of entries in the SD-files was still too high for visual inspection, it was decided to compare the shape of the docking poses with the most popular Toll-like receptor 8 agonist in research, resiquimod. This was done on the assumption that matching shapes with this agonist would improve shape complementarity with the binding pocket. Subsequently, ROCS screenings were conducted for the remaining docking poses.

An SD-file of resiquimod was created for reference purposes. A CSV-file containing a single entry for resiquimod with its SMILES-string was generated from the EC_{50} activity data CSV-file downloaded from ChEMBL from Chapter 3.1, as it was present there. This CSV-file was subsequently loaded into MOE and its protonation in the environment of the body calculated in accordance with the procedure described in Chapter 3.1.2. This structure was saved as an SD-file. Subsequently, the pharmacophore created from multiple resiquimod poses in different PDB-files in Chapter 3.2.4.4.4 was loaded into LigandScout's structure-based view. The SD-file of the resiquimod molecule was then inserted and aligned with the pharmacophore using the alignment function under "Molecule". The changes were then saved to the SD-file of resiquimod, resulting in a file containing only resiquimod in a pose aligned to the pharmacophore.

For ROCS, the data used had to be in the oeb-format. Consequently, the SD-files containing the docking poses as well as the one containing only resiquimod were converted using OpenBabel. The ROCS screening was performed, yielding three oeb-files containing the remaining docking poses that exhibited shape similarity to the reference resiquimod molecule. These files were converted back into SD-format using OpenBabel.[147][156] After ROCS, there were 404 docking poses in set 1 and 500 in both set 2 and set 3.

3.2.4.4.7 Filtering by RMSD compared to Resiquimod

In order to reduce the number of remaining docking poses, it was decided to compare their substructures with that of resiquimod, since it is a well-known and established agonist on TLR8. This was done in RDKit by aligning each docking pose with the resiquimod SD-file obtained in Section 3.2.4.4.4, searching for a common substructure between resiquimod and each docking pose and then calculating the RMSD values between their atoms. (See Chapter 2.3.2.1.1) After some trials, a common substructure of seven molecules proved to give the best results as it was neither too strict nor too weak a constraint. The script that was used returned a list of unique names of docking poses that had an RMSD value of 0, indicating a very good match of the common substructure between them and resiquimod. This was the case for the majority of molecules. A further Python 3 script was used to delete all docking poses from the SD-files containing the remaining docking poses from the previous chapter that were not included in the list generated in this step, thus discarding all molecules with an RMSD greater than 0. Again, this step was performed separately for all three SD-files with remaining docking poses. The script used to perform the RMSD analysis is provided in appendix Ic. With the completion of this step, there were 298 docking poses in set 1, 411 in set 2 and 391 in set 3.

3.2.4.4.8 First visual inspection

In this step, the docking poses were subject to a visual review, with any that did not meet the required standard removed based on personal judgement. This was completed in LigandScout's structure-based view. The protein crystal structure used in the docking process was opened. The hydrophobic pocket that had been identified as important in Toll-like receptor 8 activation in Chapter 3.2.4.4.5 was highlighted by changing its render style. The SD-files containing the remaining docking poses were inserted separately and each docking pose was thoroughly inspected with respect to the following considerations:

- Good shape complementarity with the hydrophobic pocket is desired.
- Hydrophobic interactions with the amino acids of the hydrophobic pocket that were found to be important for receptor activation are desired.
- Directed interactions preferably close to the hydrophobic moiety of the molecule that protrudes into the hydrophobic pocket are desired.
- At least two directed interactions in different parts of the molecule should be present.
- No large portions in a molecule without any kind of purpose or interaction should be present.
- No cisamide structures should be present, as they are known to be unstable in that position.[203]
- Flourine atoms do not form hydrogen bonds as well as aromatic structures do.
- Nitril groups are only weak hydrogen bond acceptors.
- Steric hinderances should be avoided.

This first visual inspection eliminated all but 43 docking poses in set 1, 59 in set 2 and 71 in set 3.

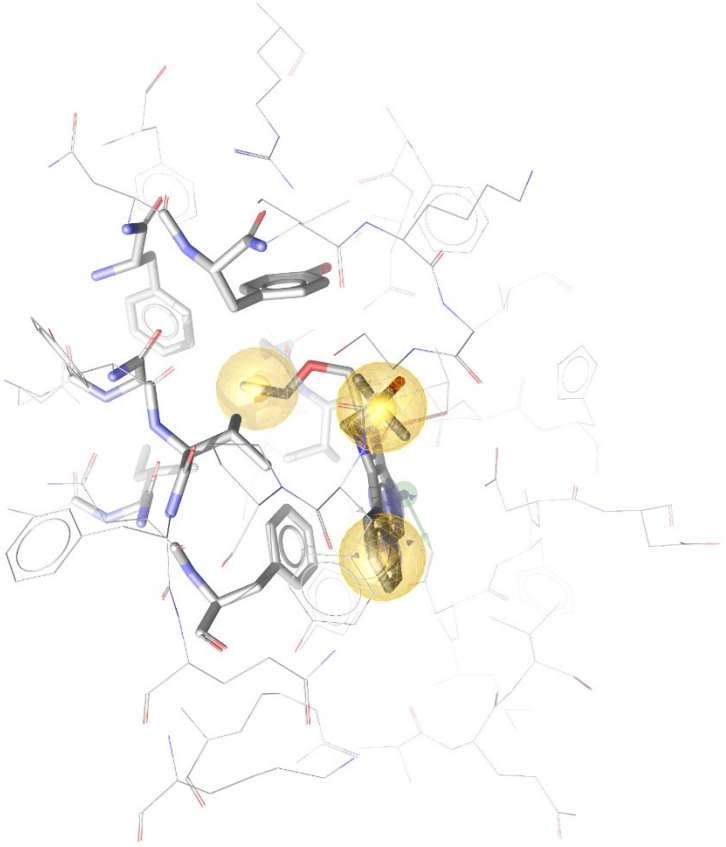


Figure 22: *An example of a molecular binding pose. The molecule is resiquimod in the uridine binding site of TLR8. The residues forming the hydrophobic pocket important for receptor activation are rendered as sticks. For a legend of the interaction features, please refer to Chapter 2.1.1.1. Adopted from Tanji et al., 2013 (PDB: 3W3N), rendered in LigandScout.*[14][50]

3.2.4.4.9 Consensus docking

Following the initial visual inspection, another method of identifying the most active compounds within the remaining docking poses was employed: consensus docking. This refers to the process of conducting dockings with other scoring functions than the one that was used earlier and evaluating which compounds produced plausible docking poses under all scoring functions.[204] To achieve this, the SD-files with the remaining docking poses were docked into the prepared protein crystal structure as described in Chapter 3.2.4. However, this time the scoring function was changed. For each file of remaining docking poses from the original docking, two dockings were performed: one with the scoring function set to goldscore and one with the scoring function set to chemscore.[146][205] The results were then visually inspected along with the original SD-files of remaining docking poses to determine which compounds produced plausible docking poses in all three attempts. Therefore, there were now nine SD-files to be visually inspected in the next step.

- Docking poses of set 1, plp score
- Docking poses of set 1, chemscore
- Docking poses of set 1, goldscore
- Docking poses of set 2, plp score
- Docking poses of set 2, chemscore
- Docking poses of set 2, goldscore
- Docking poses of set 3, plp score
- Docking poses of set 3, chemscore
- Docking poses of set 3, goldscore

It was only discovered at a later stage in the process that an error had been made at this step. It was not appropriate to perform the consensus docking with the SD-files of remaining docking poses from the original docking. This was because several molecules had more than one docking pose in these SD-files. This resulted in a disproportionate number of docking poses for the molecules in question in the newly obtained docking results, which introduced a bias in the visual inspection. This issue was duly noted and the relevant molecules in the final selection were subjected to further scrutiny to ensure that the effects were mitigated.

For the SD-files obtained from the dockings using the chemscore and goldscore scoring function, the interacting features to the protein were calculated in accordance with the methodology described in Chapter 3.2.4.4.2. Afterwards, like in Chapter 3.2.4.4.2, all entries that did not show at least two hydrophobic interactions and one hydrogen bond donor interaction were deleted.

3.2.4.4.10 Final visual inspection

The final step in identifying the most promising compounds was to visually inspect all nine SD-files obtained in the previous step and manually select the best binding modes. The same considerations as in Chapter 3.2.4.4.8 were applied, but with greater rigour. Most remaining docking poses were removed from the sets in this step and the changes saved.

Subsequently, the docking poses were evaluated to ascertain their conformational energy and likelihood of occurrence in vivo. The sets were loaded into LigandScout's ligand-based view and opened in MOE's database viewer. In MOE, under the "Calculate" tab, an energy minimisation with the MMFF94 algorithm was performed. Each molecule was then energy minimised individually in LigandScout to ascertain the difference between the docking pose and the molecule at the energy minimum.

To verify the minimisation in LigandScout, each pose was simultaneously loaded into MOE from the database viewer and compared with the molecule in LigandScout. All docking poses that were not close to their energy minimum were discarded. When discrepancies between the two programs occurred, further considerations were made as to which pose is more likely. However, if the results produced by the two programs differed significantly from the suggested docking pose, the entry was rejected.

3.2.4.4.11 Final hit selection

Once all nine SD-files had been visually inspected, the SD-files that originated from the same pharmacophore but with different scoring functions were searched for SMILES-strings of molecules that were present in all three of them. This was achieved using a Python 3 script, provided in appendix Id. These molecules were the final hits of the pharmacophore-based screening and docking approach. A thorough inspection was conducted on all docking poses of the compounds that were affected by the error that occurred in Chapter 3.2.4.4.9. It was decided that these compounds should be included in the final selection, despite the possible bias, as they demonstrated good binding modes. The final results for the pharmacophore-based screening and docking approach are provided in Chapter 4.1.

3.3 Approach 2: Machine learning

The second approach to identifying novel Toll-like receptor 8 agonists involved training a machine learning model to identify active molecules in a library of small molecules. As a parameter for measuring biological activity, EC_{50} values were identified as an appropriate target variable for the creation of the model. The objective was to predict the EC_{50} values of the compounds in Enamine's screening collection database based on patterns learned from the activity data downloaded from ChEMBL.

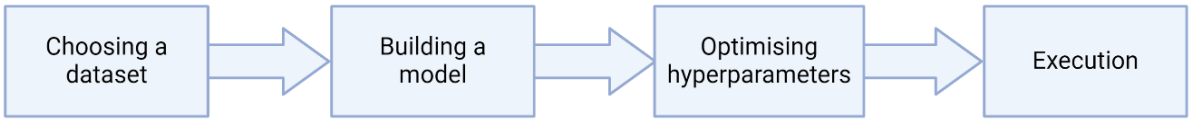


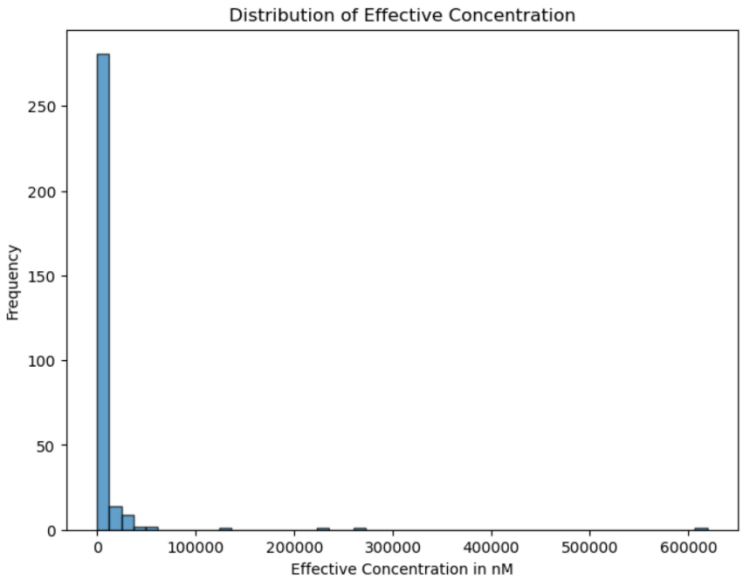
Figure 23: *Workflow for approach 2. These are the general steps that were carried out during the approach.*[13]

3.3.1 Choosing an algorithm

The objective was to develop a model that enables the selection of any number of the most effective molecules. It was therefore necessary to be able to rank the results by their predicted EC_{50} values. Given that EC_{50} values are numerical targets, it was decided to create a regressor model. An algorithm based on the decision tree ensemble methods was selected as these methods are well-suited to data sets with non-normal distributions, are resilient to outliers, require minimal data preparation, are able to identify complex relationships, and have a straightforward usability. Following the testing of several models, the gradient boosting regression tree algorithm was identified as the most effective for the dataset.

3.3.2 Choosing the dataset

In order for the model to accurately predict any EC_{50} value, it was necessary to train it on a dataset that included molecules with a diverse range of values. It was therefore decided to train the model on list 1 of Chapter 3.1.2. This list contains only agonistically active compounds, along with their EC_{50} values, regardless of whether the values indicate a highly active compound or not. However, the majority of values in this dataset were relatively low, with a few exceptionally high outliers. (Figure 24)



count	312.000000
mean	7593.003205
std	41196.908622
min	1.000000
25%	27.000000
50%	765.000000
75%	2925.000000
max	619000.000000

Figure 24: *Distribution of EC_{50} values in the dataset. The one-sided distribution has caused performance problems with the machine learning algorithm.*

Despite the chosen model's ability to handle outliers more effectively than other algorithms, the distribution remained a challenge, resulting in performance issues. As a solution, the EC_{50} values were converted to pEC_{50} values, which were then predicted and converted back to EC_{50}. This reduced the variability in the dataset, resulting in more accurate and reliable results. (Figure 25)

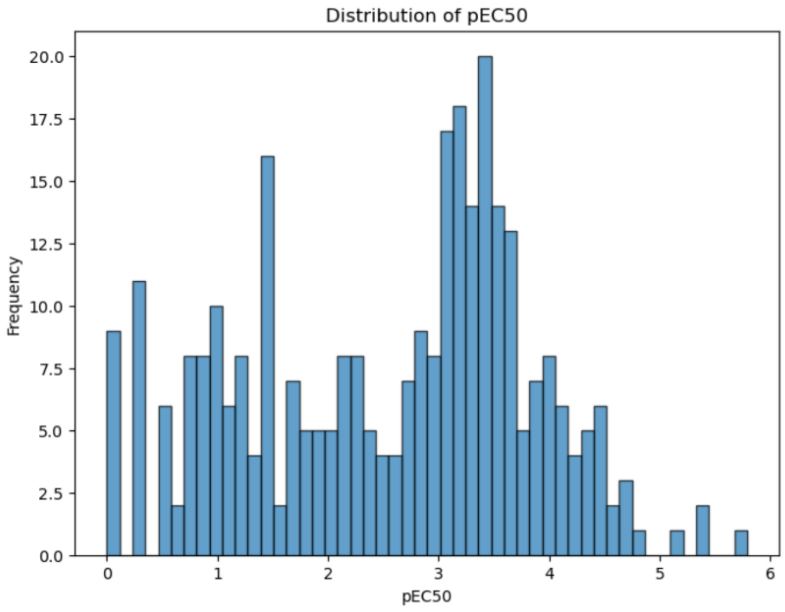


Figure 25: *Distribution of the converted pEC_{50} values. Following the conversion, the machine learning algorithm was executed without any issues.*

3.3.3 Featurisation

Given the size of the database that the model was to be applied to, which was in the region of four million molecules, it was necessary to create a compact digital representation of the molecules in order to limit the required computational effort as well as the training time. Another factor that influenced the choice of featurisation was the desire to represent a diverse range of molecular structures. Consequently, a featurisation method that incorporates as many features of a molecule as possible while being computationally efficient was preferred. Morgan fingerprints were selected as the optimal featurisation method for the molecules, as they offer a compact representation while also efficiently encoding a comprehensive picture of the molecules. Furthermore, they demonstrated superior performance compared to other featurisation methods for this specific model and dataset. Canonicalisation was employed to distinguish between enantiomers and molecules that might produce similar fingerprints.

3.3.4 Model settings and hyperparameter tuning

To gain an accurate representation of the model's functionality, it was determined that 30% of the data would be utilized as a training set. It subsequently became evident that this represented a relatively high portion of the overall dataset, as outlined in Chapter 5.3.

Gradient boosting regressors are particularly sensitive to hyperparameter tuning as described in Chapter 1.2.3.2.1.1. Therefore, a grid search with k-fold cross-validation was conducted to identify the optimal hyperparameters. This included the radius setting for the Morgan fingerprints, which was implemented using a for loop. The code is provided in appendix IIb.

The grid search returned the following hyperparameters:

- n_estimators=40: This hyperparameter defines the number of trees to be built by the model.
- max_depth=19: This limits the depth of each decision tree to 19.
- min_samples_leaf=2: A leaf node, that is a node that will not be split further, has to have at least two samples in it.
- min_samples_split=5: If a node has fewer than five samples in it, it cannot be split further.
- max_features='sqrt': To find the best feature to split a node, the square root of all samples is considered as opposed to all samples. This introduces a controlled amount of randomness to the trees which improves the generalisability of the model.[206]
- learning_rate=0.2: This controls how much each tree tries to correct the mistakes made by the previous tree
- radius=2: This setting defines the radius of atoms to be considered by the Morgan fingerprint for its vector. Setting it to two means that for each atom, the adjacent atoms as and the direct neighbours of these adjacent atoms are considered.

The hyperparameter settings were applied to the model and its performance was assessed in another k-10-fold cross-validation. As the model predicts a numeric variable, the performance metrics used were R^2, RMSD and MAE.

```
Fold 1/10
R² = 0.807, MAE = 0.393, RMSE = 0.542
Fold 2/10
R² = 0.827, MAE = 0.452, RMSE = 0.539
Fold 3/10
R² = 0.870, MAE = 0.363, RMSE = 0.444
Fold 4/10
R² = 0.819, MAE = 0.432, RMSE = 0.557
Fold 5/10
R² = 0.750, MAE = 0.431, RMSE = 0.615
Fold 6/10
R² = 0.788, MAE = 0.467, RMSE = 0.607
Fold 7/10
R² = 0.741, MAE = 0.499, RMSE = 0.658
Fold 8/10
R² = 0.692, MAE = 0.489, RMSE = 0.696
Fold 9/10
R² = 0.832, MAE = 0.428, RMSE = 0.549
Fold 10/10
R² = 0.799, MAE = 0.416, RMSE = 0.542

Average Scores Across All Folds:
R² = 0.792, MAE = 0.437, RMSE = 0.575
```

Figure 26: *k-10-fold cross-validation score for the model with grid search hyperparameters. The values of R^2, MAE and RMSD (here: RMSE) are displayed for each data fold. The performance metrics averaged across all folds can be seen at the bottom. For a detailed description of the performance metrics used, please refer to Chapter 1.2.3.4.1.2.*

Following grid search, some manual hyperparameter tuning was performed until a set of hyperparameters was found with satisfactory cross-validation scores that performed even slightly better than the hyperparameters suggested by the grid search in terms of cross-validation scores:

- n_estimators=50: This hyperparameter sets the number of trees to be built by the model.
- max_depth=17: This limits the depth of each decision tree to 17
- min_samples_leaf=3: A leaf node, that is a node that will not be split further, has to have at least three samples in it.
- min_samples_split=2: If a node has fewer than two samples in it, it cannot be split further.
- max_features='sqrt': To find the best feature to split a node, the square root of all samples is considered as opposed to all samples. This introduces a controlled amount of randomness to the trees which improves the generalisability of the model.[206]
- learning_rate=0.2: This controls how much each tree tries to correct the mistakes made by the previous tree
- radius=1: This setting defines the radius of atoms to be considered by the Morgan fingerprint for its vector. Setting it to two means that for each atom, the adjacent atoms as and the direct neighbours of these adjacent atoms are considered.

The model was run with the aforementioned hyperparameters, resulting in the following cross-validation scores:

```
Fold 1/10
R² = 0.838, MAE = 0.388, RMSE = 0.497
Fold 2/10
R² = 0.850, MAE = 0.418, RMSE = 0.501
Fold 3/10
R² = 0.843, MAE = 0.419, RMSE = 0.486
Fold 4/10
R² = 0.773, MAE = 0.476, RMSE = 0.624
Fold 5/10
R² = 0.731, MAE = 0.477, RMSE = 0.638
Fold 6/10
R² = 0.817, MAE = 0.435, RMSE = 0.563
Fold 7/10
R² = 0.689, MAE = 0.556, RMSE = 0.721
Fold 8/10
R² = 0.714, MAE = 0.488, RMSE = 0.671
Fold 9/10
R² = 0.859, MAE = 0.396, RMSE = 0.502
Fold 10/10
R² = 0.849, MAE = 0.381, RMSE = 0.470

Average Scores Across All Folds:
R² = 0.796, MAE = 0.443, RMSE = 0.567
```

Figure 27: *k-10-fold cross-validation score for the model with intelligently optimised hyperparameters. The values of R^2, MAE and RMSD (here: RMSE) are displayed for each data fold. The performance metrics averaged across all folds can be seen at the bottom. For a detailed description of the performance metrics used, please refer to Chapter 1.2.3.4.1.2.*

While the cross-validation metrics were slightly better for the manually chosen hyperparameters, there may be other reasons why the grid search algorithm chose its particular set of hyperparameters. However, it was surprising that the model with the manually optimised hyperparameters not only showed slightly better cross-validation performance, but also a slightly lower tendency to overfitting when comparing performance on the training set and the test set after execution. (Figure 28) Given that diverse molecular structures were desired, it was decided to build two models: one with the grid search hyperparameters (Model A) and one with intelligently chosen hyperparameters (Model B). It was thought that comparing the results may provide insight into the important structural elements in TLR8 agonists.

