FRIEDRICH-SCHILLER-
UNIVERSITÄT
JENA

DARLING: Deep leARning for chemicaL Information processinG

Dissertation

(KUMULATIV)

zur Erlangung des akademischen Grades
doctor rerum naturalium
(Dr. rer. nat.)

vorgelegt dem Rat der Chemisch-Geowissenschaftlichen Fakultät der
Friedrich-Schiller-Universität Jena
von

M. Sc. (Bioinformatik) Kohulan Rajan
geboren am 05.12.1990 in Nuwara-Eliya, Sri Lanka

Dissertation, Friedrich-Schiller-Universität Jena, 2021

Gutachter:

1. Prof. Dr. Christoph Steinbeck, Friedrich-Schiller-Universität, Jena, Deutschland.
2. Prof. Dr. Achim Zielesny, Westfälische Hochschule, Recklinghausen, Deutschland.
3. Prof. Dr. Egon Willighagen, Maastricht University, Maastricht, Niederlande.

Tag der öffentlichen Verteidigung: 27.10.2021

To everyone who ever loved me and believed in me

To my PhD Fathers

“It is our choices that show what we truly are, far more than our abilities.”

- Albus Dumbledore

Table of contents

Table of contents *I*

List of Figures *III*

List of Abbreviations *IV*

Zusammenfassung *VI*

Summary *X*

1. Introduction *1*

1.1 General overview of this project **1**
1.1.1 The aim of this project 1
1.1.2 Graphical abstract 3
1.1.3 The existing approaches for chemical image translation 3
1.1.4 Database generation and IUPAC name generator 4

1.2 Deep learning in chemistry **5**
1.2.1 Machine learning 5
1.2.2 Deep learning 6
1.2.3 Recent developments in deep learning 13
1.2.4 Implementations of Deep Learning in chemistry 14

1.3 Overview of the systems used in this work **16**
1.3.1 Neural Networks applied 16
1.3.2 Data, programming languages, and libraries 17
1.3.3 Hardware overview, GPUs, and TPUs 19

2. Scope of the thesis *22*

A. Development of image segmentation tool **22**

B. Deep learning for OCSR **23**

C. Development of various other methods for rediscovery of chemical information. **23**

3. Publications *24*

3.1 Publication A **24**
A review of optical chemical structure recognition tools 24

3.2 Publication B **38**
DECIMER: towards deep learning for chemical image recognition 38

3.3 Publication C **48**
Molecule Set Comparator (MSC): a CDK-based open rich-client tool for molecule set similarity evaluations 48

3.4 Publication D **54**
DECIMER-Segmentation: Automated extraction of chemical structure depictions from scientific literature 54

3.5 Publication E **64**
COCONUT online: Collection of Open Natural Products database 64

3.6 Publication F **78**
STOUT: SMILES to IUPAC Names Using Neural Machine Translation 78

3.7 Publication G **93**
DECIMER 1.0: Deep Learning for Chemical Image Recognition using Transformers 93

4. Discussion ***126***

4.1 Disadvantages of traditional tools and the need for a deep learning-based method **126**

4.2 Increasing interest in the field **128**

4.3 DECIMER and STOUT: Implementation and Applications **128**
4.3.1 DECIMER - Segmentation 129
4.3.2 DECIMER - Image Transformer 129
4.3.3 COCONUT database and STOUT 132

4.4 Future developments in software, hardware, and DECIMER **133**

5. References ***136***

6. Erklärungen ***XIII***

Selbstständigkeitserklärung **XIII**

7. Curriculum vitae ***XV***

Personal data **XV**

Education **XV**

Publications **XV**

Teaching and Supervision **XVI**

8. Acknowledgements ***XVII***

List of Figures

Figure 1: The applications developed during the progress of this thesis 03
Figure 2: Schema of a simple neural network 07
Figure 3: Schema of a simple convolutional neural network 09
Figure 4: Schema of a single LSTM unit and a GRU unit 11
Figure 5: Schema of a transformer network 12

List of Abbreviations

2D	Two Dimension
ACS	American Chemical Society
AI	Artificial Intelligence
ANN	Artificial Neural Network
ASIC	Application Specific Integrated Circuit
BERT	Bidirectional Encoder Representations from Transformers
CDK	Chemistry Development Kit
CLiDE	Chemical Literature Data Extraction
CNN	Convolutional Neural Network
COCONUT	COlleCtion of Open Natural Products
CPU	Central Processing Unit
CSR	ChemSchematicResolver
CUDA	Compute Unified Device Architecture
CVA	Conditional Variational Autoencoder
DECIMER	DEep lEarning for Chemical IMagE Recognition
FFN	Feed Forward Networks
GPT	Generative Pre-trained Transformer
GPU	Graphical Processing Unit
GCS	Google Cloud Storage
GUI	Graphical User Interface
I/O	Input and Output
IC	Integrated Circuit
InChI	International Chemical Identifier
IUPAC	International Union of Pure and Applied Chemistry
KNN	K-Nearest Neighbors algorithm
MNIST	Modified National Institute of Standards and Technology database
MSC	Molecule Structure Comparator
NLP	Natural Language Processing
NMT	Neural Machine Translation
NP	Natural Products
OCR	Optical Character Recognition
OCSR	Optical Chemical Structure Recognition
OPSIN	Open Parser for Systematic IUPAC Nomenclature

OROCS	Optical Recognition Of Chemical graphicS
OS	Operating System
OSRA	Optical Structure Recognition Application
PDF	Portable Document Format
R-CNN	Region Based Convolutional Neural Networks
RAM	Random Access Memory
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SDG	Structure Diagram Generator
SELFIES	SELF-referencIng Embedded Strings
SMILES	Simplified Molecular-Input Line-Entry System
SOTA	State-Of-The-Art
STOUT	SMILES TO IUPAC Translator
SVM	Support Vector Machine
TFRecord	TensorFlow Record
TPU	Tensor Processing Unit
VGG	Visual Geometry Group
VM	Virtual Machine
VRAM	Video Random Access Memory
XLA	Accelerated Linear Algebra

Zusammenfassung

Große Mengen an wissenschaftlichen Informationen sind in der wissenschaftlichen Primärliteratur verborgen und nicht als kuratierte Daten in wissenschaftlichen Datenbanken verfügbar. Solche Informationen öffentlich zugänglich zu machen, um offene Wissenschaft und Innovation zu unterstützen, ist eine Herausforderung, die es zu lösen gilt. Diese Problematik kann mit Deep-Learning-Ansätzen angegangen werden. Eine große Hürde beispielsweise für die Naturstoffforschung, die kleine Moleküle (Metaboliten) in Organismen untersucht, ist, dass nur ein geringer Anteil der Strukturen von Metaboliten bekannt ist. Typischerweise macht der Anteil an bekannten Strukturen weniger als 30% eines Metaboloms aus. Abgesehen von völlig unbekannten Metaboliten gibt es eine Fülle von Informationen, welche Wissenschaftler bereits über Metaboliten und ihre Strukturen in der wissenschaftlichen Primärliteratur veröffentlicht haben. Diese Informationen sind jedoch in vielen Fällen nie in frei zugänglichen Datenbanken mit Angaben zur Spezies und den Teilen des Organismus, in denen die Naturstoffe gefunden wurden, eingetragen worden. Erschwerend kommt hinzu, dass die chemischen Strukturen der Metaboliten von Wissenschaftlern als 2D-Strukturdiagramme veröffentlicht werden, die im Wesentlichen Bilder von chemischen Graphen in Form von Bitmaps sind. Die ursprüngliche Strukturinformation ist zwar noch immer auf dem Computer des publizierenden Wissenschaftlers vorhanden, aber der maschinenlesbare chemische Graph wird im Publikationsprozess zerstört.

Der Prozess der Umwandlung eines Bitmap-Bildes einer chemischen Struktur in ein computerlesbares Format wird als *Optical Chemical Structure Recognition* (OCSR) bezeichnet. In den letzten drei Jahrzehnten ist eine Vielzahl von Methoden entwickelt worden, um diese Aufgabe zu lösen. Die meisten Applikationen, die bis 2019 veröffentlicht wurden, sind regelbasierte Systeme. Von all diesen Programmen sind nur drei quelloffen und für jeden frei verfügbar. Das Aufkommen von Deep-Learning-Ansätzen hat zu einem erneuten Interesse an diesem Bereich beigetragen. Diese Arbeit adressiert das oben genannte Problem des Visual Computing unter Verwendung von Deep-Learning-Methoden. Zusätzlich ist die Umsetzung dieses Projekts von Teilprojekten begleitet worden, die zur Entwicklung von Anwendungen zur Generierung von Daten für die dabei begegneten Aufgaben und deren Analyse geführt haben.

Die gesamte Arbeit gliedert sich in ein Hauptprojekt und zwei Teilprojekte. Das Zentrum dieser Arbeit ist das Projekt *Deep Learning for Chemical Image Recognition* (DECIMER). Hierbei handelt es sich um einen neuartigen Deep-Learning-basierten Algorithmus, der entwickelt wurde, um die Herausforderung der visuellen Datenverarbeitung zu lösen, Abbildungen

chemischer Strukturen in den Volltextversionen der chemischen Primärliteratur zu erkennen und sie in maschinenlesbare chemische Graphen zu übersetzen.

Der zentrale Teil des Projekts besteht aus zwei Komponenten: 1) DECIMER-Segmentation: die erste Deep-Learning-basierte Anwendung mit offenem Quellcode zur automatischen Erkennung und Segmentierung von Abbildungen chemischer Strukturen aus der wissenschaftlichen Literatur. 2) DECIMER-Image-Transformer: ein Deep-Learning-basiertes Netzwerk, das ein *State-Of-The-Art* (SOTA) konvolutionelles neuronales Netzwerk verwendet, um Bildmerkmale zu extrahieren, und ein Transformer-Netzwerk, um die Bildmerkmale in ein computerlesbares Dateiformat, wie zum Beispiel SMILES, zu decodieren.

Die Anwendung DECIMER-Segmentation kann ein Bitmap-Bild einer gescannten Seite einer Publikation automatisch verarbeiten, Abbildungen chemischer Strukturen selbstständig erkennen und diese als separate Bilder speichern. Es kombiniert für diese Aufgaben ein Mask-RCNN-Modell und einen selbst entwickelten Maskenexpansionsalgorithmus. Durch Training mit nur 9992 annotierten Regionen aus weniger als 1000 Publikationen können die hier entwickelten Methoden Abbildungen chemischer Strukturen aus der gedruckten Literatur mit einer Genauigkeit von mehr als 90% erkennen und segmentieren. DECIMER-Segmentation ist ebenfalls als Webanwendung verfügbar unter der Domain "decimer.ai". DECIMER-Image-Transformer kann die Abbildungen chemischer Strukturen automatisch in eine SMILES-Zeichenkette übersetzen. Das System kombiniert ein EfficientNet-B3-Modell für die Extraktion von Bildmerkmalen und ein Transformer-Modell für die Übersetzung von Bildmerkmalen in SMILES. Millionen von Datenpunkten - gesammelt aus öffentlichen Datenbanken und sorgfältig mit Filterregeln kuratiert - wurden zum Trainieren der DECIMER-Image-Transformer-Modelle verwendet. Die Anwendung zeigte eine Genauigkeit von ca. 89% für die Übersetzung von Abbildungen chemischer Strukturen mit stereochemischen Informationen und Ionen. Für die Bilder ohne stereochemische Informationen und Ionen lag die Genauigkeit bei 96%.

