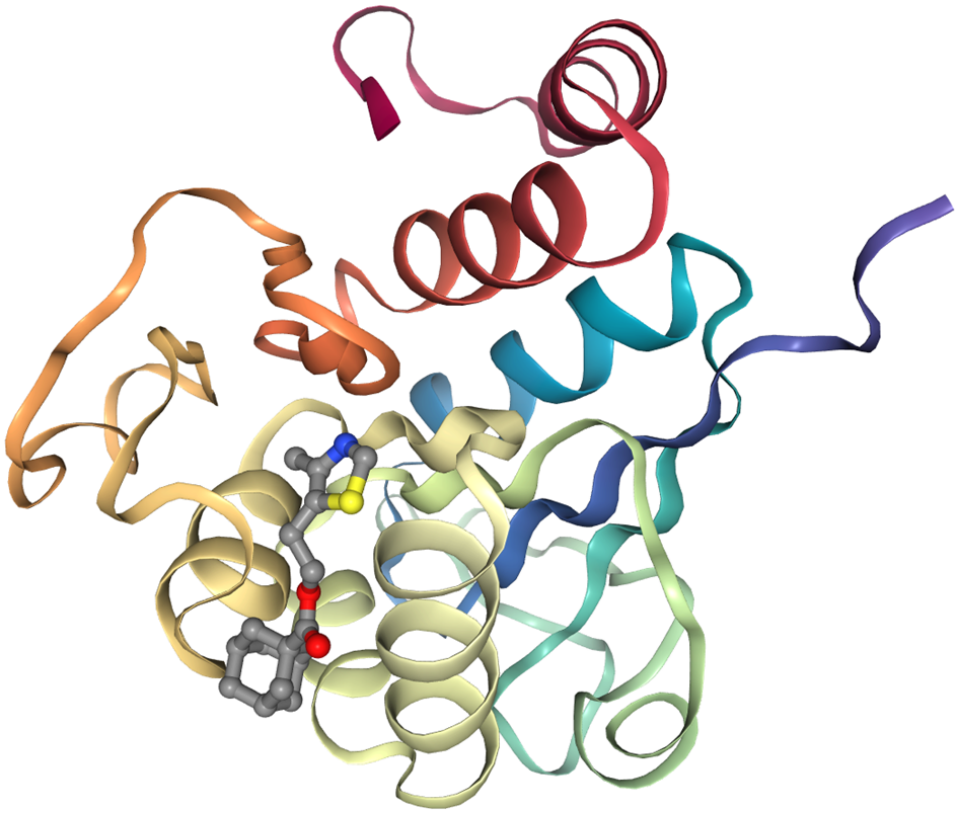
Protein-Liganden-Docking

Kevin Kretz, German Esaulkov, Leander Schäfer

12. Januar 2022



Inhaltsverzeichnis

1 Kurzfassung

Tropenkrankheiten stellen in ihren Verbeitungsgebieten eine extreme Bedrohung für die dortige Bevölkerung dar. Gemessen an ihrer Bedeutung - Malaria ist z. B. mit 200 Millionen Fällen pro Jahr die häufigste Infektionskrankheit der Welt - erfahren sie in nicht betroffenen Industrieländern nur wenig Aufmerksamkeit in Medien, in Form von Forschungsprojekten und in den Entwicklungsabteilungen von Pharmafirmen.

Aufgrund der Veröffentlichung des AlphaFold-Papers im Juli 2021 und der gleichzeitig veröffentlichten Datenbank von dreidimensionalen Proteinmodellen, sowie dem UseGalaxy-Server der Universität Freiburg haben wir gute Voraussetzungen bekommen, um mithilfe von Protein-Liganden-Docking nach möglichen Wirkstoffen gegen Tropenkrankheiten zu suchen.