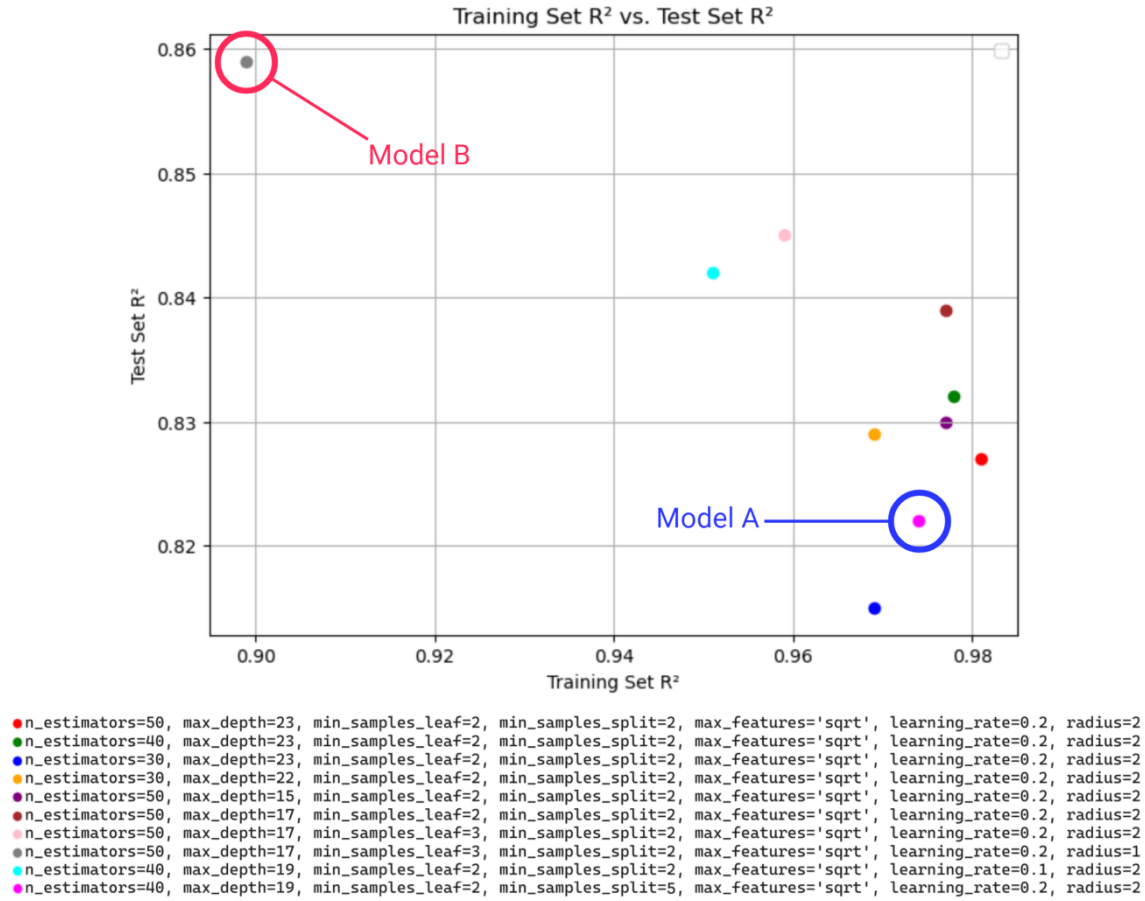


Figure 28: *Comparison of the performance in the training set and test set of the models with different hyperparameter settings. The magenta-coloured dot, highlighted in blue, represents Model A. The grey dot, highlighted in red, represents Model B. The other dots are different models with different combinations of hyperparameters that were tested in the optimisation process.*

3.3.5 Running the models and retrieving the results

Both models' scripts were executed. The model with the grid search hyperparameters (Model A) achieved a R^2 of 0.974 on the training set and 0.822 on test set. The model with the intelligently chosen hyperparameters (Model B) achieved a R^2 value of 0.899 for the training set and 0.859 for the test set. The figures demonstrate that approximately 97.4% and 90% of the training set values and 82% and 86% of the test set values could be explained by the models, respectively. Although Model A shows a very slight tendency, it can be said that both models demonstrated a relatively low level of overfitting, as evidenced by the fact that the R^2 values for training and test sets are not drastically far apart from each other.

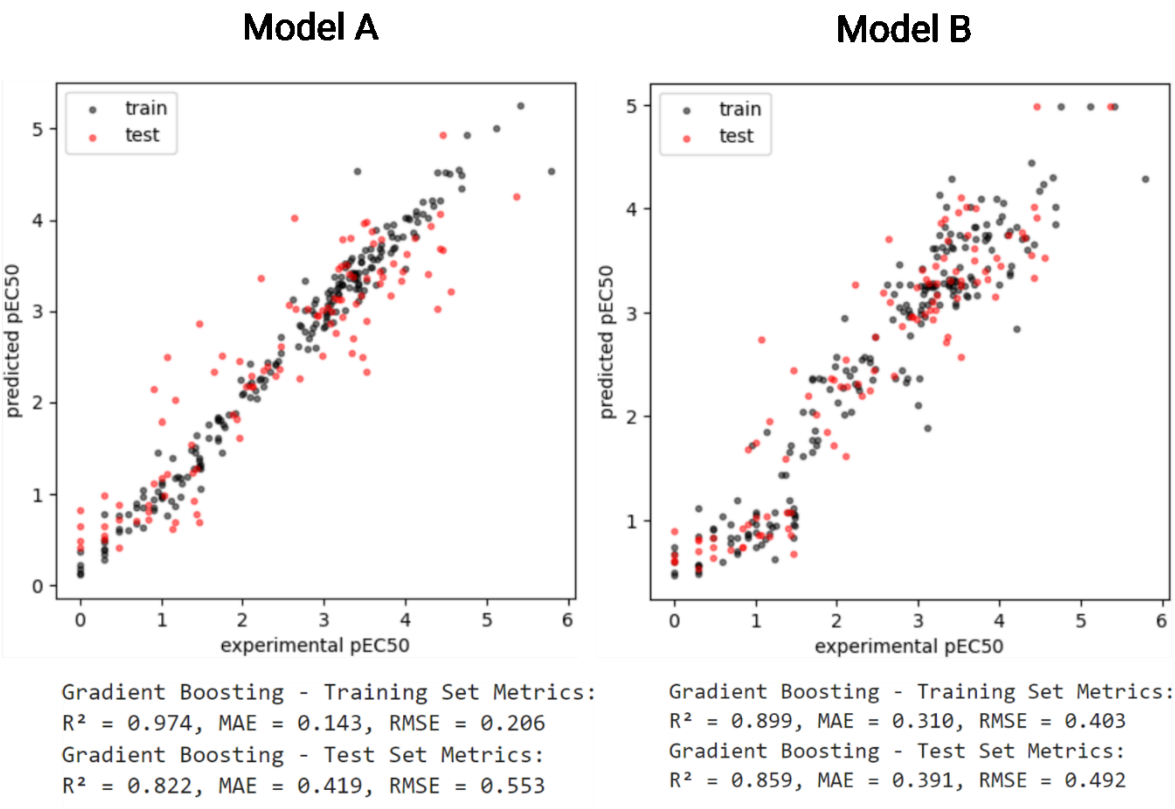


Figure 29: *Scatterplots showing the performance on the training set and test set of the two models. Model A is shown on the left and Model B on the right. The respective performance metrics are shown below the plots. For a detailed description of the performance metrics used, please refer to Chapter 1.2.3.4.1.2.*

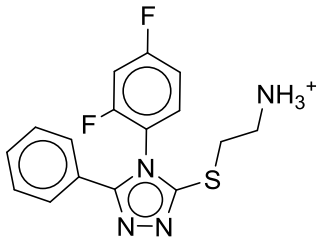
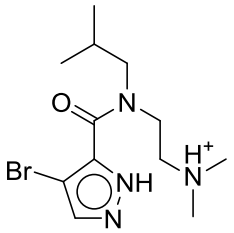
For each model, a CSV-file containing the molecules of Enamine's screening collection database, version of may 2022, along with their SMILES-string and predicted pEC_{50} value was obtained. For the following in-vitro studies, it was decided to order 30 molecules from the results of the machine learning models. Therefore, for both models, the fifteen molecules for which the lowest EC_{50} values were predicted by the respective model were selected for the final results. Another script was executed on both obtained CSV-files that converted the pEC_{50} values back to EC_{50} values, ranked the results and created another CSV-file with only the ten entries with the lowest EC_{50} values. These molecules represent the final selection of the machine learning approach. The script used to convert the pEC_{50} values back to EC_{50} as well as retrieve the results is provided in appendix IIId. The results for the machine learning models are presented in Chapter 4.

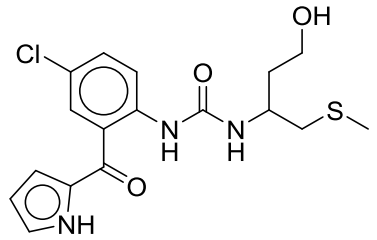
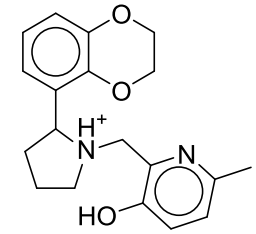
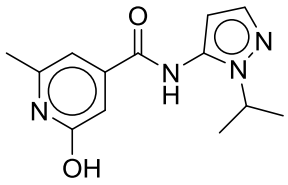
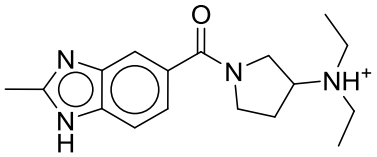
4 Results

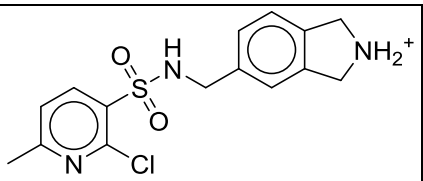
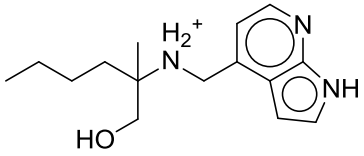
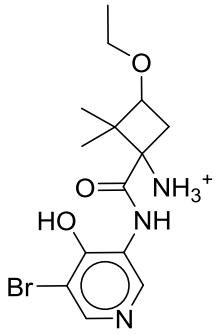
As illustrated in Figure 13, three distinct groups of molecules have been identified as potential agonists of Toll-like receptor 8 during the course of this work. One set for the molecules from approach 1, the pharmacophore-based screening and docking, another one from the machine learning model with the grid search hyperparameters (Model A) and a third one from the machine learning model with the intelligently designed hyperparameters (Model B). The molecules are presented in the following subchapters with their structures, molecular weight, chemical formula, their identification numbers in the catalogue of the chemical service provider Enamine, the EC_{50} values predicted by both machine learning models as well as their SMILES-strings.

4.1 Pharmacophore-based screening and docking set

Table 2 shows the structures of the molecules derived from pharmacophore-based screening and docking. In addition to the information mentioned in Chapter 4, it also notes the pharmacophore used in the screening that the molecule originated from. Molecules from all three pharmacophores were deliberately included in this set to ensure greater molecular diversity. Molecules affected by the bias described in Chapter 3.2.4.4.9 are highlighted in bold.

Compound	Enamine ID	EC_{50} (Model A)	EC_{50} (Model B)	SMILES
 MW = 422.4 g/Mol Chemical Formula: $C_{18}H_{16}F_2N_4O_4S$ Pharmacophore: TP78_FP0	Z56936829	2175.16 nM	9020.83 nM	[NH3+]CCSc1nnc(-c2ccccc2)n1-c1ccc(F)cc1F
 MW = 317.2 g/Mol Chemical Formula: $C_{12}H_{21}BrN_4O$ Pharmacophore: TP125_FP45	Z1251893025	3131.11 nM	565.31 nM	CC(C)CN(CC[NH+](C)C)C(=O)c1[nH]ncc1Br

 MW = 381.9 g/Mol Chemical Formula: $C_{17}H_{20}ClN_3O_3S$ Pharmacophore: TP81_FP6	Z1539671458	2950.40 nM	1045.14 nM	CSCC(CCO)NC(=O)Nc1ccc(Cl)cc1C(=O)c1ccc[nH]1
 MW = 326.4 g/Mol Chemical Formula: $C_{19}H_{22}N_2O_3$ Pharmacophore: TP78_FP0	Z1568272665	1749.69 nM	6276.62 nM	Cc1ccc(O)c(C[NH+]2CCCC2c2cccc3c2OCCO3)n1
 MW = 260.3 g/Mol Chemical Formula: $C_{13}H_{16}N_4O_2$ Pharmacophore: TP81_FP6	Z1609219007	1520.92 nM	6123.38 nM	Cc1cc(C(=O)Nc2ccnn2C(C)C)cc(O)n1
 MW = 300.4 g/Mol Chemical Formula: $C_{17}H_{24}N_4O$ Pharmacophore: TP125_FP45	Z1625506699	711.40 nM	2921.32 nM	CC[NH+](CC)C1CCN(C(=O)c2ccc3[nH]c(C)nc3c2)C1

 MW = 374.3 g/Mol Chemical Formula: $C_{15}H_{17}Cl_2N_3O_2S$ Pharmacophore: TP125_FP45	Z2040113071	2147.97 nM	2181.48 nM	Cc1ccc(S(=O)(=O)NCc2ccc3c(c2)C[NH2+]C3)c(Cl)n1
 MW = 261.4 g/Mol Chemical Formula: $C_{15}H_{23}N_3O$ Pharmacophore: TP125_FP45	**Z2443483104**	10929.87 nM	1409.69 nM	**CCCCC(C)(CO)[NH2+]Cc1ccnc2[nH]ccc12**
 MW = 358.2 g/Mol Chemical Formula: $C_{14}H_{20}BrN_3O_3$ Pharmacophore: TP78_FP0	**Z2452532380**	2555.28 nM	2430.59 nM	**CCOC1CC([NH3+])(C(=O)Nc2cncc(Br)c2O)C1(C)C**

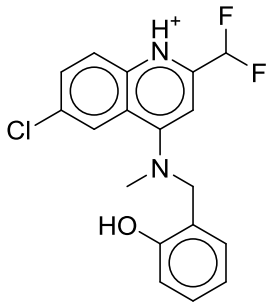
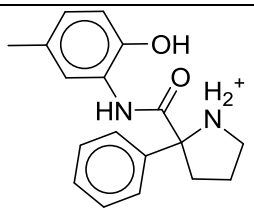
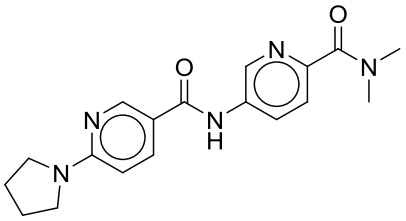
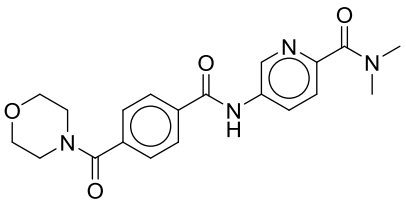
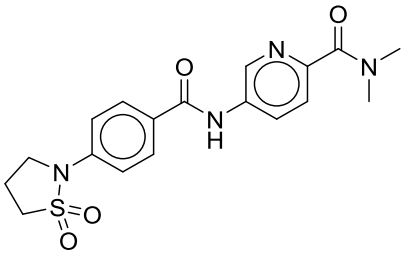
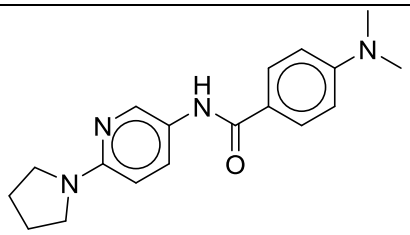
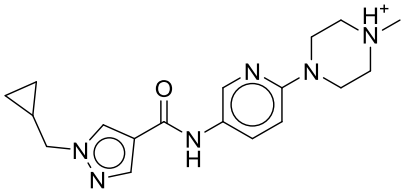
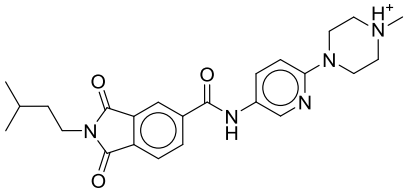
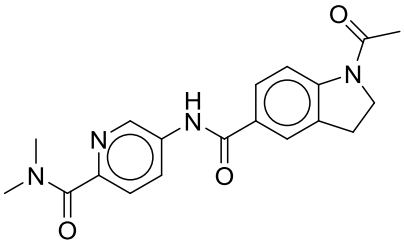
 MW = 348.8 g/Mol Chemical Formula: $C_{18}H_{15}ClF_2N_2O$ Pharmacophore: TP125_FP45	**Z2744963592**	2383.26 nM	377.30 nM	**CN(Cc1ccccc1O)c1cc(C(F)F)[nH+]c2ccc(Cl)cc12**
 MW = 296.4 g/Mol Chemical Formula: $C_{18}H_{20}N_2O_2$ Pharmacophore: TP125_FP45	Z3063578001	2876.40 nM	1307.60 nM	Cc1ccc(O)c(NC(=O)C2(c3ccccc3)CCC[NH2+]2)c1

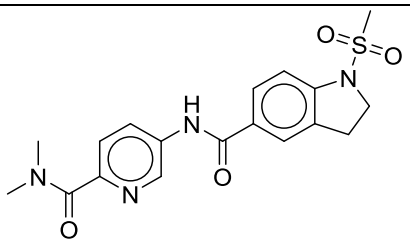
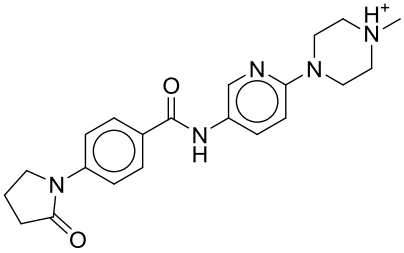
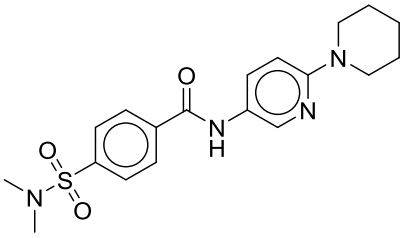
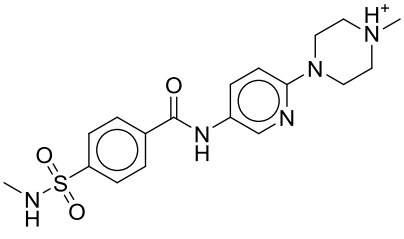
Table 2: *Final results of the pharmacophore-based screening and docking approach. Molecules affected by the bias described in Chapter 3.2.4.4.9 are highlighted in bold.*

4.2 Machine Learning Model A

Table 3 shows the structures of the 15 molecules from Enamine's database, that were predicted to have the lowest EC_{50} values by the machine learning model with hyperparameters from the grid search (Model A).