Die DECIMER-Image-Transformer-Modelle wurden vollständig auf der Google-Cloud-Plattform trainiert, unter Verwendung der neu entwickelten *Tensor Processing Units* (TPUs). TPUs sind zur Beschleunigung von Anwendungen *künstlicher Intelligenz* (KI) entwickelt worden, speziell für das Training großer und komplexer Machine-Learning-Modelle, welche ohne sie deutlich mehr Zeit benötigen würden. Ein Modell, das auf einer GPU wochenlang trainiert wurde, kann auf einer TPU innerhalb von Tagen trainiert werden.

Die Analyse der Trainings- und Testdaten ist normalerweise ein langwieriger Prozess. Daher ist es mühsam, die Vorhersagen der Modelle mit den Testdaten zu vergleichen und wertvolle Analysen zu erhalten. Zu Beginn dieser Arbeit war es notwendig, chemische Deskriptoren aus den Datensätzen abzuleiten und diese zu vergleichen. Zusätzlich war die visuelle Bewertung und ein Vergleich der Datensätze notwendig, ein zeitaufwändiger und sich wiederholender Prozess. Um diese Probleme zu lösen, wurde der *Molecule Structure Comparator* (MSC) entwickelt. Dabei handelt es sich um eine in Java geschriebene, auf dem *Chemistry Development Kit* (CDK) basierende Anwendung zur Analyse der Datensätze und Ergebnisse. Darüber hinaus bietet die grafische Benutzeroberfläche (GUI) dem Endanwender viel Flexibilität. Der Benutzer kann die chemischen Strukturen mithilfe der GUI betrachten, um jedes chemische Strukturpaar visuell zu analysieren. MSC arbeitet auf allen wichtigen Betriebssystemen wie Linux, Mac OS und Windows. Die hohe Parallelisierbarkeit von MSC ermöglicht eine einfache Skalierung auf einem gut ausgestatteten Computer oder einem Server zur Analyse großer Datensätze innerhalb von Minuten.

Die größte Schwierigkeit für Forscher, die an Naturstoffen oder naturstoffbasierter Wirkstoffforschung arbeiten, war das Fehlen einer gut gepflegten Open-Access-Datenbank für Naturstoffe. Viele der online verfügbaren Daten über Naturstoffe wurden von Forschern in mehreren verschiedenen Datenbanken zusammengestellt, von denen einige bis heute nicht zugänglich sind. Die *COlleCtion of Open NatUral producTs* (COCONUT) dient als Online-Datenbank, die alle verfügbaren und frei zugänglichen Informationen über Naturstoffe an einem Ort vereint, um die laufende Forschung an Naturstoffen zu erleichtern. Die von DECIMER generierten Daten werden in Zukunft in die COCONUT-Datenbank eingespeist.

Ein Problem, das bei dieser Arbeit aufgetreten ist, war die Erstellung von IUPAC-Namen für chemische Verbindungen, die noch keine hatten. Es standen nur wenige kommerzielle, regelbasierte Ansätze zur Verfügung, die ausschließlich über eine exklusive Lizenz erhältlich waren. Diese Lizenzierung kann zu einer Barriere für Forscher werden, indem sie ihnen den Zugang zu solchen Tools verwehrt. Aufgrund der Komplexität, eine Anwendung zu entwickeln, die in der Lage ist, alle von der *International Union of Pure and Applied Chemistry* (IUPAC) definierten Regeln zur Benennung von Verbindungen anzuwenden, gab es keine Open-Source-Methoden.

Daher wurde das zweite Teilprojekt initiiert, um dieses Problem durch die Entwicklung eines Open-Source-Tools auf der Basis von neuronaler maschineller Übersetzung (NMT) zu lösen. Um IUPAC-Namen aus einer gegebenen chemischen Struktur ohne vordefinierte Regeln zu generieren, wurde im Rahmen dieser Arbeit *SMILES to IUPAC* (STOUT) entwickelt, ein auf Deep Learning basierender Übersetzer. STOUT kann verwendet werden, um IUPAC-Namen für eine chemische Struktur zu generieren, die in Form einer SMILES-Darstellung übergeben wird. Darüber hinaus kann STOUT auch die IUPAC-Namen zurück in SMILES-Strings übersetzen.

Die Arbeit erforscht modernste Deep-Learning-Methoden und wendet sie an, um die aktuellen Probleme mit chemischen Information zu lösen. DECIMER und seine Unterprojekte werden unter Verwendung der neuesten Versionen der benutzen Bibliotheken und Programmiersprachen erstellt. Sie werden der wissenschaftlichen Gemeinschaft offen zur Verfügung gestellt, um weitere offene Forschung zu ermöglichen.

Summary

Vast quantities of scientific information are hidden in primary scientific publications and not available as curated data in scientific databases. Making such information publicly available to support open science and open innovation is a challenge that has to be solved. This problem can be addressed with deep learning approaches. A significant hurdle, for example, for natural product research, that studies small molecules (metabolites) in organisms, is that only a fraction of metabolite structures are known. Typically, it is less than 30% of a metabolome. Besides entirely unknown metabolites, there is a wealth of information that scientists already reported about the metabolites and their structures in the primary scientific literature. In many cases, however, they have never been curated into open-access databases with information regarding the species and organism parts in which the natural products have been found. To make things worse, the chemical structures of metabolites are published by scientists as 2D structure diagrams which are essentially images of chemical graphs in the form of bitmaps. The original structural information still exists on the publishing scientist's personal computer, but the machine-readable chemical graph is destroyed in the publishing process.

The process of converting a bitmap image of a chemical structure into a computer-readable format is known as *Optical Chemical Structure Recognition* (OCSR). Over the last three decades, a variety of tools has been developed to address this task. Most tools that have been published until 2019 are rule-based systems. Out of all these tools, only three were open-source and freely available to everyone. The rise of deep learning approaches has contributed to a renewed interest in this field. This thesis addresses the aforementioned visual computing problem using deep learning methods. Additionally, the implementation of this project was accompanied by sub-projects that resulted in tools for generating data related to tasks faced in the process and analysing it.

The entire thesis is divided into a main project and two sub-projects. The centre of this thesis is the *Deep Learning for Chemical Image Recognition* (DECIMER) project, a novel deep-learning-based algorithm built to solve the visual computing challenge of detecting such images in the full-text versions of the primary chemical literature and translate them into a machine-readable chemical graph. The central part of the project has two components: 1) DECIMER-Segmentation: the first open-source deep-learning-based tool for automatically recognising and segmenting the chemical structure depictions from the scientific literature. 2) DECIMER-Image transformer: a deep learning-based network that uses a State-Of-The-Art

(SOTA) convolutional neural network to extract the image features and a transformer network to decode the image features into a computer-readable file format, such as SMILES. The DECIMER-Segmentation tool can automatically process a bitmap image of a scanned journal page, automatically recognise chemical structure depictions, and save them as separate images. It combines a Mask-RCNN model and an in-house developed mask expansion algorithm for these tasks. Training on only 9992 annotated regions from less than 1000 publications has allowed the methods developed here to recognise and segment chemical structure depictions from the printed literature with more than 90% accuracy. The DECIMER-Segmentation tool is also available as a web application at the domain "decimer.ai". The DECIMER-Image transformer can automatically translate the chemical structure depictions into a SMILES string. It combines an EfficientNet-B3 model for image feature extraction and a transformer model for translation from image features to SMILES. Millions of data points collected from public databases and carefully curated with filtering rules and were used to train the DECIMER-Image transformer models. The tool demonstrated an accuracy of approximately 89% for the translation of chemical structure images with stereochemical information and ions. For the images without stereochemical information and ions, it was 96%. The DECIMER-Image transformer models were trained entirely on the Google cloud platform, utilising the newly developed *Tensor Processing Units* (TPUs). TPUs are made to be *Artificial Intelligence* (AI) accelerators specially developed for training big and complex machine learning models that take significantly more time to train without them. A model which trained on a GPU for weeks was trained on a TPU within days. Analysing the training and testing data is usually a lengthy process. Hence, comparing the predictions of the models with the testing data and getting valuable analyses is cumbersome. At the beginning of this work, it was necessary to infer chemical descriptors from the datasets and compare them. Additionally, the visual assessment and comparison of the datasets were necessary, a time-consuming and repetitive process. The *Molecule Structure Comparator* (MSC) was developed to address these problems. It is a *Chemistry Development Kit* (CDK) based tool written in Java for analysing the datasets and results. Furthermore, the *Graphical User Interface* (GUI) provides much flexibility to the end-user. The user can view the chemical structures using the GUI to visually analyse each chemical structure pair. MSC works on all the major operating systems, such as Linux, Mac OS, and Windows. MSC's high parallelizability allows easy scaling on a well-equipped computer or server to analyze large data sets within minutes.

The most significant difficulty faced by researchers working on natural products or natural product-based drug discovery was the lack of a well-maintained open-access database for natural products. Many of the data available online on natural products has been compiled by researchers in multiple different databases, some of which are inaccessible to date. Serving as an online database that combines all the available open-source information on natural products in one location, the *COlleCtion of Open NatUral producTs* (COCONUT) aims to facilitate the ongoing research on natural products. Data generated by DECIMER will be fed into the COCONUT database in the future. A problem encountered during this work was the creation of IUPAC names for chemical compounds that did not yet have any. There were few commercial, rule-based approaches available that could only be obtained through an exclusive license. This licensing can become a barrier for researchers by making them unable to access such tools. Due to the complexity of building a tool that is able to apply all the rules defined by the *International Union of Pure and Applied Chemistry* (IUPAC) for naming compounds, there were no open-source methods.

Hence, the second sub-project got initiated to resolve this issue by developing an open-source tool based on *Neural Machine Translation* (NMT). To generate IUPAC names from a given chemical structure without any predefined rules, *SMILES to IUPAC* (STOUT), a translator that uses deep learning, was created as part of this thesis. STOUT can be used to generate IUPAC names for a chemical structure passed in the form of a SMILES representation. In addition, STOUT can also translate the IUPAC names back into SMILES strings.

The thesis explores state-of-the-art deep learning methods and applies them to solve the current problems in chemical information. DECIMER and its sub-projects are built using the most recent versions of the utilised libraries and programming languages. They are made openly available to the scientific community to facilitate further open research.