2 Einleitung

Tropenkrankheiten stellen in Entwicklungsländern verheerende Schäden an und fordern viele Opfer ein. So ist z.B. Malaria mit 200 Millionen Fällen pro

Abbildung 1: Logo von Galaxy Europe

Jahr die häufigste Infektionskrankheit der Welt. Trotz der massiven Schäden, die Tropenkrankheiten, wie z.B. Malaria, die Afrikanische Schlafkrankheit und Hepatitis A- anrichten, schenken Industrieländer und die dort hiesigen Pharmakonzerne diesen Bedrohungen nur wenig Beachtung. Die Entwicklung und der Verkauf von Medikamenten gegen diese weit verbreiteten Krankheiten erschien den Konzernen als nicht lukrativ genug, obwohl der Markt dafür vorhanden wäre und wurde demnach vernachlässigt. (Doch mittlerweile gibt es Vereinigungen, die gegen diese reale Gefahr kämpfen. Wir möchten uns dem anschließen.) Nun versuchen wir mit unserem Projekt mithilfe von Bioinformatik (weitere) helfende Arzneimittel zu finden und unter Umständen auch eine neue Entdeckung zu machen. Da uns keine Laborforschungsmittel oder Supercomputer zur Verfügung stehen, müssen wir auf öffentlich zugängliche Ressourcen zurückgreifen. Hierfür gibt es die webbasierte Plattform für Computational Science, mit Fokus auf Biologie, namens „Galaxy“. Damit kann man selbstverständlich kein fertiges Medikament entwickeln, dennoch erhoffen wir uns damit eine Grundlage für weitere Forschung zu schaffen. Dort versuchen wir mithilfe der zur Verfügung gestellten Werkzeuge Protein-Liganden- Docking zu simulieren und damit einen wirkungsvollen Stoff gegen die Krankheiten zu finden. Doch was genau ist dieses Protein-Liganden-Docking? Dieses Verfahren ist eine „molecular modelling“-Technik, wobei mittels Bioinformatik versucht wird herauszufinden, welche Liganden mit welchen Proteinen an welcher Stelle binden. Man benötigt dementsprechend Daten über den Liganden und den Rezeptor des Proteins. Jetzt hat dieses Verfahren eine pharmazeutische Bedeutung, da man zum einen mögliche Wirkstoffe finden kann, die vitale Proteine des Krankheitserregers außer Kraft setzen können. Weiter erleichtert es auch die Suche, da das Auswahlverfahren nun „in silico“ erfolgen kann und nicht alle möglichen Kandidaten im Labor getestet werden müssen.

Auch wir möchten dieses bioinformatische Verfahren anwenden. Doch warum haben wir ausgerechnet diese Krankheiten gewählt?

Malaria ist eine vor allem in den tropischen Regionen Afrikas anzutreffen, aber auch in Südostasien und in den nördlichen Teilen Südamerikas zu finden. Wie bereits erwähnt ist Malaria, auch Sumpf- oder Tropenfieber bekannt, die häufigste Tropenkrankheit. Der wichtigste Überträger der Krankheit ist die weibliche Anophelesmücke. Malaria kann Symptome wie Fieber, Er-

Abbildung 2: Die Anophelesmücke beim Blutsaugen CDC/James Gathany

brechen, Gelbsucht und Krämpfe umfassen. Vor allem die von uns gewählte Variante der Falciparum-Malaria ruft schwere Symptome wie Lähmung oder Koma hervor. Über Lungen- oder Nierenversagen führt die Krankheit zum Tod.

Ebenfalls eine durch ein Insekt verbreitete Krankheit ist die Afrikanische Trypanosomiasis, oder auch Afrikanische Schlafkrankheit, dessen Erreger Trypanosoma brucei durch die Tsetse-Fliege übertragen wird. Man kann aktuell mit ca. 500 000 Betroffenen in Afrika rechnen. Über Fieber, Gliederschmerzen, Lymphknotenschwellung und Anämie führt die Krankheit zum namengebenden Dämmerzustand und anschließend zum Tod.

