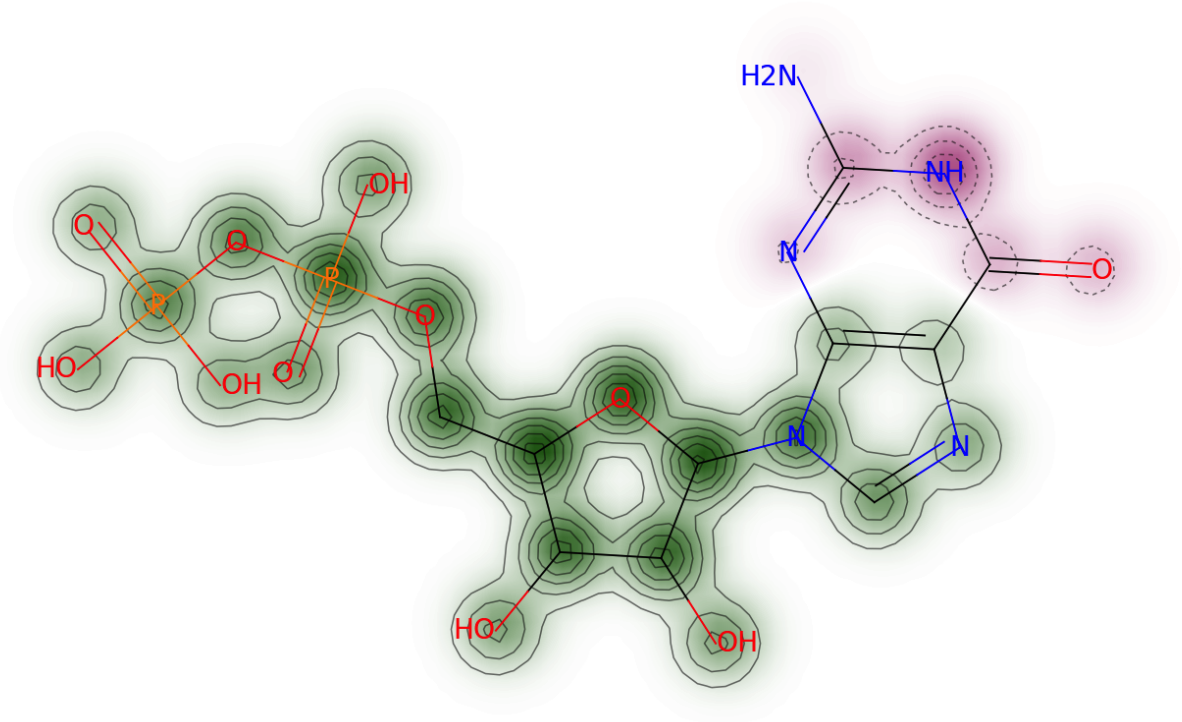
CHALMERS



Beräkning av kemisk likhet mellan substrat för karaktärisering av ett enzyms substratpromiskuitet

Kandidatarbete BBTX01-19-03

DAVID HÖGBERG
EMMA RYDHOLM
ISABELLA VOJSKOVIC
OSKAR PETTERSSON
JONATAN PERSSON
RASMUS MARTAENG

Institutionen för Biologi och bioteknik
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2019

Beräkning av kemisk likhet mellan substrat för karaktärisering av ett enzyms substratpromiskuitet

DAVID HÖGBERG, EMMA RYDHOLM, ISABELLA VOJSKOVIC,
JONATAN PERSSON, OSKAR PETTERSSON, RASMUS MARTAENG

Handledare: Martin Engqvist, Avdelningen för Systembiologi
Examinator: Claes Niklasson, Avdelningen för Kemisk reaktionsteknik

Kandidatarbete BBTX01-19-03
Institutionen för Biologi och bioteknik
Avdelningen för Systembiologi
Chalmers tekniska högskola
412 96 Göteborg
Telefon 031 772 1000

Omslag: Figuren illustrerar likheten mellan referensmolekylen ADP i jämförelse med GDP i en ”Similarity Map”. Gröna områden indikerar strukturer som bidrar till likheter och rosa områden strukturer som bidrar till olikheter.

Institutionen för Biologi och bioteknik
Chalmers tekniska högskola
Göteborg, Sverige 2019

Sammanfattning

Den traditionella synen på enzym är att de är specifika för vilka reaktioner de katalyserar och vilka substrat de kan transformera, men i allmänhet uppvisar enzym en högre grad av mångsidighet än så. Egenskapen hos enzym att kunna transformera flera strukturellt olika substrat kallas ofta för substratpromiskuitet, vilket bland annat är viktigt att studera för att bättre förstå metabola processer och hur enzym evolutionärt kan utveckla nya funktioner. Med tillgång till information om enzym och deras substrat i enzymdatabaser kan den här typen av promiskuitet undersökas kvantitativt, givet ett sätt att mäta hur strukturellt lika substraten är.

Fokus för det här projektet är att först undersöka relevanta kemoinformatiska metoder för att beräkna kemisk likhet mellan molekyler och sedan med hjälp av metoderna utveckla kod som beräknar strukturella likheter mellan ett enzyms substrat. Den slutliga koden är skriven i Python och tar en lista med substratnamn för en enzymklass som input. Koden kan sedan beräkna olika likhetsvärden mellan substraten samt skapa visualiseringar av de kemiska strukturerna och deras likheter. Förhoppningen är att koden ska kunna vidareutvecklas och användas av forskare på avdelningen för Systembiologi på Chalmers tekniska högskola för att vidare studera enzym och deras promiskuitet. Källkoden finns även tillgänglig på webbplattformen Github.

Nyckelord: enzym, enzympromiskuitet, substratpromiskuitet, molekylär likhet, kemoinformatik.

Calculating substrate similarity for characterisation of substrate promiscuity in an enzyme

Abstract

The traditional view on enzymes is that they are specific for the reactions they catalyze and for the substrate they convert into products. However, in general, enzymes show a greater versatility than that. The ability of an enzyme to convert many structurally diverse substrates is referred to as substrate promiscuity, which is important to study for a better understanding of metabolic processes and how enzymes can evolve to have new functions. With access to information about enzymes and substrates in enzyme databases, this type of promiscuity can be analyzed quantitatively, given a method to measure the structural similarity between substrates.

This project focuses on exploring relevant methods in the field of cheminformatics to measure chemical similarity between molecules, and then use these methods in the development of computer code capable of calculating and visualizing structural differences between an enzyme's substrates in various ways. The aim is that the product will be further developed and utilized by scientists at the division of Systems biology at Chalmers University of Technology for further studies of enzymes and their promiscuity. The source code will also be available at the website Github.

Keywords: enzyme, enzyme promiscuity, substrate promiscuity, molecular similarity, cheminformatics.

Förord

Den här rapporten beskriver ett kandidatarbete som har genomförts år 2019 på Institutionen för Biologi och bioteknik på Chalmers tekniska högskola. Projektgruppen vill tacka Martin Engqvist för hans tid och engagemang under hela projektets gång.

Ordlista

BRENDA:	BRaunschweig ENzymeDatabase, en enzymdatabas.
Data Frame:	En datastruktur i tabellformat med namngivna kolonner och rader.
Dictionary:	En datastruktur som sparar information i nyckel-värde-par för snabb åtkomst av data, likt ett lexikon.
EC:	Enzyme Commission. Enzym kategoriseras efter EC-nummer baserat på reaktionen de katalyserar.
Enzympromiskuitet:	Egenskapen hos enzym att avvika från en huvudsaklig funktion.
Fingerprint:	En bitsträng med ettor på positioner motsvarande vissa strukturer i en molekyl.
Git:	Ett versionshanteringssystem för kodutveckling.
Jupyter Notebook:	En webbapplikation som möjliggör interaktiv output från kod.
Kemoinformatik:	Ett ämnesområde med fokus på användning av metoder inom informatik på problem inom kemi.
MCS:	Maximum Common Substructure, den största gemensamma delstrukturen mellan två grafer.
MDS:	Multidimensional Scaling, en metod för dimensionsreducering av data.
PCA:	Principal Component Analysis, en metod för dimensionsreducering av data.
Python:	Ett programmeringsspråk.
RDKit:	Ett programmeringsverktyg utvecklat för tillämpningar inom kemoinformatik.
SMILES:	Simplified Molecular Input Line Entry System, ett kemiskt notationssystem för molekyler.
t-SNE:	t-distributed Stochastical Neighbor Embedding, en metod för dimensionsreducering av data.

Innehåll

1 | Inledning

1.1 Bakgrund

Enzym och deras katalytiska effekt är en förutsättning för liv, då de flesta reaktioner som sker i celler katalyseras av ett enzym. Katalysen möjliggörs av en sänkt aktiveringsenergi för omvandlingen av substrat till produkter, utan att enzymet konsumeras, vilket resulterar i en snabbare reaktion. Enzym har också många användningsområden som katalysatorer inom exempelvis läkemedelsindustrin, livsmedelsindustrin och papperstillverkning [1].

Den generella uppfattningen är att enzym utvecklas evolutionärt till att katalysera specifika reaktioner för specifika substrat, men det har på senare tid blivit tydligt att enzym uppvisar en högre grad av mångsidighet än vad som tidigare förmodats [2]. Enzym som på det här sättet avviker från sin huvudsakliga funktion kallas ofta för *promiskuösa*, där promiskuitet kan uppstå både med avseende på transformationen som utförs och på den mångfald substrat som kan ingå i reaktionen [3]. Även om fenomenet i sig har varit känt under en lång tid, så finns det idag en brist på vedertagna kvantitativa mått för att karaktärisera ett enzyms promiskuitet [4].

I takt med den teknologiska utvecklingen finns det samtidigt en allt större tillgång på biokemisk data om molekyler och deras egenskaper i olika databaser, vilket i kombination med fritt tillgänglig beräkningsprogramvara erbjuder stora möjligheter till kvantitativa analyser av kemisk information [5]. Hur den kemiska informationen bäst representeras, kvantifieras och bearbetas för datordriven analys har studerats inom området *kemoinformatik*, med tillämpningar inom exempelvis läkemedelsutveckling. Vanliga applikationer är då att studera relationer mellan kemiska strukturer och deras egenskaper, samt att söka efter molekyler som liknar en given struktur som kandidater till nya läkemedel. Det här förutsätter i sin tur att den kemiska likheten mellan molekylerna kan kvantifieras och beräknas på ett effektivt sätt [6].

Kemoinformatik kan därmed erbjuda verktyg för att kvantitativt undersöka likheter mellan ett visst enzyms substrat, som grund för att karaktärisera enzymets promiskuitet på substratnivå [7]. En sådan karaktärisering skulle kunna bidra till en ökad förståelse för oönskade sidoeffekter i metabola processer [8], [9] och eventuellt leda till en enzymklassificering som tydligare avspeglar enzymets funktion och egenskaper.

I den här rapporten kommer kemoinformatiska metoder för likhetsberäkning mellan substrat studeras för att potentiellt kunna användas till analys av promiskuitet på substratnivå för en given enzymklass.

1.2 Syfte

Syftet med det här projektet är att först undersöka vilka kemoinformatiska verktyg som idag finns tillgängliga för att beräkna likheter mellan molekyler, för att sedan använda verktygen i utveckling av datorkod som kan beräkna och visualisera likheter mellan de substrat som binder till ett visst enzym. Koden kommer att göras allmänt tillgänglig med avsikten att den ska kunna användas för likhetsberäkningar i framtida studier, där till exempel enzymets promiskuitet analyseras.

1.3 Avgränsningar

Litteraturstudien begränsas till att hitta de kodbibliotek som behövs för att bygga upp kodpipelinen. Den begränsas även i tid och kommer avslutas när kodbibliotek för samtliga uppgifter i pipelinen är insamlade och deras funktioner är testade.

Projektet begränsas även till användandet av BRENDA (BRaunschweig ENzyme Database) som enzymdatabas, som innehåller 7100 olika enzymklasser och har över 7,2 miljoner enzymsekvenser tillgängliga [10]. Det är möjligt att ladda hem listor från BRENDA över substrat vars reaktioner katalyseras av olika enzym, men pipelinen kommer inte hämta listor direkt från databasen. Den stora mängden enzymklasser i BRENDA kombinerat med begränsad tid gör att endast en exempelanalys av två enzymklasser ur datan från BRENDA kommer genomföras och redovisas.

Programmeringsspråket som används är Python, då det är open source och vanligt förekommande inom kemoinformatik. Python har även en stor fördel i att det finns så kallade Pythonbindningar, vilket gör det möjligt att använda kodbibliotek som är skrivna i andra programmeringsspråk genom Python. Projektet kommer använda sig av kodbibliotek som finns fritt tillgängliga som open source, eftersom projektet har en begränsad budget.

2 | Teori

2.1 Enzympromiskuitet

Den generella uppfattningen är att enzym är specifika med avseende på vilka substrat de binder till och vilka reaktioner de katalyserar, men i själva verket finns det mycket som talar för en högre grad av mångsidighet hos enzym än vad som tidigare förmodats. Det här har konsekvenser för vår förståelse kring hur enzym har utvecklats evolutionärt och vad de potentiellt skulle kunna användas till [2], [11]. Enzym som på olika sätt gör saker de inte förväntas göra kallas ofta för *promiskuösa*. Det här skiljer sig från exempelvis *multifunktionella* enzym, vilka har fler än ett aktivt säte.