Compounds	Enamine ID	EC_{50} (Model A)	EC_{50} (Model B)	SMILES
 MW = 339.4 g/Mol Chemical Formula: $C_{18}H_{21}N_5O_2$	Z2928736856	8.61 nM	2205.15 nM	CN(C)C(=O)c1ccc(NC(=O)c2ccc(N3CCCC3)nc2)cn1
 MW = 382.4 g/Mol Chemical Formula: $C_{20}H_{22}N_4O_4$	Z2928726078	9.00 nM	2486.13 nM	CN(C)C(=O)c1ccc(NC(=O)c2ccc(C(=O)N3CCOCC3)cc2)cn1
 MW = 388.4 g/Mol Chemical Formula: $C_{18}H_{20}N_4O_4S$	Z2928738714	9.14 nM	2205.15 nM	CN(C)C(=O)c1ccc(NC(=O)c2ccc(N3CCCS3(=O)=O)cc2)cn1

 MW = 310.4 g/Mol Chemical Formula: $C_{18}H_{22}N_4O$	Z230704720	9.31 nM	2546.58 nM	CN(C)c1ccc(C(=O)Nc2ccc(N3CCCC3)nc2)cc1
 MW = 340.4 g/Mol Chemical Formula: $C_{18}H_{24}N_6O$	Z2764206490	9.69 nM	2842.89 nM	C[NH+]1CCN(c2ccc(NC(=O)c3cnn(CC4CC4)c3)cn2)CC1
 MW = 435.5 g/Mol Chemical Formula: $C_{24}H_{29}N_5O_3$	Z223833178	11.49 nM	7895.00 nM	CC(C)CCN1C(=O)c2ccc(C(=O)Nc3ccc(N4CC[NH+](C)CC4)nc3)cc2C1=O
 MW = 352.4 g/Mol Chemical Formula: $C_{19}H_{20}N_4O_3$	Z2928738672	11.65 nM	1894.93 nM	CC(=O)N1CCc2cc(C(=O)Nc3ccc(C(=O)N(C)C)nc3)ccc21

 MW = 388.4 g/Mol Chemical Formula: $C_{18}H_{20}N_4O_4S$	Z2928739057	12.02 nM	1894.93 nM	CN(C)C(=O)c1ccc(NC(=O)c2ccc3c(c2)CCN3S(C)(=O)=O)cn1
 MW = 379.5 g/Mol Chemical Formula: $C_{21}H_{25}N_5O_2$	Z221484126	12.13 nM	3831.57 nM	C[NH+]1CCN(c2ccc(NC(=O)c3ccc(N4CCCC4=O)cc3)cn2)CC1
 MW = 388.5 g/Mol Chemical Formula: $C_{19}H_{24}N_4O_3S$	Z227262780	12.30 nM	2546.58 nM	CN(C)S(=O)(=O)c1ccc(C(=O)Nc2ccc(N3CCCCC3)nc2)cc1
 MW = 389.5 g/Mol Chemical Formula: $C_{18}H_{23}N_5O_3S$	Z230202390	13.27 nM	3965.01 nM	CNS(=O)(=O)c1ccc(C(=O)Nc2ccc(N3CC[NH+](C)CC3)nc2)cc1

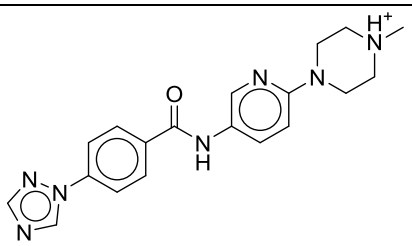
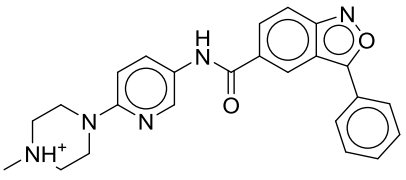
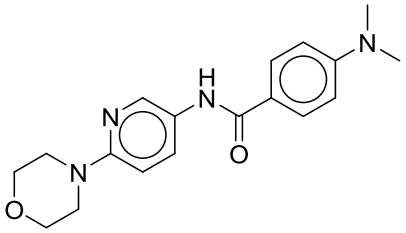
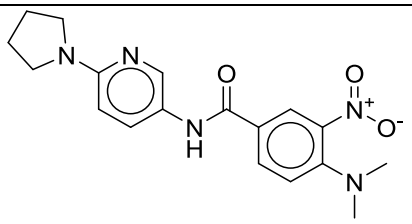
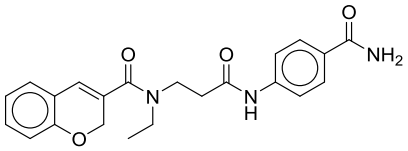
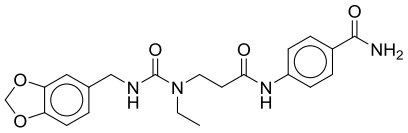
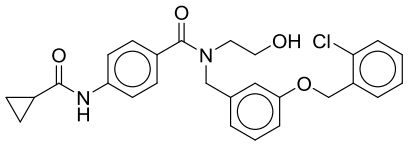
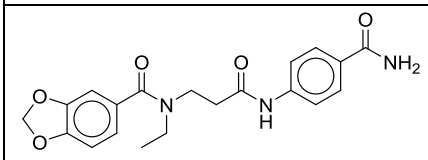
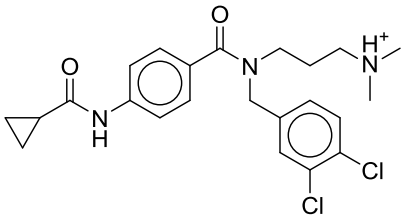
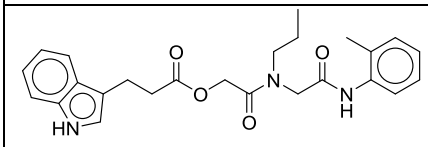
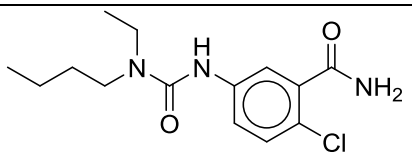
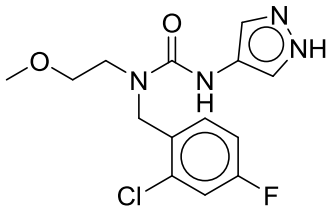
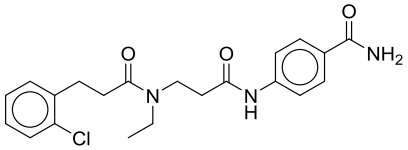
 MW = 363.4 g/Mol Chemical Formula: $C_{19}H_{21}N_7O$	Z230197300	13.43 nM	6445.44 nM	C[NH+]1CCN(c2ccc(NC(=O)c3ccc(-n4cncn4)cc3)cn2)CC1
 MW = 413.5 g/Mol Chemical Formula: $C_{24}H_{23}N_5O_2$	Z230198598	13.72 nM	4308.56 nM	C[NH+]1CCN(c2ccc(NC(=O)c3ccc4noc(-c5ccccc5)c4c3)cn2)CC1
 MW = 326.4 g/Mol Chemical Formula: $C_{18}H_{22}N_4O_2$	Z229816512	13.72 nM	3636.10 nM	CN(C)c1ccc(C(=O)Nc2ccc(N3CCOCC3)nc2)cc1
 MW = 355.4 g/Mol Chemical Formula: $C_{18}H_{21}N_5O_3$	Z230699144	13.89 nM	2188.33 nM	CN(C)c1ccc(C(=O)Nc2ccc(N3CCCC3)nc2)cc1[N+](=O)[O-]

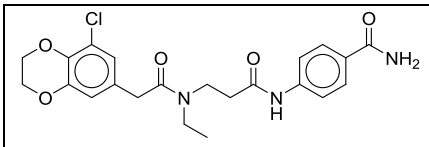
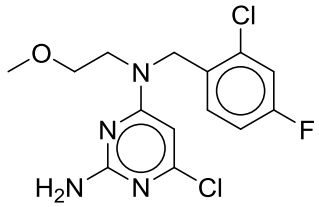
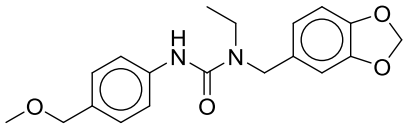
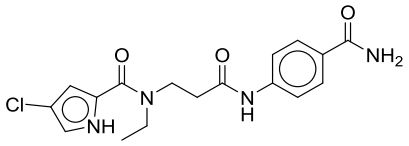
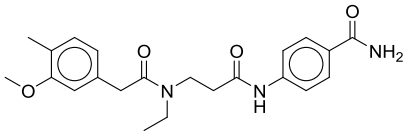
Table 3: *The final results for machine learning Model A. They include the fifteen molecules in Enamine's Screening Collection database, version of may 2022, that were predicted by Model A to have the lowest EC_{50} values.*

4.3 Machine Learning Model B

Table 4 shows the structures of the 15 molecules from Enamine's database, that were predicted to have the lowest EC_{50} values by the machine learning model with intelligently designed hyperparameters (Model B).

Compound	Enamine ID	EC_{50} (Model A)	EC_{50} (Model B)	SMILES
 MW = 393.4 g/Mol Chemical Formula: $C_{22}H_{23}N_3O_4$	Z285541642	153.10 nM	15.57 nM	CCN(CCC(=O)Nc1ccc(C(N)=O)cc1)C(=O)C1=Cc2ccccc2OC1
 MW = 412.4 g/Mol Chemical Formula: $C_{21}H_{24}N_4O_5$	Z573712576	3968.89 nM	20.34 nM	CCN(CCC(=O)Nc1ccc(C(N)=O)cc1)C(=O)NCc1ccc2c(c1)OCO2
 MW = 479 g/Mol Chemical Formula: $C_{27}H_{27}ClN_2O_4$	Z167207688	540.25 nM	20.45 nM	O=C(Nc1ccc(C(=O)N(CCO)Cc2ccccc(OCc3ccccc3Cl)c2)cc1)C1CC1
 MW = 383.4 g/Mol Chemical Formula: $C_{20}H_{21}N_3O_5$	Z285540094	1207.72 nM	24.09 nM	CCN(CCC(=O)Nc1ccc(C(N)=O)cc1)C(=O)c1ccc2c(c1)OCO2

 MW = 448.4 g/Mol Chemical Formula: $C_{23}H_{27}Cl_2N_3O_2$	Z826838296	912.00 nM	24.79 nM	C[NH+](C)CCCN(Cc1ccc(Cl)c(Cl)c1)C(=O)c1ccc(NC(=O)C2CC2)cc1
 MW = 435.5 g/Mol Chemical Formula: $C_{25}H_{29}N_3O_4$	Z13758572	1478.66 nM	26.86 nM	CCCN(CC(=O)Nc1ccccc1C)C(=O)COC(=O)CCc1c[nH]c2ccccc12
 MW = 297.8 g/Mol Chemical Formula: $C_{14}H_{20}ClN_3O_2$	Z414266466	1842.69 nM	27.58 nM	CCCCN(CC)C(=O)Nc1ccc(Cl)c(C(N)=O)c1
 MW = 326.8 g/Mol Chemical Formula: $C_{14}H_{16}ClFN_4O_2$	Z1149172628	2800.96 nM	28.50 nM	COCCN(Cc1ccc(F)cc1Cl)C(=O)Nc1cn[nH]c1
 MW = 401.9 g/Mol Chemical Formula: $C_{21}H_{24}ClN_3O_3$	Z290498678	935.14 nM	28.67 nM	CCN(CCC(=O)Nc1ccc(C(N)=O)cc1)C(=O)CCc1ccccc1Cl

 MW = 445.9 g/Mol Chemical Formula: $C_{22}H_{24}ClN_3O_5$	Z285541906	1292.68 nM	29.06 nM	CCN(CCC(=O)Nc1ccc(C(N)=O)cc1)C(=O)Cc1cc(Cl)c2c(c1)OCCO2
 MW = 345.2 g/Mol Chemical Formula: $C_{14}H_{15}Cl_2FN_4O$	Z1430528549	640.23 nM	29.12 nM	COCCN(Cc1ccc(F)cc1Cl)c1cc(Cl)nc(N)n1
 MW = 342.4 g/Mol Chemical Formula: $C_{19}H_{22}N_2O_4$	Z1892841216	1525.88 nM	30.05 nM	CCN(Cc1ccc2c(c1)OCO2)C(=O)Nc1ccc(COC)cc1
 MW = 362.8 g/Mol Chemical Formula: $C_{17}H_{19}ClN_4O_3$	Z285581258	442.57 nM	30.33 nM	CCN(CCC(=O)Nc1ccc(C(N)=O)cc1)C(=O)c1cc(Cl)c[nH]1
 MW = 397.5 g/Mol Chemical Formula: $C_{22}H_{27}N_3O_4$	Z285541880	1742.69 nM	30.60 nM	CCN(CCC(=O)Nc1ccc(C(N)=O)cc1)C(=O)Cc1ccc(C)c(OC)c1

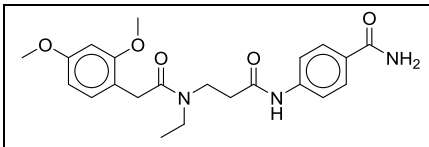
 MW = 413.5 g/Mol Chemical Formula: $C_{22}H_{27}N_3O_5$	Z285541410	3715.71 nM	30.62 nM	CCN(CCC(=O)Nc1ccc(C(N)=O)cc1)C(=O)Cc1ccc(OC)cc1OC

Table 4: The final results for machine learning model B. They include the fifteen molecules in Enamine's Screening Collection database, version of May 2022, that were predicted by Model B to have the lowest EC_{50} values.

5 Discussion

When comparing the three sets of molecules, it is noticeable that the molecules from the machine learning models are much more similar to each other than those from the pharmacophore-based screening and docking. This is partly due to the human choices made in the design process, but also because deliberate choices were made in the pharmacophore-based screening and docking that resulted in greater molecular diversity.

The large differences in predicted EC_{50} values for the same molecules between the two machine learning models also highlight the importance of hyperparameter tuning for gradient boosted regression models. It is also noticeable that machine learning Model A tends to predict slightly lower EC_{50} values for the molecules in its final results than Model B does for its set.

5.1 Common substructures

In order to find commonalities between the molecules that might be important for the activation of TLR8, the common substructures within each set were calculated using the findMCS function in RDKit. Appendix IIIb, IIIc and IIId show the molecule sets with their highlighted common substructures, appendix IIIa shows the Python script used. Figure 30 shows the maximum common substructure of all molecules within each of the sets of final results.

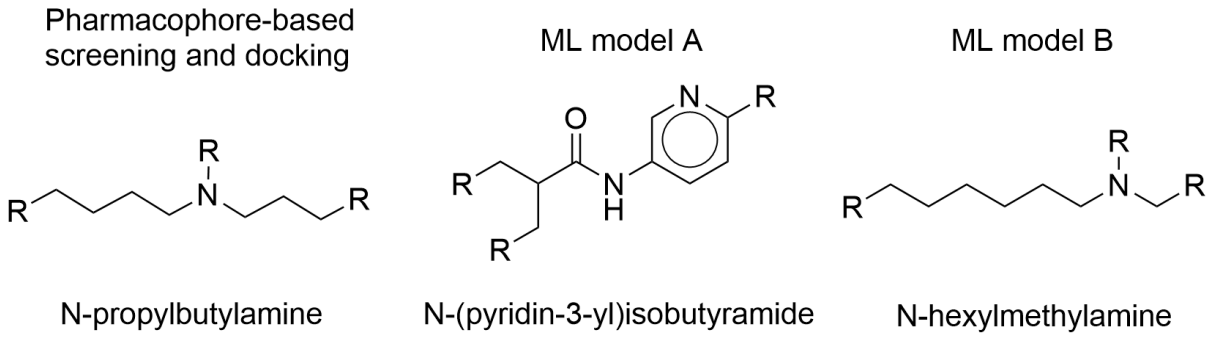


Figure 30: *The maximum common substructures for the molecules from pharmacophore-based screening and docking and from machine learning Model A and machine learning model B. Machine learning Model A shows a distinct substructure that may be important as a scaffold for agonists at TLR8.*

While the common substructure for the molecules from the pharmacophore-based screening and docking is rather unspecific, the set of molecules from machine learning Model A show distinct substructures that could potentially be important for TLR8 activation. The maximum common substructure of the molecules in the final results of machine learning Model B is also rather unspecific. However, if only the substructure of the ten molecules with the lowest EC_{50} value is analysed, a similar structure to Model A can be seen.

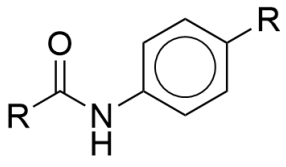


Figure 31: *The maximum common substructure of the ten molecules of set 2 (Model B) for which the lowest EC_{50} values were predicted.*

Whether these substructures themselves show the required molecular interactions or are only suitable as a basic framework for agonists due to their size cannot be subsequently clarified in this work. Further detailed studies will be required to address this issue.

5.2 Selectivity estimation using PLIF

Many known TLR8 agonists are actually dual TLR7/8 agonists, including the well-known agonist resiquimod, which has served as a model for this work on several occasions.[48] TLR7 is phylogenetically the most similar receptor to TLR8. It is also an endosomal PAMP receptor that is primarily found in cells of the innate immune system.[15] Its ligands are also single-stranded RNA molecules, but with different motifs.[48] TLR7 is also involved in viral recognition, but recognises different viruses as TLR8.[15] The profile of cytokines released by TLR7 activation is different from that of TLR8. Due to the high sequence homology of the two receptors, it is difficult to selectively drug TLR8.[138][207] Therefore, it was decided to test the selectivity of the molecules found between TLR7 and TLR8. For this purpose, they were docked into a prepared crystal structure of TLR7 and then analysed with PLIF to determine whether they showed interactions with those amino acid residues of TLR7 that are important for its activation.

5.2.1 Execution

One SD-file was prepared for each set of molecules in the final results. This was achieved by writing the SMILES of the molecules to a CSV-file, which was then opened in MOE and protonated using the wash function as described in Chapter 3.2.4.4.5. The resulting SD-files were saved. The correct three-dimensional structures in the gas phase were then calculated with Corina. The protein structure 5GMF was downloaded from the PDB and prepared in the same way as described in Chapter 3.2.4.1.[208][209]

The three SD-files were then docked separately with GOLD into the previously prepared crystal structure, using the same settings as in Section 3.2.4.2. The obtained SD-files with the docking poses for TLR7 and the prepared structure of TLR7 were opened in MOE, and the PLIF analysis was performed in accordance with the methodology outlined in Chapter 3.2.4.4.5.

5.2.2 Results

In publications, interactions with the amino acids Thr586, Asp555, and Phe408 were identified as crucial for the activation of TLR7.[210] Given that almost all molecules found interact with almost all amino acid residues important for the activation of TLR8, special attention was paid to molecules that also interact with all amino acids important for the activation of TLR7. While most molecules interacted with at least one residue in TLR7, there was only one molecule in each of the three sets that interacted with Thr586, Asp555 and Phe408 simultaneously. Figure 6 shows the molecules involved.

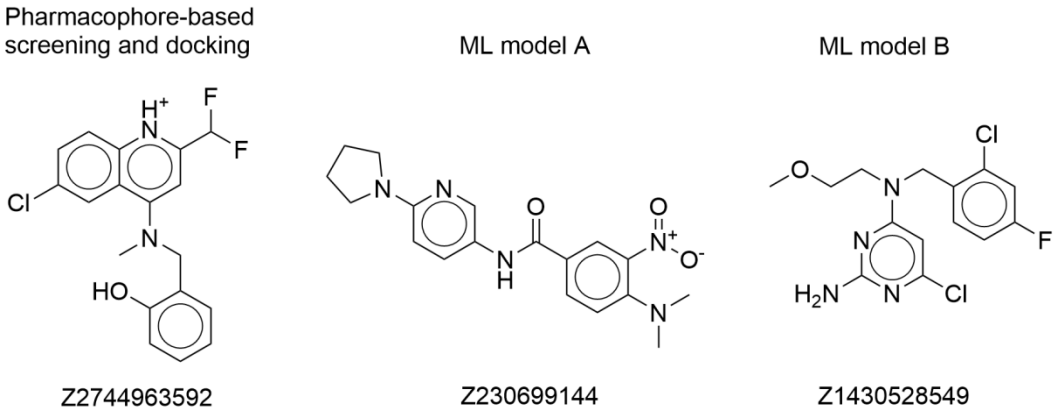


Figure 32: *All molecules from all sets of final results that have interacted with all amino acid residues of TLR7 that are important for its activation.*

Z2744963592 is also one of the molecules affected by the bias described in Chapter 3.2.4.4.9. Although these molecules demonstrated activity at TLR7, they were not excluded from the final results. One reason for this is that dual TLR7/8 agonists are also excellent drug candidates. Another reason is that the PLIF analysis carried out only allows a very superficial estimate of activity. It cannot be concluded with certainty that these are molecules with activity at TLR7. Nevertheless, the results should be considered in future research into these molecules.

5.3 Limitations

During the course of this project, a number of limitations were identified that may influence the interpretation of the results. A number of these decisions were made due to the author's lack of experience in the field of computer-aided drug design, as all the techniques used had to be learned and were performed for the first time.

The first limitation concerns the number of decoys used to validate the pharmacophores in Chapter 3.2.2.2. Discussions with Prof. Wolber revealed that, in comparison to current common practice in CADD, the number of decoys that was used in the validation process is rather small. Furthermore, it would have been beneficial to perform validations with and without decoys. This would have resulted in the creation of even more effective pharmacophore models. Additionally, with better programming expertise, it would have been feasible to devise a methodology for integrating a data set of decoys into the machine learning models as well.

An irreversible error that became apparent only at a later stage in the process is the distortion described in Chapter 3.2.4.4.9. This introduced a bias into the set of pharmacophore-based screening and docking. Despite careful scrutiny, it cannot be ruled out that the final molecule selection of this set may have been influenced by this.

Another issue concerns the machine learning models. When splitting the data sets into training and test sets, it was assumed that the test set must be larger than would have actually been necessary. In discussions with Dr. Nunes-Alves at TU Berlin, it was established that more data could have been used to train the models, which might also have influenced the predictive power of the models.