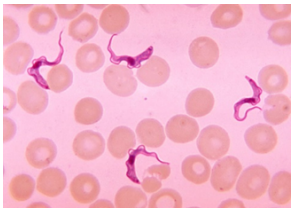


Abbildung 3: Trypanosoma, die Erreger der Schlafkrankheit (CDC/Dr. Myron G. Schultz)

Als dritte Krankheit betrachten wir die Chagas-Krankheit, auch Südamerikanische Trypanosomiasis genannt. Der Erreger Trypanosoma cruci, ein Verwandter der Trypanosoma brucei, wird durch den Kot verschiedener Raubwanzen, aber vor allem von Triatoma infestans, übertragen. Aktuell gibt es ca. 18 Millionen Erkrankte, wobei jedes Jahr 50.000 dazukommen. Zahl der Todesfälle beträgt jährlich um die 15.000. Die Krankheit verursacht Ödeme, chronisches Herzversagen und dem Absterben von Nervenzellen im Darm. Dies führt mitunter zum Tod durch Darmverschluss oder Darmdurchbruch.

Sowohl die Chagas-Krankheit als auch die Afrikanische Trypanosomiasis sind von der Weltgesundheitsorganisation als „neglected tropical diseases“ (NTD´s) anerkannt.

Mit Galaxy haben wir schon eine gute Grundlage zur Erforschung und Durchführung des Protein-Liganden-Dockings, jedoch fehlen uns die Daten über den Arzneistoff, also den Liganden, und das Organell, sprich ein vitales Protein, welches im Erreger außer Kraft gesetzt werden soll, sodass er stirbt. Man benötigt also Datenbanken, mit den entsprechenden Sequenzen bzw. Strukturen. Für das Protein gibt es die die Proteinstrukturdatenbank AlphaFold, welches von DeepMind und EMBL-EBI entwickelt wurde. Da werden Proteinstrukturen aufgeführt, die von einer AI auf Grundlage der Aminosäuresequenz ermittelt bzw. vorher-

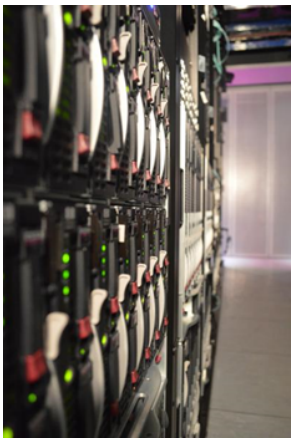


Abbildung 4: Datencenter des EMBL-EBI (EMBL-EBI)

gesagt wurden. Dabei sind diese Vorhersagen sehr präzise. Hat man dann eine mögliche Struktur für den Liganden gefunden, kann man nach realen Chemikalien in der EMBL-EBI Datenbank suchen. Das „European Molecular Biology Laboratory Bioinformatics Institute“ beherbergt die größte öffentlich zugängliche biologische Datenbank und bietet gleichzeitig auch bioinformatische Dienste für Forschende aus aller Welt an. Nur mithilfe von Galaxy, AlphaFold und EMBL-EBI können wir an unserem Projekt forschen. Und wie genau das abläuft wird im weiteren Verlauf erläutert.

3 Vorgehensweise, Materialien und Methode

3.1 Protein- und Ligandenstruktur eingeben

Auf der Galaxy-Website kann man ein neues Programm (History genannt) erstellen. Danach luden wir in der History als erstes über den „Upload-Data“-Button die PDB -Datei des Pro-teins Plasmodium Falciparum von unserem lokalen PC hoch.

SMILES-String-Herkunft erklären!!!

Dann brauchten wir eine Compound library, in welcher Moleküle aufgelistet sein sollten, die dem Molekül des natürlichen Liganden vom Plasmodium Falciparum ähnelten. Diese erstellten wir, indem wir das auf Galaxy bereitstehende Tool „Search ChEMBL database“ nutzten. Die-ses Tool ermöglicht es, die ChEMBL-Datenbank mit Hilfe des Tanimoto-Algorithmus nach Molekülen zu durchsuchen, die ähnlich zu einem Molekül sind, dass man über einen einzuge-benden SMILES-String übermittelt. Wir gaben also den SMILES-String `(c1(c(ncs1)C)CCO)` des natürlichen Liganden (5-(2-hydroxyethyl)-4-methylthiazole) als SMILES-Input eingegeben. In dem Tool konnte man auswählen, wie hoch der „Tanimoto cutoff score“ sein soll. Dieser Score entscheidet, wie hoch die Ähnlichkeit zwischen einem Liganden aus der Datenbank und dem eingegeben Vergleichs-Liganden sein muss, um in die Compound library aufgenommen zu werden. Ein Bereich von 40 (niedrige Ähnlichkeit ist ausreichend) bis 100 steht zur Verfü-gung Außerdem haben wir ausgewählt, dass der „Filter for Lipinski‘s Rule of Five“ ange-wendet werden soll. Die Rule of Five ist eine Faustregel, die Moleküle auf ihre orale Biover-fügbarkeit überprüft, also auf die Frage, ob ein Molekül prinzipiell als oral-einzunehmendes Arzneimittel geeignet wäre. Evt. einzelne Kriterien und Entwickler nennen. Nach dem Ausführen dieses Befehls haben wir nun also eine Compound library im Format einer SMILES-Liste, die aus