Promiskuösa egenskaper hos enzym kan förekomma i många olika former, och vad som avses kan därmed variera från fall till fall. I grunden finns det ändå tre huvudsakliga sätt på vilka ett enzym kan avvika från sin primära funktion, vilket kan antas definiera tre kategorier av promiskuitet [3]:

1. Katalytisk promiskuitet: Enzymet katalyserar olika kemiska transformationer med samma substrat.
2. Betingad promiskuitet: Katalysen kan ske under avvikande reaktionsförhållanden, så som extrema temperaturer eller pH-värden.
3. Substratpromiskuitet: Enzymet katalyserar samma kemiska transformationer fast med olika substrat.

Projektet fokuserar på metoder för att karaktärisera och kvantifiera den sistnämnda typen av promiskuitet.

Information och forskningsresultat om enzym och deras substrat finns tillgängliga i enzymdatabaser. En vanligt förekommande databas i det här sammanhanget är BRENDA [10], med information om enzym kategoriserat under EC-nummer (Enzyme Commission number) som baseras på reaktionen de katalyserar. En konsekvens av den här typen av indelning är att varje klass kan innehålla från en upp till flera tusen olika enzymsekvenser, så länge de katalyserar samma reaktion.

Varje enzymklass i BRENDA inkluderar en lista med motsvarande substratnamn, vilket möjliggör en kvantitativ analys av hur specifika enzymen är på substratnivå, givet ett sätt att uppskatta graden av likhet mellan molekylerna. Med runt 7100 enzymklasser i BRENDA [10] blir det här ett problem som lämpar sig väl för kemoinformatiska metoder.

2.2 Kemoinformatik

Från att de första databaserna för kemiska strukturer utvecklades på 60-talet har mängden lagrad data från experimentella resultat ökat kraftigt, vilket i kombination med förbättrad datorprestanda erbjuder stora möjligheter för kvantitativa analyser. Kemoinformatik som område handlar i stora drag om hur metoder inom informatik kan appliceras på den data som finns tillgänglig om kemiska föreningar och deras egenskaper [6].

Information om kemiska föreningar lagras ofta i form av deras kemiska strukturer, men i grunden är det motsvarande kemiska egenskaper och vad de kan användas till som är av intresse. En återkommande frågeställning inom kemi är därmed vilka slutsatser som kan dras om kemiska egenskaper baserat på den kemiska strukturen. Med tillgång till större mängder data kan det här studeras kvantitativt för att hitta *Quantitative Structure-Property Relationships* (QSPR). Givet en önskad egenskap hos en molekyl, exempelvis för utveckling av ett nytt läkemedel, blir sedan ett vanligt kemoinformatiskt problem att i databaser söka efter potentiella strukturer som antas uppvisa liknande egenskaper [6].

Gemensamt för de ovan nämnda metoderna är att de utgår ifrån den kemiska strukturen och det grundläggande antagandet att kemiskt lika molekyler med avseende på struktur också uppvisar likheter i egenskaper och aktivitet, vilket ofta betecknas *Similar Property Principle* [12]. Anledningen till antagandet är att kemisk struktur är enklare att kvantifiera och studera än de egenskaper och aktiviteter som i slutändan är målet att förstå och utnyttja, exempelvis för utveckling av nya läkemedel med särskilda egenskaper. Det här är däremot ett antagande som behöver motiveras i varje enskilt fall, då det finns situationer där små skillnader i kemisk struktur ger oväntat stora skillnader i aktivitet [13].

Initialt utvecklades de kemoinformatiska verktygen av specifika företag och institutioner med sina respektive sätt att lagra och hämta information, men på senare tid har det blivit allt vanligare med initiativ för öppna databaser och datorverktyg som är fritt tillgängliga för forskningssyften [5]. Det här skapar i sin tur nya potentiella applikationer och målgrupper för användning av kemoinformatiska metoder, vilket är fokus i den här rapporten.

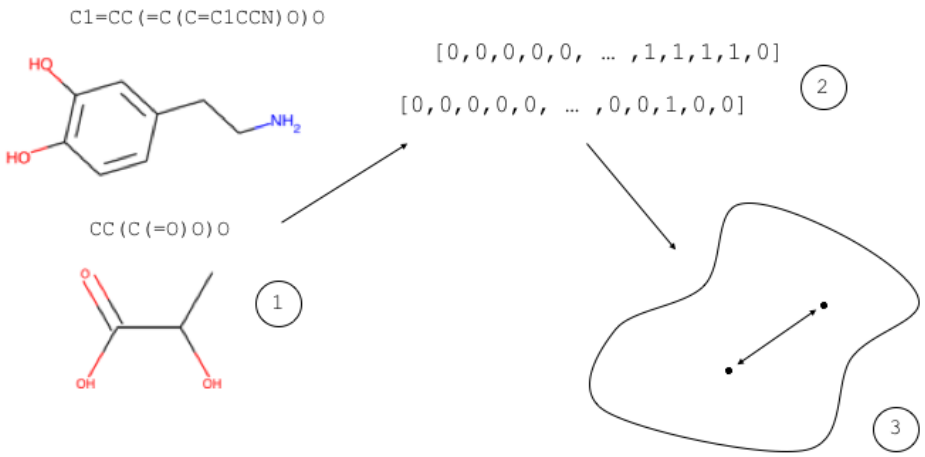
2.3 Kemisk likhet

En förutsättning för att kunna söka efter kemiskt lika strukturer i en databas, eller undersöka förhållandet mellan strukturer och egenskaper, är att det finns metoder för att jämföra och kvantifiera likheterna mellan strukturerna. Eftersom kemiska strukturer alltid behöver jämföras *med avseende på* någon viss struktur, finns det inget entydigt sätt att göra jämförelsen på. Det är därmed ett återkommande problem inom kemoinformatik just hur kemisk likhet kan kvantifieras och beräknas på ett effektivt sätt i form av ett likhetsmått.

I huvudsak är det tre komponenter som tillkommer för varje likhetsmått: (1) den kemiska strukturen behöver representeras i ett datoranpassat format, (2) den representerade informationen behöver viktas och kvantifieras baserat på hur likheten ska värderas, (3) den viktade informationen behöver jämföras och omvandlas till ett kvantitativt mått i form av en likhetskoefficient (vanligen mellan 0 och 1) [12].

Vad gäller att kvantifiera informationen om molekylerna ges den mest uttömmande beskrivningen av molekylens kvantmekaniska vågfunktion, som beskriver elektrondensiteten. All annan representation kommer därmed att innebära förlorad information på något sätt [13]. Ur ett kemoinformatiskt perspektiv är de kvantkemiska beräkningar som krävs för den här typen av information däremot för krävande, då det ofta är stora dataset av molekyler som ingår i analysen.

Vanligare är istället att sätta numeriska värden på särskilt utvalda egenskaper hos molekylerna (vikta den representerade informationen), och sedan representera dessa som punkter i ett vektorrum. Givet ett avståndsmått i vektorrummet kan sedan likheten mellan molekylerna uppskattas utifrån hur nära varandra de hamnar som punkter i rummet. På samma sätt blir avståndet i sig ett mått på olikheten mellan molekylerna [13]. Se Figur 2.1 för en illustration av varje delsteg.



Figur 2.1: *Likhetsberäkningar för molekylära strukturer i tre steg: (1) strukturerna representeras i ett datoranpassat format, (2) egenskaper hos molekylerna kvantifieras och viktas i vektorformat, (3) avståndet mellan motsvarande punkter i vektorrummet beräknas med ett avståndsmått.*

Nedan följer beskrivningar av varje delsteg i den här processen, det vill säga först hur informationen normalt representeras i ett datoranpassat format (2.3.1), sedan hur den representerade informationen kan viktas för likhetsberäkningen (2.3.2), och till sist vilka koefficienter som är vanligt förekommande som likhetsmått (2.3.3). Eftersom kemiska föreningar ofta representeras som grafstrukturer på olika sätt är det även vanligt att ge ett lokalt mått på likheten baserat på den största gemensamma delstrukturen i respektive graf, vilket beskrivs närmre i avsnitt 2.3.4.

2.3.1 Kemisk representation

Det finns ett flertal sätt att representera molekylära strukturer i ett datoranpassat format. Utöver att namnge molekylerna på olika sätt är det vanligt att konstruera unika sekvenser av textsymboler som ger information om hur molekylens atomer är ordnade och hur de är bundna till varandra [6].

En sådan typ av representation är *Simplified Molecular Input Line Entry System* eller SMILES. SMILES är ett kemiskt notationssystem speciellt utvecklat för datoranvändning av kemister, där molekyler skrivs om till textsymboler efter atomernas ordningsföljd [14]. SMILES är ett brett använt notationssystem och flertalet kemikaliedatabaser registrerar SMILES för respektive molekyl. Det här gör att representationen kan hämtas direkt från databaserna istället för att de måste genereras av kemisten själv.

Ett liknande notationssystem är IUPACs *International Chemical Identifier* eller InChI [15]. InChI:s skrivs likt SMILES ut som textsymboler, men de byggs upp av helt andra regler. InChI:s generaras utifrån den kemiska strukturens utformning på ett hierarkiskt skiktat sätt i så kallade lager. Att en InChI är hierarkiskt skiktad betyder att lagrena alltid skrivs ut i samma ordning och varje nytt lager representeras med ett snedstreck. Exempel på SMILES och InChI för laktat visas i Tabell 2.1.

Tabell 2.1: Exempel på SMILES och InChI för laktat.

Molekyl	Laktat
SMILES	CC(O)C(=O)O
InChI	1S/C3H6O3/c1-2(4)3(5)6/h2,4H,1H3,(H,5,6)/t2-/m0/s1

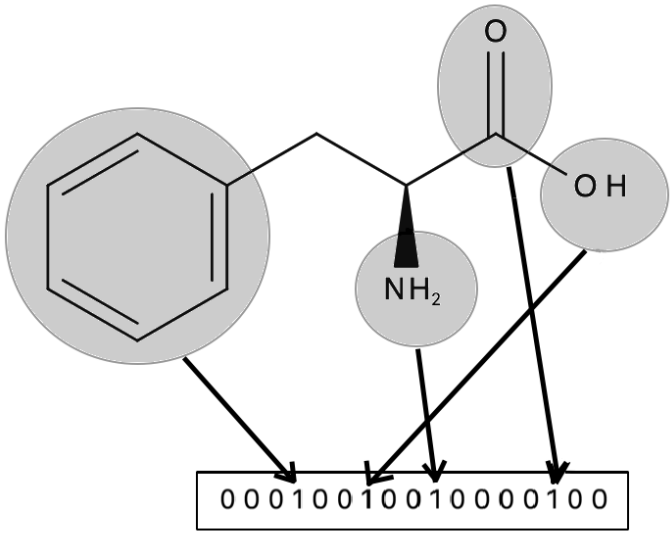
Ett annat intuitivt sätt att representera kemiska strukturer är med molekylgrafer, vilket har en matematisk motsvarighet inom fältet grafteori. En graf definieras då som en mängd noder och kanter, där noderna motsvarar atomer och kanterna motsvarar bindningar i en molekylär struktur [16]. I en dator representeras grafer ofta i form av matriser med element för de positioner som har kopplingar till varandra, men eftersom storleken på den här typen av matriser växer snabbt med antal atomer är det vanligare att lista atomerna och deras bindningar i tabellformat, i så kallade ”connection tables” [6].

2.3.2 Deskriptorer och fingerprints

Efter att informationen har representerats i ett datoranpassat format behöver egenskaper hos strukturerna kvantifieras för likhetsjämförelserna. Deskriptorer är beskrivningar av molekyler som till exempel smältpunkt, dipolmoment, joniseringsenergi och antal kolatomer. Ett mer komplext sätt att beskriva en molekyl är att kombinera flera deskriptorer till ett molekylärt fingerprint, som här kommer beskrivas mer ingående [17].

Ett fingerprint är en bitsträng som representerar vilka substrukturer som finns hos molekylen. Substrukturerna kan till exempel vara förekomsten av en viss atom, bindning eller funktionell grupp, där dess närvaro representeras av en 1:a och frånvaro av en 0:a. Längden på fingerprints varierar, men de är typiskt 150-2500 bitar långa [6], [17]. Det finns olika sätt att transformera en molekyl till ett fingerprint där några av de vanligaste är *nyckel-baserade-, topologiska-* och *cirkulära fingerprints.*

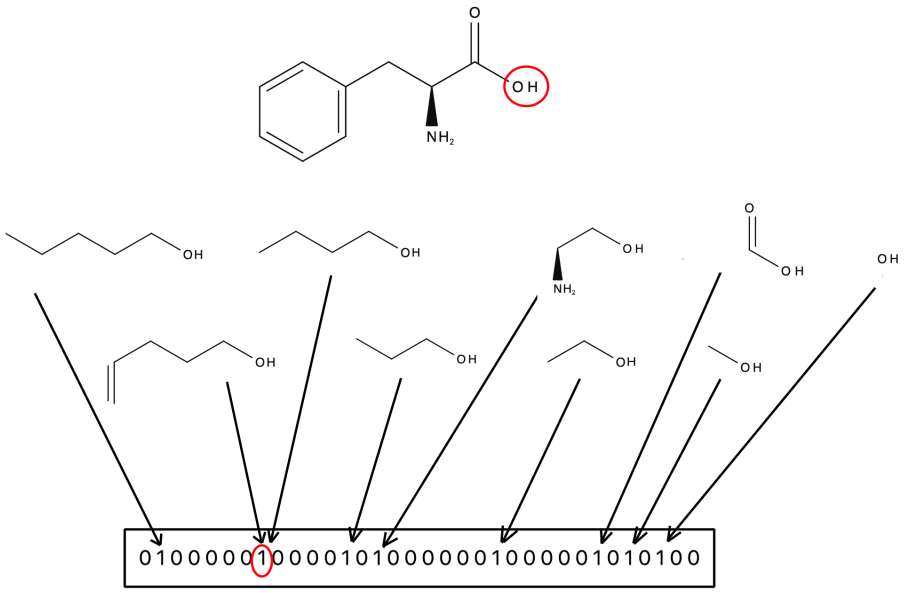
Nyckel-baserade fingerprints utgår ifrån förekomsten av förutbestämda substrukturer. Varje bit representerar en specifik substruktur och molekylen sökes igenom för att se om den innehåller den specifika strukturen. Om så är fallet sätts den biten till en 1:a [17], se Figur 2.2 för exempel. *MACCS keys* är ett sådant fingerprint som vanligtvis är 166 bitar långt. Trots sin korta längd så inkluderas de viktigaste substrukturerna [18].