1. Introduction

1.1 General overview of this project

1.1.1 The aim of this project

Throughout the history of chemistry, there have been thousands of scientific articles published on chemical compounds, their structures and properties. Mining this information from the published literature (semi-)automatically [1, 2] and making it publicly available in open-access databases [3] is a current challenge. This thesis aims to develop a set of deep neural networks capable of translating chemical structure depictions published as bitmaps, as can be found in the past 50 years of primary literature, back into the machine-readable form of SMILES [4] strings. These SMILES strings can then be used for chemical database similarity searches, nourishing the existing databases with more information with the translated IUPAC names [5] and aiding the ongoing research. Thus, a wealth of chemical information about metabolites and natural products becomes accessible.

The high relevance of the extraction of structural information about chemical compounds from the printed literature can be illustrated in the current context of the rapidly expanding field of metabolomics [6]. As it is one of the prominent pillars of omics technologies, a renewed interest has emerged in the structures and properties of small molecules in living organisms [7, 8] - primary and secondary metabolites. Often referring to "known knowns, known unknowns and unknown unknowns" - a humorous reference to a quote[1] by Donald Rumsfeld [9] - we are facing the similar situation that only a small percentage of metabolites are well-documented today and available in public databases. The known knowns can routinely be assigned to spectroscopic features visible in metabolomics data because those metabolites are documented in publicly accessible databases [10]. For a more significant percentage, spectroscopic features indicate the presence of compounds (the known unknowns) and potential biomarkers. Still, neither chemical structures nor matching spectroscopic data are available in open-access databases [11, 12]. The third category, the unknown unknowns, is inaccessible to the analytical methods employed due to their physicochemical properties, inappropriate cultivation conditions, or low concentration. They are not the subject of this work

---

[1] https://en.wikipedia.org/wiki/There_are_known_knowns

for now but might become known unknowns and later known knowns due to further research and technological advances.

This project attempts to address the problem of assigning known unknowns by re-discovering the uncurated wealth of structural information about metabolites hidden in the primary literature **(Publications A, B, D, and G).** Publications such as the Journal of Natural Products (ACS publication) or Phytochemistry (Elsevier) are looking back at 40 years of rich information on natural products. While text mining approaches have been widely described [13–16], the problem of translating bitmap images of chemical structure depictions is still not completely solved (**Publication A**).

The core part of this thesis is to solve this problem involving the use of Deep Learning methodologies. This includes the training of multiple deep learning models to detect, segment, and translate chemical structure depictions in the printed literature into computer-readable file formats (**Publications B, D and G**). A further contribution of this project was to explore the possibilities of having an open-source tool for statistically evaluating the methods applied in the process **(Publication C)**. Another sub-project in this thesis explores the non-availability of an open-source based tool for chemical language translation, and it has been addressed using deep learning methodologies (**Publication F**).

The ultimate goal of this thesis is to combine the image mining methods with text mining approaches to generate information regarding the chemical structure, chemical names, species, organism parts, spectra and other properties of published compounds. This information also enhances open-access databases **(Publication E).** The source code of the applications developed for this project is openly accessible to encourage and facilitate other ongoing research in this field.

1.1.2 Graphical abstract

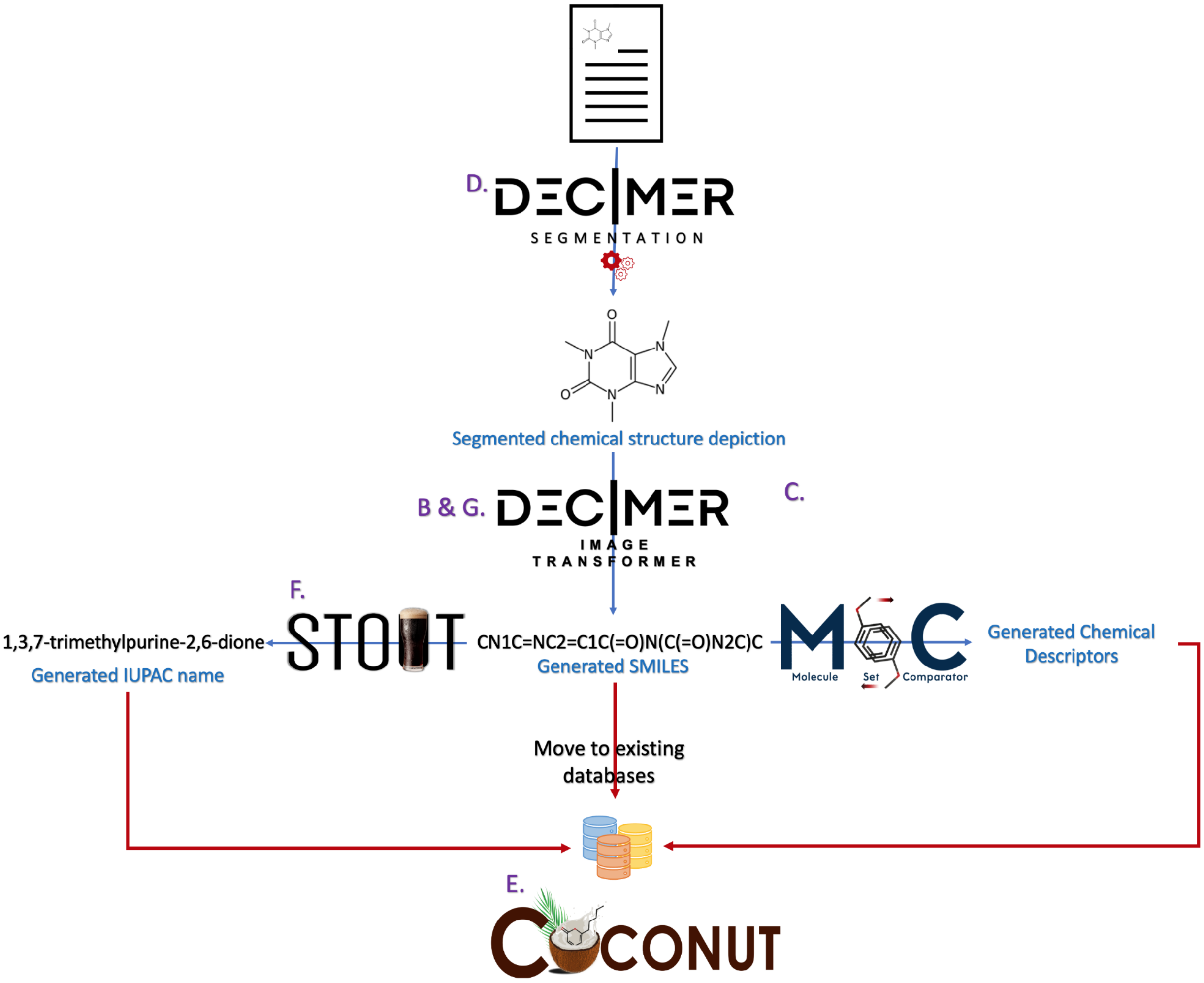

D.
DECIMER
SEGMENTATION
Segmented chemical structure depiction
B & G.
DECIMER
IMAGE TRANSFORMER
C.
F.
STOUT
1,3,7-trimethylpurine-2,6-dione
Generated IUPAC name
CN1C=NC2=C1C(=O)N(C(=O)N2C)C
Generated SMILES
Molecule Set Comparator
Generated Chemical Descriptors
Move to existing databases
E.
COCONUT

Figure 1: The applications developed during the progress of this thesis: The projects are represented by corresponding project logos and are arranged in a systematic way in the sequence within the workflow. Each project has a corresponding publication, shown in purple letters. (**Publications B, C, D, E, F and G**)

1.1.3 The existing approaches for chemical image translation

The process of translating a chemical structure depiction into a machine-readable file is known as *Optical Chemical Structure Recognition* (OCSR). For nearly three decades [17, 18], research in this demanding field has developed, starting with the first complete working system Kekulé [19] in 1992. The most common OCSR approaches consist of rule-based workflows (**Publication A**). The critical step is the vectorization of the chemical structure depiction and the subsequent interpretation of the vectors and nodes as bonds and atoms. The rule-based approach has also been implemented into commercial and open-source systems to extract 2D bitmaps into connection tables or SMILES representations **(Publication A)**.

These commercial software systems include Kekulé [19], *Optical Recognition Of Chemical graphicS* (OROCS) by IBM [20], *Chemical Literature Data Extraction* (CLiDE) [21] and the open-source approaches are, *Optical Structure Recognition Application* (OSRA) [22], Imago [23] and MolVec [24]. More details about all of the approaches and systems are available in Publication A. To solve the problem, all of the packages above employ the following steps:

1. Scanning
2. Vectorization
3. Searching for dashed lines and dashed wedges
4. Character recognition
5. Graph compilation
6. Post-processing
7. Display and editing

A system like this needs to be carefully tuned manually, both individually and in combination with the other steps. The incorporation of new methods for image feature detection is a time-consuming process and the curation of these data is error prone.

Besides the traditional rule-based approaches, few other methods were developed over the years. Staker *et al* [25] published the first complete deep learning-based method in 2019. Even though the approach seemed interesting, the system was not made openly available (see **Publication A**).

Since much attention has been paid to this field lately (**Publication A**) and with the new developments seen in deep learning, more deep learning methods have been published (**Publications A and G**).

1.1.4 Database generation and IUPAC name generator

Data mined using the OCSR methods must be stored in a database so that it can be used later and distributed to the public. Considering this, work on an in-house database was initiated (**Publication E**). Following the curation of data, one of the difficulties was generating IUPAC names for molecules that did not have an existing name found in publicly available databases. Surprisingly, there were no open-source methods available. In search of available and reliable methods, all the found tools were rule-based methods, such as the commercial software ChemAxon MolConvert [26]. It is available to researchers with an academic license for free.

A set of rules and guidelines have been established by the International Union of Pure and Applied Chemistry (IUPAC) to simplify the chemical nomenclature process. More information can be found in Publication F.

Developing an open-source method based upon all these guidelines and rules can be a time-consuming process, therefore the Neural Machine Translation (NMT) tool was developed instead **(Publication F)**.

1.2 Deep learning in chemistry

1.2.1 Machine learning

Machine learning is a field of computer science focused on algorithms that can improve by learning the patterns of data given to it. In 1997, Tom Mitchell published the following definition in his book ‘Machine learning’: “Machine Learning is the study of computer algorithms that improve automatically through experience” [27]. With the use of data, one can train an algorithm to understand a specific problem by constructing a model without explicitly programming it to solve the problem [28]. Later, this trained model can be deployed to predict solutions to the given problem.

This approach has been explored in various fields for decades [29]. One of the earliest works was done by Arthur.L. Samuel in 1959. According to Samuel, teaching the game of checkers to a computer can create a system that can beat a human player by only learning to play for 8-10 hours [28]. Today, machine learning is used for various tasks, such as email filtering, optical character recognition (OCR), and computer vision, where specifically programming algorithms are complex processes [30].