in diesem Fall 50 Molekülen besteht, die dem natürlichen Ligan-den des Plasmodium Falciparums ähnlich sind und außerdem als Arzneimittel theoretisch ge-eignet wären. Die Datei sieht folgendermaßen aus: In jeder Zeile ist ein Molekül aufgelistet, zunächst mit dem SMILES-String und dann mit dem Titel, also der ChEMBL-ID des jeweili-gen Moleküls. Folgendes sind die Strukturformeln der Compound library und des Natürlichen Liganden. Alle Moleküle aus der Compound library haben als Titel ihre ChEMBL-ID, der Natürliche Ligand ist schlicht mit „ligand“ betitelt. Ähnlichkeiten sind zum Teil klar erkennbar.

3.2 Dateien für das Docking vorbereiten

Danach konvertierten wir das Protein, also den Rezeptor, in eine PDBQT-Datei (wir nannten sie „Prepared receptor“), die für das Docking benötigt wird. Dann haben wir das fpocket-Tool genutzt, um überhaupt die Stelle auszumachen, an der der Ligand an das Protein docken sollte. Fpocket ist die wohl beste Möglichkeit, Bindungsstellen für kleine Moleküle zu ermitteln. Es ist ein Open-Source-Programm, das mit Hilfe der -Sphären-Theorie arbeitet. Input war die PDB-Datei des Proteins, Output war zum einen eine Datei namens „Pocket properties“, in der die verschiedenen Eigenschaften der Taschen aufge-listet waren. Am wichtigsten war ihr Gesamtscore, nach dem wurden sie auch sortiert, pocket1 war mit einem Score von rund 0,4 die beste. Evt. Alpha-Sphären erklären. Zum anderen kam eine Dateiliste heraus, die alle acht Taschen im PDB-Format umfasste, also die Atome in Kontakt mit jeder Tasche. Danach konnten wir mit dem Tool „Calculate the box parameters using RDKit“ eine Konfigu-rationsdatei für das Docking erstellen, die als Input pocket1 hatte und unter anderem die Grö-ße des für das Docking benötigten Bereichs beinhaltet, also x-, y- und z-Puffer. Das Tool nutzt RDKit, eine Sammlung an Programmen für Chemieinformatik und Maschinelles Lernen. Die Datei heißt „Docking config file“ Dann mussten wir nur noch die Compound library mit dem Tool „Compound conversion“, einem Dateiformat-Konvertierungstool, die Compound library in eine SDF-Datei namens „Prepared ligands“ umwandeln, und das Docking konnte beginnen: Als Docking-Programm haben wir AutoDock Vina verwendet, das bei Galaxy über das Tool „Vina Docking“ zur Verfügung steht. AutoDock Vina ist ein Open-Source-Programm, das von The Scripps Research Institute entwickelt wurde. In Tests, in denen es mit anderen Moleku-lar-Docking-Simulations-Programmen verglichen wurde, schnitt es gut bis sehr gut ab.

AutoDock Vina erklären!!!