Figur 2.2: *Ett hypotetiskt nyckel-baserat fingerprint konstruerat av substrukturnycklar, där de fyra bitar som visar 1 är aktiverade av de inringade områdena på molekylen.*

Topologiska fingerprints konstrueras genom att alla linjära fragment i molekylen, upp till en viss längd, går igenom en hash-funktion och sparas som en 1:a eller 0:a i bit-strängen. Hashningen innebär att en bit kan sättas till en 1:a av flera olika fragment men också att varje bit inte kan spåras tillbaka till en specifik egenskap som den kan i till exempel MACCS [17]. Figur 2.3 visar ett exempel på hur linjära, hashade fingerprints fungerar. Hashning innebär också att alla molekyler kan få ett meningsfullt fingerprint då det medför en flexibilitet i längden på fingerprintet. En stor fördel med de här typerna av fingerprint är att bitarna inte är fördefinierade och de kan därmed representera en större mängd olika strukturer. *Daylight fingerprint* är den vanligaste typen av de här fingerprinten [18].

Cirkulära fingerprints liknar de topologiska men istället för att generera linjära fragmenten så inkluderas området inom en viss diameter, se Figur 2.4. Diametern definierar det maximala antalet bindningar som kan förekomma i en linjär följd inom



Figur 2.3: *Ett hypotetisk, linjärt, hashat fingerprint. I illustrationen visas alla linjära substrukturer som utgår från den inringade atomen, med ett maximalt avstånd på 5 bindningar. Den inringade 1:an är ett exempel på en bit-kollision, där mer än en substruktur leder till att en bit sätts till 1.*

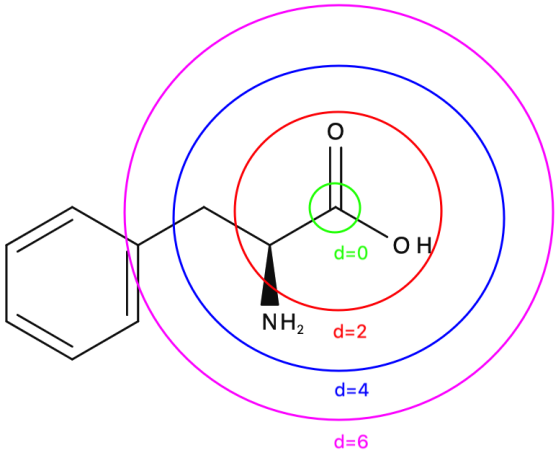
varje sökområde [19]. Vanligt är att en diameter på 4 alternativt 6 används [18]. En typ av cirkulära fingerprint är *Extended-connectivity fingerprints* (ECFPs), som är baserade på Morgan-algoritmen [20] som utvecklades på 1960-talet.

2.3.3 Likhetsmått

Likheten mellan två fingerprints beräknas med ett likhets- eller avståndsmått, som oftast ger ett värde mellan 0 och 1, där 0 indikerar full olikhet och 1 motsvarar identiska fingerprints [18].

För likhetsberäkningar av molekylära fingerprints är det standard att använda *Tanimoto-koefficienten*. Det finns även andra som fungerar väl, till exempel *Dice-koefficienten*, *Cosinuslikhet* eller *Soergels avstånd*. Avståndsmått som däremot har visat sig fungera sämre i sammanhanget är *Euklidiskt avstånd* samt *Manhattan*, vilka inte bör användas som enskilda mått [21]. En sammanställning av de diskuterade likhets- och avståndsmåtten finns i Tabell 2.2, och en mer utförlig tabell går att hitta i Appendix A.

Med värden för likhet mellan två molekyler kan sedan ett medelvärde för en mängd molekyler beräknas. Baserat på en sådan medellikhet föreslår Nath och Atkins ett kvantitativt index av substratpromiskuitet som också tar hänsyn till värden för k_{cat} och K_M för reaktionen som katalyseras [22]. Indexet ger ett värde mellan 0 och 1 på promiskuiteten som utöver likheten mellan substraten också väger in hur effektivt reaktionen sker.



Figur 2.4: *Ett exempel som illustrerar diameterns betydelse i cirkulära fingerprint. Strukturen inom varje diameter inkluderas upp till den givna diametern, detta upprepas med alla atomer som centralatom.*

Valet av både avståndsmått och fingerprint avgör det resulterade likhetsvärdet mellan två molekyler och olika kombinationer kan resultera i olika värden [18]. Genom att kombinera likhetsmåtten kan man potentiellt uppnå ett bättre resultat [21]. Det utförandet kallas *datafusion* och inkluderar alla metoder som kombinerar olika datakällor med följden att kombinationen är mer informativ än vad den enskilda datakällan hade varit [23].

Datafusion på likhetsmått är en procedur som kan genomföras på olika sätt. Likhetsmåtten kan exempelvis kombineras direkt efter det att måtten har normaliserats, vilket de flesta har då de ligger mellan 0 och 1. Ett annat sätt är att resultatet av likhetssberäkningar på ett dataset med molekyler rankas efter likhet med avseende på en referensmolekyl. [24]

2.3.4 Maximum Common Substructure (MCS)

Ett alternativt sätt att studera likheten mellan molekyler är att använda sig av metoder från grafteori och jämföra representationer av molekyler som matematiska grafer. För att identifiera graden av likhet mellan två grafer används ofta algoritmer för att hitta isomorfa delstrukturer. Två grafer sägs vara isomorfa om deras hörn stämmer exakt överens med varandra samt att hörnen är bundna med likadana kanter. En inducerad delgraf är en graf vars hörn med respektive kanter är en delmängd av en annan grafs hörn och kantmängd. MCS-algoritmer (Maximum Common Substructructure) kan utnyttjas för att hitta de molekyler som har störst antal inducerade delgrafer gemensamt med en referensmolekyl [16].

Tabell 2.2: Likhets- och avståndsmått som kan användas till att beräkna likheter mellan molekylära fingerprints. a = antal 1-bitar i fingerprint A, b = antal 1-bitar i fingerprint B, c = antal 1-bitar gemensamt för både A och B, m = totalt antal bitar i fingerprint A och B. D står för *distance* och S för *similarity*.

Likhetsmått	Ekvation	Intervall
Tanimoto	$S_{A,B} = \frac{c}{a+b-c}$	[0, 1]
Dice	$S_{A,B} = \frac{2c}{a+b}$	[0, 1]
Cosinuslikhet	$S_{A,B} = \frac{c}{\sqrt{ab}}$	[0, 1]
Soergel	$D_{A,B} = 1 - \frac{c}{a+b-c}$	[0, 1]
Sokal	$S_{A,B} = \frac{c}{2a+2b-3c}$	[0, 1]
Euklidiskt	$D_{A,B} = \sqrt{a+b-2c}$	[0, N]
Manhattan	$D_{A,B} = a+b-2c$	[0, N]

2.4 Klustring och visualisering

Klustring är en metod för att gruppera data efter inbördes likheter och är hjälpsam i analysprocessen. Fingerprints kan vara tusentals bitar långa, och varje bit symboliserar en ny dimension. Därför blir analys av denna typ av data ofta väldigt komplex med mer konventionella medel.

För att visualisera klustren bör datan dimensionsreduceras [25], något som kan göras med Principal Component Analysis (PCA), t-distributed Stochastical Neighbor Embedding (t-SNE) eller Multidimensional Scaling (MDS).

PCA [26] och MDS [27] är två likartade deterministiska metoder medan t-SNE [28] är en stokastisk metod som utnyttjar maskininlärning. t-SNE är icke-linjär, vilket innebär att den aktivt kan flytta datapunkterna för att förtydliga resultatet. Det här kringgår problemet i att högdimensionellt data lätt kollapsar ner på en väldigt liten yta, men det innebär också att avstånden i klusterplotten inte har matematisk vikt, även om de fortfarande illustrerar likhet/olikhet. t-SNE är också stokastisk och kan se olika ut från gång till gång.

Efter dimensionsreducering kan klustring påbörjas. Klusterverktyg kan mäta avstånd mellan en mängd datapunkter och söker därefter bästa sättet att gruppera dem [29]. En önskvärd gruppering är en där punkterna i varje grupp (kluster) har så korta parvisa avstånd som möjligt, samtidigt som antalet kluster blir så litet som möjligt. Förhoppningsvis upptäcks naturliga grupperingar på detta vis. I fallet med fingerprints blir alltså avstånden i fråga skillnaderna mellan dem; liknande molekyler hamnar i samma grupper. Det här kan sedan plottas för en intuitiv presentation.

2.5 Verktyg och databaser

Istället för att skriva ny datorkod från grunden går det att utnyttja programpaket som är specialiserade på ett visst tillämpningsområde, i det här fallet kemoinformatik. Programmerare kan då använda redan befintlig kod för att få ett bättre utgångsläge att utveckla sin egen kod. Nedan presenteras vanliga kodbibliotek och verktyg relaterade till kemoinformatik, samt exempel på databaser för tillgång till kemisk information. En översiktlig tabell över undersökta verktyg hittas även i Appendix B.

Exempel på vanliga paket för hantering av molekyler och deras egenskaper är RDKit och JChem. RDKit är öppen källkod skriven i C++ med ett Python-gränssnitt för att exempelvis generera olika fingerprints, beräkna likhet mellan fingerprints med ett antal likhetskoefficienter, läsa och behandla strukturer i olika filformat samt rita molekylära strukturer [30]. JChem från Chemaxon erbjuder i sin tur verktyg i programmerinspråket JAVA, och utbudet av fingerprints skiljer sig en del från RDKits. Inte alla bibliotek som Chemaxon erbjuder är öppen källkod och kan därmed kräva licens.

Andra programmeringsverktyg som ger tillgång till kemoinformatiska metoder är Cinfony och KNIME. Cinfony [31] är ett gemensamt Python-gränssnitt för paket som RDKit, Open Babel och JChem medan KNIME [32] är en mjukvaruprodukt för dataanalys med kopplingar till flera kemoinformatiska paket.

För konvertering mellan olika kemiska representationer, så som namn till en SMILES-sträng, går det att utnyttja webbportaler som OPSIN [33] och Chemical Identifier Resolver (CIR) [34], mjukvara som Open Babel [35], eller databaser som PubChem [36]. För CIR finns även ett gränssnitt för Python, CIRpy, som gör det möjligt att interagera med CIR i Python-kod [37]. På samma sätt erbjuder PubChemPy möjligheter att i Python interagera med PubChem, för att till exempel hitta SMILES-strängar givet ett molekylnamn eller söka efter molekyler med en viss struktur i databasen [38].

För klustring och dimensionsreducering är scikit-learn [39] ett vanligt programpaket för Python, baserat på maskininlärning. Exempel på en klustersalgoritm från scikit-learn är DBSCAN.

Databaser som tillhandahåller information om molekyler växer ständigt och det finns många alternativ att välja mellan [5]. Som tidigare nämnts är BRENDA en av de mest omfattande databaserna med information om enzym och deras substrat [10]. Andra exempel på enzymdatabaser är KEGG, PeroxiBase och KinBase, men de tillhandahåller antingen information om en viss specialiserad enzymfunktion eller endast specifika enzymklasser [40].

3 | Metod och genomförande

Genomförandet inleddes med en litteraturstudie, följt av kompetensutveckling inom programmering och avslutades med utveckling av själva kodprodukten.

3.1 Litteraturstudie

Syftet med den inledande litteraturstudien var att undersöka dels vilka kemoinformatiska metoder som kan utnyttjas för beräkning av kemisk likhet och dels vilka programmeringsverktyg som finns tillgängliga för den här typen av beräkningar.

Databaser och webbsidor som har använts för litteratursökningen är bland annat *Web of Science*, *PubMed*, *Chalmers bibliotek* och *Google scholar*. För att validera källornas relevans och trovärdighet har vi tagit hänsyn till vilka artiklar som ofta har refererats till av andra samt hur många citeringar artiklarna har. Artikelns publiceringsår har även noterats för att säkerställa att vi använder oss av aktuell forskning.

För att få grundkunskaper inom ämnesområdet har information om kemoinformatik inhämtats framförallt från textböcker och översiktsartiklar. För mer specifik forskning och metoder har vetenskapliga artiklar varit huvudkällan till information, med särskilt fokus på metodavsnitten i artiklarna.

Litteraturstudien ligger till grund för teoriavsnittet i rapporten och har bidragit till valen av kodbibliotek som använts. En tabell över kodbibliotek som har hittats och undersökts under processen finns i Appendix B.