The Machine Learning methods are differentiated based on the learning process. Supervised learning is the process of training by mapping input and output data points [31]. The data used here is labelled and is generally input and output pairs. A model can be built by training the algorithm to map its input data to its paired output data [32, 33]. This trained model can then predict the outcomes for unseen input data.

Unsupervised learning is detecting patterns in the data without any prior information and human interference [34]. The data points used here are unlabelled. Here, the algorithms learn on their own to identify hidden patterns of the data and cluster the data points with similar patterns together [35] [36].

Another learning methodology is semi-supervised learning, where both labelled and unlabelled data are used while training the algorithms [37]. Usually, less labelled data is used compared to unlabelled data. The semi-supervised learning method comes into play when the amount of labelled data is limited [38].

The reinforcement learning process is another machine learning approach. Unlike the other learning processes, reinforcement learning uses a reward-based algorithm. Here, the algorithm finds the actions it should take to minimize error and maximize reward by using trial and error methods [39]. Reinforcement learning does not require labelled data for training. This makes it different from supervised learning [40].

In general, only one method of learning is used in a single system. However, multiple methods of learning are also possible. Machine Learning methods have become the most used methods to solve specific problems in recent years. Currently, deep learning is playing a dominant part in the machine learning field [41].

1.2.2 Deep learning

Deep learning is also known as deep machine learning, hierarchical learning or deep structured learning. Deep learning falls under the broad theme of machine learning, where Artificial Neural Networks (ANNs) are primarily used. Deep learning uses deep NNs (deep = much bigger) than the normal ANNs, and the algorithms to train them needed significant improvements and much bigger data. Artificial neural networks are loosely based on biological neurons of living systems (brains) [42]. By modelling a network that can mimic the processes of a biological neural network [42], they can be trained to solve a particular problem. A Feed-Forward Network is the simplest ANN, where the information flows perpetually forward from the top layer to the bottom layer. It contains connections between artificial neurons and works like synapses between neurons in the brain. These are composed of input neurons connected to a set of neurons in the hidden layers, and they terminate with a group of output neurons (Figure 2).

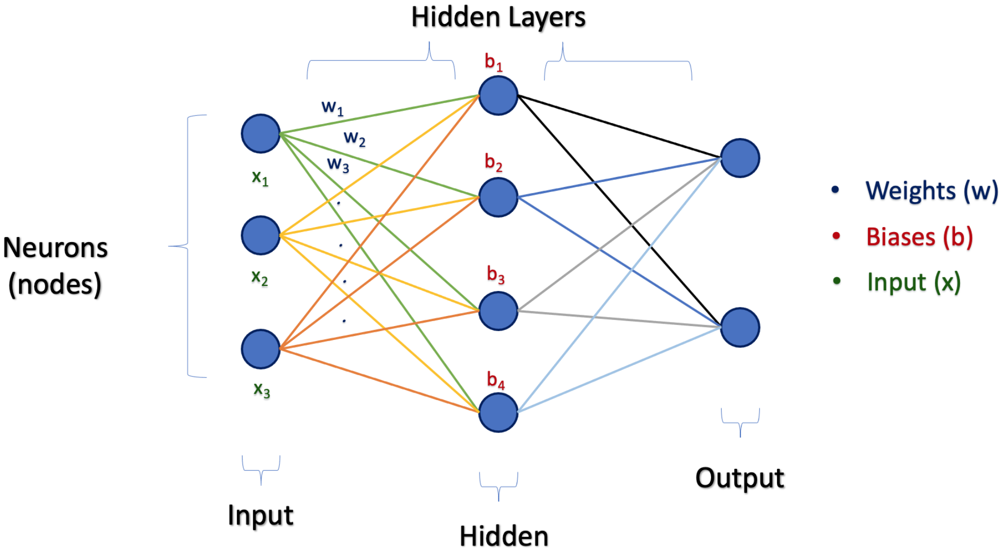


Figure 2: Schema of a simple neural network.

The information which is given to the ANN is fed forward through the layers in the network. Each neuron computes the weighted sum of all values of input vectors (x) that are passed to it. During training, the weights (w) are adjusted by the learning process in order to fit the actual output to the desired outcome. Every neuron in each layer is connected to every neuron in the previous and next layer in a fully connected neural network. One neuron in the adjacent layer processes the information passed to it by the previous layer as a weighted sum, see equation 1.

$$\sum_{i=1}^{n} x_i w_i = (x_1 w_1 + x_2 w_2 + \dots\dots + x_n w_n) \qquad eq. 1$$

Once the weighted sum is calculated, it can be any value. In some cases, it can be beneficial when the values are in a particular range (*e.g.,* 0-1). To map the calculated value to a given range and to introduce non-linear properties into the network, the neuron uses a non-linear function known as an activation function (f) [43]. The magnitude of the weighted sum determines the activation of a neuron. The neuron does not need to be always active whenever it receives a more considerable calculated value; it might need to be inactive below a certain threshold. Also, sometimes the calculated value can be lower, and the neuron has to be activated. Hence, a bias (b) is added to the weighted sum to keep the neuron active or inactive. This is done after the weighted sum and before the activation function is applied [44], see equation 2.

$$x_{(i+1)} = f\left(\sum_{i=1}^{n} x_i w_i + b_i\right) \qquad eq.2$$

Similarly, as explained above, all neurons except the input neurons receive a weighted sum, a bias is added, and an activation function is applied. The final values correspond to whether or not the neuron should be activated. If the last calculated value is 0, the neuron is inactive. If a neuron is activated, its final calculated value is sent through the next connection to be multiplied with another weight. This process continues until an output neuron is reached.

By comparing the true value and the calculated value, the output neuron determines the final result. If the value does not match up, the neuron sends information back to the output network to alter the weights and biases accordingly. This is done by using a backpropagation algorithm [45].

The squared values of the differences between the true values (t) and the predicted values (p) added up together (equation 3) is defined as the cost of a single training example.

$$cost = \sum_{i=1}^{n} (p_i - t_i)^2 \qquad eq.3$$

To get an optimal result, the cost has to be minimized. The function used to calculate this cost is known as the loss function or cost function [46]. The training of the network aims to reduce the loss to get better accuracy. The process of minimizing the loss of the network is an optimization problem, and it is done by using an optimization algorithm [46]. During the application of the optimization algorithm, the weights and biases are adjusted accordingly to reduce the loss. A network is optimized to perform better in multiple training steps over multiple epochs. The amount by which the weights and biases are adjusted is called the learning rate [47]. Throughout the process, every component mentioned above gets adjusted to reduce the loss and improve overall accuracy. This whole process is known as training.

The networks used in deep learning are typically much bigger, i.e., consisting of more layers [48]. The word "deep" here means that there are multiple layers present in the artificial neural network. These layers can process information provided and learn the representation of the data with multiple levels of abstraction [49, 50].

The deep neural networks implemented in this thesis are A) Convolutional Neural Networks (CNN), B) Recurrent Neural Networks (RNN), and C) Transformers.

Also, a combination of these networks is used to create different models. The CNN and RNN are combined to create an encoder-decoder model architecture [51] for captioning images(**Publication B**) and a model of language translation [52] using an RNN-based encoder-decoder architecture(**Publication F**).

A. Convolutional Neural Network (CNN)

CNNs or ConvNets are a collection of layers of neural networks specialising in processing the data in a matrix or a grid-like fashion, such as a 2D Image. Digital images are built using millions of square-like shapes in a grid, which are called pixels. A pixel contains the colour values, exposure details and other metadata as numerical values that make up the overall image [53]. A CNN is designed based on the visual cortex of the human brain [54]. The layers of CNNs are arranged so that they can perceive the visual information, such as colours, edges, objects, and interpret them into a piece of valuable information, see figure 3.

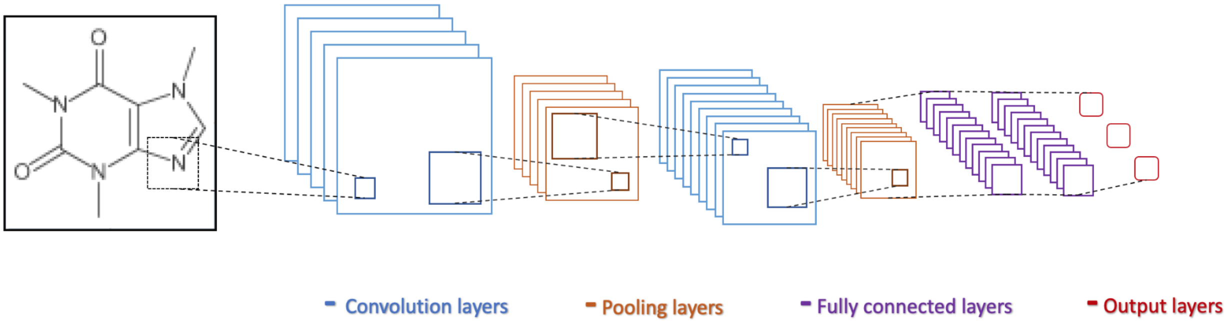


Figure 3: Schema of a simple convolutional neural network.

CNNs are built using stakes of convolutional layers, pooling layers and fully connected layers. Together with all of them, a CNN is formed [55]. Feature extraction is done through convolutional and pooling layers, whereas final outputs are generated with the fully connected layers. During the training, linear operations are passed through a nonlinear activation function. The common nonlinear activation function used in a CNN is the Rectified Linear Unit (ReLU), which computes the result of $f(x) = \max(0, x)$. These networks can automatically learn the spatial hierarchies of the image features, start recognizing patterns and learn low and high levels of details of the images. [56]. Krizhevsky, Sutskever, and Hinton demonstrated the first commendable achievement using CNNs to classify images in 2012 with their work on AlexNet [57]. The benefits of using CNNs in the field of computer vision make them suitable for extracting image features, object detection, and image recognition tasks.

In this thesis, three different types of CNN architectures are examined. First, for image feature extractions, two of the state-of-the-art networks, Inception-V3 [58] and EfficientNet-B3 [59], are used (**Publications B and G**).

For the segmentation work, the unaltered implementation of the Mask R-CNN (Region-Based Convolutional Neural Networks) [60] is used, a general framework created for instance-segmentation. Inherently, CNN is used for image feature extraction and for object detection while creating segmentation masks for the detected objects that can later be used to segment them from the original image (**Publication D**).

B. Recurrent Neural Networks (RNN)

Another category of neural networks are Recurrent Neural Networks (RNNs), which is one of the oldest deep neural network architectures in use [61]. These networks form connections between nodes like a feed-forward network but also have feedback loops between nodes [62]. This makes them suitable to work on temporal problems such as natural language processing, image captioning and speech recognition tasks. RNNs can be trained using sequential or time-series data [63]. The network has a stored state called the memory, which can store information from a prior input to influence the current input and output. This previous information helps the RNNs to determine the outcome in the sequence.