On a final note, all computer-aided methods in drug discovery are merely models based on various assumptions and therefore, by their very nature, cannot be applied without restriction.

5.4 Final remarks

The molecules discovered in the course of this project will be ordered from Enamine by the FU Berlin over the next few months and tested for their biological activity in in-vitro studies by a contractual partner. The results of these tests are not yet available. Appendix IV provides a table of the molecules that have already been ordered for further testing.

Nevertheless, the project was able to identify some indications of their activity and provide promising candidates. Another goal that has been achieved is the acquisition of expertise in different methods of computer-aided drug design by the author.

Sources

[1]Takeuchi, O., Akira, S. Toll-like receptors; their physiological role and signal transduction system. 1, 0–635, 2001, https://doi.org/10.1016/s1567-5769(01)00010-8

[2]Hansson, G.K., Edfeldt, K. Toll to be paid at the gateway to the vessel wall. Arterioscler Thromb Vasc Biol 25, 1085-1087, 2005, https://doi.org/10.1161/01.ATV.0000168894.43759.47

[3]Cervantes, J.L., Weinerman, B., Basole, C., Salazar, J.C. TLR8: the forgotten relative revindicated. Cell Mol Immunol 9, 434-438, 2012, https://doi.org/10.1038/cmi.2012.38

[4]Sameer, A.S., Nissar, S. Toll-Like Receptors (TLRs): Structure, Functions, Signaling, and Role of Their Polymorphisms in Colorectal Cancer Susceptibility. Biomed Res Int 2021, 2021:1157023, https://doi.org/10.1155/2021/1157023.

[5]Javaid, N., Choi, S. Toll-like Receptors from the Perspective of Cancer Treatment. Cancers (Basel) 12(2), 297, 2020, doi: 10.3390/cancers12020297.

[6]Armant, M.A., Fenton, M.J. Toll-like receptors: a family of pattern-recognition receptors in mammals. Genome Biol 3(8), REVIEWS3011, 2002, doi: 10.1186/gb-2002-3-8-reviews3011.

[7]Kato, J., Svensson, C.I. Role of extracellular damage-associated molecular pattern molecules (DAMPs) as mediators of persistent pain. Prog Mol Biol Transl Sci 131, 251-279, 2015, https://doi.org/10.1016/bs.pmbts.2014.11.014

[8]Li, D., Wu, M. Pattern recognition receptors in health and diseases. Signal Transduct Target Ther 6, 1, 291, 2021, https://doi.org/10.1038/s41392-021-00687-0

[9]Nouri, Y., Weinkove, R., Perret, R. T-cell intrinsic Toll-like receptor signaling: implications for cancer immunotherapy and CAR T-cells. J ImmunoTher Cancer 2021, 9:e003065, https://doi.org/10.1136/jitc-2021-003065.

[10]Sun, H., Li, Y., Zhang, P. et al. Targeting toll-like receptor 7/8 for immunotherapy: recent advances and prospectives. Biomark Res 10, 89, 2022, https://doi.org/10.1186/s40364-022-00436-7

[11]Chuang, T.H., Ulevitch, R.J. Cloning and characterization of a sub-family of human toll-like receptors: hTLR7, hTLR8 and hTLR9. Eur Cytokine Netw 2000, 11(3), 372-378. PMID: 11022120.

[12]Akira, S., Takeda, K. Toll-like receptor signalling. Nat Rev Immunol 2004, 4(7), 499-511, https://doi.org/10.1038/nri1391.

[13]Created with BioRender.com

[14]Tanji, H., Ohto, U., Shibata, T., Miyake, K., Shimizu, T. Structural Reorganization of the Toll-Like Receptor 8 Dimer Induced by Agonistic Ligands. Science 2013, 339(6126), 1426-1429, https://doi.org/10.1126/science.1229159.

[15]Kawai, T., Akira, S. The role of pattern-recognition receptors in innate immunity: update on Toll-like receptors. Nat Immunol 11, 5, 373-384, 2010, https://doi.org/10.1038/ni.1863

[16]Jiménez-Dalmaroni, M.J., Gerswhin, M.E., Adamopoulos, I.E. The critical role of toll-like receptors--From microbial recognition to autoimmunity: A comprehensive review. Autoimmun Rev 15, 1, 1-8, 2016, https://doi.org/10.1016/j.autrev.2015.08.009

[17]Cohen, P. The TLR and IL-1 signalling network at a glance. J Cell Sci 127, Pt 11, 2383-2390, 2014, https://doi.org/10.1242/jcs.149831

[18]Tan, R.S., Ho, B., Leung, B.P., Ding, J.L. TLR cross-talk confers specificity to innate immunity. Int Rev Immunol 33, 443-453, 2014, https://doi.org/10.3109/08830185.2014.921164

[19]Piccinini, A.M., Midwood, K.S. DAMPening inflammation by modulating TLR signalling. Mediators Inflamm 2010, 672395, 2010, https://doi.org/10.1155/2010/672395

[20]Chen, K., Huang, J., Gong, W., Iribarren, P., Dunlop, N.M., Wang, J.M. Toll-like receptors in inflammation, infection and cancer. Int Immunopharmacol 7, 10, 1285-1299, 2007, https://doi.org/10.1016/j.intimp.2007.05.016

[21]Urban-Wojciuk, Z., Khan, M.M., Oyler, B.L. et al. The Role of TLRs in Anti-cancer Immunity and Tumor Rejection. Front Immunol 2019, 10:2388, https://doi.org/10.3389/fimmu.2019.02388.

[22]Grimmig, T., Matthes, N., Hoeland, K. et al. TLR7 and TLR8 expression increases tumor cell proliferation and promotes chemoresistance in human pancreatic cancer. Int J Oncol 2015, 47, 857-866, https://doi.org/10.3892/ijo.2015.3069.

[23]Tanji, H., Ohto, U., Shibata, T. et al. Toll-like receptor 8 senses degradation products of single-stranded RNA. Nat Struct Mol Biol 2015, 22, 109-115, https://doi.org/10.1038/nsmb.2943.

[24]Heil, F., Hemmi, H., Hochrein, H. et al. Species-specific recognition of single-stranded RNA via toll-like receptor 7 and 8. Science 2004, 303(5663), 1526-1529, https://doi.org/10.1126/science.1093620.

[25]Ganguly, D., Chamilos, G., Lande, R. et al. Self-RNA-antimicrobial peptide complexes activate human dendritic cells through TLR7 and TLR8. J Exp Med 2009, 206(9), 1983-1994, https://doi.org/10.1084/jem.20090480.

[26]Melchjorsen, J., Jensen, S.B., Malmgaard, L. et al. Activation of innate defense against a paramyxovirus is mediated by RIG-I and TLR7 and TLR8 in a cell-type-specific manner. J Virol 2005, 79(20), 12944-12951, https://doi.org/10.1128/JVI.79.20.12944-12951.2005.

[27]Triantafilou, K., Orthopoulos, G., Vakakis, E. et al. Human cardiac inflammatory responses triggered by Coxsackie B viruses are mainly Toll-like receptor (TLR) 8-dependent. Cell Microbiol 2005, 7, 1117-1126, https://doi.org/10.1111/j.1462-5822.2005.00537.x.

[28]Moen, S.H., Ehrnström, B., Kojen, J.F., Yurchenko, M., Beckwith, K.S., Afset, J.E., Damås, J.K., Hu, Z., Yin, H., Espevik, T., Stenvik, J. Human Toll-like Receptor 8 (TLR8) Is an Important Sensor of Pyogenic Bacteria, and Is Attenuated by Cell Surface TLR Signaling. Front Immunol 10, 1209, 2019, https://doi.org/10.3389/fimmu.2019.01209

[29]McDonald, G., Cabal, N., Vannier, A. et al. Female Bias in Systemic Lupus Erythematosus is Associated with the Differential Expression of X-Linked Toll-Like Receptor 8. Front Immunol 2015, 6:457, https://doi.org/10.3389/fimmu.2015.00457.

[30]de Groot, N.G., Bontrop, R.E. COVID-19 pandemic: is a gender-defined dosage effect responsible for the high mortality rate among males? Immunogenetics 2020, 72(5), 275-277, https://doi.org/10.1007/s00251-020-01165-7.

[31]Schön, M.P., Schön, M. TLR7 and TLR8 as targets in cancer therapy. Oncogene 2008, 27(2), 190-199, https://doi.org/10.1038/sj.onc.1210913.

[32]Kawai, T., Akira, S. Innate immune recognition of viral infection. Nat Immunol 2006, 7(2), 131-137, https://doi.org/10.1038/ni1303.

[33]Farrugia, M., Baron, B. The Role of Toll-Like Receptors in Autoimmune Diseases through Failure of the Self-Recognition Mechanism. Int J Inflam 2017, 2017:8391230, https://doi.org/10.1155/2017/8391230.

[34]Cook, D.N., Pisetsky, D.S., Schwartz, D.A. Toll-like receptors in the pathogenesis of human disease. Nat Immunol 2004, 5, 975-979, https://doi.org/10.1038/ni1116.

[35]Gangloff, S.C., Guenounou, M. Toll-like receptors and immune response in allergic disease. Clin Rev Allergy Immunol 2004, 26, 115-125, https://doi.org/10.1007/s12016-004-0006-0.

[36]Tanji, H., Ohto, U., Shimizu, T. (2013). Crystal Structures of Innate Immune RNA Receptor TLR8. PF Activity Report 2013 #31, University of Tokyo. https://pfwww.kek.jp/acr2013pdf/part_a/13ah4_1.pdf, accessed 06.02.2024, 19:30

[37]Godfroy, J.I. 3rd, Roostan, M., Moroz, Y.S., Korendovych, I.V., Yin, H. Isolated Toll-like receptor transmembrane domains are capable of oligomerization. PLoS One 7(11), e48875, 2012, doi: 10.1371/journal.pone.0048875.

[38]Lannoy, V., Côté-Biron, A., Asselin, C., Rivard, N. TIRAP, TRAM, and Toll-Like Receptors: The Untold Story. Mediators Inflamm 2023, 2899271, 2023, https://doi.org/10.1155/2023/2899271

[39]Protein Data Bank. Crystal structure of human TLR8 in complex with ssRNA40 (PDB ID: 4R08). 2015, https://doi.org/10.2210/pdb4r08/pdb

[40]Protein Data Bank. Crystal structure of the Toll/interleukin-1 receptor (TIR) domain of human IL-1RAPL (PDB ID: 1T3G). 2005, https://doi.org/10.2210/pdb1t3g/pdb

[41]Khan, J.A., Brint, E.K., O'Neill, L.A., Tong, L. Crystal structure of the Toll/interleukin-1 receptor domain of human IL-1RAPL. J. Biol. Chem. 279, 30, 31664-70, 2004, https://doi.org/10.1074/jbc.M403434200

[42]Tanji, H., Ohto, U., Motoi, Y. et al. Autoinhibition and relief mechanism by the proteolytic processing of Toll-like receptor 8. Proc Natl Acad Sci USA 2016, 113(11), 3012-3017, https://doi.org/10.1073/pnas.1516000113.

[43]Yim, M. The Role of Toll-Like Receptors in Osteoclastogenesis. J Bone Metab 27, 4, 227-235, 2020, https://doi.org/10.11005/jbm.2020.27.4.227

[44]Ishii, N., Funami, K., Tatematsu, M. et al. Endosomal localization of TLR8 confers distinctive proteolytic processing on human myeloid cells. J Immunol 2014, 193(10), 5118-5128, https://doi.org/10.4049/jimmunol.1401375.

[45]Lee, J., Tian, Y., Chan, S.T. et al. TNF-alpha Induced by Hepatitis C Virus via TLR7 and TLR8 in Hepatocytes Supports Interferon Signaling via an Autocrine Mechanism. PLoS Pathog 2015, 11, https://doi.org/10.1371/journal.ppat.1004937.

[46]Ma, Y., Li, J., Chiu, I. et al. Toll-like receptor 8 functions as a negative regulator of neurite outgrowth and inducer of neuronal apoptosis. J Cell Biol 2006, 175, 209-215, https://doi.org/10.1083/jcb.200606016.

[47]Protein Data Bank. Crystal structure of human TLR8 with an uncleaved Z-loop (PDB ID: 5HDH). 2016, https://doi.org/10.2210/pdb5hdh/pdb

[48]Sakaniwa, K., Shimizu, T. Targeting the innate immune receptor TLR8 using small-molecule agents. Acta Crystallogr D Struct Biol 76, Pt 7, 621-629, 2020, https://doi.org/10.1107/S2059798320006518

[49]Greulich, W., Wagner, M., Gaidt, M.M., Stafford, C., Cheng, Y., Linder, A., Carell, T., Hornung, V. TLR8 Is a Sensor of RNase T2 Degradation Products. Cell 179(6), 1264-1275.e13, 2019, doi: 10.1016/j.cell.2019.11.001.

[50]Protein Data Bank. Crystal structure of human TLR8 in complex with Resiquimod (R848) crystal form 3 (PDB ID: 3W3N). 2013, https://doi.org/10.2210/pdb3w3n/pdb

[51]Martínez-Espinoza, I., Guerrero-Plata, A. The Relevance of TLR8 in Viral Infections. Pathogens 11, 2, 134, 2022, https://doi.org/10.3390/pathogens11020134

[52]Nimma, S., Gu, W., Maruta, N. et al. Structural Evolution of TIR-Domain Signalosomes. Front Immunol 12, 784484, 2021, https://doi.org/10.3389/fimmu.2021.784484

[53]O'Neill, L.A., Bowie, A.G. The family of five: TIR-domain-containing adaptors in Toll-like receptor signalling. Nat Rev Immunol 7(5), 353-64, 2007, doi: 10.1038/nri2079.

[54]Landström, M. The TAK1-TRAF6 signalling pathway. Int J Biochem Cell Biol 42, 5, 585-589, 2010, https://doi.org/10.1016/j.biocel.2009.12.023

[55]Deliz-Aguirre, R., Cao, F., Gerpott, F.H.U., Auevechanichkul, N., Chupanova, M., Mun, Y., Ziska, E., Taylor, M.J. MyD88 oligomer size functions as a physical threshold to trigger IL1R Myddosome signaling. J Cell Biol 220, 7, e202012071, 2021, https://doi.org/10.1083/jcb.202012071

[56]Kawasaki, T., Kawai, T. Toll-like receptor signaling pathways. Front Immunol 2014, 5:461, https://doi.org/10.3389/fimmu.2014.00461.

[57]Takaesu, G., Kishida, S., Hiyama, A., Yamaguchi, K., Shibuya, H., Irie, K., Ninomiya-Tsuji, J., Matsumoto, K. TAB2, a novel adaptor protein, mediates activation of TAK1 MAPKKK by linking TAK1 to TRAF6 in the IL-1 signal transduction pathway. Mol Cell 5, 4, 649-658, 2000, https://doi.org/10.1016/s1097-2765(00)80244-0

[58]Walsh, M.C., Lee, J., Choi, Y. Tumor necrosis factor receptor-associated factor 6 (TRAF6) regulation of development, function, and homeostasis of the immune system. Immunol Rev 266, 1, 72-92, 2015, https://doi.org/10.1111/imr.12302

[59]Gorden, K.B., Gorski, K.S., Gibson, S.J. et al. Synthetic TLR agonists reveal functional differences between human TLR7 and TLR8. J Immunol 2005, 174(3), 1259-1268, https://doi.org/10.4049/jimmunol.174.3.1259.

[60]Karlson, P., Doenecke, D., Koolman, J., Fuchs, G., Gerok, W. Karlsons Biochemie und Pathobiochemie. Georg Thieme Verlag, 15th edition, 2005, ISBN: 3-13-357815-4, p.684-686.

[61]Ayithan, N., Tang, L., Tan, S.K. et al. Follicular Helper T (TFH) Cell Targeting by TLR8 Signaling For Improving HBsAg-Specific B Cell Response In Chronic Hepatitis B Patients. Front Immunol 12, 735913, 2021, https://doi.org/10.3389/fimmu.2021.735913

[62]Study of Oral TLR8 Agonist Selgantolimod on HBsAg in Participants With Both Chronic Hepatitis B and HIV, Clinicaltrials.gov identifier: NCT05551273, https://clinicaltrials.gov/study/NCT05551273?cond=TLR8&rank=1, accessed 03.18.2024, 14:00

[63]Wang, C.H., Eng, H.L., Lin, K.H., Chang, C.H., Hsieh, C.A., Lin, Y.L., Lin, T.M. TLR7 and TLR8 gene variations and susceptibility to hepatitis C virus infection. PLoS One 6, 10, e26235, 2011, https://doi.org/10.1371/journal.pone.0026235

[64]Li, S., Yao, J.C., Li, J.T., Schmidt, A.P., Link, D.C. TLR7/8 agonist treatment induces an increase in bone marrow resident dendritic cells and hematopoietic progenitor expansion and mobilization. Exp Hematol 96, 35-43.e7, 2021, https://doi.org/10.1016/j.exphem.2021.02.001

[65]Zhang, P., Cox, C.J., Alvarez, K.M., Cunningham, M.W. Cutting edge: cardiac myosin activates innate immune responses through TLRs. J Immunol 183, 1, 27-31, 2009, https://doi.org/10.4049/jimmunol.0800861

[66]Gambuzza, M.E., Sofo, V., Salmeri, F.M., Soraci, L., Marino, S., Bramanti, P. Toll-like receptors in Alzheimer's disease: a therapeutic perspective. CNS Neurol Disord Drug Targets 13, 9, 1542-1558, 2014, https://doi.org/10.2174/1871527313666140806124850

[67]Moller-Larsen, S., Nyegaard, M., Haagerup, A. et al. Association analysis identifies TLR7 and TLR8 as novel risk genes in asthma and related disorders. Thorax 63, 1064–1069, 2008, https://doi.org/10.1136/thx.2007.094128

[68]Niazi, S.K., Mariam, Z. Computer-Aided Drug Design and Drug Discovery: A Prospective Analysis. Pharmaceuticals (Basel) 17, 22, 2023, https://doi.org/10.3390/ph17010022

[69]Langer, T., Wolber, G. Pharmacophore definition and 3D searches. Drug Discov Today Technol 1, 3, 203-207, 2004, https://doi.org/10.1016/j.ddtec.2004.11.015

[70]Yu, W., MacKerell, A.D. Jr. Computer-Aided Drug Design Methods. Methods Mol Biol 1520, 85-106, 2017, https://doi.org/10.1007/978-1-4939-6634-9_5

[71]Poongavanam, V., Ramaswamy, V. Computational Drug Discovery: Methods and Applications. Wiley-VCH GmbH, 2024, ISBN: 978-3-5278-4074-8, Volume 1, Part 2

[72]Poongavanam, V., Ramaswamy, V. Computational Drug Discovery: Methods and Applications. Wiley-VCH GmbH, 2024, ISBN: 978-3-5278-4074-8, Volume 1, Part 3

[73]Poongavanam, V., Ramaswamy, V. Computational Drug Discovery: Methods and Applications. Wiley-VCH GmbH, 2024, ISBN: 978-3-5278-4074-8, Volume 2, Part 6

[74]Klebe, G., Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen, Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.102f

[75]Lees, J.A., Dias, J.M., Han, S. Applications of Cryo-EM in small molecule and biologics drug design. Biochem Soc Trans. 49(6): 2627–2638, 2021, https://doi.org/10.1042/BST20210444

[76]Vázquez, J., López, M., Gibert, E. et al. Merging Ligand-Based and Structure-Based Methods in Drug Discovery: An Overview of Combined Virtual Screening Approaches. Molecules 25, 4723, 2020, https://doi.org/10.3390/molecules25204723

[77]Ross, G.A., Morris, G.M., Biggin, P.C. One Size Does Not Fit All: The Limits of Structure-Based Models in Drug Discovery. J Chem Theory Comput 9, 4266-4274, 2013, https://doi.org/10.1021/ct4004228

[78]Khedkar, S.A., Malde, A.K., Coutinho, E.C., Srivastava, S. Pharmacophore modeling in drug discovery and development: an overview. Med Chem 3, 2, 187-197, 2007, https://doi.org/10.2174/157340607780059521

[79]Klebe, G., Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen, Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.249ff

[80]Seidel, T., Wieder, O., Garon, A., Langer, T. Applications of the Pharmacophore Concept in Natural Product inspired Drug Design. Mol Inform 39(11), e2000059, 2020, doi: 10.1002/minf.202000059.

[81]Seidel, T., Ibis, G., Bendix, F., Wolber, G. Strategies for 3D pharmacophore-based virtual screening. Drug Discov Today Technol 7(4), e221-e228, 2010, doi: 10.1016/j.ddtec.2010.11.004.

[82]Giordano, D., Biancaniello, C., Argenio, M.A., Facchiano, A. Drug Design by Pharmacophore and Virtual Screening Approach. Pharmaceuticals (Basel) 15(5), 646, 2022, doi: 10.3390/ph15050646.

[83]Graves, A.P., Brenk, R., Shoichet, B.K. Decoys for docking. J Med Chem 48(11), 3714-28, 2005, doi: 10.1021/jm0491187.