3.2 Metoder för kodutveckling

Programmeringsspråket som användes i det här projektet var Python, under den senaste versionen Python 3. Python är brett använt inom kemoinformatik och dataanalys, vilket gjorde det enkelt att hitta tillgängliga kodbibliotek. Det är även gratis att installera och använda, vilket är viktigt för att fler ska kunna använda slutprodukten samt för att projektet inte har några resurser för att inhandla programkod.

För versionshantering användes Git, som är ett distribuerat versionshanteringssystem där utvecklare av ett kodprojekt kan arbeta på egna kopior av projektet som sedan sammanfogas [41]. Vi utnyttjade även Github (www.github.com), som är en webbplattform för att skapa och hantera Git-projekt.

För att kunna genomföra de kemoinformatiska analyserna krävdes det en kunskapsbas inom programmering, som hos projektdeltagarna inte fanns från början, i det här fallet inom programmeringsspråket Python. Grundläggande kunskaper inom Python

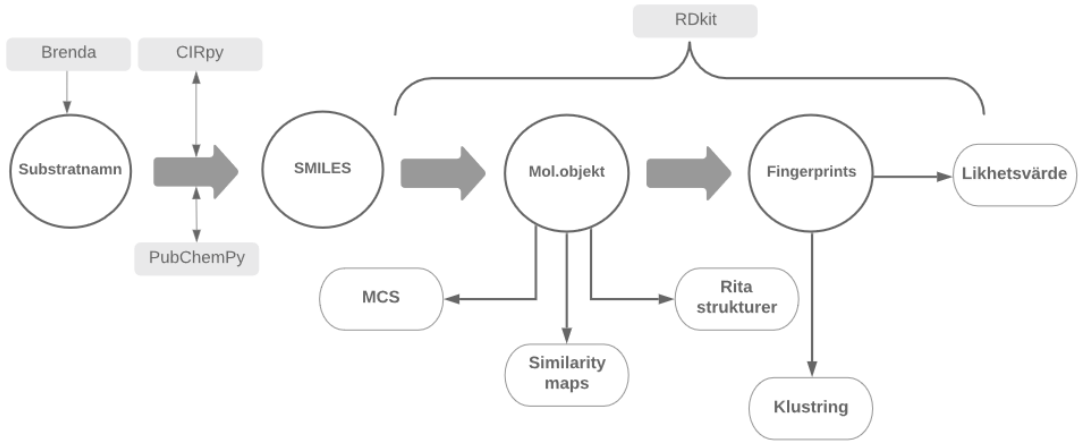
erhölls genom Codecademy [42], ett utbildningsföretag som erbjuder en onlinebaserad platform med programmeringskurser. Inlärningsprocessen i Python kombinerades med genomgångar med handledaren, både teoretiska och praktiska. Samma kompetensutecklingsstrategier användes för Git. Kurserna som togs var ”Learn Python 2”och ”Learn Git”.

Kodprodukten kommer publiceras som öppen källkod under licensen *GNU General Public License version 3* eller GPLv3. Licensen GPLv3 betyder att koden får modifieras för att uppfylla nya syften, samt friheten att vidare distribuera den modifierade koden i syfte att hjälpa andra.

3.3 Kodpipeline

Det huvudsakliga målet med slutprodukten, det vill säga kodpipelinen, var att först gå från en lista med substratnamn till en representation av molekylerna, för att därefter mäta likheten mellan substraten. Efter en omfattande litteraturstudie och sökande efter lämpliga kodbibiliotek kunde funktioner implementeras i koden och en pipeline växte successivt fram.

Den slutliga pipelinen tar som indata listor med ett enzyms substrat, översätter substratnamnen till SMILES, och kan därifrån rita ut deras 2D-struktur samt beräkna deskriptorer, fingerprints och likheter mellan strukturerna, se Figur 3.1.



Figur 3.1: *Ett flödesschema som representerar den slutliga pipelinen och de kodbibliotek som använts.*

Koden skrevs i separata moduler för varje funktion som behövde utföras, där output för en funktion sedan kunde användas som input för en annan. Exempelvis kunde kod som kräver en SMILES-sträng som input utvecklas separat under antagandet att det finns andra funktioner som ger korrekta SMILES-strängar som output med givna indata. På så sätt kunde arbetet delas upp i gruppen för respektive modul

och sedan sammanfogas, med hjälp av Git, som ett fullständigt paket i slutet. Själva sammanfogandet av varje bidrag från gruppmedlemmarna sköttes i slutändan av handledaren.

3.3.1 Databehandling

Rådatan för projektet tillhandahölls av projekthandledaren och bestod av listor över substratnamn från BRENDA sorterade efter EC-nummer och ursprungsorganism. Eftersom substratlistorna även inkluderade kofaktorer så antogs det första substratet i reaktionsformeln vara huvudsubstratet vid databehandlingen. Datan var därtill indelad i två dataset: samtliga substrat (observerade *in vivo* och *in vitro*) och naturliga substrat (endast observerade *in vivo*). Kod implementerades för att kunna välja mellan dem. Substratnamnen filtrerades även på information om stereokemi, så som inledande 'D-' eller 'L-', för att undvika dubbletter eller andra problem vid konverteringen till SMILES.

3.3.2 Substratnamn till SMILES

För representation av substratens kemiska struktur valdes SMILES, på grund av dess större utbredning och enkelhet relativt InChI. De programpaket som användes för beräkning av fingerprints kunde även ta SMILES som input.

För att konvertera givna substratnamn till SMILES valdes paketen CIRpy och PubChemPy [37], [38] (se Appendix B). Båda paketen användes i implementeringen då de kompletterade varandra vid konverteringen av vissa substratnamn. Därtill implementerades en funktion som testade varje hämtat SMILES i RDKit (som används i nästa steg); om ett SMILES hämtat från CIRpy inte fungerade i RDKit testades ett SMILES hämtat från PubChem istället. I de fall ingen metod gav giltigt SMILES sorterades substraten undan i en separat lista.

Utöver detta hittades också runt 30 SMILES via manuell sökning på PubChem. De hittades alltså inte av sökalgoritmen i PubChemPy.

3.3.3 Användning av RDKit

Huvuddelen av funktionerna för likhetsberäkningar implementerades med hjälp av RDKit [43], då det erbjöd den funktionalitet som behövdes för projektet tillsammans med ett tydligt gränssnitt för Python. RDKit har en egen typ molekylrepresentation i form av molekylobjekt som kan skapas med en SMILES-sträng som input, och som i sin tur kan användas för att rita ut molekylens struktur eller beräkna olika typer av fingerprints. Baserat på framtagna fingerprints för molekylstrukturerna finns det därefter olika typer av likhetsmått som kan användas för att uppskatta likheten.

3.3.4 Beräkningar av fingerprints

RDKit erbjuder ett Daylight-liknande fingerprint (topologiskt), ett MACCS-fingerprint och ett Morgan-fingerprint (cirkulärt), som alla har implementerats i projektet och kan specificeras genom att ange ett parametervärde (se Appendix C). De här fingerprinten motsvarar de tre huvudkategorierna av fingerprints som presenterades i avsnitt 2.3.2. Ett av RDKits fingerprint implementerades inte, nämligen *Atom pairs and Topological Torsions*, då det inte genererade en bitsträng och därför inte var anpassat för fortsatta applikationer.

Valet av vilket fingerprint som passar bäst för respektive situation överlåtes till användaren av koden. Däremot tillhandahålls en funktion som jämför fingerprinten utifrån vilka som överlag genererar högre/lägre likhetsvärden.

3.3.5 Likhetsberäkningar

För likhetsberäkningar mellan fingerprints erbjöd RDKit ett flertal likhetskoefficienter, däribland Tanimoto, Dice, Sokal och Cosinus. De implementerade koefficienterna hittas i Appendix A och kan anges med ett parametervärde till funktionerna (se Appendix C). Här är det återigen upp till användaren vilket mått som ska användas.

För en mängd molekyler implementerades funktioner för att beräkna en likhetsmatris innehållande likhetsvärde mellan varje kombination av utvalda molekyler. En funktion, som beräknar minimum, maximum, summan, medianen, medelvärdet och standard avvikelsen, tillämpades på likhetsmatrisen och kan betraktas som en kortare sammanställning av likhetsmatrisen.

Genom datafusion var det möjligt att använda sig av flera koefficienter för ett viktat mått. Alla likhetsvärde summerades då ihop för en given molekyl och ett medelvärde beräknades. Som nämnts i teorin kunde även resultatet av varje likhetberäkning rankas och summeras för att sedan jämföras. Det är samma princip och slutsatsen är densamma i båda lägen. Det förstnämnda förenklar dock studerandet av likhetsberäkningarna då man får ut ett likhetsvärde istället för en rang.

Vi valde att använda oss av Tanimoto, Dice, Sokal och Cosinus för att beräkna likheten mellan två fingerprints vid datafusion. Anledningen till de här fyra var för att de var likvärdigt normaliserade.

3.3.6 Generering av deskriptorer

Då fingerprints och likhetsberäkningar kan bli väldigt abstrakta och svårtolkade utvecklades även funktioner för att ta fram mer konkreta deskriptorer som exempelvis molekylmassa, antal OH-grupper och antal aromatiska ringar. De här kan genereras direkt från RDKits molekylobjekt. En komplett lista över vilka deskriptorer som kan erhållas finns i Appendix C.

3.3.7 Visualisering av substrat

Utöver numeriska resultat kan det vara av intresse att visualisera de olika substratens 2D-struktur samt vilka strukturer som är gemensamma för alla substrat i en enzymklass. Med hjälp av funktioner i RDKit implementerades metoder för att rita ut 2D-strukturen för antingen enskilda substrat eller för samtliga substrat i en enzymklass. Till den senare inkluderades även en RDKit-funktion för att beräkna MCS och markera den i samtliga 2D-strukturer.

RDKit erbjöd även en funktion för att rita ut en så kallad *Similarity Map* [44] mellan två molekyler. En Similarity Map konstrueras genom att en av molekylerna först väljs som referensmolekyl. Den andra molekylens 2D-struktur ritas sedan ut och markeras med gröna eller röda områden, där de gröna områdena symboliserar likheter med referensmolekylen, medan de röda visar olikheter.

3.3.8 Klustring och datavisualisering

För att få en bättre överblick över vilka grupperingar av lika molekyler som kan finnas i ett dataset implementerades klusteralgoritmer i koden. Molekylerna klustras med avseende på beräknade likhetsmått mellan alla molekyler.

Klustringsmetoderna som inkluderades i koden är Butina och DBSCAN [45], [46]. Butina är en klustringsmetod som finns i RDKit, vilket kan göra den mer kompatibel med tidigare kod. DBSCAN är väldigt snarlik, men är bättre lämpad för att hitta outliers, det vill säga datapunkter som inte hör hemma i något specifikt kluster.

I koden finns funktioner för att plotta klustren med dess molekyler i en 2D-plot om så önskas. För att åstadkomma det har varje dataset av fingerprints dimensionsreducerats med hjälp av PCA, t-SNE eller MDS (se avsnitt 2.4). Vilken av metoderna som lämpar sig bäst för vilken data är inget som diskuteras i den här rapporten, samtliga tillhandahålls i koden som valbara verktyg. Klustren representeras i plotten med en specifik färg, där alla punkter med samma färg tillhör samma kluster. Dimensionsreduceringen finns med i koden för att ge en överskådlig bild av klustren, men kan även användas för datavisualisering utan klustring.

3.3.9 Exempelanalys

Kodens funktionalitet exemplifierades med hjälp av analyser av två enzymklasser med relativt olika promiskuösa egenskaper. En mer promiskuös klass med avseende på sina substrat (EC 2.5.1.18) valdes med stöd i litteraturen [11], [22], medan en mer specifik klass (EC 3.6.1.5) valdes genom att först sortera alla enzymklasser efter medelvärde för likhet och sedan undersöka vilka av de med hög medellikhet som lämpade sig bra för visualisering. Hänsyn togs även till antalet substrat i varje klass och hur många av dem som kunde konverteras till SMILES. Själva analysen utfördes sedan med ett urval av funktionerna i en Jupyter Notebook [47], som möjliggör stegvis exekvering av Python-kod och direkt output av bilder, som ett exempel på hur koden skulle kunna användas i praktiken.

4 | Resultat

Den slutliga kodprodukten[1] består av två huvudsakliga moduler i filerna ”brenda.py” respektive ”cheminfo.py”, för att separera dataimporten från själva analysen av substraten. Funktionaliteten i cheminfo som avser själva analysen kan på så sätt användas för listor av molekyler från andra källor än BRENDA.

Den första modulen, `brenda`, har tillgång till data från BRENDA i form av två textfiler, en för endast naturliga substrat (endast observerade *in vivo*) och en för alla listade substrat. I koden finns två klasser för respektive alternativ, `BrendaNaturalMols` och `BrendaMols`, där motsvarande objekt kommer att innehålla listor med substrat för respektive EC-klass. Givna data kan sedan returneras i form av antingen ett dictionary eller en dataframe, vilka motsvarar två olika typer av datastrukturer.

Den andra modulen, `cheminfo`, består av klasserna `NameToSmile` respektive `SmileToData`, för att separera konverteringen från namn till SMILES och den analys som sedan använder SMILES-representationen som input.