Some common RNN architectures are the Long Short-Term Memory (LSTM) [64] models and the Gated Recurrent Units (GRUs) [65]. An LSTM consists of three gates, one input gate, one output gate, and one forget gate [66], see figure 4. These gates are connected like a feed-forward neural network (FFN) but have feedback connections as well. The cells work based on a concept called the cell state that can retain the input information for a short period (memory) and carry it throughout the processing of a sequence. During training, the gates learn what information to keep or not to keep. LSTMs are used in handwriting recognition [67] and speech recognition [68]. LSTMs are mainly developed to deal with the vanishing gradient problem [69].

The GRUs were introduced in 2014 as a gating mechanism for the RNNs [70]. They are similar to an LSTM but have only two gates: a reset gate and an update gate, see figure 4. This does not consist of the cell state but uses a hidden state. The update gate decides which information to keep and what not to keep. The reset gate decides which past information it should forget.

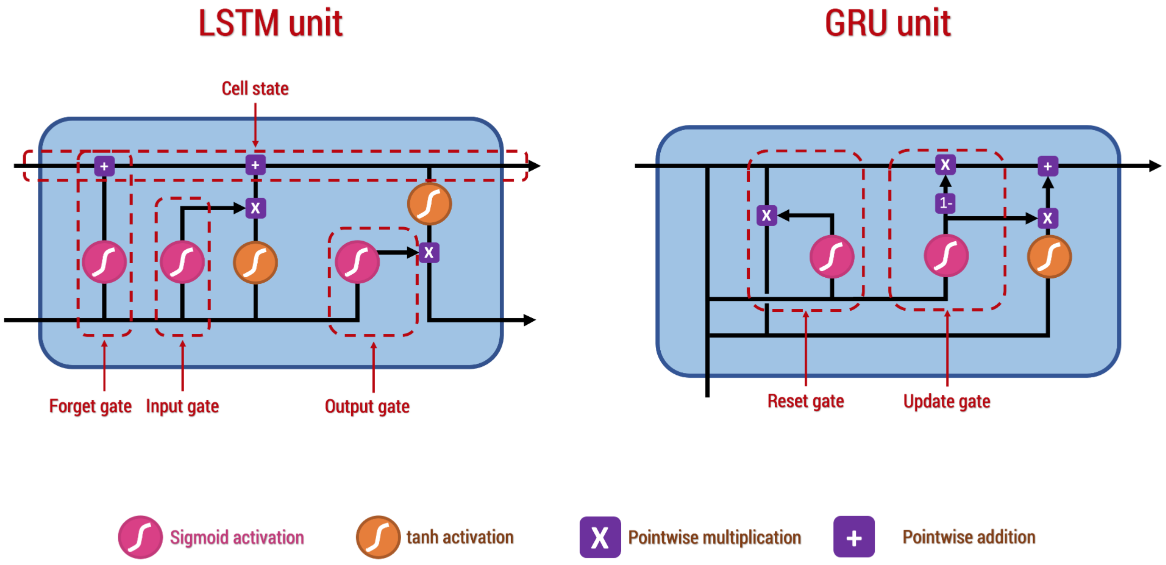


Figure 4: Schema of a single LSTM unit and a GRU unit.

In this work, the GRUs are used for the image captioning encoder-decoder network. The encoder-decoder network in this work uses a CNN-based encoder that uses image features as inputs and transforms them into a vector in latent space. An RNN-based decoder combines encoder output and decoder input to learn which image features should be used in predicting the sequence of SMILES tokens [51, 71]. After the training, the trained model uses only image features as inputs to predict a SMILES string (**Publication B**). The encoder-decoder network also consists of an attention mechanism [52]. The way that attention works is similar to how humans focus on details of an image to construct meaningful sentences to describe it. The attention identifies which image details should be used to predict a specific set of words, rather than using the whole image features as inputs [72]. During the learning process, attention determines the set of image features which corresponds to the right sequence of words **(Publication B)**.

In the field of language translation, most published models were based on the encoder-decoder architectures [73]. These networks had limitations in dealing with long-range dependencies, and they only worked sequentially. The development of the transformer networks [74] was aimed to solve these limitations overall.

C. Transformer Networks

Transformer networks entirely depend on the attention mechanism, making no use of recurrence as in RNNs or convolution as in CNNs at all. These networks are made of a stack of encoder blocks and a stack of decoder blocks. Each encoder block has a multi-head attention layer connected with a feed-forward layer. Each decoder block has a masked multi-head attention layer connected with a multi-head attention layer and then connected with a feed-forward layer [74]. An encoder stack consists of multiple encoder blocks, while a decoder stack consists of multiple decoder blocks. The encoder and decoder stacks each have the same number of units in a transformer, see figure 5.

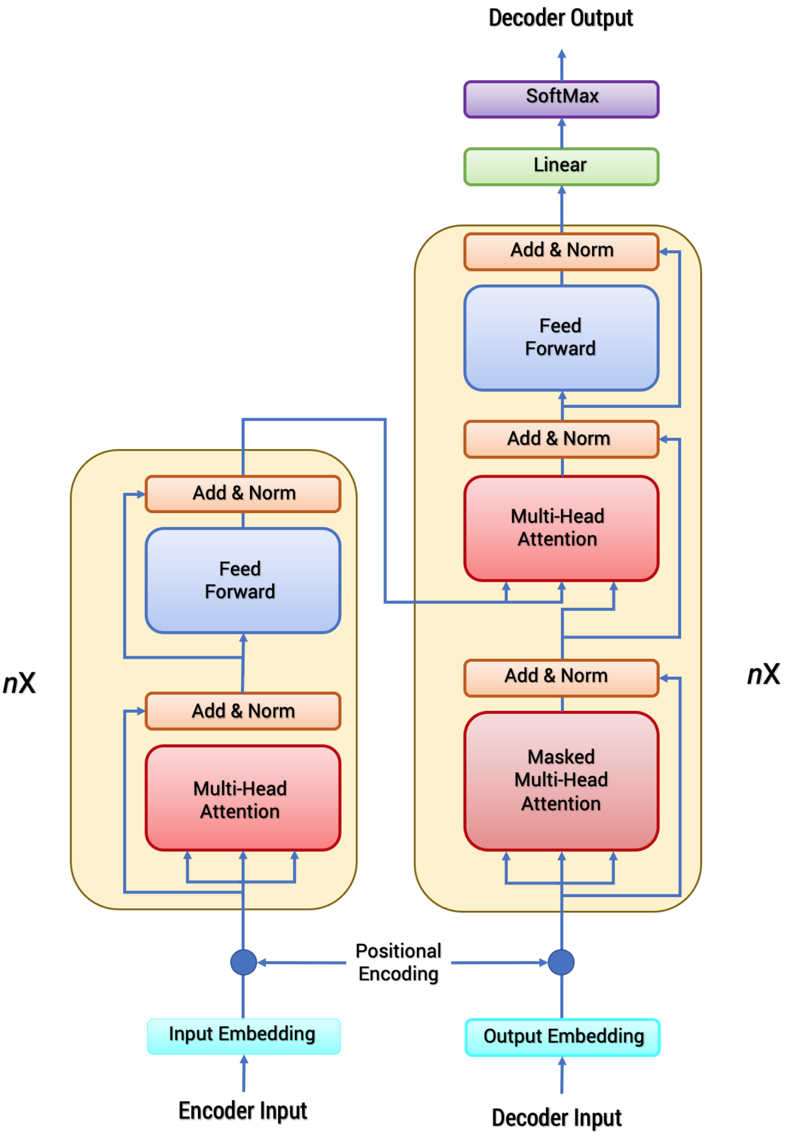


Figure 5: Schema of a transformer network according to the paper “Attention is all you need” [74].

The input sequences are embedded with the positional information before they are fed into the network. The positional information contains details about the order of a sequence (sentence), how the words are arranged, and how far apart the words are placed from each other. This information, along with the input data, is used by multiple attention mechanisms in the network, such as self-attention, encoder-decoder attention, and multi-head attention, to learn to focus on the input sequences to predict the correct output sequences. Transformers are parallelizable, making the network train much faster [74] (**Publication G**). These properties of transformers helped in developing multiple state-of-the-art models like Google's BERT [75] and Open AI's GPT-3 [76].

The deep learning-based models work with high dimensional data allowing them to solve complex problems. The ability to solve intricate issues led to the use of deep learning in many domains, such as business, economics, science, and chemistry. The use of deep learning and artificial neural networks in many areas has been growing exponentially. For the last few years, it has been applied to diverse areas, such as natural language processing [77], computer vision [78], stock market predictions [79], computational photography [80], drug discovery [81], bioinformatics [82], and quantum computing [83].

1.2.3 Recent developments in deep learning

Recent advances in deep learning have made it clear that the technology can be used for numerous applications. The most successful machine learning breakthroughs of the past few years benefited from large to indefinite training corpora. An example is AlphaGo Zero, where a small set of rules for the game of Go and an easy rule for determining the winning party of a game of Go enabled two deep learning-based machines to play a potentially unlimited number of games against each other until the machine achieved superhuman playing strength [84]. The authors recently reported a generalisation of the scheme in AlphaZero (note the missing “Go”), which also achieved a superhuman level of play in chess and shogi (Japanese chess) in only 24 hours [85].

Advancements in hardware and the development of software APIs [86, 87] such as TensorFlow [88, 89] are facilitating this growth.

For decades, the shallow three-layer feedforward perceptron has been the most prominent representative of artificial neural networks. In contrast to elementary perceptrons, the shallow three-layer variant is already computationally universal, i.e., capable in principle of approximating an arbitrarily complex non-linear model function by adjusting the number of neurons of its single hidden layer. More recent deep neural networks [90] with multiple layers of hidden neurons are not more “computationally universal” than the preceding shallow ones. However, they have performed better for difficult learning tasks with concrete data sets [91]. Thus “computational universality” as a fundamental characteristic does not guarantee practical learning success - where the intrinsic relation between shallow and deep networks is still under investigation [92]. The recent deep learning advances [93] in computer vision [78] and natural language processing [94] also profit from several methodological advances that address their long known severe problems like vanishing gradients and exploding gradients [50, 95] during the learning process. While deep networks are still more challenging to train than their shallow relatives because of a considerably extended hyper-parameter optimisation [96, 97], the earlier mentioned prohibitive difficulties have nowadays been effectively solved to allow a successful [98] - and in contrast to shallow networks often superior - model approximation.

The analysis of images, in terms of both classification and the detection of features in images, has benefitted particularly from the advances mentioned above, both on the informatics and the hardware side. The application of convolutional neural networks (CNNs) to analyse the MNIST dataset of handwritten digital images [99] was found to be one of the best solutions to detect and analyse the handwritten digits and visual imagery [100].