[84]Stein, R.M., Yang, Y., Balius, T.E., O'Meara, M.J., Lyu, J.K., Young, J., Tang, K., Shoichet, B.K., Irwin, J.J. Property-Unmatched Decoys in Docking Benchmarks. J. Chem. Inf. Model. 2021, https://doi.org/10.1021/acs.jcim.0c00598

[85]Klebe, G., Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen, Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.227ff

[86]Poongavanam, V., Ramaswamy, V. Computational Drug Discovery: Methods and Applications. Wiley-VCH GmbH, 2024, ISBN: 978-3-5278-4074-8, Volume 1, Part 1

[87]Poongavanam, V., Ramaswamy, V. Computational Drug Discovery: Methods and Applications. Wiley-VCH GmbH, 2024, ISBN: 978-3-5278-4074-8, Volume 2, Part 5

[88]Klebe, G. Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen, Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.262

[89]Clark, D.E., Mannhold, R., Kubinyi, H., Timmerman, H. Evolutionary Algorithms in Molecular Design - Methods and Principles in Medicinal Chemistry, WILEY-VCH Verlag GmbH, 2000, ISBN: 3-527-30155-0, p.31ff

[90]Rothenaigner, I., Hadian, K. Brief Guide: Experimental Strategies for High-Quality Hit Selection from Small-Molecule Screening Campaigns. SLAS Discov 26, 851-854, 2021, https://doi.org/10.1177/24725552211008862

[91]Klebe, G., Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen, Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.49f

[92]Koshland, D.E. Application of a Theory of Enzyme Specificity to Protein Synthesis. Proc Natl Acad Sci U S A 44, 98–104, 1958, https://doi.org/10.1073/pnas.44.2.98

[93]Böhm, H.-J., Schneider, G. Protein ligand interactions - From molecular recognition to drug design. Methods and principles in medicinal chemistry; Vol. 19. Wiley-VCH, 2003, ISBN: 3-527-30521-1, p.3.

[94]Klebe, G., Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen, Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.54ff

[95]Klebe, G. Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen. Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.116.

[96]Johnson, K.A., Goody, R.S. The Original Michaelis Constant: Translation of the 1913 Michaelis–Menten Paper. Biochemistry 50(39), 8264–8269, 2011, doi: 10.1021/bi201284u.

[97]Du, X., Li, Y., Xia, Y.L. et al. Insights into Protein-Ligand Interactions: Mechanisms, Models, and Methods. Int J Mol Sci 17, 144, 2016, https://doi.org/10.3390/ijms17020144

[98]Seo, S., Choi, J., Park, S., Ahn, J. Binding affinity prediction for protein-ligand complex using deep attention mechanism based on intermolecular interactions. BMC Bioinformatics 22, 542, 2021, https://doi.org/10.1186/s12859-021-04466-0

[99]Klebe, G., Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen, Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.63ff

[100]Wermuth, C.G., Aldous, D., Raboisson, P., Rognan, D. The Practice of Medicinal Chemistry, 4th edition. Academic Press, 2015, ISBN: 978-0-12-417205-0, p.21.

[101]Klebe, G., Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen, Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.66

[102]Boström, J., Norrby, P.O., Liljefors, T. Conformational energy penalties of protein-bound ligands. J Comput Aided Mol Des 12, 383-396, 1998, https://doi.org/10.1023/a:1008007507641

[103]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. Wiley-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.94ff

[104]Halgren, T.A. Merck molecular force field. I. Basis, form, scope, parameterization, and performance of MMFF94. J Comput Chem 17, 490–519, 1996, https://doi.org/10.1002/(sici)1096-987x(199604)17:5/6<490::aid-jcc1>3.0.co;2-p

[105]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. Wiley-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.279ff

[106]Jász, A., Rák, A., Ladjánszki, I., Cserey., G. Optimized GPU implementation of Merck

Molecular Force Field and Universal Force Field. Journal of Molecular Structure, Volume 1188, 227-233, 2019, https://doi.org/10.1016/j.molstruc.2019.04.007

[107]Lin, FY., MacKerell, A. D., Force fields for small molecules, Methods Mol Biol. 2022: 21–54., 2019, https://doi.org/10.1007/978-1-4939-9608-7_2

[108]Kuentz, M.T., Arnold, Y. Influence of molecular properties on oral bioavailability of lipophilic drugs – Mapping of bulkiness and different measures of polarity. Pharm Dev Technol 14(3), 312–320, 2009, doi: 10.1080/10837450802626296.

[109]Klebe, G. Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen. Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.119f.

[110]Klebe, G. Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen. Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.291-292.

[111]Klebe, G. Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen. Spektrum Akademischer Verlag Heidelberg, 2. Auflage, 2009, ISBN: 978-3-8274-2046-6, p.292.

[112]Lipinski, C.A., Lombardo, F., Dominy, B.W., Feeney, P.J. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. Advanced Drug Delivery Reviews, Volume 46, Issues 1–3, Pages 3-26, 2001, https://doi.org/10.1016/S0169-409X(00)00129-0

[113]Witten, I.H., Frank, E., Hall, M.A. Data Mining – Practical Machine Learning Tools and Techniques, 3rd Edition. Morgan Kaufmann, 2011, ISBN: 978-0-12-374856-0, p.xxi

[114]Müller, A.C., Guido, S. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, 2018, ISBN: 978-9-35-213457-1, p.1-5

[115]Shi, Y.-F., Yang, Z.-X., Ma, S., Kang, P.-L., Shang, C., Hu, P., Liu, Z.-P. Machine Learning for Chemistry: Basics and Applications. Engineering 27, 70-83, 2023, doi: 10.1016/j.eng.2023.04.013.

[116]Lee, C.H., Huang, H.C., Juan, H.F. Reviewing ligand-based rational drug design: the search for an ATP synthase inhibitor. Int J Mol Sci 12(8), 5304-18, 2011, doi: 10.3390/ijms12085304.

[117]Müller, A.C., Guido, S. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, 2018, ISBN: 978-9-35-213457-1, p.25

[118]Witten, I.H., Frank, E., Hall, M.A. Data Mining – Practical Machine Learning Tools and Techniques, 3rd Edition. Morgan Kaufmann, 2011, ISBN: 978-0-12-374856-0, p.40

[119]Bruce, P., Bruce, A., Gedeck, P. Practical Statistics for Data Scientists - 50+ Essential Concepts Using R and Python, 2nd Edition. O'Reilly Media, 2020, ISBN: 978-1-492-07294-2, p.1-36

[120]Witten, I.H., Frank, E., Hall, M.A. Data Mining – Practical Machine Learning Tools and Techniques, 3rd Edition. Morgan Kaufmann, 2011, ISBN: 978-0-12-374856-0, p.51ff

[121]Müller, A.C., Guido, S. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, 2018, ISBN: 978-9-35-213457-1, p. 45-68

[122]Witten, I.H., Frank, E., Hall, M.A. Data Mining – Practical Machine Learning Tools and Techniques, 3rd Edition. Morgan Kaufmann, 2011, ISBN: 978-0-12-374856-0, p.61-83

[123]Müller, A.C., Guido, S. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, 2018, ISBN: 978-9-35-213457-1, p. 70-92

[124]Atteia, G.E., Abdelsamee, N., Mengash, H. Evaluation of using Parametric and Non-parametric Machine Learning Algorithms for Covid-19 Forecasting. Int J Adv Comput Sci Appl 12, 1071, 2021, https://doi.org/10.14569/IJACSA.2021.0121071

[125]Witten, I.H., Frank, E., Hall, M.A. Data Mining – Practical Machine Learning Tools and Techniques, 3rd Edition. Morgan Kaufmann, 2011, ISBN: 978-0-12-374856-0, p.352-362.

[126]Natekin, A., Knoll, A. Gradient boosting machines, a tutorial. Front Neurorobot 7, 21, 2013, doi: 10.3389/fnbot.2013.00021.

[127]Ramsundar, B., Eastman, P., Walters, P., Pande, V. Deep Learning for the Life Sciences: Applying Deep Learning to Genomics, Microscopy, Drug Discovery, and More. O'Reilly Media, 2019, ISBN: 978-1-49-203983-9, p.41-51

[128]Moriwaki, H., Tian, YS., Kawashita, N. et al. Mordred: a molecular descriptor calculator. J Cheminform 10, 4, 2018. https://doi.org/10.1186/s13321-018-0258-y

[129]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods, WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.76-79

[130]Brown, N. In Silico Medicinal Chemistry: Computational Methods to Support Drug Design. The Royal Society of Chemistry, 2016, ISBN: 978-1-78262-260-4, p. 55-60

[131]Müller, A.C., Guido, S. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, 2018, ISBN: 978-9-35-213457-1, p. 26-29

[132]Müller, A.C., Guido, S. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, 2018, ISBN: 978-9-35-213457-1, p.251-275.

[133]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.412.

[134]Witten, I.H., Frank, E., Hall, M.A. Data Mining – Practical Machine Learning Tools and Techniques, 3rd Edition. Morgan Kaufmann, 2011, ISBN: 978-0-12-374856-0, p.147ff.

[135]Witten, I.H., Frank, E., Hall, M.A. Data Mining – Practical Machine Learning Tools and Techniques, 3rd Edition. Morgan Kaufmann, 2011, ISBN: 978-0-12-374856-0, p.159ff.

[136]Yang, L., Shami, A. On Hyperparameter Optimization of Machine Learning Algorithms: Theory and Practice. Neurocomputing, 2020, doi: 10.1016/j.neucom.2020.07.061.

[137]Witten, I.H., Frank, E., Hall, M.A. Data Mining – Practical Machine Learning Tools and Techniques, 3rd Edition. Morgan Kaufmann, 2011, ISBN: 978-0-12-374856-0, p.180ff.

[138]Wang, X., Chen, Y., Zhang, S., Deng, J.N. Molecular dynamics simulations reveal the selectivity mechanism of structurally similar agonists to TLR7 and TLR8. PLoS One 17, 4, e0260565, 2022, https://doi.org/10.1371/journal.pone.0260565

[139]Wolber, G., Langer, T. LigandScout: 3-D Pharmacophores Derived from Protein-Bound Ligands and Their Use as Virtual Screening Filters. J Chem Inf Model 45, 160–169, 2005, https://doi.org/10.1021/ci049885e

[140]Inte:Ligand GmbH (2010), LigandScout User Manual, Version 2010-04-15, Inte:Ligand GmbH, https://www.inteligand.com/ligandscout3/downloads/ligandscout-manual-2010-04-15.pdf, accessed 01.30.2024, 21:40 h

[141]Molecular Operating Environment (MOE), 2022.02 Chemical Computing Group ULC, 910-1010 Sherbrooke St. W., Montreal, QC H3A 2R7, Canada, 2024.

[142]Da, C., Kireev, D. Structural protein-ligand interaction fingerprints (SPLIF) for structure-based virtual screening: method and benchmark study. J Chem Inf Model 54(9), 2555-2561, 2014, doi: 10.1021/ci500319f.

[143]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.470.

[144]Cambridge Crystallographic Data Centre. GOLD Software, accessed 03.04.2024, 17:50. https://www.ccdc.cam.ac.uk/solutions/software/gold/.

[145]Cambridge Crystallographic Data Centre. About Us, accessed 03.04.2024, 17:55. https://www.ccdc.cam.ac.uk/about-us/.

[146]Jones, G., Willett, P., Glen, R.C., Leach, A.R., Taylor, R. Development and Validation of a Genetic Algorithm for Flexible Docking. J. Mol. Biol. 267, 727-748, 1997, https://doi.org/10.1006/jmbi.1996.0897

[147]Hawkins, P.C.D., Skillman, A.G., Nicholls, A. Comparison of Shape-Matching and Docking as Virtual Screening Tools. J Med Chem 50, 74-82, 2007. doi: 10.1021/jm0603365

[148]Hawkins, P.C.D., Stahl, G. Ligand-Based Methods in GPCR Computer-Aided Drug Design. Methods Mol Biol. 2018;1705:365-374., 2018, doi: 10.1007/978-1-4939-7465-8_18.

[149]Mills, J.E., Dean, P.M. Three-dimensional hydrogen-bond geometry and probability information from a crystal survey. J Comput Aided Mol Des 10(6), 607-622, 1996, doi: 10.1007/BF00134183.

[150]OpenEye Scientific Software. ROCS Documentation, accessed 02.27.2024, 16:55. https://docs.eyesopen.com/applications/rocs/rocsreport.html.

[151]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.247.

[152]Leach, A.R. Molecular Modelling – Principles and Applications. Pearson Education, 2nd edition, 2001, ISBN: 0-582-38210-6, p.676-678.

[153]Klebe, G. Wirkstoffdesign – Entwurf und Wirkung von Arzneistoffen. Spektrum Akademischer Verlag Heidelberg, 2nd edition, 2009, ISBN: 978-3-8274-2046-6, p.257.

[154]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.92.

[155]Molecular Networks GmbH. CORINA 3.0 Documentation (User Manual), accessed 02.29.2024, 19:15. https://ics.uci.edu/~dock/manuals/corina_manual.pdf.s

[156]O'Boyle, N.M., Banck, M., James, C.A., Morley, C., Vandermeersch, T., Hutchison, G.R. Open Babel: An open chemical toolbox. J Cheminform 3, 33, 2011, doi: 10.1186/1758-2946-3-33.

[157]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.70.

[158]Stein, R.M., Yang, Y., Balius, T.E. et al. Comparison of Shape-Matching and Docking as Virtual Screening Tools. J Chem Inf Model 61(2), 699-714, 2021, doi: 10.1021/acs.jcim.0c00598.

[159]Zdrazil, B., Felix, E., Hunter, F. et al. The ChEMBL Database in 2023: a drug discovery platform spanning multiple bioactivity data types and time periods. Nucleic Acids Res 52(D1), D1180-D1192, 2024, doi: 10.1093/nar/gkad1004.

[160]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.58.

[161]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.198ff.

[162]Enamine Ltd. About Us, accessed 03.04.2024, 16:50. https://enamine.net/about-us.

[163]OpenAI. ChatGPT (Versions from October 2023 to March 2024) [Large language model]. https://chat.openai.com/chat.

[164]OpenAI. ChatGPT Help Documentation, accessed 03.05.2024, 12:50. https://help.openai.com/en/collections/3742473-chatgpt.

[165]Jain, S. ChatGPT is a new AI chatbot that can find mistakes in your code or write a story for you. Business Insider India, December 5, 2022, accessed 03.05.2024, 13:00. https://www.businessinsider.in/tech/news/what-is-chatgpt-and-how-does-it-work/articleshow/95994901.cms.

[166]BioRender. https://biorender.com, accessed 04.30.2024, 15:20.

[167]DeepL Translator. https://deepl.com, accessed 04.30.2024, 16:40.

[168]ChemDraw. https://revvitysignals.com/products/research/chemdraw, accessed 05.21.2024, 12:15

[169]Oliphant, T.E. Python for Scientific Computing. Comput Sci Eng 9(3), 10-20, 2007, doi: 10.1109/MCSE.2007.58.

[170]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.222.

[171]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.484f.

[172]RDKit Documentation. https://www.rdkit.org/new_docs/source/rdkit.Chem.rdMolAlign.html, accessed 03.01.2024, 17:30.

[173]RDKit Documentation. https://www.rdkit.org/docs/source/rdkit.Chem.MCS.html, accessed 04.30.2024, 15:10.

[174]Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E. Scikit-learn: Machine Learning in Python. J Mach Learn Res 12, pp. 2825-2830, 2011. https://doi.org/10.48550/arXiv.1201.0490

[175]Hunter, J.D. Matplotlib: A 2D Graphics Environment. Comput Sci Eng 9(3), 90-95, 2007. doi: 10.1109/MCSE.2007.55

[176]McKinney, W. Data Structures for Statistical Computing in Python. Python in Science Conference, 2010, doi: 10.25080/Majora-92bf1922-00a.

[177]Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357-362, 2020, doi: 10.1038/s41586-020-2649-2.

[178]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.44ff.

[179]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.52-58.

[180]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.58ff

[181]wwPDB. wwPDB Documentation, accessed 05.07.2024, 17:45. https://www.wwpdb.org/documentation/file-format-content/format33/sect9.html.

[182]Voet, D., Voet, J.G., Pratt, C.W. Lehrbuch der Biochemie. WILEY-VCH Verlag GmbH & Co. KGaA, 2002, ISBN: 3-527-30519-X, p.38

[183]Cappel, D., Dixon, S.L., Sherman, W. et al. Exploring conformational search protocols for ligand-based virtual screening and 3-D QSAR modeling. J. Comput. Aided Mol. Des. 29, 165–182, 2015, https://doi.org/10.1007/s10822-014-9813-4

[184]Gasteiger, J., Engel, T. Chemoinformatics – Basic concepts and Methods. WILEY-VCH Verlag GmbH & Co. KGaA, 2018, ISBN: 978-3-527-33109-3, p.96

[185]Luo, L., Zhong, A., Wang, Q., Zheng, T. Structure-Based Pharmacophore Modeling, Virtual Screening, Molecular Docking, ADMET, and Molecular Dynamics (MD) Simulation of Potential Inhibitors of PD-L1 from the Library of Marine Natural Products. Mar Drugs 20(1), 29, 2021, doi: 10.3390/md20010029.

[186]Wieder, M., Perricone, U., Seidel, T. et al. Comparing pharmacophore models derived from crystal structures and from molecular dynamics simulations. Monatsh Chem 147, 553-563, 2016, doi: 10.1007/s00706-016-1674-1.

[187]Triballeau, N., Acher, F., Brabet, I., Pin, J.P., Bertrand, H.O. Virtual Screening Workflow Development Guided by the “Receiver Operating Characteristic” Curve Approach. Application to High-Throughput Docking on Metabotropic Glutamate Receptor Subtype 4. J. Med. Chem. 48, 7, 2534–2547, 2005, https://doi.org/10.1021/jm049092j

[188]Koes, D.R., Camacho, C.J. Pharmer: efficient and exact pharmacophore search. J. Chem. Inf. Model. 51, 6, 1307-14, 2011, https://doi.org/10.1021/ci200097m

[189]Petoukhov, M.V., Eady, N.A., Brown, K.A., Svergun, D.I. Addition of missing loops and domains to protein models by x-ray solution scattering. Biophys J. 83, 6, 3113-25, 2002, https://doi.org/10.1016/S0006-3495(02)75315-0

[190]Fiser, A., Do, R.K., Sali, A. Modeling of loops in protein structures. Protein Sci. 9, 9, 1753-73, 2000, https://doi.org/10.1110/ps.9.9.1753

[191]Protein Data Bank. Thermotoga maritima family 1 Glycoside hydrolase complexed with Carba-Cyclophellitol transition state mimic (PDB ID: 5N6S). 2017, https://doi.org/10.2210/pdb5n6s/pdb

[192]Beenakker, T.J.M., Wander, D.P.A., Offen, W.A., Artola, M., Raich, L., Ferraz, M.J., Li, K.Y., Houben, J.H.P.M., van Rijssel, E.R., Hansen, T., van der Marel, G.A., Codée, J.D.C., Aerts, J.M.F.G., Rovira, C., Davies, G.J., Overkleeft, H.S. Carba-cyclophellitols Are Neutral Retaining-Glucosidase Inhibitors. J. Am. Chem. Soc. 139, 19, 6534-6537, 2017, https://doi.org/10.1021/jacs.7b01773

[193]Sund-Levander, M., Forsberg, C., Wahren, L.K. Normal oral, rectal, tympanic and axillary body temperature in adult men and women: a systematic literature review. Scand J Caring Sci. 16, 2, 122-8, 2002, https://doi.org/10.1046/j.1471-6712.2002.00069.x

[194]GOLD user guide. https://www.ccdc.cam.ac.uk/media/Documentation/0C5D99BC-7CC3-49B6-8319-06BEA8CA342D/GOLD_User_Guide_2020_1.pdf, accessed 05.21.2024, 13:35

[195]Korb, O., Stützle, T., Exner, T.E. An ant colony optimization approach to flexible protein–ligand docking. Swarm Intell. 1, 115–134, 2007, https://doi.org/10.1007/s11721-007-0006-9

[196]Baell, J.B., Holloway, G.A. New Substructure Filters for Removal of Pan Assay Interference Compounds (PAINS) from Screening Libraries and for Their Exclusion in Bioassays. J. Med. Chem. 53, 7, 2719–2740, 2010, https://doi.org/10.1021/jm901137j

[197]Yoo, E., Salunke, D.B., Sil, D., Guo, X., Salyer, A.C., Hermanson, A.R., Kumar, M., Malladi, S.S., Balakrishna, R., Thompson, W.H., Tanji, H., Ohto, U., Shimizu, T., David, S.A. Determinants of activity at human Toll-like receptors 7 and 8: quantitative structure-activity relationship (QSAR) of diverse heterocyclic scaffolds. J. Med. Chem. 57, 19, 7955-70, 2014, https://doi.org/10.1021/jm500744f