`NameToSmile` tar en lista med namn på molekyler som input, och ger ett objekt med namnen på molekylerna och deras motsvarande SMILES-format. Namnen görs om till gemener, och eventuell information om stereokemi filtreras bort för att undvika dubbletter och problem vid konverteringen. I vissa fall kommer konverteringen att misslyckas, så det går att få listor över alla eller endast konverterade namn genom att ange en parameter. Eftersom konverteringen är relativt tidskrävande sparas tidigare översättningar kontinuerligt i textfiler för snabbare åtkomst.

I `SmileToData` finns sedan all funktionalitet relaterad till likhetsberäkningar och visualiseringar av molekylerna. Givet en lista med namn och motsvarande SMILES går det exempelvis att rita ut molekylerna, beräkna en matris med likhetsvärden, beräkna olika medelvärden, visualisera gemensamma delstrukturer och utföra dimensionsreduceringar. Mer dokumentation om tillgängliga funktioner finns i readme-filen på Github[2].

4.1 Exempel på analys av GST (EC 2.5.1.18)

För att illustrera kodpipelinens funktionalitet följer här ett exempel på analys av Glutation S-Transferas (GST), med EC-nummer 2.5.1.18, vars primära funktion är detoxifiering genom att koppla toxiska föreningar till antioxidanten glutation [48]. Den här klassen av enzym har uppvisat promiskuösa egenskaper på substratnivå (se till exempel [11]).

[1] `https://github.com/EngqvistLab/cheminformatics`

[2] `https://github.com/EngqvistLab/cheminformatics/blob/master/package_usage.ipynb`

4.1.1 Dataimport

Till att börja med behöver funktionaliteten importeras från filerna ”brenda.py”, för import av hämtad data från BRENDA, och från ”cheminfo.py”, för själva analysen av substraten. I det här fallet väljer vi att hämta naturliga substrat och spara molekyldata i form av en dictionary:

```
In [1]: import brenda, cheminfo
        mol_obj = brenda.BrendaNaturalMols(typeof='substrate')
        mol_data = mol_obj.data_dict()
```

Vår dictionary innehåller nu nyckel-värde-par i form av EC-nummer som nycklar och motsvarande listor med substrat som värden. Med EC-nummer 2.5.1.18 som nyckel returneras därmed en lista på naturliga substrat i den klassen:

```
In [2]: mols = mol_data['2.5.1.18']
        display(mols)
```

```
['1,1,1-trichloro-2,2-bis-(4-chlorophenyl)ethane',
 '12-oxo-phytodienoic acid',
 '2-hydroxychromene-2-carboxylic acid',
 'brostallicin',
 'glutathione',
 'harderoporphyrinogen',
 'protoporphyrinogen',
 'rac-4-hydroxynonenal',
 'rx']
```

4.1.2 SMILES-konvertering

Listan med substrat, i variabeln `mols` ovan, kan sedan användas som input i `NameToSmile` från `cheminfo` för konvertering till SMILES. Objektet motsvarande variabeln `chem_obj` nedan kommer att innehålla listor med namn och motsvarande SMILES, där det går att välja om alla eller endast konverterade namn ska anges med parametern `exclude_none`. Nedan skrivs konverterade SMILES och de namn som har exkluderats ut i listor med display-funktionen:

```
In [3]: chem_obj = cheminfo.NameToSmile(names=mols, retest_none=False)
        names = chem_obj.names()
        smiles = chem_obj.smiles()
        names_all = chem_obj.names(exclude_none=True)

        display(smiles)
        display([x for x in names_all if x not in names])
```

```
['Clc1ccc(cc1)C(c2ccc(Cl)cc2)C(Cl)(Cl)Cl',
 'CCC=CCC1C(C=CC1=O)CCCCCCCC(=O)O',
 'C1=CC=C2C(=C1)C=CC(O2)(C(=O)O)O',
 'CN1C=C(C=C1C(=O)NCCN=C(N)N)NC(=O)C2=CC(=CN2C)NC(=O)C3=CC(=CN3C)NC(
  =O)C4=CC(=CN4C)NC(=O)C(=C)Br',
 'N[C@@H](CCC(=O)N[C@@H](CS)C(=O)NCC(O)=O)C(O)=O',
 'CC1=C2CC3C(=C(C(=N3)CC4C(=C(C(=N4)CC5=NC(CC(=C1CCC(=O)O)N2)C(=C5C)
  CCC(=O)O)C)C=C)C)CCC(=O)O',
 'CC1=C2CC3=C(C(=C(N3)CC4=C(C(=C(N4)CC5=C(C(=C(N5)CC(=C1CCC(=O)O)N2)
  CCC(=O)O)C)C=C)C)C=C)C',
 'CCCCCC(O)C=CC=O']

['rx']
```

Substratet 'rx' var det enda som inte kunde konverteras. Redan här går det att se på SMILES-strängarna att de motsvarar relativt olika strukturer.

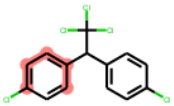
4.1.3 Likhetsanalys

I nästa steg kan listorna med namn och SMILES användas för att konstruera ett `SmileToData`-objekt. Här anges vilket typ av fingerprint (deskriptor) och likhetsmått som ska användas. Tillgängliga deskriptorer och likhetmått kan returneras med metoderna `valid_descriptors()` och `valid_metrics()`, och finns angivna i Appendix C. I det här fallet väljs fingerprint 'rdkit' samt likhetsmått 'tanimoto':

```
In [4]: data_obj = cheminfo.SmileToData(names, smiles,
                              descriptor='rdkit', metric='tanimoto')
```

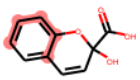
I första hand kan det vara intressant att rita upp strukturerna, vilket kan göras i ett rutnät med `draw_structures()`. Här finns också möjligheten att visualisera den största gemensamma delstrukturen (MCS) i molekylgrafen genom att ange parametern `highlight_substructure`:

```
In [5]: data_obj.draw_structures(highlight_substructure=True)
```

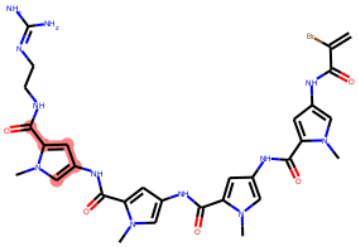


1,1,1-trichloro-2,2-bis-(4-chlorophenyl)ethane

12-oxo-phytodienoic acid

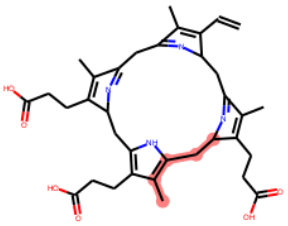


2-hydroxychromene-2-carboxylic acid

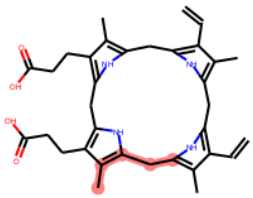


brostallicin

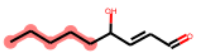
glutathione



harderoporphyrinogen



protoporphyrinogen



rac-4-hydroxynonenal

Likheter kan beräknas parvis mellan alla strukturer med metoden `similarity()`, som utnyttjar tidigare angivna fingerprints och likhetsmått och returnerar en likhetsmatris:

```
In [6]: data_obj.similarity()
```

	1,1,1-tr..	12-oxo-p..	2-hydrox..	brostall..	glutathi..	harderop..	protopor..	rac-4-hy..
1,1,1-tr..	1.0000	0.1614	0.2179	0.2285	0.1690	0.2438	0.2474	0.0895
12-oxo-p..	0.1614	1.0000	0.2000	0.1929	0.1887	0.2270	0.2058	0.2749
2-hydrox..	0.2179	0.2000	1.0000	0.4281	0.2631	0.4760	0.3824	0.0912
brostall..	0.2285	0.1929	0.4281	1.0000	0.2743	0.5802	0.4809	0.0962
glutathi..	0.1690	0.1887	0.2631	0.2743	1.0000	0.3237	0.2776	0.1018
harderop..	0.2438	0.2270	0.4760	0.5802	0.3237	1.0000	0.5862	0.0986
protopor..	0.2474	0.2058	0.3824	0.4809	0.2776	0.5862	1.0000	0.0932
rac-4-hy..	0.0895	0.2749	0.0912	0.0962	0.1018	0.0986	0.0932	1.0000

Eftersom Tanimoto-koefficienten varierar mellan 0 och 1, där 0 indikerar olikhet, tyder värdena i matrisen på ganska små grader av likhet i de flesta fall.

Funktionen `global_similarity_stats()` ger i sin tur en sammanfattning över globala likhetsegenskaper för hela mängden av molekyler. I det här fallet är likheten som minst 0.089 och som mest 0.586. Den största likheten nås mellan ”harderoporphyrinogen” och ”protoporphyrinogen” enligt matrisen ovan, vilka även uppvisar likheter i tidigare figur. Medellikheten för alla strukturer är 0.257, vilket tyder på att enzymklassen är relativt promiskuös med avseende på sina substrat.

```
In [7]: data_obj.global_similarity_stats()
```

```
{'min': 0.08945686900958466,
 'max': 0.5862445414847162,
 'sum': 7.200275672108634,
 'median': 0.22777422078009707,
 'mean': 0.25715270257530837,
 'stdev': 0.14323756938020502}
```

Med tanke på att ’harderoporphyrinogen’ och ’protoporphyrinogen’ ändå uppvisar vissa likheter, kan det vara intressant att undersöka mer specifikt vad som skiljer dem åt. För det syftet går det att få en översikt över ett fåtal egenskaper hos molekylerna, antingen i form av en dataframe eller ett dictionary. Nedan visas två dictionaries för respektive molekyl. Förklaringar till respektive variabel finns i Appendix C.

```
In [8]: prop_dict = data_obj.data_dict()
        display(prop_dict['harderoporphyrinogen'])
        display(prop_dict['protoporphyrinogen'])
```

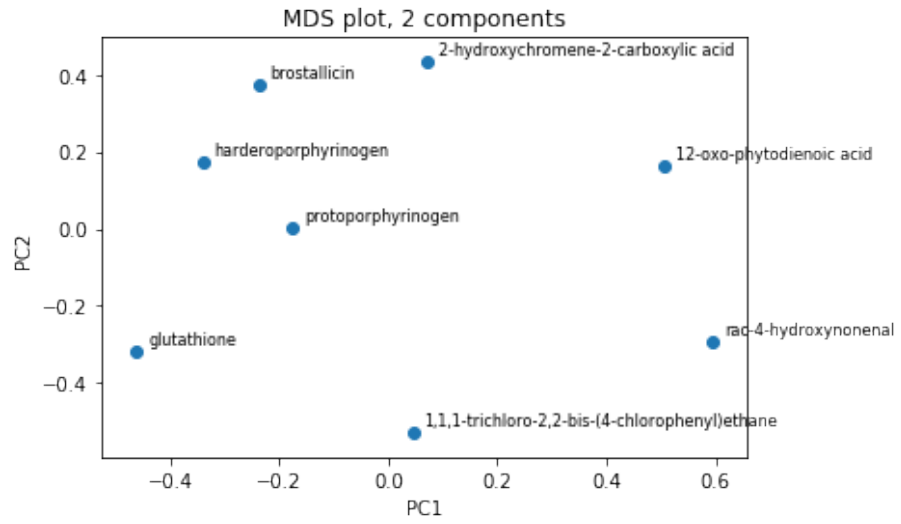
```
{'smile': 'CC1=C2CC3C(=...
 'molwt': 614.3104,
 'num_alirings': 4,
 'num_arorings': 1,
 'num_hbond_acceptors': 6,
 'num_hbond_donors': 4,
 'num_hetatom': 10,
 'num_nhoh': 4,
 'num_no': 10,
 'num_rotbond': 10,
 'tpsa': 164.7699}
```

```
{'smile': 'CC1=C2CC3=C(...
 'molwt': 568.3050,
 'num_alirings': 1,
 'num_arorings': 4,
 'num_hbond_acceptors': 2,
 'num_hbond_donors': 6,
 'num_hetatom': 8,
 'num_nhoh': 6,
 'num_no': 8,
 'num_rotbond': 8,
 'tpsa': 137.76}
```

Här framgår bland annat att de två molekylerna är relativt lika i storlek, men att den till höger (protoporphyrinogen) är mer aromatisk, då den har fler aromatiska ringar.

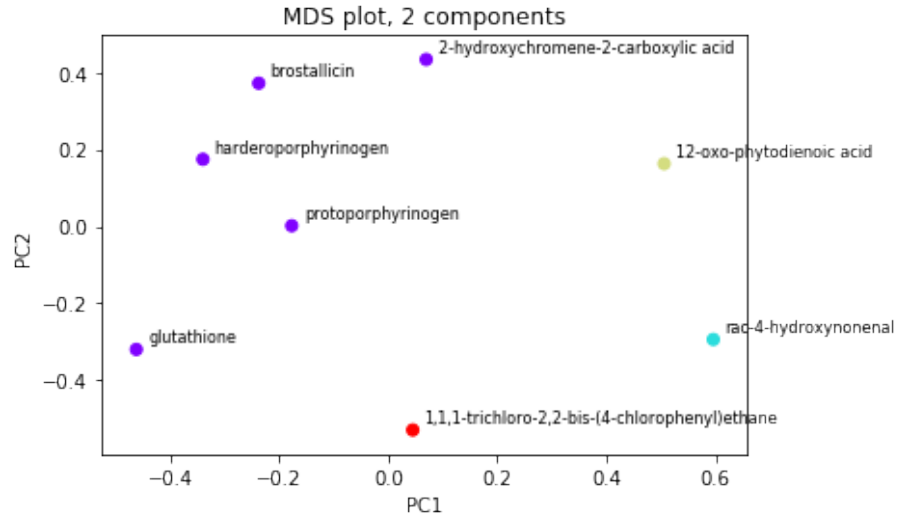
Till sist kan de inbördes avstånden mellan strukturerna visualiseras med exempelvis en MDS-plot, som efterliknar avstånden i likhetsmatrisen ovan:

```
In [9]: data_obj.mds(include_labels=True)
```



I det här fallet finns inga tydliga outliers, det vill säga strukturer som ligger väldigt långt ifrån alla andra i figuren. Däremot hamnar substraten i övre vänstra hörnet relativt nära varandra, vilket kan indikera att de har fler egenskaper gemensamt med varandra än med övriga substrat. Med hjälp av en klusteralgoritm kan strukturerna tilldelas kategorier baserat på deras inbördes likheter, vilka sedan kan anges för att färgkoda punkterna i vår MDS-plot:

```
In [10]: clusters = data_obj.cluster_butina()
         data_obj.mds(include_labels=True, color_categories=clusters)
```



Klustringen antyder därmed att substraten markerade med lila är inbördes relativt lika, medan de övriga tre substraten inte visar någon större inbördes likhet då de hamnat i egna kluster.