In the large-scale visual image recognition competition ImageNet, deep convolutional networks achieved unsurpassed low error rates with very large networks of 650,000 neurons [57]. A continuous stream of methodological advancements followed [58], complemented by GPU hardware developments providing an order of magnitude faster training of CNNs [101]. Also, further developments such as Google's Tensor Processing Units (TPUs) [102] or the recently developed neuromorphic hardware aside from architectural changes are geared towards helping deep learning algorithms achieve substantially lower precision than traditional von Neumann architectures [103].

1.2.4 Implementations of Deep Learning in chemistry

Deep learning in chemistry is mostly either concerned with the prediction of properties of molecules or substances or with the generation of molecular structures with desired properties [104].

Advances in modern hardware equipment allow researchers to effectively compare drug discovery and toxicology prediction methods widely used in cheminformatics, such as K-Nearest Neighbour (KNN), Support Vector Machine (SVM), random forest etc. with deep learning to see how successful it is at the same tasks [105]. In many areas, deep learning outperforms all the known methods for classification and prediction [106]. Since most of these methods typically use different datasets, a study in 2017 [107] utilised a standardised dataset from ChEMBL [108] to compare deep learning with other methods in a standardised way. It could be shown that deep learning networks are the best-performing classifiers.

In 2017, Olivecrona *et al.* [109] used recurrent neural networks and reinforcement learning to design compounds with desired properties in the context of drug discovery based on the ChEMBL database. The authors employed a so-called *prior* network that was trained with

ChEMBL SMILES codes. Together with a scoring function, this *prior* network informs an *agent* network about the likelihood that a generated SMILES [4] representation possesses the desired property.

The encoding of chemical connection tables (graphs) as linear text notations in SMILES is a suitable data structure that can be used to train models using deep learning. Frequently, however, the trained models produce syntactically and semantically invalid SMILES. To overcome this problem and to avoid issues such as unbalanced parenthesis and pairing of ring closure symbols during learning and model building, an adaptation of SMILES was recently introduced in the form of DeepSMILES [110]. Although DeepSMILES performed better than SMILES, the results still contained semantically invalid DeepSMILES (**Publication B**). A new type of string representation was introduced to overcome this problem, known as SELFIES [111]. Chemical structures can be represented and used in deep learning efficiently using SELFIES.

To achieve the *de novo* molecular design of drug-like molecules, in 2018, Lim *et al.* used a conditional variational autoencoder (CVA) and SMILES as an output format to generate molecules with desired properties. An encoder learns a representation of a training set of known molecules and their respective descriptors. Based on that, a decoder then learns to generate SMILES with a desired subset of these properties [112].

Segler, Preuss, and Waller in 2018 addressed the problem of computer-aided retrosynthesis by combining a Monte-Carlo tree search with several deep learning models trained on a comprehensive set of published reactions. The system showed a higher ability to solve retrosynthetic problems than the available traditional approaches, which used a rule-based method. Its results were deemed equivalent to synthesis routes designed by humans in a double-blind test [113].

Traditional methods are dominating the mining of chemical information from scientific text. In 2018, Corbett and Boyle [114] examined deep neural networks, which can be used as an alternative to the conditional random fields-based approaches traditionally used in chemical named entity recognition. Here also, deep neural networks were applied with superior performance [115].

Goh *et al.* in 2017 [116] used chemical graphs depicted as 2D images to develop a CNN which predicts chemical properties. The 2D image was mapped onto a grid of 80x80 pixels to solve inherent memory problems and is then used for training. Fernandez *et al.* in 2018 [117] used modified chemical images to predict the toxicity of chemicals. Their approach, however, was based on colouring the chemical diagrams, thereby adding information that is not usually contained in 2D diagrams found in traditional chemical publications.

1.3 Overview of the systems used in this work

1.3.1 Neural Networks applied

A challenge of this work was choosing a network architecture for DECIMER-Image Transformer (**Publications B and G**). ANNs have seen many developments in the last few years as deep learning has developed. While using image data, unlike the traditional Feed-Forward Networks (FFNs), using CNNs generally leads to a more accurate extraction of important features [118]. The extracted information was found to be very useful for classification in many cases [57]. This project aimed to transform the image features into a sentence, for example into a SMILES string. This goal can be achieved by taking advantage of developments in Image Captioning Networks (ICN) [119] [120] and Natural Language Processing (NLP) [121].

Initially, a network inspired by the publication 'Show, Attend and Tell: Neural Image Caption Generation with Visual Attention' was used in this work [51]. The network encoder-decoder network model included a combination of a CNN and an RNN. The performance and scalability were explored further in publication B. Here, image captioning techniques are employed, in which image features are extracted using a CNN. The extracted features are then transformed into meaningful sentences using a language model. The feature extraction is done using state-of-the-art image classification networks such as Inception-V3 [122] in Publication B and EfficientNet [59] in Publication G.

Inception-V3 is a widely used CNN-based network for image recognition and numerous other image feature extraction tasks [123]. Like Inception-V3, EfficientNet is also a CNN-based network, but the model is optimized for better performance by balancing the network's depth, width, and resolution of the images [59].

Transforming the extracted image features into meaningful sentences is done by the language models. Generally, in language translation, a language model is used to translate one language into another. This work uses a set of image features as a language, and a string of SMILES tokens as a second language. Here, an encoder-decoder model is used for this purpose [51]. In this project, the encoder encodes an image's features into a latent space, and the decoder uses the encoded information and the decoder input to predict a meaningful sentence (SMILES). In NLP, a similar approach is used for language translations [52]. A significant improvement has been achieved in recent years by using transformer-based networks in language translation. This includes Google's BERT [75] and Open AI's GPT models [76, 124, 125].

Based on the transformer-based approach, a novel network model for chemical structure depiction to SMILES has been developed. It is explained in detail in publication G**.** The transformer network used here is the network presented in the 2017 publication ‘Attention is all you need’ [74].

SMILES to IUPAC name translation works similar to language translations. Using the encoder-decoder model for translation [52], a translator was developed to translate SMILES to IUPAC name and the reverse. Here, rather than using two different languages, SMILES and IUPAC names were used (**Publication F**).

Another challenge to overcome was developing a tool to extract chemical structure depictions from the printed literature. It is possible to overcome this problem by using a segmentation tool that can detect the chemical structures within a page of printed literature and then separate them into individual images. This has been addressed using a segmentation method that integrates Mask-RCNN (Region-based Convolutional Neural Networks) [126] as the main component. Publication D provides a detailed description of the implementation.

1.3.2 Data, programming languages, and libraries

The key objective of this project was to collect data from reliable sources and curate them properly before training the models. Doing so would enable the training parameters to be used as efficiently as possible. Generally, the work involved the use of synthetic data in which chemical structure depictions were generated using in-house software. For this work, the entire PubChem database [127] was utilised. Initially, the ChEMBL database [108] was used but, the amount of data found there was deemed insufficient because the trained models could not reach a satisfying level of accuracy.

The downloaded data was curated using several rulesets to build a balanced dataset where the training and test datasets had similar molecules and covered the same chemical space. Also, it was necessary to remove infrequently appearing features of molecules, such as disconnected atoms or isotopes, which could cause problems during training (**Publications B, F and G**). Using the Chemistry Development Kit (CDK) [128] as the backbone, self-written Java code was used to curate the data. Using CDK, the curated data was then converted to isomeric and canonical SMILES strings. More about the data curation and usage is explained in the publications B, F and G.

The generated SMILES strings were used to depict chemical structures using the CDK Structure Diagram Generator (SDG). CDK depiction generation made it possible to create large amounts of data, which prevented manually curated data from being a bottleneck during further development. With the included Java ImageIO libraries, it was possible to render production quality bitmaps when using the CDK. As a result, it was relatively straightforward to make any changes concerning the image quality to the entire training set.

The textual data used in this project mainly consisted of SMILES strings. At the initial stage of the research, predictions in many cases were semantically and syntactically invalid. This reduced the overall accuracy of the predictions. Two different ways of adapting SMILES were recently developed to overcome this difficulty, DeepSMILES [110] and SELFIES [111]. In publication B, the use of DeepSMILES and its benefits were discussed, but later it was found that the machine learning models were capable of handling SELFIES better than DeepSMILES.

Furthermore, all of the SELFIES predicted referred to ~99.99% of valid chemical structures. In our testing results for structures with stereochemistry, a fraction of the SMILES decoded from the SELFIES were invalid, which is 46 out of 3.7 million SMILES. In publications F and G, the implementation and the results using SELFIES are explained in detail. Converting the SMILES into DeepSMILES and SELFIES was done using Python 3 [129].

The deep neural networks (DNNs) were implemented in Python 3 using TensorFlow 2 [88] at the backend. The decision to use TensorFlow 2 was made because it was straightforward to migrate the models onto the Google cloud platform to train. Also, most of the reference models that were used throughout the project were primarily written using TensorFlow.

All of the training data for the models were converted into TFRecords before training, except for the training data used in the publication D. TFRecords are files capable of storing a sequence of binary records [130], which is used in TensorFlow. They are mainly used to increase input efficiency and to train models faster by reducing the I/O bottlenecks. Compared to raw data, TFRecords can be read more efficiently, resulting in faster training models. The TFRecords used in this work were generated on University servers and moved to the Google Cloud Storage (GCS) buckets afterwards. Publications B, F, and G provide further details about the data, each model architecture, and the implementation.

The predictions of the DECIMER Image Transformer were evaluated using the Tanimoto similarity index [131] and isomorphic matches of InChIs [132]. These evaluations were performed using scripts written in Java using the Chemistry Development Kit (CDK). Additionally, various chemical descriptors were calculated for the predicted structures and compared to the original data. For this, custom scripts using Java had to be written. Hence, a standalone software based on CDK was developed to ease this evaluation process, explained in publication C.

For the segmentation work in this project, a corpus of published literature was gathered from the ACS Journal of Natural Products and Elsevier’s Phytochemistry. Annotating these printed pages for identifying chemical structure depictions has been done using the Visual Geometry Group (VGG) Image Annotator [133]. Publication D provides a detailed description of how this data was used to train and evaluate the models. The models were also assessed with publications from the Journal MDPI Molecules.

1.3.3 Hardware overview, GPUs, and TPUs

During the work on DECIMER, a server equipped with an NVIDIA GEFORCE GTX 1080Ti graphics card with 11GB VRAM (Video Random Access Memory) was initially used to train small networks with small datasets in the order of thousands of images to evaluate the methods. To scale up the network, fit more data per batch, obtain the training results faster and test various other network architectures, GPUs with more VRAM and CUDA (Compute Unified Device Architecture) cores were necessary. Thus, the training was moved to a server with a much faster NVIDIA Tesla V100 GPU. To reduce memory bottlenecks during training and facilitate the CPU-related tasks for training the bigger models in the servers, they were equipped with 384 GB of RAM and two Intel(R) Xeon(R) Gold 6230 CPUs. More about the specific GPU-equipped hardware configurations are also described in publications B and D.