Als Input verwendeten wir für den Rezeptor die PDBQT-Datei „Prepared receptor“, für die Liganden die SDF-Datei „Prepared ligands with errors“ (Weshalb dieser Name wird gleich erklärt.) und als Box-Konfiguration die TXT-Datei „Docking config file“. Den pH-Wert setz-ten wir auf 7.4, den pH-Wert von menschlichem Blut. Das Ergebnis, eine SDF-Dateiliste, war jedoch als Fehler markiert, da acht der 50 SMILES-Strings nicht kompatibel waren. In der Fehlermeldung haben wir nachgeschaut, welche Lig-anden Probleme machten. Dort waren sieben Liganden aufgelistet, ein weiterer Ligand wurde zwar nicht in der Fehlermeldung aufgeführt, wurde jedoch auch nicht ausgeführt und war dementsprechend auch nicht kompatibel. Wir haben uns daraufhin die Compound library her-untergeladen, mit dem Windows-Editor geöffnet und alle Problem-Liganden (es waren die Liganden Nummer 6, 9, 12, 20, 27, 30, 39 und 45) manuell gelöscht, die Datei, die nun also noch 42 Liganden umfasste, lokal gespeichert und wieder in der Galaxy-History hochgeladen. Dort haben wir sie dann wieder in eine SDF-Datei konvertiert, und das Docking mit densel-ben Inputs (selbstverständlich abgesehen von den Liganden) wie davor wiederholt. Nun war das Ergebnis eine funktionierende Dateiliste. Danach haben wir eine SMI-Datei des natürlichen Liganden erstellt, in dem wir den schon am Anfang verwendeten SMI-String mit dem Titel „ligand“ über den Button „Upload Data“ -¿ „Paste/Fetch data“ und der Einstellung „Convert spaces to tabs“ eingegeben haben. Die Datei nannten wir „Natural ligand“. Dann nutzen wir das Tool „Concatenate datasets“, um die Datasets „Natural ligand“ und „Compound library without errors“ zu einer SMI-Datei mit 43 Liganden zusammenzuführen. Dieses Dataset nannten wir „Labelled compound library“.

Dieses visualisierten wir mit dem Tool „Visualisation“, das uns eine SVG-Datei mit den Strukturformeln der Liganden brachte. Mit dem Tool „Molecule to fingerprint” erstellten wir die Labelled compound library in eine Open Babel FP2 fingerprints-Datei. Molekulare Fingerprints kodieren Molekülstrukturen in Bits. Mit ihnen kann man Moleküle auf ihre Ähnlichkeit untersuchen. Danach konnten wir mit der Fingerprints-Datei als Input die Liganden mit dem Tool „Taylor Butina clustering“ in Cluster einsortieren. Der Taylor-Butina Algorithmus erstellt Cluster, in denen Moleküle aufgelistet sind, die zu dem zentralen Molekül des Clusters eine bestimmte Ähnlichkeit aufweisen. Den Ähnlichkeits-Schwellenwert kann man im Tool angeben, wir ha-ben 0,8 verwendet. Das Taylor-Butina Clustering ist (im Gegensatz zum nächsten Clustertool, welches wir verwenden werden) nicht hierarchisch, es gibt also keine Über- und Untercluster. Dann haben wir noch mit dem Tool „NxN clustering“ ein Dendogramm erstellt, das erzeugt wurde, indem das Tool eine Selbstähnlichkeitsmatrix verwendet hat, um die

Ähnlichkeit zu ermitteln. Um die Ergebnisse des Dockings, die bisher noch in unterschiedlichen SDF-Dateien lagen, übersichtlicher zu machen, verwendeten wir das Tool „Extract values from an SD-file“. Input waren die im Docking-Schritt erstellte Kollektion aus SDF-Dateien, Output eine Kollektion aus Tabular-Dateien, die die Informationen anders darstellten. Um eine Datei mit allen Ergebnissen zu bekommen, nutzen wir das Tool „Collapse Collec-tion“. Dieses reihte alle vom gerade benutzen Tool erstellten Dateien in einer Tabular-Datei hintereinander auf. Dann sortierten wir die Liganden mit dem Tool „Sort Dataset“ nach dem Docking-Score. Dann hatten wir also eine übersichtliche Datei, in der alle Ergebnisse des Dockings nach Bin-dungsstabilität sortiert waren. Von den fünf Liganden, die die stabilsten Bindungen erzielt hatten, konvertierten wir danach noch die SDF-Dateien des Dockings mit „Compound conversion“ in PDB-Dateien, um mit dem NGL-Viewer 3D-Visualisierungen von ihnen vornehmen zu können, das ist direkt in Ga-laxy möglich. Auf der Website von NGL konnten wir dann auch direkt die PDB-Datei des Proteins und die des Liganden hochladen, und so beide in der beim Docking bewerteten Position visualisiert betrachten.