4.2 Exempel på analys av Apyras (EC 3.6.1.5)

För att kontrastera föregående analys följer här en liknande analys av Apyras (EC-nummer 3.6.1.5), som katalyserar omvandlingen av olika typer av nukleosidtrifosfater och -difosfater, och därmed kan tänkas inkludera en mångfald relativt lika substrat.

```
In [11]: mols = mol_data['3.6.1.5']
         display(mols)
```

```
['adp', 'amp', 'atp', 'cdp', 'ctp', 'gdp', 'gtp', 'ttp', 'udp', 'utp']
```

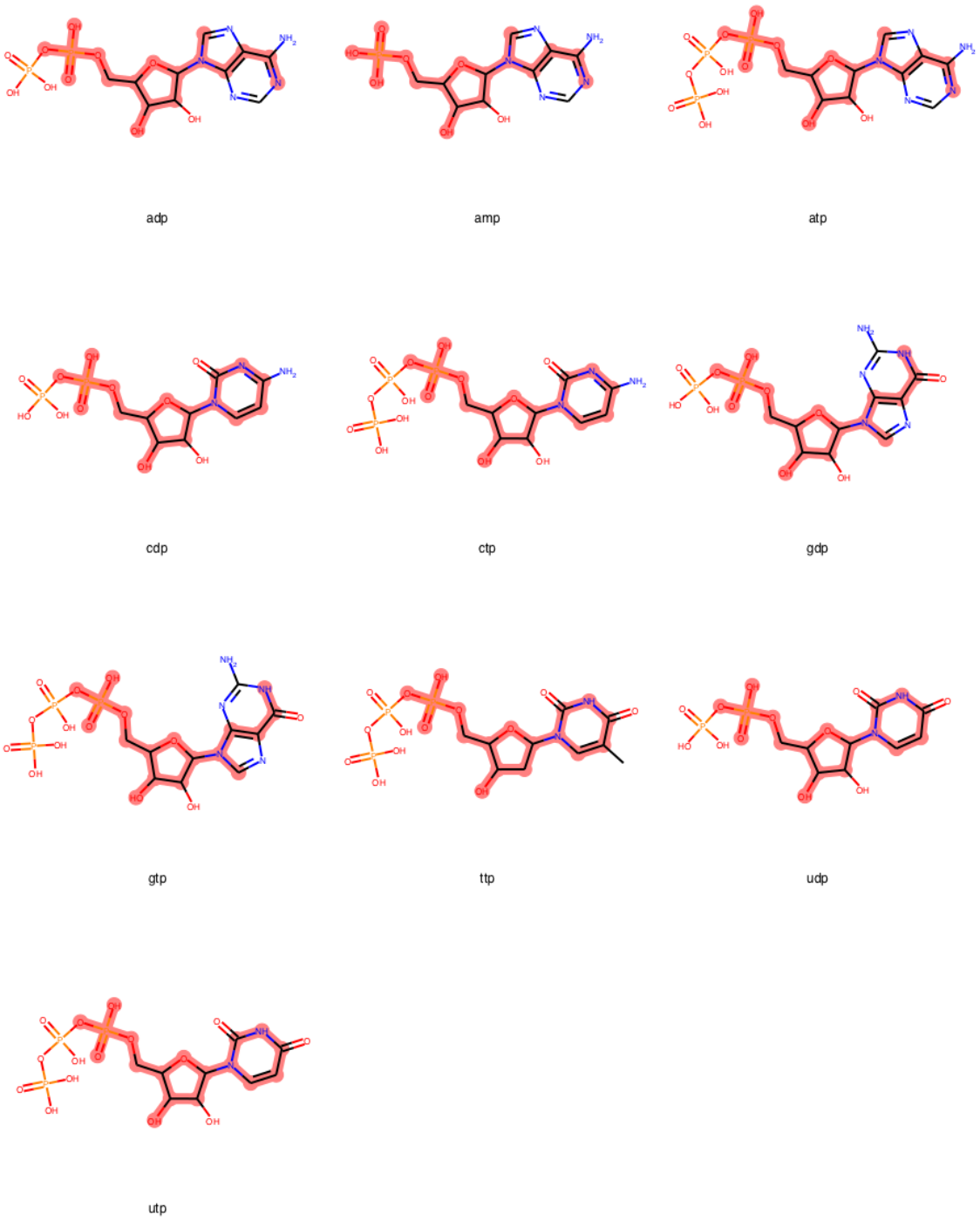
Här finns inga egentliga problem med molekylnamnen, så konverteringen lyckas för alla substrat. Redan i SMILES-format uppvisar strukturerna större likheter än i föregående analys av GST:

```
In [12]: chem_obj = cheminfo.NameToSmile(names=mols, retest_none=False)
         names = chem_obj.names()
         smiles = chem_obj.smiles()
         display(smiles)
```

```
['C1=NC2=C(C(=N1)N)N=CN2C3C(C(C(O3)COP(=O)(O)OP(=O)(O)O)O)O',
 'C1=NC2=C(C(=N1)N)N=CN2C3C(C(C(O3)COP(=O)(O)O)O)O',
 'C1=NC2=C(C(=N1)N)N=CN2C3C(C(C(O3)COP(=O)(O)OP(=O)(O)OP(=O)(O)O)O)O',
 'C1=CN(C(=O)N=C1N)C2C(C(C(O2)COP(=O)(O)OP(=O)(O)O)O)O',
 'C1=CN(C(=O)N=C1N)C2C(C(C(O2)COP(=O)(O)OP(=O)(O)OP(=O)(O)O)O)O',
 'C1=NC2=C(N1C3C(C(C(O3)COP(=O)(O)OP(=O)(O)O)O)O)N=C(NC2=O)N',
 'C1=NC2=C(N1C3C(C(C(O3)COP(=O)(O)OP(=O)(O)OP(=O)(O)O)O)O)N=C(NC2=O)N',
 'CC1=CN(C(=O)NC1=O)C2CC(C(O2)COP(=O)(O)OP(=O)(O)OP(=O)(O)O)O',
 'C1=CN(C(=O)NC1=O)C2C(C(C(O2)COP(=O)(O)OP(=O)(O)O)O)O',
 'C1=CN(C(=O)NC1=O)C2C(C(C(O2)COP(=O)(O)OP(=O)(O)OP(=O)(O)O)O)O']
```

Med namn och SMILES som input till ett `SmileToData`-objekt kan strukturerna ritas ut med största gemensamma delstruktur. Här används samma fingerprints och likhetsmått som i den tidigare analysen.

```
In [13]: data_obj = cheminfo.SmileToData(names, smiles,
                                descriptor='rdkit', metric='tanimoto')
         data_obj.draw_structures(highlight_substructure=True)
```



Likhetsmatrisen nedan indikerar likheter mellan 0.744 och 0.990, med medelvärde 0.846, vilket tyder på en betydligt mer specifik enzymklass med avseende på sina substrat, jämfört med EC 2.5.1.18:

```
In [14]: data_obj.similarity()
```

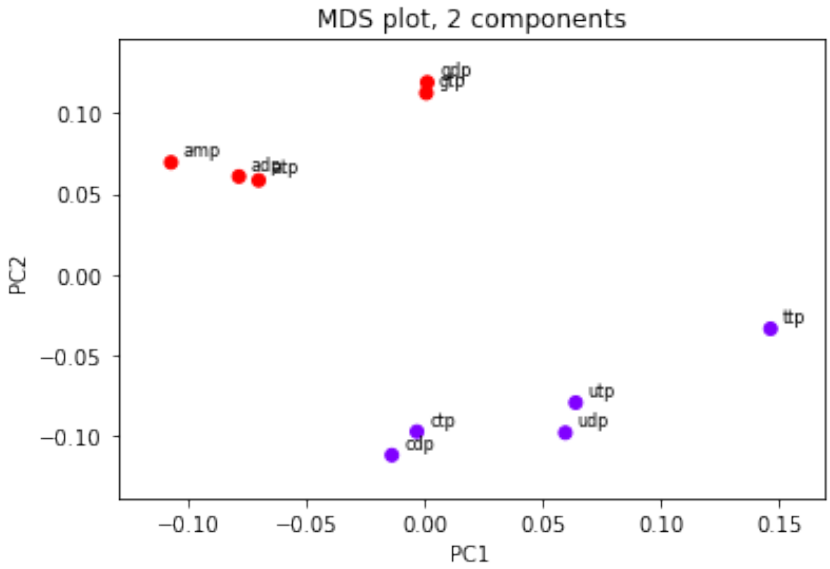
	adp	amp	atp	cdp	ctp	gdp	gtp	ttp	udp	utp
adp	1.0000	0.9731	0.9896	0.8102	0.8157	0.9022	0.8965	0.7686	0.7956	0.8011
amp	0.9731	1.0000	0.9630	0.7847	0.7903	0.8769	0.8715	0.7438	0.7702	0.7758
atp	0.9896	0.9630	1.0000	0.8102	0.8256	0.8954	0.9063	0.7784	0.7956	0.8110
cdp	0.8102	0.7847	0.8102	1.0000	0.9823	0.7840	0.7833	0.8275	0.9181	0.9094
ctp	0.8157	0.7903	0.8256	0.9823	1.0000	0.7876	0.7983	0.8439	0.9094	0.9266
gdp	0.9022	0.8769	0.8954	0.7840	0.7876	1.0000	0.9887	0.7628	0.7891	0.7927
gtp	0.8965	0.8715	0.9063	0.7833	0.7983	0.9887	1.0000	0.7734	0.7883	0.8035
ttp	0.7686	0.7438	0.7784	0.8275	0.8439	0.7628	0.7734	1.0000	0.8800	0.8970
udp	0.7956	0.7702	0.7956	0.9181	0.9094	0.7891	0.7883	0.8800	1.0000	0.9821
utp	0.8011	0.7758	0.8110	0.9094	0.9266	0.7927	0.8035	0.8970	0.9821	1.0000

```
In [15]: data_obj.global_similarity_stats()
```

```
{'min': 0.7437837837837837,
 'max': 0.9895833333333334,
 'sum': 38.07902885187518,
 'median': 0.810989010989011,
 'mean': 0.8462006411527818,
 'stdev': 0.07273984237970946}
```

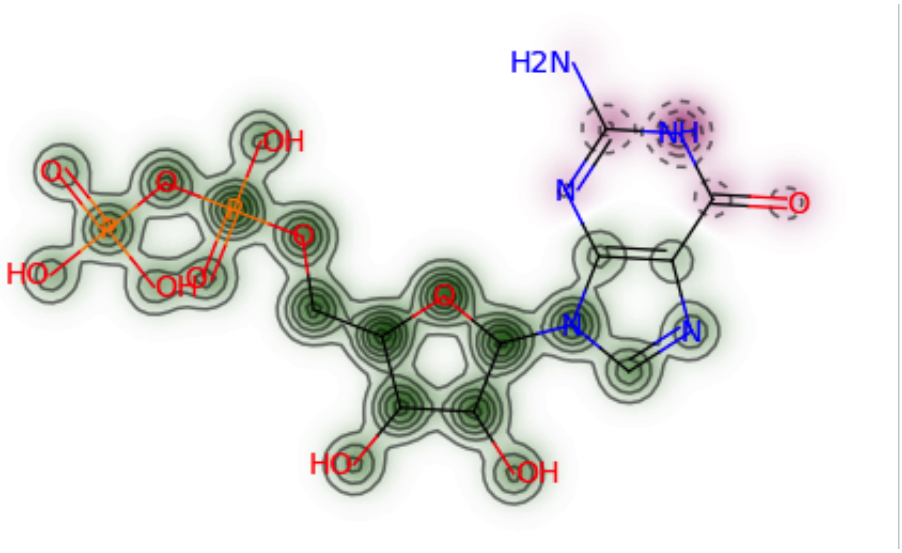
Eftersom substraten är så pass lika gav klusteralgoritmen endast ett kluster i det här fallet med samma parametervärden som tidigare. Genom att sänka parametern `cutoff` i klustermetoden så minskar avståndskravet mellan grupperingarna, vilket möjliggör klustring även för mer lika molekyler. I det här fallet ger en sänkning i cutoff från 0.7 till 0.15 en tydlig uppdelning i två kluster, motsvarande pyrimidinerna (lila) och purinerna (röda). Det här är att förvänta, då den främsta skillnaden mellan dem båda är att pyrimidiner har en ringstruktur medan puriner har två.

```
In [16]: clusters = data_obj.cluster_butina(cutoff = 0.15)
         data_obj.mds(include_labels=True, color_categories=clusters)
```



Istället för att undersöka likheter och skillnader mellan två specifika strukturer genom att lista deras egenskaper, som i tidigare avsnitt, kan vi rita ut en Similarity Map (se avsnitt 3.3.7) med funktionen `draw_mol_comparison`.

```
In [17]: data_obj.draw_mol_comparison(refmol='adp', mol='gdp')
```



Här ritas strukturen för GDP ut, med gröna områden för de strukturer som bidrar mest till likheten med referensmolekylen (ADP), och rosa områden för de strukturer som skulle öka likheten om de exkluderades från strukturen. Utifrån nomenklaturen (guanosindifosfat och adenindifosfat) framgår det att enbart kvävebasen skiljer dem åt. I fallet ovan registreras enbart olikhet vid kvävebasen. Metoden har alltså korrekt utrönt vilken del av molekylen som divergerar från referensmolekylen.