Nvidia's V-series GPUs were explicitly designed to drive deep learning and other GPU-related tasks. The GPU's higher number of CUDA cores [134] and the increased memory bandwidth helped the models train faster [135] than with a GTX 1080 Ti [136]. Due to the heat generated by the card's core, GTX 1080Ti was prone to slow down under larger workloads. Additionally, a V100 GPU with 32 GB of VRAM allowed the use of larger batch sizes. With larger models, training time plays a crucial role and thus, it is essential to use the correct hardware to speed up the training process. Training time increases exponentially as more data is used to train larger models **(Publication B)**. To train the large models efficiently, multiple GPUs must be used. When training on multiple GPUs the scaling can be nonlinear. Also, training efficiency deteriorates when incrementing the number of GPUs [137].

This project needed a minimum of 4 GPUs to achieve the maximum speed up of training, which was impossible to do in-house. Since a wide range of GPU solutions was available with Google Cloud Platform for deep learning, the entire training process was moved there instead of purchasing and installing complete hardware in-house.

With the development of Tensor Processing Units (TPUs) [138] by Google [139], scaling a model on the cloud became much easier [140]. A TPU is a specialized piece of hardware developed and custom built by Google for accelerating Artificial Intelligence (AI) applications.

These are built using Integrated Circuits (ICs) designed to do one specific task (an Application-Specific Integrated Circuit (ASIC) [141])[2].

Generally, a typical training process using TensorFlow includes Input and Output (I/O) operations to read and write the data, pre-processing the training data, initializing model variables and parameters, Initializing the code, the model, loss function, and gradient function, summary operations to monitor the training process, save and restore operations for checkpointing the training. When the training is performed using cloud TPUs, the centre part of the code which includes the model, loss functions and the gradient subgraphs gets compiled using XLA[3](Accelerated Linear Algebra [142]) and is executed on the TPUs. The rest of the training code will run on the cloud TPU server (host machine) as part of the regular distributed TensorFlow session, which includes the reading of training data (I/O operations), data pre-processing steps, and all other training management functions, such as saving and restoring checkpoints. Executing only the main training script on TPUs, while distributing it to many nodes, makes it possible to train the models faster.

In this way, TPUs proved to be a practical solution to train larger models within days that require training on GPUs for weeks or months (**Publication F and G**). Also, TPUs were found to be well-suited to train CNN-based models efficiently [143]. The training performance was scaled up as stated in publication F and G without losing any accuracy in the initial test. A single TPU V3-8 device has eight nodes and 128 GB of memory, which can accommodate large batches that are distributed among the eight nodes. Based on training time and performance, the batch size of 1024 images allow the best results. Meanwhile, to fit the same batch size of 1024 images, a minimum of four GPUs (64 GB memory) were required. Given the training time, TPUs are also more cost-effective than GPUs. Although currently, a single TPU device was utilised for training the models, in the future, it can be scaled up by using much larger TPU pods with up to 2048 nodes and 32 TB of memory.

The training data in this work was stored on Google Cloud Storage (GCS) buckets in TFRecord format. By storing them on a storage bucket, training multiple models on multiple Virtual Machines (VMs) at the same time was possible. This enables all the training data to be placed in one single location, which will minimize the time and cost associated with transferring the data to the virtual machines.

Virtual machines and the TPUs were launched in the nearest Google cloud servers, "Europe-west-4a". The network overhead associated with data transfers could be reduced by having them all at the same location. Overall, using the Google cloud platform enabled all the work

---

[2] https://en.wikipedia.org/wiki/Application-specific_integrated_circuit

[3] https://www.tensorflow.org/xla

to move forward at a much faster rate than expected. Throughout the entire work, millions of images were used to train multiple models with Google's TPUs. Without Google's cloud platform, this would be impossible within the required time frame. Training the models with the TPUs is comprehensively addressed in publications F and G.

2. Scope of the thesis

In recent years, advancements in deep learning have enabled breakthroughs in computer vision and natural language processing technologies. Deep learning as an approach for image recognition [144] and language translation [73] leads to near-perfect results. It has been shown that machines trained on millions of data points surpass human performance in certain domains [58, 84], including the image captioning networks [120] and language transformers [75] which are the core of this thesis.

An initial assessment of the available OCSR tools was done to assess how deep-learning approaches compare with the traditional rule-based approach **(Publication A)**. A benchmarking study was done on the existing open-source tools during this process **(Publication A)**. Unfortunately, all offered methods were rule-based.

This thesis aimed to make deep learning-based end-to-end methods for a) Converting the published literature and pages from printed literature into high-resolution images and segmenting out only the chemical structure depictions, b) the development of a deep learning-based method to convert chemical structure depictions into a computer-readable representation, such as SMILES, and c) the development of a large molecule set comparison tool and a deep learning-based method for SMILES to IUPAC name translation to facilitate the data enrichment of the newly developed Natural Products (NP) database.

A. Development of image segmentation tool

In this project, the main objective was to create a tool that would process a bitmap image of a scanned journal page, automatically recognize chemical structure depictions and save them as separate images. The first step was to convert pages in the file into a series of high-resolution images, followed by the detection of chemical structures and the generation of masks that describe their positions using a deep learning model. Then, for chemical structures which are not completely covered by the masks created by the deep learning model, a mask expansion algorithm was applied to expand the masks so that they are covered completely. Finally, the detected chemical structure depictions were segmented and saved as separate image files (**Publication D**). The whole algorithm is made available as open-source software to the public. The tool is also available as a web application where a PDF file can be uploaded to detect and segment the chemical compounds depicted in it.

B. Deep learning for OCSR

This part of the work aimed to develop an OCSR method using the image captioning-based deep learning models, which can recognize chemical structure depictions and convert them into a machine-readable format (**Publications B and G**). This first model was built on the encoder-decoder architecture **(Publication B)** using large synthetic datasets. It has also been improved by increasing the size of the training data **(Publication B)**. The next step involved training the models on Tensor Processing Units (TPUs), an optimized hardware for deep learning. After that, the focus shifted towards creating and training transformer-based models to improve overall performance.

Further, images depicting stereochemistry and ions were also added to the training dataset, which was not part of the initial training datasets. Finally, artificial image augmentations were added to the training dataset to reflect real-world chemical structure depictions. All models were trained with more than 30 Mio images (**Publication G**), and the trained models have been made publicly available (**Publication B and G**).

C. Development of various other methods for rediscovery of chemical information.

This part of the work describes the sub-projects that have arisen during the development of the OCSR tool. Initially, the testing results were mainly evaluated using Java scripts that used the CDK library to calculate chemical descriptors and Tanimoto similarity indices. Calculations such as these are tedious and time-consuming. During the evaluation process, it was discovered that a tool capable of performing these calculations quicker and providing visual analysis of the results would simplify these evaluations substantially. Due to the lack of open-source methods, a software application with above said properties was developed (**Publication C**) to perform the required calculations. The OCSR method (**Publication D and G**) eventually will be used to extract chemical structure depictions from printed material. The retranslated SMILES strings and the chemical structure depictions should be deposited in an open-access database for future use. This was made possible by developing one of the largest open-access databases for natural products (**Publication E**). During data curation work, it was discovered that many compounds did not have IUPAC names, and there were also no open-source methods to generate these names as well. This inspired the development of a deep learning-based toolkit which translates the SMILES strings to IUPAC names, called STOUT (**Publication F**).

3. Publications

3.1 Publication A

A review of optical chemical structure recognition tools

Rajan, K.[1], Brinkhaus, H.O.[2#], Zielesny, A.[3] & Steinbeck, C[4].

J Cheminform 12, 60 (2020)
DOI: 10.1186/s13321-020-00465-0

# The contribution of Brinkhaus, H.O. was part of his master thesis.

Author contributions
(Underlined numbers refer to PhD students)

Author No	**1**	**2**	**3**	**4**
Conceptual research design	x	x	x	x
Planning of research activities	x	x		
Reviewing the tools	x	x		
Data analysis and interpretation	x	x		
Manuscript writing	x	x	x	x
Suggested publication equivalence value	0.5			

Rajan *et al. J Cheminform (2020) 12:60*
https://doi.org/10.1186/s13321-020-00465-0

Journal of Cheminformatics

REVIEW **Open Access**

A review of optical chemical structure recognition tools

Kohulan Rajan[1], Henning Otto Brinkhaus[1], Achim Zielesny[2] and Christoph Steinbeck[1*]

Abstract

Structural information about chemical compounds is typically conveyed as 2D images of molecular structures in scientific documents. Unfortunately, these depictions are not a machine-readable representation of the molecules. With a backlog of decades of chemical literature in printed form not properly represented in open-access databases, there is a high demand for the translation of graphical molecular depictions into machine-readable formats. This translation process is known as Optical Chemical Structure Recognition (OCSR). Today, we are looking back on nearly three decades of development in this demanding research field. Most OCSR methods follow a rule-based approach where the key step of vectorization of the depiction is followed by the interpretation of vectors and nodes as bonds and atoms. Opposed to that, some of the latest approaches are based on deep neural networks (DNN). This review provides an overview of all methods and tools that have been published in the field of OCSR. Additionally, a small benchmark study was performed with the available open-source OCSR tools in order to examine their performance.

Keywords: Optical chemical structure recognition, Named entity recognition, Data mining, Chemical data extraction, Chemical structure, Open data, Machine learning

Introduction

A vast amount of knowledge is still hidden in the primary scientific literature, which is not accessible because the information is not properly curated and stored in open-access databases. Rediscovering this information and making it public is a complicated task that many data scientists took up as a challenge. This problem cannot be solved easily as the data available on chemistry and chemical structures are exponentially increasing by the day. While there is hope that the knowledge in future publications is readily deposited in a semantically well-annotated form in public archives, there is still a backlog of decades of chemical literature to be curated and stored in open-access databases.

In the synthetic sciences, natural products research, drug discovery and many other fields, there is a renewed interest to find more information about the known chemical structures and about the small molecules which are already published. Chemical knowledge that is collected in a time-consuming and expensive process often does not end up as structured information in databases. This way it remains inaccessible for re-use and databases may remain incomplete.

The literature can contain chemical information in various forms. Even though the textual information is presented in an unstructured manner, it does contain valuable information on the chemical compounds. In order to decode this information and create structured data, researchers developed chemical Named Entity Recognition (NER) systems [1]. On the other hand, decoding chemical structure information is a very different task with the goal of turning a graphical depiction into a machine-readable format. This research field is known as Optical Chemical Structure Recognition (OCSR).