[198]Beesu, M., Caruso, G., Salyer, A.C., Khetani, K.K., Sil, D., Weerasinghe, M., Tanji, H., Ohto, U., Shimizu, T., David, S.A. Structure-Based Design of Human TLR8-Specific Agonists with Augmented Potency and Adjuvanticity. J. Med. Chem. 58, 19, 7833-49, 2015, https://doi.org/10.1021/acs.jmedchem.5b01087

[199]Tanji, H., Ohto, U., Shimizu, T. Crystal structure of human TLR8 in complex with MB-343. To be published, crystal structure available on PDB, structure Deposited: 2015-09-27 Released: 2016-10-05, https://doi.org/10.2210/pdb5AZ5/pdb

[200]Beesu, M., Caruso, G., Salyer, A.C.D., Khetani, K.K., Sil, D., Weerasinghe, M., Tanji, H., Ohto, U., Shimizu, T., David, S.A. Structure-based Design of Human TLR8-specific Agonists with Augmented Potency and Adjuvanticity. To be published, crystal structure available on PDB, structure Deposited: 2015-07-03 Released: 2016-07-20, https://doi.org/10.2210/pdb5AWC/pdb

[201]Protein Data Bank. Crystal structure of human TLR8 in complex with Resiquimod (R848) crystal form 2 (PDB ID: 3W3M). 2013, https://doi.org/10.2210/pdb3w3m/pdb

[202]Protein Data Bank. Crystal structure of human TLR8 in complex with Resiquimod (R848) crystal form 1 (PDB ID: 3W3L). 2013, https://doi.org/10.2210/pdb3w3l/pdb

[203]Kumari, S., Carmona, A.V., Tiwari, A.K., Trippier, P.C. Amide Bond Bioisosteres: Strategies, Synthesis, and Successes. J. Med. Chem. 63, 21, 12290-12358, 2020, https://doi.org/10.1021/acs.jmedchem.0c00530

[204]Houston, D.R., Walkinshaw, M.D. Consensus docking: improving the reliability of docking in a virtual screening context. J. Chem. Inf. Model. 53, 2, 384-90, 2013, https://doi.org/10.1021/ci300399w

[205]Eldridge, M.D., Murray, C.W., Auton, T.R., Paolini, G.V., Mee, R.P. Empirical scoring functions: I. The development of a fast empirical scoring function to estimate the binding affinity of ligands in receptor complexes. J. Comput. Aided Mol. Des. 11, 5, 425-45, 1997, https://doi.org/10.1023/a:1007996124545

[206]Breiman, L. Random Forests. Machine Learning 45, 5–32, 2001, https://doi.org/10.1023/A:1010933404324

[207]Zhang, S., Hu, Z., Tanji, H., Jiang, S., Das, N., Li, J., Sakaniwa, K., Jin, J., Bian, Y., Ohto, U., Shimizu, T., Yin, H. Small-molecule inhibition of TLR8 through stabilization of its resting state. Nat. Chem. Biol. 14, 1, 58-64, 2018, https://doi.org/10.1038/nchembio.2518

[208]Zhang, Z., Ohto, U., Shibata, T., Krayukhina, E., Taoka, M., Yamauchi, Y., Tanji, H., Isobe, T., Uchiyama, S., Miyake, K., Shimizu, T. Structural Analysis Reveals that Toll-like Receptor 7 Is a Dual Receptor for Guanosine and Single-Stranded RNA. Immunity 45, 4, 737-748, 2016, https://doi.org/10.1016/j.immuni.2016.09.011

[209]Protein Data Bank. Crystal structure of monkey TLR7 in complex with guanosine and polyU (PDB ID: 5GMF). 2016, https://doi.org/10.2210/pdb5gmf/pdb

[210]Abiri, A., Rezaei, M., Zeighami, M.H., Vaezpour, Y., Dehghan, L., Khorram, Ghahfarokhi, M. Discovery of new TLR7 agonists by a combination of statistical learning-based QSAR, virtual screening, and molecular dynamics. Inform Med Unlocked 27, 100787, 2021, https://doi.org/10.1016/j.imu.2021.100787

Appendix

I Pharmacophore-based screening and docking

Ia Complete figures of the pharmacophores in Chapter 3.2.2.3

Complete figures with exclusion volumes of the three pharmacophores featured in Chapter 3.2.2.3 are shown below.

Pharmacophore 1 (TP78-FP0):

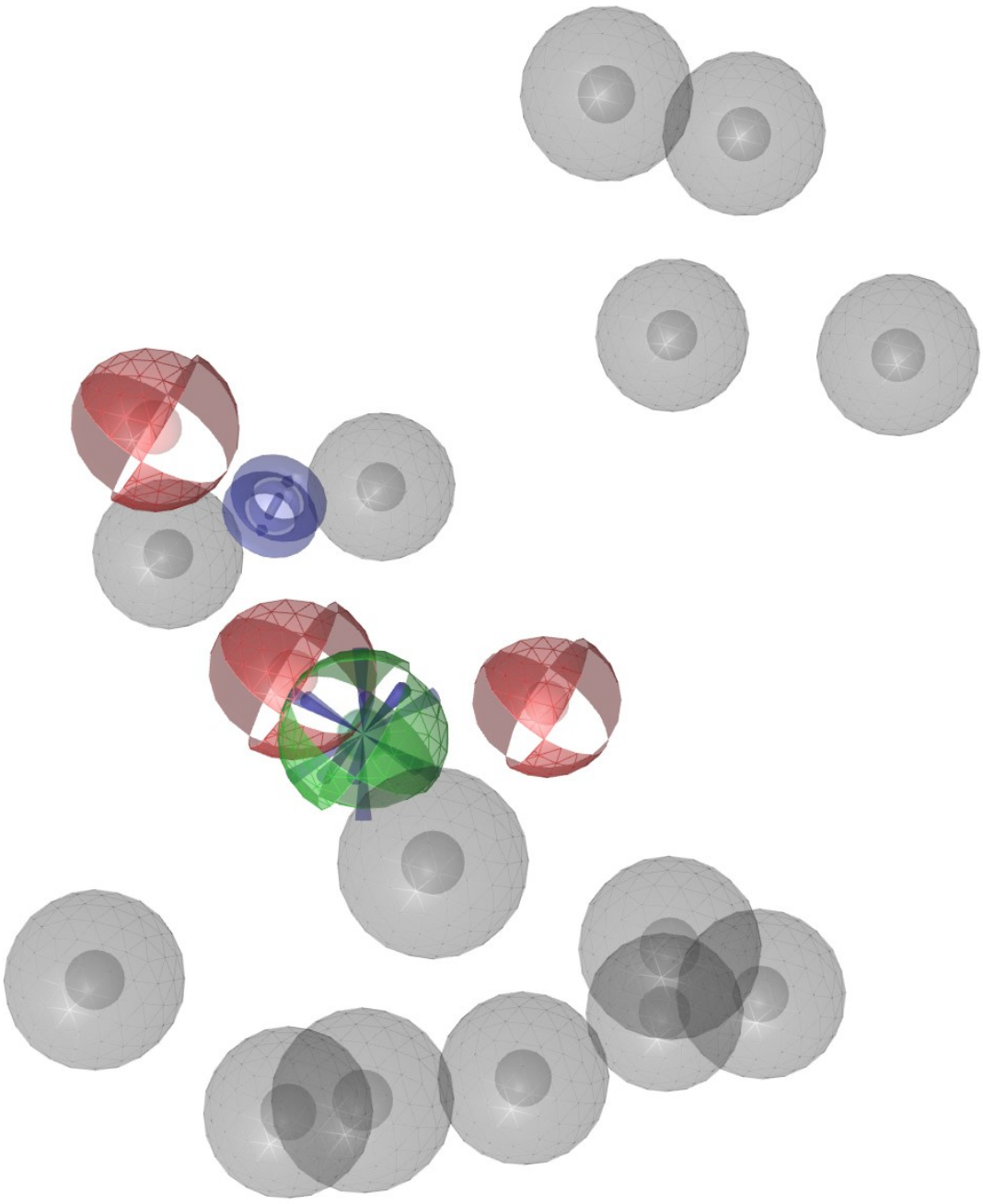


Figure 33: *Figure of pharmacophore 1 with exclusion volumes*

Pharmacophore 2 (TP81-FP6):

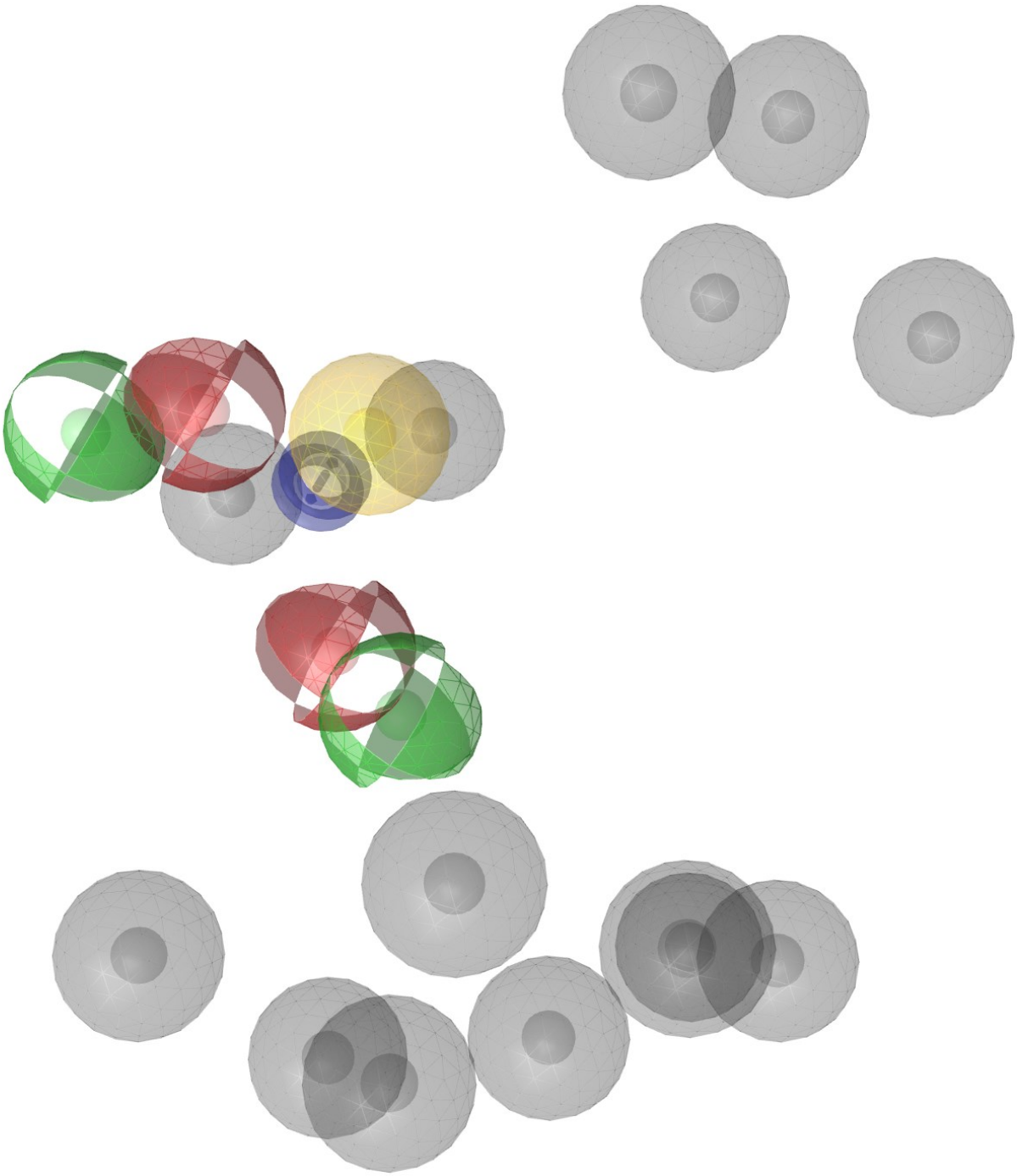


Figure 34: *Figure of pharmacophore 2 with exclusion volumes*

Pharmacophore 3 (TP125-FP45):

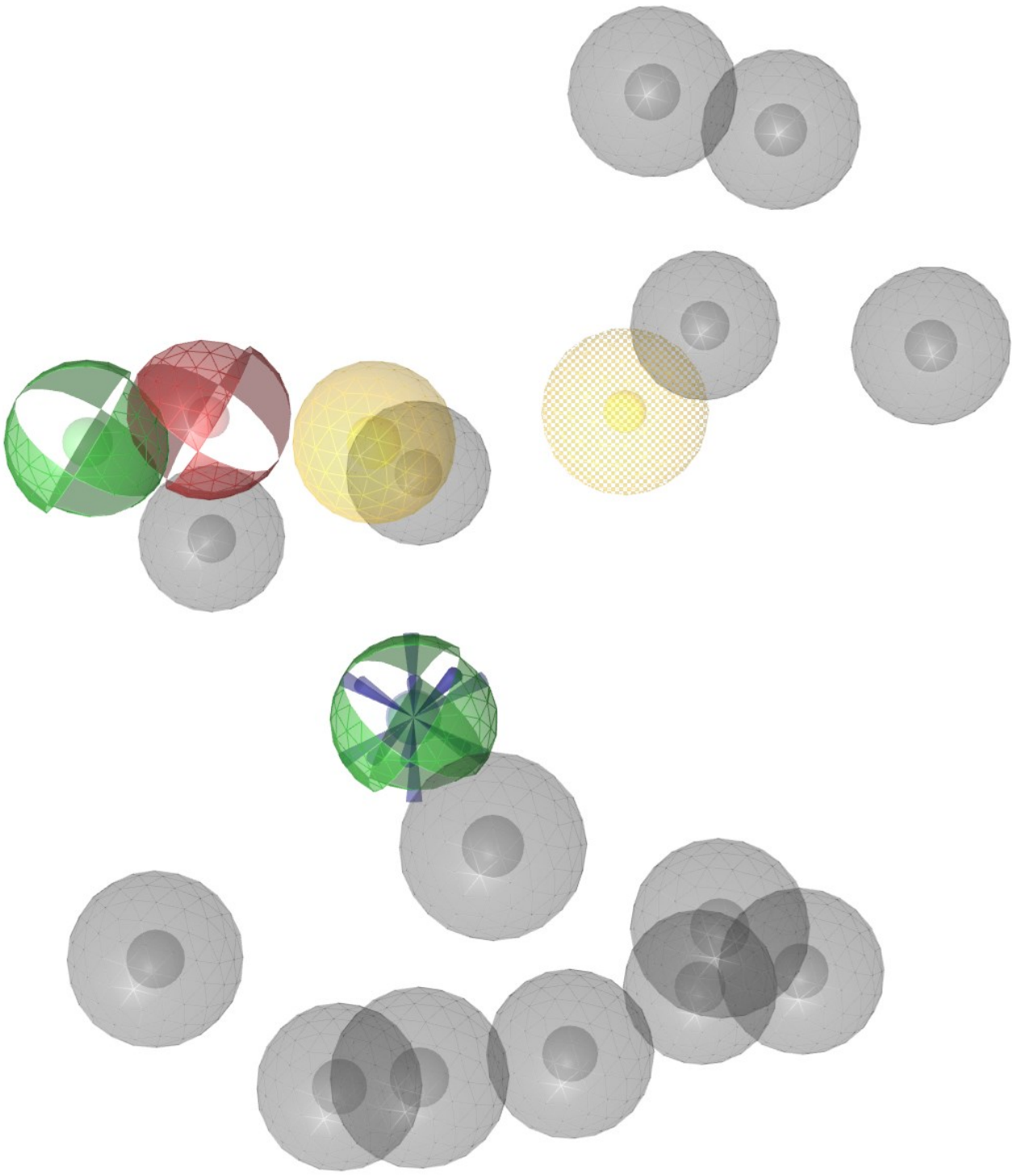


Figure 35: *Figure of pharmacophore 3 with exclusion volumes*

Ib PAINS filter script

The following script was used in Chapter 3.2.4.4.1 for filtering PAINS. It was already available on the workgroup server. The file paths have been adapted accordingly.

```
from rdkit import Chem
from rdkit.Chem import AllChem, PandasTools
from rdkit.Chem.FilterCatalog import FilterCatalog, FilterCatalogParams
import pandas as pd


#input SMARTS patterns within '' marks with , as a delimiter

smarts =
['[$([#6](~[#1])(~[#1])),$([#6](~[#1])(~[#6])),$([#6](~[#6])(~[#6]))]=[#6]~[#6,#7,#
15,#16](=[#7,#8,#16])~[#6,#7X3,#8X2]','[$([#6](~[#1])(~[#1])),$([#6](~[#1])(~[#6]))
,$([#6](~[#6])(~[#6]))]=[#6]~[#6]#[#7]','[#6]1~[#6](~[#9,#17,#35,#53])~[#6](~[#9,#1
7,#35,#53])~[#6](~[#9,#17,#35,#53])~[#6](~[#9,#17,#35,#53])~[#6]~1(~[#9,#17,#35,#53
])','[#7]~[#6](=[#16])~[#7]','[#8]=[#6]1~c2c(~[#8]~[#6]=,:[#6]~1)cccc2','[#6]~[#16X
2&H1]','[#7&R0]=[#7&R0]','[#6]~[#6H1&R0]=[#8]','[$([#6]=[#7&R0]~[#7,#8]),$([#6]=[#7
]~[#7&R0,#8&R0])]','[#6]=[#7&R0]~[#6]','[$([#6](=[#8,#16])~[#7&R0]-
[#7,#8]),$([#6](=[#8,#16])~[#7]-[#7&R0,#8&R0])]','[#7&R0]-
[#7&R0]','[#7]1~[#6](=[#8])~[#7]~[#7]~[#7]~[#6]~1','[#6]1(=[#7,#8,#16])~[#6]=,:[#6]
~[#6](=[#7,#8,#16])~[*]~[*]~1','[#7,#16]-[#6X4]-[#6]-
[#17,#35,#53]','[#6]~[#6](=[#16])~[#8,#16]','[#6]~[#6](=[#8])~[#16]','[#5](~[#8])~[
#8]']


#input sdf containing all molecules to be filtered


sdf = 'input.sdf'


#select name for molecules from input sdf


name = 'ID'


#filterin step dividing a set of molecules into matching (unwanted) and clean set


matches = {}
clean = {}


for i in smarts:
 patt = Chem.MolFromSmarts(i)
 with Chem.SDMolSupplier(sdf) as suppl:

  for mol in suppl:
   if mol.HasSubstructMatch(patt):
    keym = Chem.MolToSmiles(mol)
    valuem = mol.GetProp(name)
    matches.update({keym: valuem})


with Chem.SDMolSupplier(sdf) as suppl:
 for mol in suppl:
  keyc = Chem.MolToSmiles(mol)
  valuec = mol.GetProp(name)
```

```
  clean.update({keyc: valuec})

uclean = {keyc:valuec for keyc,valuec in clean.items() if keyc not in matches}


# get the clean set

uclean_items = uclean.items()
uclean_list = list(uclean_items)

#print(uclean_list)
with open('test_output_clean.smi', 'w') as fp:
 fp.write("\n".join('%s %s' % x for x in uclean_list))


# get the matched set

umatch_items = matches.items()
umatch_list = list(umatch_items)

#print(uclean_list)
with open('test_output_unwanted.smi', 'w') as fp:
 fp.write("\n".join('%s %s' % x for x in umatch_list))
```

Ic RMSD script

The following script was used for the RMSD analysis in Section 3.2.4.4.7. The file paths were adapted to the respective SD-file.

```
from rdkit.Chem import rdFMCS
from rdkit.Chem import AllChem
from rdkit import Chem
from pathlib import Path


def align_molecules(reference, pose, min_substructure_size=7):

    # Find the Maximum Common Substructure (MCS)
    mcs = rdFMCS.FindMCS([reference, pose])


    # Check if the size of the aligned substructure is below the minimum

    if mcs.numAtoms < min_substructure_size:
        return None  # Skip this pair of molecules


    # Create molecule objects from the MCS

    ref_mcs = Chem.MolFromSmarts(mcs.smartsString)
    pose_mcs = Chem.MolFromSmarts(mcs.smartsString)


    # Align the molecules using the MCS

    AllChem.Compute2DCoords(ref_mcs)
    AllChem.Compute2DCoords(pose_mcs)


    # Align the full molecules to the MCS

    rmsd = AllChem.GetBestRMS(ref_mcs, pose_mcs)
```

```
    return rmsd


# Load the reference molecule

ref_sdf_path = Path("reference_molecule.sdf")

ref_supplier = Chem.SDMolSupplier(str(ref_sdf_path))

reference_molecule = ref_supplier[0]  # Assuming there's only one molecule in the
reference SDF


# Load the docking poses

docking_sdf_path = Path("Docking_poses.sdf")

docking_supplier = Chem.SDMolSupplier(str(docking_sdf_path))


# Output SDF file for poses with RMSD of 0.0

output_sdf_path = Path("output.sdf")

output_supplier = Chem.SDWriter(str(output_sdf_path))

min_substructure_size = 7  # Set the desired minimum substructure size
for i, docking_pose in enumerate(docking_supplier):
    if docking_pose is not None:

        # Align the docking pose onto the reference molecule using MCS
        rmsd = align_molecules(reference_molecule, docking_pose,
min_substructure_size)


        # Check if rmsd is not None (meaning the comparison was not skipped)

        if rmsd is not None:

            if rmsd == 0.0:

                # Save the 3D structure to the output SDF file
                molecule_name = docking_pose.GetProp("_Name")
                print(f"Molecule Name: {molecule_name}")
                output_supplier.write(docking_pose)


output_supplier.close()
```