5 | Diskussion

5.1 Slutprodukt/Kodpipeline

Den färdiga pipelinens funktioner testades på en del av rådatan från BRENDA med goda resultat. Utöver det demonstreras ett urval av kodens funktioner gällande likhetsberäkning och visualisering av substrat på två utvalda EC-klasser, 2.5.1.18 (GST) och 3.6.1.5 (apyras) i en exempelanalys, se Resultatet (avsnitt 4).

I exempelanalysen kan vi se att koden fungerar väl för sitt syfte, vi kan tydligt se att GST är mer promiskuöst än apyras utifrån likhetsmatriserna och sammanställningarna av dem. Apyras genererar högre maximum-, minimum- och medelvärde än GST. Medelvärdet för apyras är ungefär 0.85 jämfört med medelvärdet för GST som är cirka 0.26. Att substraten som katalyseras av apyras är mer lika styrks också av MCS-plotten, där apyras substrat har en större gemensam struktur jämfört med GST. Syftet med valet av enzymen är att visa på kontraster mellan ett promiskuöst och ett specifikt enzym. I litteraturen beskrivs GST som mer promiskuöst [11], vilket exempelanalysen också styrker. Vidare visar exempelanalysen på olika sätt att jämföra molekylerna, som till exempel beräkna deskriptorer samt Similarity Maps. De här funktionerna tror vi ska kunna ge en djupare analys av skillnaderna mellan substraten.

5.1.1 Behandling av indata

Det första steget i kodpipelinen var att konvertera substratnamnen till SMILES, vilket visade sig vara något problematiskt. Av omkring 60 000 substratnamn från BRENDA kunde endast 23 000 konverteras till SMILES. På de 37 000 icke konverterade substraten genomfördes en överskådlig filtrering baserad på nyckelord som bland annat ”protein” och ”rna”. Filtreringen visade att minst 6700 av de icke konverterade substraten var proteiner, RNA, DNA och diverse polypeptider. Det här beror troligen på att sådana molekyler är för stora eller har för varierande kemisk sammansättning för att det ska vara praktiskt möjligt att representera dem i SMILES-format.

Utöver det fanns även substratnamn som var för generella för att kunna ge någon SMILES. Ett exempel kan ses i analysen av EC 2.5.1.18, där ”rx” är ett substrat som beskrivs av BRENDA som att ”R kan vara en alifatisk, aromatisk eller heterocyklisk grupp; X kan vara en sulfat-, nitril-, eller halidgrupp” [49].

En manuell sökning mot PubChem av 70 icke konverterade substrat gav, som nämnts i genomförande, omkring 30 sökträffar med giltiga SMILES, så det finns tydliga fall där problemet ligger hos sökalgoritmerna. Det här behöver tas i åtanke som en felkälla i form av saknad data vid framtida analys, då en manuell kontroll av

omkring 30 000 substratnamn inte är praktiskt genomförbart.

5.1.2 Likhetsberäkningar

Likhetsanalyserna som har genomförts baseras i grunden på fingerprintberäkningar, vilka är snabba att beräkna och vanligt förekommande vid analys av kemisk likhet, varför de är en bra utgångspunkt för denna analys. Däremot kommer eventuella begränsningar med fingerprints för representation av molekylär information också att innebära begränsningar i analysen. Ett oundvikligt problem är att en viss mängd information går förlorad genom att reducera komplexa strukturer hos en molekyl till ettor och nollor i en vektor. Vilken information som går förlorad beror i sin tur av vilken information som fingerprintet innehåller. Valet av fingerprint påverkar alltså resultatet, men vilket som är ”det rätta valet” är inte självklart och kan behöva avgöras från fall till fall. Koden tillhandahåller medel för att jämföra vilka fingerprints som överlag genererar högre eller lägre likhet, vilket kan vara relevant information vid val av fingerprint.

En annan grundläggande utgångspunkt för projektet, och inom kemoinformatik i stort, är som nämnt Similar Property Principle (se avsnitt 2.2, det vill säga att lika molekyler med avseende på struktur också kan antas uppvisa likheter i egenskaper. Eftersom beräkningarna av fingerprints baseras på den kemiska strukturen kommer avvikelser från den här principen att förbises i analyser som utförs med hjälp av koden, vilket kan vara en osäkerhet. Likhetsanalysen i sammanhanget utgår alltså från strukturell likhet, snarare än funktionell, vilket är viktigt att komma ihåg.

I den genomförda exempelanalysen har endast Tanimoto använts som likhetsmått, men koden inkluderar även andra likhetsmått. Som nämnts kan de ge olika utslag. För att få ett bättre och mer enhetligt mått på likhet kan datafusion tillämpas på de olika måtten.

5.1.3 Visualerisering av substrat

Funktionerna för att rita upp 2D-strukturerna för samtliga substrat i en EC-klass fungerade till största delen bra. Särskilt bra fungerade funktionen som markerar MCS vid ritningen, då den tydliggjorde var likheten mellan samtliga substrat låg. Utritningen tycktes dock fungera sämre för en del molekyler, särskilt stora och mer komplexa sådana. Mer specifikt tycks 5- och 6-kolringar i komplexa molekyler bli något ihoptrycka.

Similarity Map-funktionen fungerade också väl, och tydliggjorde ytterligare var likheter och olikheter mellan två molekyler låg. Vid analys av stora molekyler gav den dock olika resultat beroende på referensmolekyl och därför behövs individuell bedömning kring vilken referens som ger rimliga resultat.

5.1.4 Vidareutveckling

I nuläget kan koden beräkna likhetsvärden mellan substrat och visualisera strukturerna på olika sätt, men förhoppningen är att den ska kunna utvecklas för vidare applikationer.

Kod som beräknar ett kvantitativt index för promiskuitet, som definierat i [22], utvecklades men togs inte med i vår slutliga kodpipeline. Anledningen var att indexet även inkluderade värden för K_M och k_{cat} för varje reaktion, två parametrar som är organismspecifika och som i de flesta fall inte fanns med i den ursprungliga datan från BRENDA. Därmed skulle många enzymklasser och substrat behöva exkluderas från analysen, som också blir beroende av vilken organism som avses.

Eftersom information om hur bra reaktionerna katalyseras för varje substrat rimligen borde spela roll för enzymets promiskuitet, i kombination med likhetsberäkningar för substraten, är det något som skulle kunna utforskas mer. Med BRENDAs supplementfunktion KENDA (Kinetic ENzyme DAta) kan till exempel mer fullständig organismspecifik kinetisk data tas fram. Den processen hade dock blivit ett mycket större åtagande och lämnades därmed outforskad i det här projektet.

Då det finns många olika fingerprints att välja mellan hade det varit rimligt att använda fler fingerprints för samma dataset av molekyler för att sedan göra datafusion på resultaten. På så sätt skulle sannolikt ett mer informativt värde på likheten mellan substraten fås och eventuella problem med att välja ett specifikt fingerprint kan undvikas.

5.2 Metoder och verktyg

Python fungerade väl för projektets ändamål, i och med att det fanns många användbara kodpaket för intresseområdet. Språket var också relativt enkelt att lära sig och med tanke på det stora utbudet av kodbibliotek utvecklade för Python är det tydligt att väldigt mycket kan göras med relativt lite programmeringsvana med språket.

Gruppen som helhet hade troligen kunnat dra bättre nytta av Git än vad som var fallet. Uppdateringar mot GitHub förekom ganska sporadiskt och borde ha skett betydligt oftare. Det hade troligen resulterat i ett effektivare arbete och färre osäkerheter kring vad andra i gruppen hade gjort. Det uppstod dock inga större problem då alla arbetade i separata filer.

RDKit kom att bli den centrala komponenten i koden, då det visade sig att det utförde de flesta av de funktioner som var av intresse. Det här visar värdet i att utföra en grundlig litteraturstudie för att identifiera tidigare arbete på ett område. Det var överlag enkelt att använda RDKit och hemsidan gav ofta exempel på användande som det gick att ta hjälp av.
Ett återkommande problem med RDKit var dock att vid MDS, PCA och utritning av molekyler blev textstorleken väldigt liten, något som gjorde det svårt att läsa. Att kunna välja textstorlek visade sig vara väldigt invecklat att implementera, så

denna möjlighet utelämnades. Det var också stundvis svårt att hitta någon utförlig dokumentation om vad varje specifik funktion eller parameter i RDKit gjorde.

5.3 Fortsatt forskning

Avsaknaden av en allmänt accepterad definition för promiskuitet har möjligen hållit forskningsområdet tillbaka. Det vore även önskvärt att ta fram ett index över promiskuitet som gäller generellt för en hel enzymklass och inte bara per organism. Det index som föreslås i [22] beror av k_{cat} och k_M, och de värdena skiljer sig så mycket mellan olika organismer att ett enkelt aritmetiskt medelvärde inte är tillräckligt.

BRENDA är en stor databas, men mer data kommer alltid generera bättre resultat. Det är därför viktigt att ständigt sträva efter att upptäcka nya substrat, även för de enzymklasser som redan är väl kartlagda. Likhetsmått och promiskuitetsuppskattningar kan vara viktiga hjälpmedel för avgörandet av vilka nya substrat som ska testas experimentellt.

5.4 Arbetsprocess

Arbetsprocessen för projektet har överlag flutit på bra. I början av projektet hade ingen i gruppen några kunskaper inom kemoinformatik och vi fick därför börja med att lära oss grunderna i området. Då ämnet var väldigt brett var det lätt att hitta vetenskapliga källor, men till en början inte helt lätt att reda ut vilka som var relevanta för projektets ändamål.

Gruppdiskussionerna mellan varje litteratursökning var värdefulla för att sortera ut vilken information och vilka kodbibliotek som var av intresse och har även bidragit till en ökad förståelse för ämnet.

Även programmeringen har varit givande och lärorik då flera i gruppen saknade erfarenhet av programmering i Python sedan tidigare. Det här projektet har varit ett bra tillfälle att få utveckla och fördjupa befintliga programmeringskunskaper.

5.5 Slutsats

Den presenterade koden är en fullständig pipeline som utför de uppgifter som syftet åtog sig. RDKit har bidragit med huvuddelen av de faktiska funktionerna, men många verktyg, såsom CIRpy, PubChemPy och Scikit, har varit nödvändiga för att nå slutprodukten. En användare kan utan någon djupare förståelse för programmering få en övergripande förståelse för koden och bruka den för att beräkna likhet mellan substrat. Kodprodukten kan ge specifika likhetsvärden mellan substratpar på kemoinformatiskt klassiskt vis, men pipelinen är främst designad för att ge användaren en överblick över den rådande likheten i ett större dataset, samt att hitta tendenser i datan. Därtill kan koden vidareutvecklas och användas i en framtida fullskalig analys av hela BRENDA.

Litteratur

[1] N. S. Punekar, *ENZYMES: Catalysis, Kinetics and Mechanisms*. Singapore: Springer Singapore, 2018, ISBN: 978-9811307843.

[2] "Protein promiscuity and its implications for biotechnology", *Nature Biotechnology*, årg. 27, nr 2, s. 157–167, 2009, ISSN: 15461696. DOI: 10.1038/nbt1519.

[3] K. Hult och P. Berglund, "Enzyme promiscuity: mechanism and applications", *Trends in Biotechnology*, årg. 25, nr 5, s. 231–238, 2007, ISSN: 01677799. DOI: 10.1016/j.tibtech.2007.03.002.

[4] S. Martínez Cuesta, S. A. Rahman, N. Furnham och J. M. Thornton, "The Classification and Evolution of Enzyme Function", *Biophysical Journal*, årg. 109, nr 6, s. 1082–1086, 2015, ISSN: 15420086. DOI: 10.1016/j.bpj.2015.04.020.

[5] M. González-Medina, J. J. Naveja, N. Sánchez-Cruz och J. L. Medina-Franco, "Open chemoinformatic resources to explore the structure, properties and chemical space of molecules", *RSC Advances*, årg. 7, nr 85, s. 54 153–54 163, 2017, ISSN: 20462069. DOI: 10.1039/c7ra11831g.

[6] J. Gasteiger och T. Engel, *Chemoinformatics: a textbook*. Weinheim: Wiley-VCH; Chichester: John Wiley, 2003, ISBN: 3-527-30681-1.