Nach dem exakt selben Verfahren haben wir die Afrikanische Schlafkrankheit, genauer das für sie wichtige Hitze-Schock Protein 83 (HSP 83) behandelt. Unterschiede lagen darin, dass das Protein die Compound library 52 Moleküle umfasste und diese von Anfang an alle funkti-onstüchtig waren, der bei Malaria notwendige Schritt des manuellen Löschens von Problem-Liganden, erneute Aufbereitung und erneutes Docking fielen also weg. HSP 83 besaß wesent-lich mehr Taschen als P. Falciparum (57).

3.3 SMI...

Wir schrieben hierzu folgendes Python-Script:

```
1  # Jugend forscht 2021 - Kevin Kretz, German Esaulkov, Leander
       Sch fer
2  import os
3  import xml.etree.ElementTree as ET
4
5  import requests
6
7  input_directory = "Input smi-files/"
8
9
10 def main():                                              # !
       Hauptfunktion
11     for smi_filename in os.listdir(input_directory):    # F r
       jede SMI-Datei im Inputverzeichnis wird Folgendes getan:
```

```
12          smi_path = input_directory + smi_filename       # Der
    Name der Krankheit wird eingelesen.
13          disease_name = smi_filename.rsplit(".")[0]
14          disease_dir = disease_name + "/"
15          if not os.path.exists(disease_dir):
16              os.makedirs(disease_dir)
17          print(disease_name)
18          run_for_disease(smi_path, disease_dir)          # Die
    Funktion run_for_disease wird f r die Krankheit
    ausgef hrt.
19          print("*****\n\n")
20
21
22  def read_in_strings(filename):
23      file = open(filename)
24      list_txt = file.readlines()
25
26      strings = list(one_strings.rsplit("\t") for one_strings in
    list_txt)
27
28      return strings
29
30
31  def substructure_search(strings_data, directory):
32      strings_input = strings_data[0]
33      ChEMBL_id_input = strings_data[1][:-1]
34
35      substructure_results_request = requests.get('https://www.
    ebi.ac.uk/chembl/api/data/substructure/' + strings_input)
36      substructure_results = substructure_results_request.content
    .decode()
37
38      filename = "substructure_results_" + ChEMBL_id_input + ".
    xml"
39      filepath = directory + filename
40
41      xml_file = open(filepath, "w")
42      xml_file.write(substructure_results)
43      xml_file.close()
44
45      tree = ET.parse(filepath)
46
47      # tree = ET.fromstring(substructure_results)
48      root = tree.getroot()
49
50      for molecule in root.iter("molecule"):
51          for molecule_chembl_id_ in molecule.iter("
    molecule_chembl_id"):
52              molecule_chembl_id = molecule_chembl_id_.text
```

```
        for max_phase_ in molecule.iter("max_phase"):
            if max_phase_.text:
                max_phase = int(max_phase_.text)
                print("FOUND:     " + molecule_chembl_id + "\
    nFROM:      " + ChEMBL_id_input + "\nmax_phase: " + str(
                    max_phase))
            else:
                max_phase = -1


def run_for_disease(smi_path, disease_dir):          # !
    strings = read_in_strings(smi_path)

    for string in strings:
        substructure_search(string, disease_dir)


main()

# molecules_raw = substructure_results.rsplit("<molecules>")
    [1].rsplit("</molecules>")[0]
# molecules_list = list(molecule for molecule in molecules_raw.
    split("</molecule>"))

# for i in molecules_list:
#    print(i)
# print(molecules_list)
```

4 Ergebnisse

Blabla

5 Ergebnisdiskussion

- Ergebnisse aus Python Script docken lassen

6 Zusammenfassung

Blabla

7 Quellen- und Literaturverzeichnis

Blabla

8 Unterstützungsleistungen

Blabla