Id Check three SD-files for common SMILES script

The following script was used to search for common molecules in the three SD-files obtained for each pharmacophore in Chapter 3.2.4.4.11. It has been adapted to the corresponding file paths.

```
from rdkit import Chem
from rdkit.Chem import AllChem
from rdkit.Chem import SDMolSupplier

def read_sdf(file_path):
    supplier = SDMolSupplier(file_path)
    molecules = [mol for mol in supplier if mol is not None]
    return molecules


def get_smiles_set(molecules):
    return {Chem.MolToSmiles(mol) for mol in molecules}


def find_common_molecules(sdf_files):
    molecules_sets = [get_smiles_set(read_sdf(file)) for file in sdf_files]
    common_molecules = set.intersection(*molecules_sets)
    return common_molecules


def main():

    # Replace with your SDF file paths

    sdf_files = ['filepath_1', 'filepath_2', 'filepath_3']

    common_molecules = find_common_molecules(sdf_files)

    print("Number of common molecules:", len(common_molecules))
    print("Common SMILES:")
    for smiles in common_molecules:
        print(smiles)

if __name__ == "__main__":
    main()
```

II Machine learning

IIa Machine learning model

The following script shows the machine learning model created in Chapter 3.3. In this version it is adapted to the model with the hyperparameters from the grid search (model A). The hyperparameters have been adjusted for model B.

```
from sklearn.ensemble import GradientBoostingRegressor
import matplotlib.pyplot as plt
import pickle
import numpy as np
import pandas as pd
from sklearn import model_selection
from sklearn import metrics
from rdkit import Chem
from rdkit.Chem import AllChem

np.set_printoptions(precision=3)
pd.options.display.precision = 3


# Read the file.

filepath =
'/home/bfitz/Documents/TLR8_compounds/csv/csv_prep/TLR8_EC50_agonist_activity_prep.
csv'

agonists_50nM = pd.read_csv(filepath)

ag50_df = pd.DataFrame(agonists_50nM)


# Create a numpy array for the SMILES.

smi = np.array(ag50_df['Smiles'])


# Create mol objects for each SMILES in the array.

mols = [Chem.MolFromSmiles(x) for x in smi]


# Choose Morgan Fingerprints as a descriptor.

radius = 2

fp_arrays = []
canonical_smiles_list = []  # List to store canonicalized SMILES


for mol in mols:

    # Canonicalize the SMILES

    canonical_smiles = Chem.MolToSmiles(mol, isomericSmiles=True)
    canonical_smiles_list.append(canonical_smiles)


    # Generate Morgan fingerprints for the canonicalised structure

    canonicalized_mol = Chem.MolFromSmiles(canonical_smiles)
    fp = AllChem.GetMorganFingerprintAsBitVect(canonicalized_mol, radius)
    fp_list = list(fp.ToBitString())
```

```
    fp_array = [int(bit) for bit in fp_list]
    fp_arrays.append(fp_array)

fp_arrays_np = np.array(fp_arrays)

print(fp_arrays_np.shape)


# Add the Morgan Fingerprints and canonicalised SMILES to the original dataframe

ag50_df['morgan_fp'] = list(fp_arrays_np)
ag50_df['canonical_smiles'] = canonical_smiles_list

ag50_df['pEC50'] = -np.log10(ag50_df['Standard Value'])


# Separate the target variable (y) and the features (X)

y = ag50_df['pEC50']
X = fp_arrays_np


# Split the data into training and testing sets

train_ind, test_ind = model_selection.train_test_split(np.arange(len(ag50_df)),
test_size=0.33, random_state=17)


# Use only the Morgan Fingerprints as features

X_train = X[train_ind, :]
X_test = X[test_ind, :]


# Model Selection

gb02 = GradientBoostingRegressor(n_estimators=40, max_depth=19, min_samples_leaf=2,
min_samples_split=5, max_features='sqrt', learning_rate=0.2, random_state=100)


# Fit the classifier using the training set.

gb02.fit(X_train, y[train_ind])


# Make predictions for the training and test set.

y_pred_train = gb02.predict(X_train)
y_pred_test = gb02.predict(X_test)


# Load the compounds from the ldb file

sdf_file_path = 'file_path.sdf'
suppl = Chem.SDMolSupplier(sdf_file_path)


# Prepare a list to store results

results = []


# Loop through each compound in the SDF file
```

```
for mol in suppl:
    if mol is not None:
        # Canonicalize the SMILES for the compound from SDF
        canonical_smiles = Chem.MolToSmiles(mol, isomericSmiles=True)

        # Calculate Morgan fingerprints for the canonicalized structure
        fp1 = AllChem.GetMorganFingerprintAsBitVect(mol, radius)
        fp1_list = list(fp1.ToBitString())
        fp1_array = [int(bit) for bit in fp1_list]

        # Make predictions using the trained model
        predicted_pEC50 = gb02.predict([fp1_array])[0]

        # Extract compound name and SMILES
        compound_name = mol.GetProp("_Name")

        # Append the results to the list
        results.append({'CompoundName': compound_name, 'SMILES': canonical_smiles,
'Predicted_pEC50': predicted_pEC50})

# Convert the results list to a DataFrame
results_df = pd.DataFrame(results)

# Save the results to a CSV-file
output_csv_path = 'output.csv'

results_df.to_csv(output_csv_path, index=False)
```

IIb Grid search script

The following script was used for the grid search for hyperparameters. It only works in conjunction with the machine learning model as it uses its variables. The grid of parameters was adapted in several cycles.

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn import model_selection
from sklearn import metrics
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV


learning_rates = [0.1, 0.2, 0,3, 0.5]


# Define the hyperparameter grid for the grid search

param_grid = {

    'n_estimators': [30, 40, 50, 60, 70],
    'max_depth': [10, 15, 17, 19, 25],
    'min_samples_leaf': [1, 2, 3, 4],
    'min_samples_split': [2, 3, 4, 5],
    'max_features': ['sqrt', 'log2'],
    'radius': [2, 3, 4],
    'learning_rate': learning_rates
}

best_params = {}


for radius in param_grid['radius']:

    fp_arrays = []

    for mol in mols:

        fp = AllChem.GetMorganFingerprintAsBitVect(mol, radius)
        fp_list = list(fp.ToBitString())
        fp_array = [int(bit) for bit in fp_list]
        fp_arrays.append(fp_array)

    fp_arrays_np = np.array(fp_arrays)

    ag50_df['log_EC50'] = np.log(ag50_df['Standard Value'])


    # Separate the target variable (y) and the features (X)

    y = ag50_df['log_EC50']
    X = fp_arrays_np


    # Split the data into training and testing sets

    train_ind, test_ind = model_selection.train_test_split(np.arange(len(ag50_df)),
test_size=0.33, random_state=17)


    # Use only the Morgan Fingerprints as features
```

```
    X_train = X[train_ind, :]
    X_test = X[test_ind, :]


    # Create GridSearchCV object

    param_grid_rf = {
        'n_estimators': param_grid['n_estimators'],
        'max_depth': param_grid['max_depth'],
        'min_samples_leaf': param_grid['min_samples_leaf'],
        'min_samples_split': param_grid['min_samples_split'],
        'max_features': param_grid['max_features'],
        'learning_rate': param_grid['learning_rate']
    }

    k5f = model_selection.KFold(n_splits=10, shuffle=True, random_state=14)
    rf_model = GradientBoostingRegressor(random_state=100)
    grid_search = GridSearchCV(rf_model, param_grid_rf, cv=k5f, scoring='r2')
    grid_search.fit(X_train, y[train_ind])


    # Get the best parameters

    best_n_estimators = grid_search.best_params_['n_estimators']
    best_max_depth = grid_search.best_params_['max_depth']
    best_min_samples_leaf = grid_search.best_params_['min_samples_leaf']
    best_min_samples_split = grid_search.best_params_['min_samples_split']
    best_max_features = grid_search.best_params_['max_features']
    best_learning_rate = grid_search.best_params_['learning_rate']

    best_params[(radius, best_n_estimators, best_max_depth, best_min_samples_leaf,
best_min_samples_split, best_max_features)] = grid_search.best_score_


# Find the best combination

best_params_combination = max(best_params, key=best_params.get)

best_radius, best_n_estimators, best_max_depth, best_min_samples_leaf,
best_min_samples_split, best_max_features = best_params_combination


# Print the best values for hyperparameters

print(f"Best n_estimators: {best_n_estimators}")
print(f"Best max_depth: {best_max_depth}")
print(f"Best min_samples_leaf: {best_min_samples_leaf}")
print(f"Best min_samples_split: {best_min_samples_split}")
print(f"Best max_features: {best_max_features}")
print(f"Best radius: {best_radius}")
print(f"Best learning_rate: {best_learning_rate}")


# Plot the results using a grid of subplots

fig, axes = plt.subplots(nrows=len(param_grid), ncols=1, figsize=(10, 5 *
len(param_grid)))


for i, (param, values) in enumerate(param_grid.items()):

    # Adjusted to handle the absence of 'param_' prefix in cv_results_

    means =
grid_search.cv_results_['mean_test_score'][grid_search.cv_results_['param_' +
```

```
param] == best_params_combination[i]]
    axes[i].plot(values, means, marker='o')
    axes[i].set_xlabel(param)
    axes[i].set_ylabel('Mean R^2')
    axes[i].grid()


plt.tight_layout()
plt.show()



# Get the best model for the final evaluation

best_model = grid_search.best_estimator_
y_pred_test = best_model.predict(X_test)


# Calculate relevant metrics for the test set.

r2_test = metrics.r2_score(y[test_ind], y_pred_test)
mae_test = metrics.mean_absolute_error(y[test_ind], y_pred_test)
rms_test = metrics.mean_squared_error(y[test_ind], y_pred_test, squared=False)
print(f"Gradient Boosting Regressor - Test Set Metrics:")
print(f"R\u00b2 = {r2_test:.3f}, MAE = {mae_test:.3f}, RMSE = {rms_test:.3f}")
```

IIc Cross-validation script

The following script was used to perform the k-10 cross-validation on Model A and B. In this version it is adapted for model B. It was adapted for Model A for its cross-validation.

```
from sklearn.model_selection import KFold


# Define the number of folds for cross-validation

num_folds = 10
kf = KFold(n_splits=num_folds, shuffle=True, random_state=42)


# Initialise lists to store evaluation metrics for each fold

r2_scores = []
mae_scores = []
rms_scores = []


# Perform cross-validation

for fold, (train_index, test_index) in enumerate(kf.split(X)):

    print(f'Fold {fold+1}/{num_folds}')


    # Split data into train and test sets for this fold

    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]


    # Train the model

    gb = GradientBoostingRegressor(n_estimators=50, max_depth=17,
min_samples_leaf=3, min_samples_split=2, max_features='sqrt', learning_rate=0.2,
random_state=100)

    gb.fit(X_train, y_train)


    # Make predictions on test set

    y_pred_test = gb.predict(X_test)


    # Calculate evaluation metrics

    r2 = metrics.r2_score(y_test, y_pred_test)
    mae = metrics.mean_absolute_error(y_test, y_pred_test)
    rms = metrics.mean_squared_error(y_test, y_pred_test, squared=False)


    # Append scores to lists

    r2_scores.append(r2)
    mae_scores.append(mae)
    rms_scores.append(rms)


    # Print metrics for this fold
```

```
    print(f'R² = {r2:.3f}, MAE = {mae:.3f}, RMSE = {rms:.3f}')


# Calculate average scores across all folds
avg_r2 = np.mean(r2_scores)
avg_mae = np.mean(mae_scores)
avg_rms = np.mean(rms_scores)


print("\nAverage Scores Across All Folds:")
print(f"R² = {avg_r2:.3f}, MAE = {avg_mae:.3f}, RMSE = {avg_rms:.3f}")
```

IId Machine learning results retrieving script

The following script was used to retrieve the fifteen molecules with the lowest predicted EC_{50} values from the predictions of the machine learning model. It also converts pEC_{50} to EC_{50} at the same time. The script has been adapted to the respective files containing the results of each model.

```
import pandas as pd


# Read the CSV-file

input_file_path = 'input.csv'

df = pd.read_csv(input_file_path)


# Convert pEC50 to EC50

df['EC50'] = 10 ** (-df['Predicted_pEC50'])


# Sort DataFrame by EC50 values

df_sorted = df.sort_values(by='EC50')


# Select the top 15 molecules with the lowest EC50

top_15_molecules = df_sorted.head(15)


# Save the result to a new CSV-file

output_file_path = 'output.csv'

top_15_molecules.to_csv(output_file_path, index=False)
```

III Maximum Common Substructure

IIIa FindMCS script

The following script was used to calculate the maximum shared substructure. In the present version it was used for the molecules from the pharmacophore-based screening and docking. For the other molecule sets the script was adapted accordingly.

```
from rdkit import Chem
from rdkit.Chem import rdFMCS, Draw


# Define molecules as SMILES strings

smiles1 = 'Cc1ccc(S(=O)(=O)NCc2ccc3c(c2)C[NH2+]C3)c(Cl)n1'
smiles2 = 'Cc1ccc(O)c(C[NH+]2CCCC2c2cccc3c2OCCO3)n1'
smiles3 = 'Cc1cc(C(=O)Nc2ccnn2C(C)C)cc(O)n1'
smiles4 = 'CC[NH+](CC)C1CCN(C(=O)c2ccc3[nH]c(C)nc3c2)C1'
smiles5 = 'CCOC1CC([NH3+])(C(=O)Nc2cncc(Br)c2O)C1(C)C'
smiles6 = 'CCCCC(C)(CO)[NH2+]Cc1ccnc2[nH]ccc12'
smiles7 = 'CN(Cc1ccccc1O)c1cc(C(F)F)[nH+]c2ccc(Cl)cc12'
smiles8 = 'Cc1ccc(O)c(NC(=O)C2(c3ccccc3)CCC[NH2+]2)c1'
smiles9 = 'CC(C)CN(CC[NH+](C)C)C(=O)c1[nH]ncc1Br'
smiles10 = 'CSCC(CCO)NC(=O)Nc1ccc(Cl)cc1C(=O)c1ccc[nH]1'
smiles11 = '[NH3+]CCSc1nnc(-c2ccccc2)n1-c1ccc(F)cc1F'


# Convert SMILES strings to RDKit molecule objects

mol1 = Chem.MolFromSmiles(smiles1)
mol2 = Chem.MolFromSmiles(smiles2)
mol3 = Chem.MolFromSmiles(smiles3)
mol4 = Chem.MolFromSmiles(smiles4)
mol5 = Chem.MolFromSmiles(smiles5)
mol6 = Chem.MolFromSmiles(smiles6)
mol7 = Chem.MolFromSmiles(smiles7)
mol8 = Chem.MolFromSmiles(smiles8)
mol9 = Chem.MolFromSmiles(smiles9)
mol10 = Chem.MolFromSmiles(smiles10)
mol11 = Chem.MolFromSmiles(smiles11)


# Find the Maximum Common Substructure (MCS)

mcs_result = rdFMCS.FindMCS([mol1, mol2, mol3, mol4, mol5, mol6, mol7, mol8, mol9,
mol10, mol11])


# Convert the MCS SMARTS string to a molecule object

mcs_mol = Chem.MolFromSmarts(mcs_result.smartsString)


# Generate substructure matches for each molecule

match1 = mol1.GetSubstructMatch(mcs_mol)
match2 = mol2.GetSubstructMatch(mcs_mol)
match3 = mol3.GetSubstructMatch(mcs_mol)
match4 = mol4.GetSubstructMatch(mcs_mol)
match5 = mol5.GetSubstructMatch(mcs_mol)
match6 = mol6.GetSubstructMatch(mcs_mol)
```

```
match7 = mol7.GetSubstructMatch(mcs_mol)
match8 = mol8.GetSubstructMatch(mcs_mol)
match9 = mol9.GetSubstructMatch(mcs_mol)
match10 = mol10.GetSubstructMatch(mcs_mol)
match11 = mol11.GetSubstructMatch(mcs_mol)


# Visualize and highlight the MCS in each molecule

img = Draw.MolsToGridImage([mol1, mol2, mol3, mol4, mol5, mol6, mol7, mol8, mol9,
mol10, mol11],

                           subImgSize=(500, 500),

                           highlightAtomLists=[match1, match2, match3, match4,
match5, match6, match7, match8, match9, match10, match11])