[7] D. A. Pertusi, M. E. Moura, J. G. Jeffryes, S. Prabhu, B. Walters Biggs och K. E. Tyo, "Predicting novel substrates for enzymes with minimal experimental effort with active learning", *Metabolic Engineering*, årg. 44, s. 171–181, 2017, ISSN: 10967184. DOI: 10.1016/j.ymben.2017.09.016.

[8] C. L. Linster, E. Van Schaftingen och A. D. Hanson, "Metabolite damage and its repair or pre-emption", *Nature Chemical Biology*, årg. 9, nr 2, s. 72–80, 2013, ISSN: 15524450. DOI: 10.1038/nchembio.1141.

[9] E. Van Schaftingen, R. Rzem, A. Marbaix, F. Collard, M. Veiga-Da-Cunha och C. L. Linster, "Metabolite proofreading, a neglected aspect of intermediary metabolism", *Journal of Inherited Metabolic Disease*, årg. 36, nr 3, s. 427–434, 2013, ISSN: 01418955. DOI: 10.1007/s10545-012-9571-1.

[10] I. Schomburg, L. Jeske, M. Ulbrich, S. Placzek, A. Chang och D. Schomburg, "The BRENDA enzyme information system – From a database to an expert system", *Journal of Biotechnology*, årg. 261, s. 194–206, 2017, ISSN: 18734863. DOI: 10.1016/j.jbiotec.2017.04.020.

[11] W. M. Atkins, "Biological messiness vs. biological genius: Mechanistic aspects and roles of protein promiscuity", *Journal of Steroid Biochemistry and Molecular Biology*, årg. 151, s. 3–11, 2015, ISSN: 18791220. DOI: 10.1016/j.jsbmb.2014.09.010.

[12] P. Willett, "Molecular Similarity Approaches in Chemoinformatics: Early History and Literature Status", i *Frontiers in Molecular Design and Chemical Information Science - Herman Skolnik Award Symposium 2015: Jürgen Bajorath*, R. J. Bienstock, V. Shanmugasundaram och J. Bajorath, utg., kap. 6, s. 67–89. DOI: 10.1021/bk-2016-1222.ch006.

[13] N. Nikolova och J. Jaworska, "Approaches to Measure Chemical Similarity - a Review", *QSAR & Combinatorial Science*, årg. 22, nr 910, s. 1006–1026, 2004, ISSN: 1611-020X. DOI: 10.1002/qsar.200330831.

[14] D. Weininger, "SMILES, a Chemical Language and Information System: 1: Introduction to Methodology and Encoding Rules", *Journal of Chemical Information and Computer Sciences*, årg. 28, nr 1, s. 31–36, 1988, ISSN: 00952338. DOI: 10.1021/ci00057a005.

[15] S. R. Heller, A. McNaught, I. Pletnev, S. Stein och D. Tchekhovskoi, "InChI, the IUPAC International Chemical Identifier", *Journal of Cheminformatics*, årg. 7, nr 1, 2015, ISSN: 17582946. DOI: 10.1186/s13321-015-0068-4.

[16] E. Duesbury, J. Holliday och P. Willett, "Maximum Common Substructure-Based Data Fusion in Similarity Searching", *Journal of Chemical Information and Modeling*, årg. 55, nr 2, s. 222–230, 2015. DOI: 10.1021/ci5005702.

[17] M. Karthikeyan och R. Vyas, *Practical Chemoinformatics*. Springer, 2014. DOI: 10.1007/978-81-322-1780-0.

[18] A. Cereto-Massagué, M. J. Ojeda, C. Valls, M. Mulero, S. Garcia-Vallvé och G. Pujadas, "Molecular fingerprint similarity search in virtual screening", *Methods*, årg. 71, s. 58–63, 2015, ISSN: 1046-2023. DOI: https://doi.org/10.1016/j.ymeth.2014.08.005.

[19] D. Rogers och M. Hahn, "Extended-Connectivity Fingerprints", *Journal of Chemical Information and Modeling*, årg. 50, nr 5, s. 742–754, 2010. DOI: 10.1021/ci100050t.

[20] H. L. Morgan, "The Generation of a Unique Machine Description for Chemical Structures-A Technique Developed at Chemical Abstracts Service.", *Journal of Chemical Documentation*, årg. 5, nr 2, s. 107–113, 1965. DOI: 10.1021/c160017a018.

[21] A. Bajusz Dávidand Rácz och K. Héberger, "Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations?", *Journal of Cheminformatics*, årg. 7, nr 1, s. 20, 2015, ISSN: 1758-2946. DOI: 10.1186/s13321-015-0069-3.

[22] A. Nath och W. M. Atkins, "A Quantitative Index of Substrate Promiscuity", *Biochemistry*, årg. 47, nr 1, s. 157–166, 2008. DOI: 10.1021/bi701448p.

[23] P. Willett, "Combination of Similarity Rankings Using Data Fusion", *Journal of Chemical Information and Modeling*, årg. 53, nr 1, s. 1–10, 2013. DOI: 10.1021/ci300547g.

[24] P. Willett, "Fusing similarity rankings in ligand-based virtual screening", *Computational and structural biotechnology journal*, årg. 5, e201302002, febr. 2013. DOI: 10.5936/csbj.201302002.

[25] S. Kaski, "Dimensionality reduction by random mapping: Fast similarity computation for clustering", i *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98CH36227)*, IEEE, vol. 1, 1998, s. 413–418.

[26] I. Jolliffe, *Principal component analysis*. Springer, 2011.

[27] I. Borg och P. Groenen, "Modern multidimensional scaling: Theory and applications", *Journal of Educational Measurement*, årg. 40, nr 3, s. 277–280, 2003.

[28] L. v. d. Maaten och G. Hinton, "Visualizing data using t-SNE", *Journal of machine learning research*, årg. 9, nr 11, s. 2579–2605, 2008.

[29] A. K. Jain, M. N. Murty och P. J. Flynn, "Data clustering: a review", *ACM computing surveys (CSUR)*, årg. 31, nr 3, s. 264–323, 1999.

[30] B. W. Kernighan och P. J. Plauger, "Software Tools", *SIGSOFT Softw. Eng. Notes*, årg. 1, nr 1, s. 15–20, 1976, ISSN: 0163-5948. DOI: 10.1145/1010726.1010728.

[31] N. M. O'Boyle och G. R. Hutchison, "Cinfony - combining open source cheminformatics toolkits behind a common interface", *Chemistry Central Journal*, årg. 2, nr 1, s. 1–10, 2008, ISSN: 1752153X. DOI: 10.1186/1752-153X-2-24.

[32] S. Beisken, T. Meinl, B. Wiswedel, L. F. de Figueiredo, M. Berthold och C. Steinbeck, "KNIME-CDK: Workflow-driven cheminformatics", *BMC Bioinformatics*, årg. 14, nr 1, 2013, ISSN: 14712105. DOI: 10.1186/1471-2105-14-257.

[33] D. Lowe, *OPSIN: Open Parser for Systematic IUPAC nomenclature*, https://opsin.ch.cam.ac.uk, Hämtad: 2019-05-13, 2019.

[34] NCI/CADD Group, *Chemical Identifier Resolver*, https://cactus.nci.nih.gov/chemical/structure, Hämtad: 2019-05-13, 2019.

[35] *Open Babel: The Open Source Chemistry Toolbox*, http://openbabel.org/wiki/Main_Page, Hämtad: 2019-05-13, 2016.

[36] U.S. National Library of Medicine, *PubChem*, https://pubchem.ncbi.nlm.nih.gov, Hämtad: 2019-05-13, 2019.

[37] M. Swain, *CIRpy*, https://cirpy.readthedocs.io/en/latest/index.html, Hämtad: 2019-05-10, 2015.

[38] M. Swain, *PubChemPy documentation*, https://pubchempy.readthedocs.io/en/latest/, Hämtad: 2019-05-10, 2014.

[39] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt och G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project", i *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, s. 108–122.

[40] D. Schomburg och I. Schomburg, ”Enzyme Databases”, i *Data Mining Techniques for the Life Sciences*, O. Carugo och F. Eisenhaber, utg. Totowa, NJ: Humana Press, 2010, s. 113–128, ISBN: 978-1-60327-241-4. DOI: 10.1007/978-1-60327-241-4_7.

[41] B. S. Scott Chacon, *Pro Git, Everything you need to know about git*. Apress, 2019.

[42] Codecademy, https://www.codecademy.com, Hämtad: 2019-05-06.

[43] RDKit, https://www.rdkit.org, Hämtad: 2019-05-07.

[44] S. Riniker och G. A. Landrum, ”Similarity maps - A visualization strategy for molecular fingerprints and machine-learning methods”, *Journal of Cheminformatics*, årg. 5, nr 9, s. 1–7, 2013, ISSN: 17582946. DOI: 10.1186/1758-2946-5-43.

[45] D. Butina, ”Unsupervised data base clustering based on daylight’s fingerprint and Tanimoto similarity: A fast and automated way to cluster small and large data sets”, *Journal of Chemical Information and Computer Sciences*, årg. 39, nr 4, s. 747–750, 1999.

[46] M. Ester, H.-P. Kriegel, J. Sander och X. Xu, ”A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”, i *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD’96, AAAI Press, 1996, s. 226–231.

[47] P. Jupyter, https://jupyter.org, Hämtad: 2019-05-11, 2019.

[48] C. K. Mathews, K. E. Van Holde, D. R. Appling och S. J. Anthony-Cahill, *Biochemistry*. Pearson, 2013, ISBN: 9780132775045.

[49] BRENDA, *Information on EC 2.5.1.18 - glutathione transferase*, https://www.brenda-enzymes.org/enzyme.php?ecno=2.5.1.18, Hämtad: 2019-05-07.

Bilagor

A | Likhetskoefficienter

Nedan visas en tabell över alla likhetsmått som diskuteras i rapporten alternativt som går att använda i koden.

Tabell A.1: Likhets/avståndsmått som kan användas till att beräkna likheter mellan molekylära fingerprints. a = antal 1-bitar i fingerprint A, b = antal 1-bitar i fingerprint B, c = antal 1-bitar gemensamt för både A och B, m = totalt antal bitar i fingerprint A och B. D står för *distance* och S för *similarity*.

Likhetsmått	Ekvation	Intervall
Tanimoto	$S_{A,B} = \frac{c}{a+b-c}$	[0,1]
Dice	$S_{A,B} = \frac{2c}{a+b}$	[0,1]
Cosinuslikhet	$S_{A,B} = \frac{c}{\sqrt{ab}}$	[0,1]
Soergel	$D_{A,B} = 1 - \frac{c}{a+b-c}$	[0,1]
Sokal	$S_{A,B} = \frac{c}{2a+2b-3c}$	[0,1]
Russel	$S_{A,B} = \frac{c}{m}$	[0,1]
Mcconnaughey	$S_{A,B} = \frac{c(a+b)-ab}{ab}$	[-1,1]
Kulczynski	$S_{A,B} = \frac{c(a+b)}{2ab}$	[0,1]
Braunblanquet	$S_{A,B} = \frac{c}{max(a,b)}$	[0,1]
Asymmetric	$S_{A,B} = \frac{c}{min(a,b)}$	[0,1]
Euklidiskt	$D_{A,B} = \sqrt{a+b-2c}$	[0,N]
Manhattan	$D_{A,B} = a+b-2c$	[0,N]

B | Undersökta kodbibliotek

Tabell B.1: De huvudsakliga kodbibliotek/program som hittades under litteraturstudien och undersöktes inom ramen för projektet.

Kodbibliotek	Beskrivning/Funktion
RDKit	Beräknar deskriptorer, fingerprints, MCS och ritar molekyler.
JChem	Javabibliotek från ChemAxon för beräkning av fingerprints.
CIRpy	Python-koppling till Chemical Identifier Resolver (CIR) för att översätta substratnamn till representation (SMILES).
PubChemPy	Python-koppling till PubChem för att översätta substratnamn till representation (SMILES).
OPSIN	Konverterar namn till struktur. Används bland annat av CIR.
Open Babel	Konverterar mellan olika kemiska filformat.
Cinfony	Gemensamt Python-interface för Open Babel, RDKit, CDK, Indigo, JChem och OPSIN.
KNIME	Mjukvara för dataanalys med kopplingar till bland annat RDKit, CDK, Indigo och CIR.

C | Parametervärden

Tillgängliga fingerprints som kan anges som input i SmileToData ges av metoden `valid_descriptors()`, där siffrorna efter 'morgan' anger val av radier för beräkning av Morgan fingerprints (se avsnitt 2.3.2):

```
['maccs', 'morgan3', 'morgan5', 'rdkit']
```

Tillgängliga likhetsmått för input i SmileToData ges av metoden `valid_metrics()`:

```
['asymmetric',
 'braunblanquet',
 'cosine',
 'dice',
 'kulczynski',
 'mcconnaughey',
 'rogotgoldberg',
 'russel',
 'sokal',
 'tanimoto']
```

Förklaringar till de egenskaper som beräknas för substraten i SmileToData ges till sist av metoden `explain_properties()`:

```
['"molwt": molecular weight',
 '"num_alirings": number of aliphatic rings',
 '"num_arorings": number of aromatic rings',
 '"num_hbond_acceptors": number of hydrogen bond acceptors',
 '"num_hbond_donors": number of hydrogen bond donors',
 '"num_hetatom": number of hetero atoms',
 '"num_nhoh": number of NH and OH groups',
 '"num_no": number of N and O atoms',
 '"num_rotbond": number of rotatable bonds',
 '"tpsa": polar surface area']
```