display(img)
```

IIIb Maximum common substructure of the final results from pharmacophore-based screening and docking approach

The following table shows the molecules from the final results of the pharmacophore-based screening and docking approach with their common substructure highlighted.

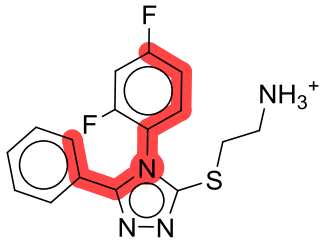
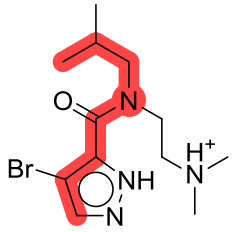
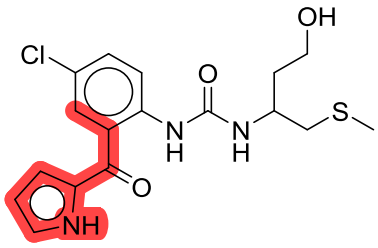
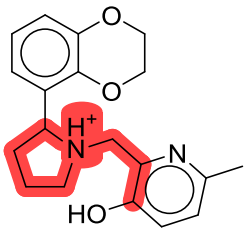
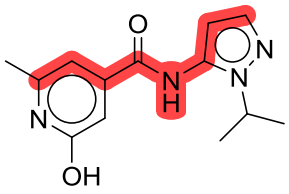
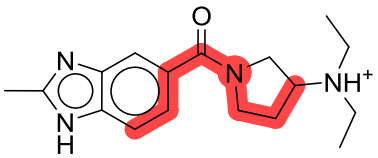
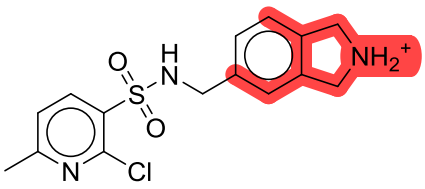
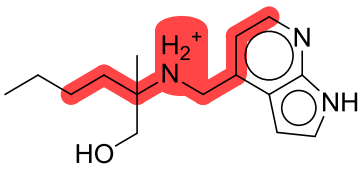
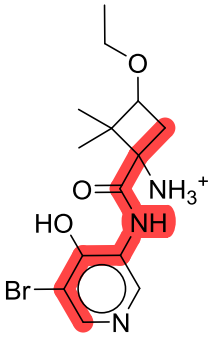
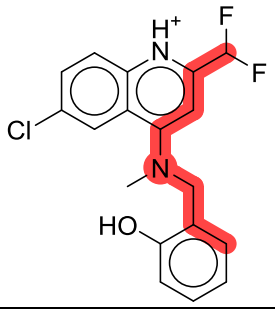
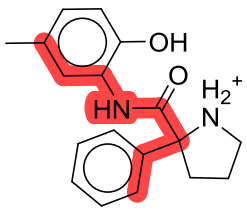
		
Z56936829	Z1251893025	Z1539671458
		
Z1568272665	Z1609219007	Z1625506699
		
Z2040113071	Z2443483104	Z2452532380
		
Z2744963592	Z3063578001	

Table 5: *Maximum common substructure of the final results from pharmacophore-based screening and docking approach.*

IIIc Maximum common substructure of the final results of ML Model A

The following table shows the molecules of the final results from machine learning Model A with their common substructure highlighted.

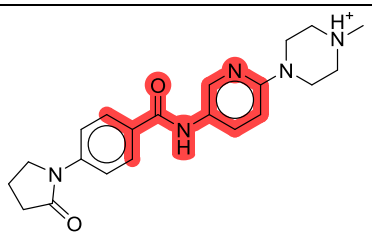
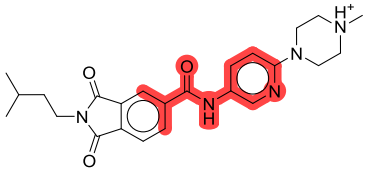
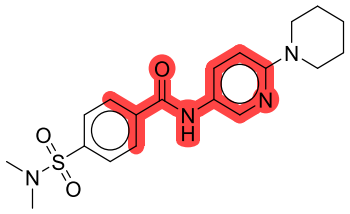
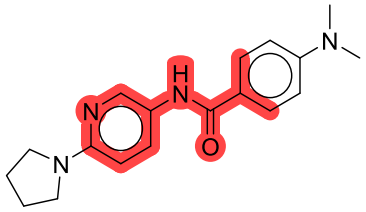
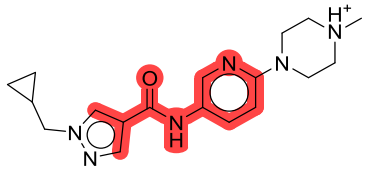
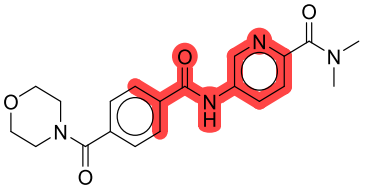
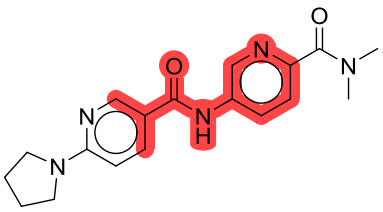
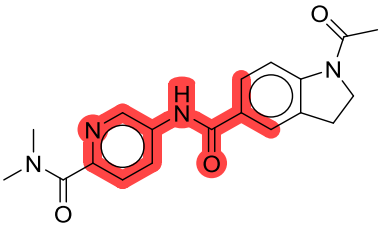
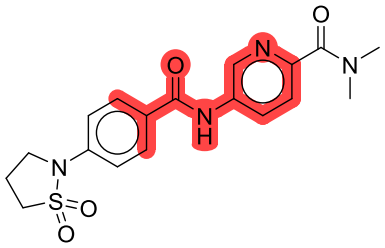
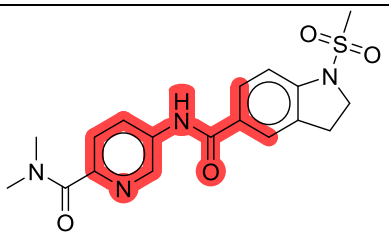
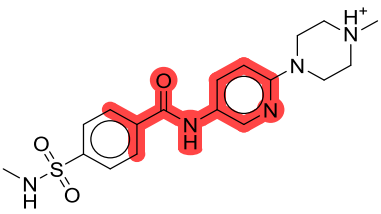
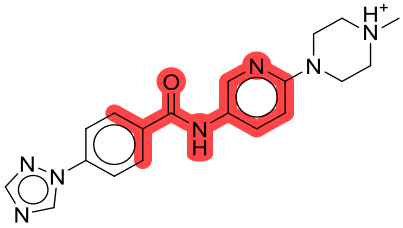
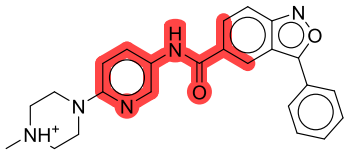
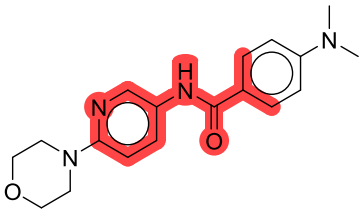
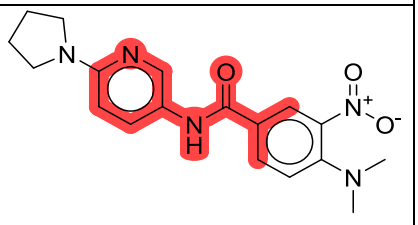
		
Z221484126	Z223833178	Z227262780
		
Z230704720	Z2764206490	Z2928726078
		
Z2928736856	Z2928738672	Z2928738714
		
Z2928739057	Z230202390	Z230197300
		
Z230198598	Z229816512	Z230699144

Table 6: *Maximum common substructure of the final results of machine learning model A.*

IIId Maximum common substructure of the final results of ML Model B

The following table shows the molecules of the final results from machine learning Model B with their common substructure highlighted.

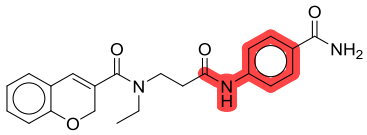
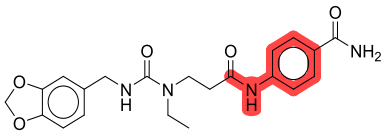
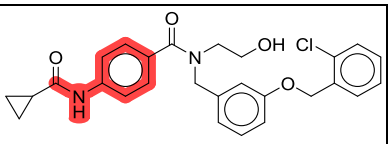
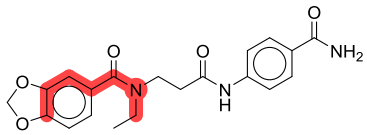
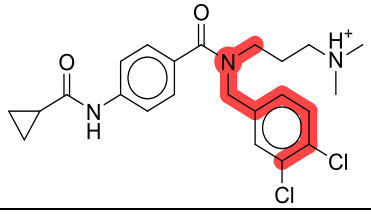
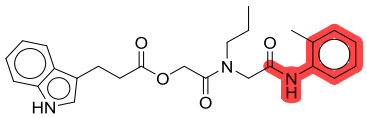
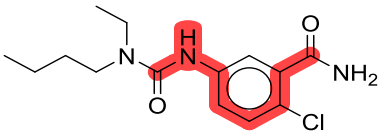
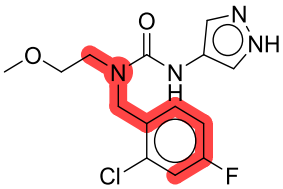
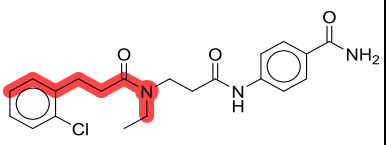
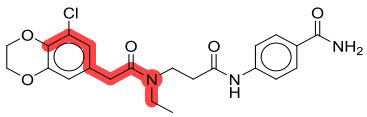
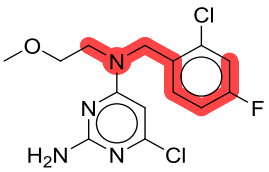
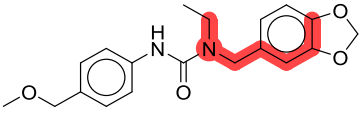
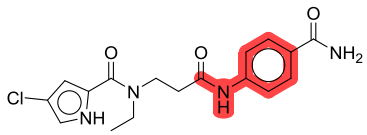
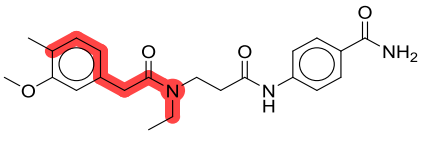
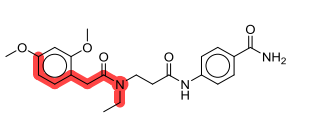
		
Z285541642	Z573712576	Z167207688
		
Z285540094	Z826838296	Z13758572
		
Z414266466	Z1149172628	Z290498678
		
Z285541906	Z1430528549	Z1892841216
		
Z285581258	Z285541880	Z285541410

Table 7: *Maximum common substructure of the final results of machine learning Model B*

IIIe Maximum common substructure of the molecules with the 10 lowest predicted EC_{50} values in the final results of ML Model B

The following table shows the ten molecules from machine learning Model B predicted by this model to have the lowest EC_{50} values, with their common substructure highlighted.

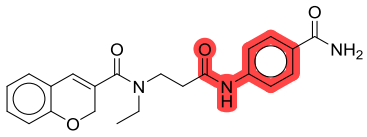
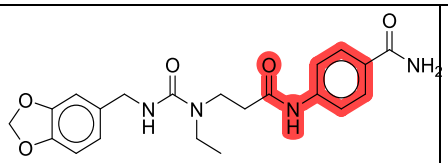
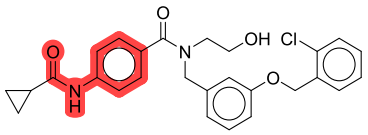
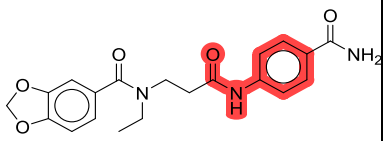
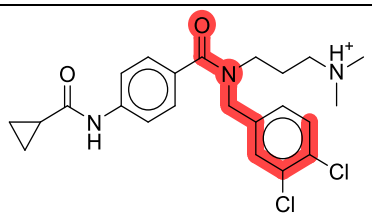
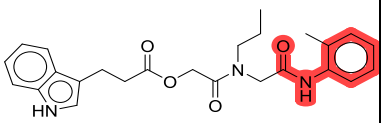
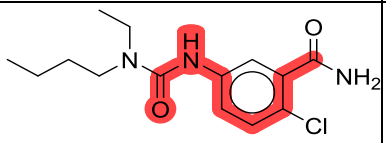
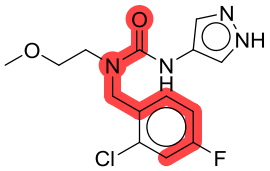
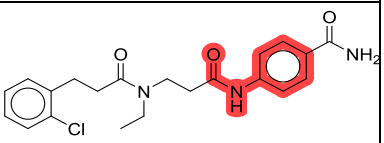
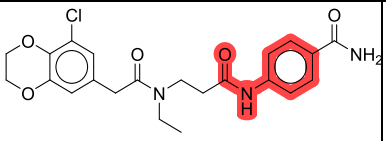
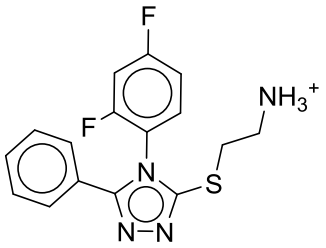
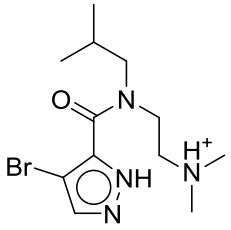
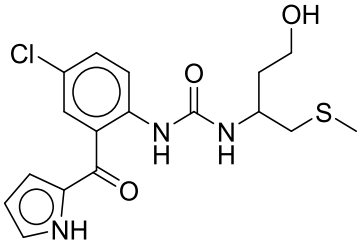
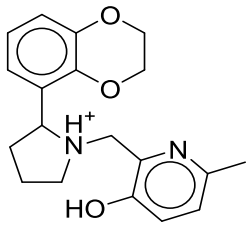
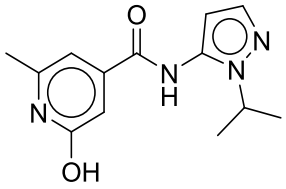
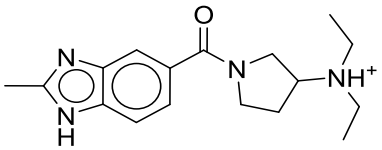
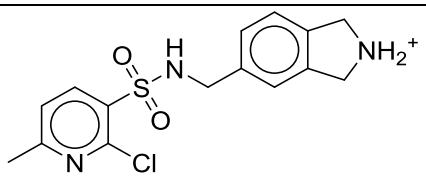
		
Z285541642	Z573712576	Z167207688
		
Z285540094	Z826838296	Z13758572
		
Z414266466	Z1149172628	Z290498678
		
Z285541906		

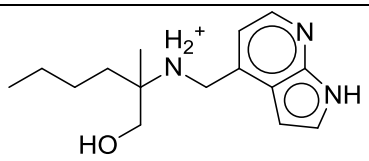
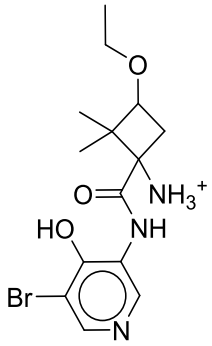
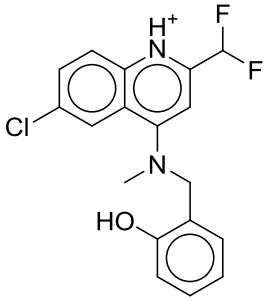
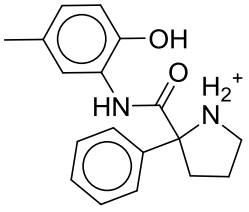
Table 8: *Maximum common substructure of the molecules with the 10 lowest predicted EC50 values in the final results of machine learning Model B.*

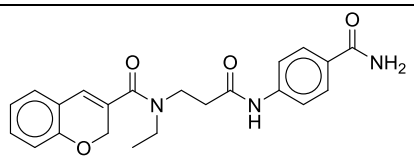
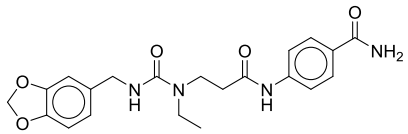
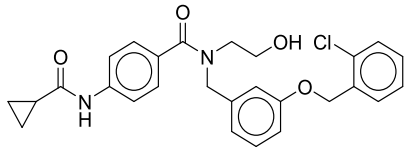
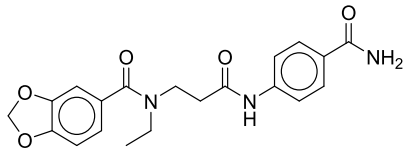
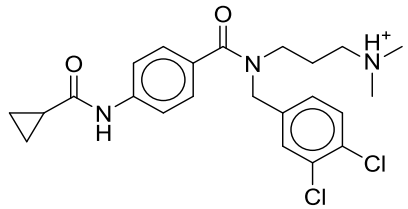
IV Molecules ordered for in-vitro testing

The following molecules have already been ordered by the contract partner for in-vitro testing.

Compound	Final results set of origin
 MW = 422.4 g/Mol Chemical Formula: $C_{18}H_{16}F_2N_4O_4S$ Enamine ID: Z56936829	Pharmacophore-based screening and docking (TP78_FP0)
 MW = 317.2 g/Mol Chemical Formula: $C_{12}H_{21}BrN_4O$ Enamine ID: Z1251893025	Pharmacophore-based screening and docking (TP125_FP45)
 MW = 381.9 g/Mol Chemical Formula: $C_{17}H_{20}ClN_3O_3S$ Enamine ID: Z1539671458	Pharmacophore-based screening and docking (TP81_FP6)

 MW = 326.4 g/Mol Chemical Formula: $C_{19}H_{22}N_2O_3$ Enamine ID: Z1568272665	Pharmacophore-based screening and docking (TP78_FP0)
 MW = 260.3 g/Mol Chemical Formula: $C_{13}H_{16}N_4O_2$ Enamine ID: Z1609219007	Pharmacophore-based screening and docking (TP81_FP6)
 MW = 300.4 g/Mol Chemical Formula: $C_{17}H_{24}N_4O$ Enamine ID: Z1625506699	Pharmacophore-based screening and docking (TP125_FP45)
 MW = 374.3 g/Mol Chemical Formula: $C_{15}H_{17}Cl_2N_3O_2S$ Enamine ID: Z2040113071	Pharmacophore-based screening and docking (TP125_FP45)

 MW = 261.4 g/Mol Chemical Formula: $C_{15}H_{23}N_3O$ Enamine ID: Z2443483104	Pharmacophore-based screening and docking (TP125_FP45)
 MW = 358.2 g/Mol Chemical Formula: $C_{14}H_{20}BrN_3O_3$ Enamine ID: Z2452532380	Pharmacophore-based screening and docking (TP78_FP0)
 MW = 348.8 g/Mol Chemical Formula: $C_{18}H_{15}ClF_2N_2O$ Enamine ID: Z2744963592	Pharmacophore-based screening and docking (TP125_FP45)
 MW = 296.4 g/Mol Chemical Formula: $C_{18}H_{20}N_2O_2$ Enamine ID: Z3063578001	Pharmacophore-based screening and docking (TP125_FP45)

 MW = 393.4 g/Mol Chemical Formula: $C_{22}H_{23}N_3O_4$ Enamine ID: Z285541642	Machine Learning Model B
 MW = 412.4 g/Mol Chemical Formula: $C_{21}H_{24}N_4O_5$ Enamine ID: Z573712576	Machine Learning Model B
 MW = 479 g/Mol Chemical Formula: $C_{27}H_{27}ClN_2O_4$ Enamine ID: Z167207688	Machine Learning Model B
 MW = 383.4 g/Mol Chemical Formula: $C_{20}H_{21}N_3O_5$ Enamine ID: Z285540094	Machine Learning Model B
 MW = 448.4 g/Mol Chemical Formula: $C_{23}H_{27}Cl_2N_3O_2$ Enamine ID: Z826838296	Machine Learning Model B

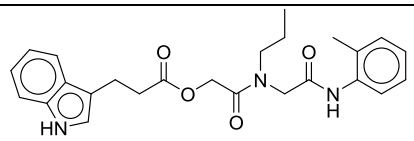
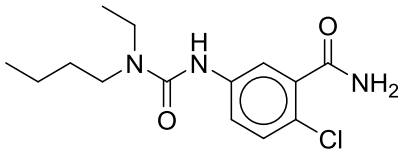
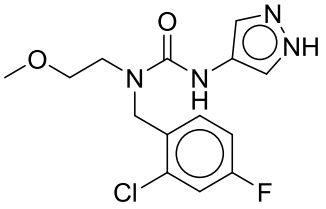
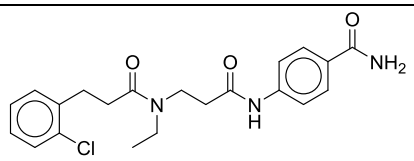
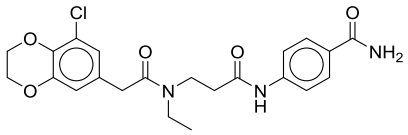
 MW = 435.5 g/Mol Chemical Formula: $C_{25}H_{29}N_3O_4$ Enamine ID: Z13758572	Machine Learning Model B
 MW = 297.8 g/Mol Chemical Formula: $C_{14}H_{20}ClN_3O_2$ Enamine ID: Z414266466	Machine Learning Model B
 MW = 326.8 g/Mol Chemical Formula: $C_{14}H_{16}ClFN_4O_2$ Enamine ID: Z1149172628	Machine Learning Model B
 MW = 401.9 g/Mol Chemical Formula: $C_{21}H_{24}ClN_3O_3$ Enamine ID: Z290498678	Machine Learning Model B
 MW = 445.9 g/Mol Chemical Formula: $C_{22}H_{24}ClN_3O_5$ Enamine ID: Z285541906	Machine Learning Model B

Table 9: *Molecules ordered for in-vitro testing.*

V Table of Figures

VI Table of Tables

VII Abbreviations

AUC	Area under the curve
ADMET	Absorption, distribution, metabolism, excretion, toxicity
ADCC	Antibody-dependent cytotoxicity
Bcl-2	B-cell lymphoma 2 gene
CADD	Computer-aided drug design
CD	Cluster of differentiation
CDCC	Cambridge Crystallographic Data Centre
COX2	Cyclooxygenase 2
COVID	Coronavirus disease
CNS	Central nervous system
DAMP	Damage-associated molecular pattern/Danger-associated molecular pattern
EC_{50}	Effective concentration for 50% of maximum effect
GOLD	Genetic Optimisation for Ligand Docking
GU-rich	Guanidine and uridine rich
HBV	Hepatitis B virus
HIV	Human immunodeficiency virus
HSC	Haematopoietic stem cells
HSPC	Haematopoietic stem and progenitor cells
IL	Interleukin
IFN	Interferon
IRAK	Interleukin-1 receptor-associated kinases
IRF3	Interferon regulatory factor 3
kDa	Kilo Dalton
log P	logarithmic partition coefficient
LRR	Leucine-rich repeat
MAL	MyD88 Adaptor-Like
MAP kinase	Mitogen-activated protein kinase
MCS	Maximum common substructure
MIP	Macrophage inflammatory protein
ML	Machine learning
MOE	Molecular operating environment
MyD88	Myeloid differentiation primary response 88
NMR	Nuclear magnetic resonance
NF-κB	Nuclear factor 'kappa-light-chain-enhancer' of activated B-cells
PAINS	Pan-assay-interference compounds
PAMP	Pathogen-associated molecular pattern
PBMC	Peripheral blood mononuclear cells
PD1	Programmed cell death Protein 1
PDL1	Programmed Cell Death 1 Ligand 1
PDB	Protein data bank
PRR	Pattern-recognition receptor
ROC curve	Receiver-operator curve
RMSD/RMSE	Root mean squared deviation/Root mean squared error
RNAse T2	Ribonuclease T2
ROS	Reactive oxygen species
SAR	Structure-activity relationship
SNP	Single nucleotide polymorphism
SMILES	Simplified Molecular Input Line Entry System
TAB	TAK1-binding-proteins
TAK	Transforming-growth-factor-β-associated-kinase
Th1	T_1-helper cell
TICAM-1	TIR domain containing adaptor molecule 1

TIR	Toll-interleukin-1-receptor
TIRAP	TIR domain containing adaptor protein
TNF-α	Tumor necrosis factor
TRAF	Tumor-necrosis-factor-receptor-associated-protein
TRAM	TRIF-related adaptor molecule
TRIF	TIR-domain-containing adapter-inducing interferon-β (=TICAM-1)
UNC93B1	Unc-93 homolog B1