The first work on OCSR was done in 1990 [2, 3] but the first complete system was published by McDaniel and Balmuth in 1992: Their program named Kekulé [4]

---

*Correspondence: christoph.steinbeck@uni-jena.de
[1] Institute for Inorganic and Analytical Chemistry, Friedrich-Schiller-University Jena, Lessingstr. 8, 07743 Jena, Germany
Full list of author information is available at the end of the article

© The Author(s) 2020. This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

[illegible]

[illegible]

[illegible] 1 [illegible]

[illegible]	[illegible]	[illegible]	[illegible] [illegible]	[illegible]	[illegible]
[illegible]	C	[illegible] sv	[illegible]	[illegible]	[illegible]
[illegible]	C	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	C	[illegible] sv	[illegible]	[illegible]	[illegible]
[illegible]	C	[illegible]	[illegible] s[a]	[illegible]	[illegible]
[illegible] C		[illegible] sv	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	C	[illegible]	[illegible] s	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible] [illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible] [illegible]		[illegible]	[illegible]	[illegible]	[illegible]

[a] [illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

graph. All these networks are based on CNNs. The segmentation networks follow the idea of dilated convolution described by Yu and Koltun [50].

The segmentation network was trained on chemical structure images generated with RDKit [51] based on data retrieved from CHEMBL [52]. The output of the segmentation network is used as an input for the classification networks to locate atoms, bonds and charges.

Every network was checked individually for accuracy and it is stated that the classification networks perform much better than the segmentation networks. The overall accuracy of the model is determined by the resulting chemical graphs. According to the authors, the system outperforms OSRA.

Comparison of the open-source OCSR tools

Materials and methods

In order to compare the results of the three available open-source OCSR tools Imago (version 2.0), MolVec (version 0.9.7) and OSRA (version 2.1.0), multiple datasets which are freely available online were analyzed according to the validation procedure of the OSRA developers [53]. The datasets were:

1. A set of 5719 images of chemical structures and the corresponding molfiles (based on data from the USPTO) obtained from the OSRA online presence [53].
2. The dataset (UOB) of 5740 images and molfiles of chemical structures developed by the University of Birmingham, United Kingdom, and published alongside MolRec [54].
3. The Conference and Labs of the Evaluation Forum (CLEF) test set of 961 images and molfiles published in 2012 [55].
4. A subset (450 images and SDfiles) of a dataset published with ChemInfty (see above) based on data from the Japanese Patent Office (JPO), obtained from the OSRA online presence [53]. (Note that this dataset contains many labels (sometimes with Japanese characters) and irregular features, such as variations in the line thickness. Additionally, some images have a poor quality and contain a lot of noise.)

The TIFF images were converted to PNG images with a resolution of 72 dpi to assure comparability, as MolVec and Imago both showed problems handling those TIFF files in batch mode.

The command-line version of Imago [56] was executed without installation by running the following command in the directory with the executable file:

```
./imago_console -dir /image/directory/path
```

MolVec was downloaded as a jar file with all of its dependencies. It was executed from the command-line by running:

```
java -cp molvec-0.9.7.jar:common-image-3.4.1.jar:common-io-
3.4.1.jar:common-lang-3.4.1.jar:commons-cli-1.4.jar:imageio-core-
3.4.1.jar:imageio-metadata-3.4.1.jar:imageio-tiff-3.4.1.jar:ncats-
common-0.3.3.jar:ncats-common-cli-0.9.1.jar gov.nih.ncats.molvec.Main -
dir /image/directory/path -outDir /output/directory/path
```

OSRA was installed in an Anaconda environment using the Conda recipe for PyOSRA which was published by Ed Beard [57]. This was done analogously to the installation instructions in the ChemSchematicResover documentation [58]. We used the PyOSRA environment because compiling OSRA from source code is excessively complex as it has a lot of dependencies that need to be compiled from their own source code as well. There is the option to obtain a commercial license to get a precompiled version of the software.

OSRA was then executed on the test data set by running the following shell command in the directory with the images. Here, it is necessary to specify the location of the dictionaries for superatoms and common spelling corrections.

```
for image in *.png; do osra -f sdf -a
/path/to/superatom/dictionary/superatom.txt -l
/path/to/spelling/corrections/dictionary/spelling.txt -w
/output/path/$image.sdf $image;done
```

The accuracies of the tools listed in Table 2 below correspond to perfectly recognized structures according to a perfect match of the Standard InChI strings [59] that were created based on the OCSR results and the reference files.

All the processing was done on a Linux workstation running with Ubuntu 20.04 LTS, which has 2 Intel(R) Xeon(R) Silver 4114 CPUs capable of handling 40 threads and with 64 GB of RAM.

Results

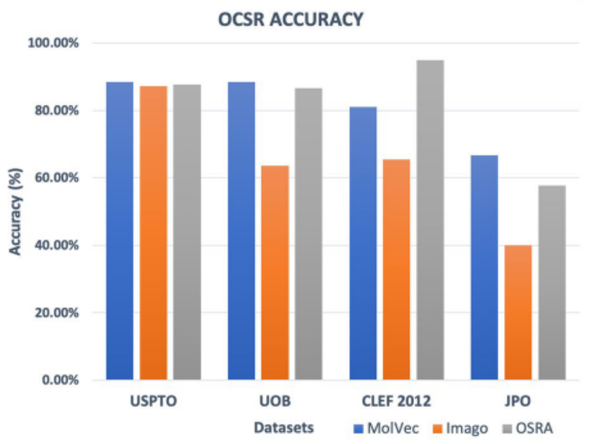
As shown in Table 2, MolVec processes the images significantly faster than its competitors. All three tools performed fairly well on the given set of images. As illustrated in Fig. 1, the proportion of accurate results produced by MolVec and OSRA with the UOB, CLEF and JPO datasets was approximately 20% higher than in the results produced by Imago. The lower overall performance of all three tools with the JPO dataset is likely due to the lower quality of the depictions, the presence of labels and other irregular features. The extraordinarily good performance of OSRA on the CLEF dataset is a notable observation. The examination of the images in the dataset reveals a set of well-segmented, clean chemical structure depictions which is seemingly handled especially well by OSRA.

Table 2 Time elapsed and accuracy reported for the open-source OCSR tools

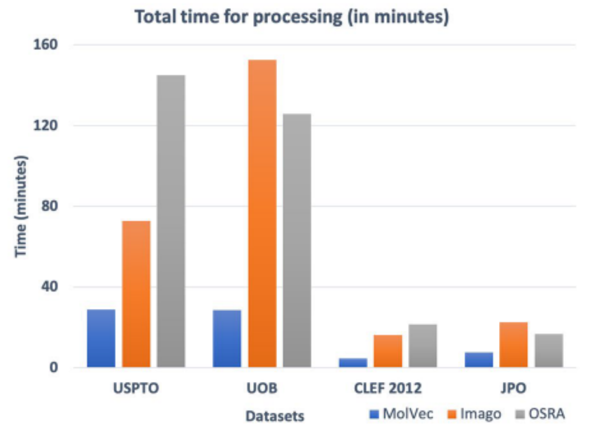
Dataset		MolVec 0.9.7	Imago 2.0	OSRA 2.1
USPTO (5719 images)	Time (min)	28.65	72.83	145.04
	Accuracy	88.41%	87.20%	87.69%
UOB (5740 images)	Time (min)	28.42	152.52	125.78
	Accuracy	88.39%	63.54%	86.50%
CLEF 2012 (961 images)	Time (min)	4.41	16.03	21.33
	Accuracy	80.96%	65.45%	94.90%
JPO (450 images)	Time (min)	7.50	22.55	16.68
	Accuracy	66.67%	40.00%	57.78%

Discussion

In the years 1992–1993, the first developments in the field of OCSR were reported. These early developments were commercial, but there have been open solutions since 2009. In this review, we analyzed all freely available tools used in OCSR. Additionally, we summarized the methods which have been described but never became available as testable software. There are different OCSR



a Accuracy of the OCSR tools



b Total time for processing the datasets by the OCSR tools

Fig. 1 **a** Accuracy (Right: higher the better) and **b** Total time for processing (Left: lower the better)

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

Published online: 07 October 2020

41. Staker J, Marshall K, Abel R, McQuaw CM (2019) Molecular structure extraction from documents using deep learning. J Chem Inf Model 59:1017–1029
42. Ronneberger O, Fischer P, Brox T (2015) U-Net: convolutional networks for biomedical image segmentation. In: Medical image computing and computer-assisted intervention—MICCAI 2015. Springer International Publishing, pp 234–241
43. Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ (eds) Advances in neural information processing systems 25. Curran Associates, Inc., pp 1097–1105
44. Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ (eds) Advances in neural information processing systems 27. Curran Associates Inc, New York, pp 3104–3112
45. Indigo Toolkit. https://lifescience.opensource.epam.com/indigo/. Accessed 25 June 2020
46. Kim S, Chen J, Cheng T et al (2019) PubChem 2019 update: improved access to chemical data. Nucleic Acids Res 47:D1102–D1109
47. Lowe D Chemical reactions from US patents (1976–Sep 2016) (2017). https://figshare.com/articles/Chemical_reactions_from_US_patents_1976-Sep2016_/5104873
48. Lowe DM (2012) Extraction of chemical structures and reactions from the literature. https://doi.org/https://doi.org/10.17863/CAM.16293
49. Oldenhof M, Arany A, Moreau Y, Simm J (2020) ChemGrapher: optical graph recognition of chemical compounds by deep learning. arXiv [stat.ML]
50. Yu F, Koltun V (2015) Multi-scale context aggregation by dilated convolutions. arXiv [cs.CV]
51. Website. RDKit: open-source cheminformatics. https://www.rdkit.org. Accessed 15 Sept 2020
52. Gaulton A, Hersey A, Nowotka M et al (2017) The ChEMBL database in 2017. Nucleic Acids Res 45:D945–D954
53. OSRA validation datasets. https://sourceforge.net/p/osra/wiki/Validation/. Accessed 24 June 2020
54. MolrecUOB Benchmark dataset. https://www.cs.bham.ac.uk/research/groupings/reasoning/sdag/chemical.php. Accessed 29 June 2020
55. CLEF-IP 2012 Structure Recognition Test Set. https://www.ifs.tuwien.ac.at/~clef-ip/download/2012/qrels/clef-ip-2012-chem-recognition-qrels.tgz. Accessed 29 June 2020
56. Imago Download. https://lifescience.opensource.epam.com/download/imago.html. Accessed 24 June 2020
57. Beard E PyosraConda Recipe. https://github.com/edbeard/conda_recipes/tree/master/pyosra. Accessed 24 June 2020
58. ChemSchematicResolver Documentation. https://www.chemschematicresolver.org/docs/install. Accessed 24 June 2020
59. Heller S, McNaught A, Stein S, Tchekhovskoi D, Pletnev I (2013) InChI - the worldwide chemical structure identifier standard. J Cheminform 5:7

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

BMC

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]			
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

Rajan *et al. J Cheminform (2020) 12:65*
https://doi.org/10.1186/s13321-020-00469-w

Journal of Cheminformatics

PRELIMINARY COMMUNICATION

Open Access

DECIMER: towards deep learning for chemical image recognition

Kohulan Rajan[1], Achim Zielesny[2] and Christoph Steinbeck[1*]

Abstract

The automatic recognition of chemical structure diagrams from the literature is an indispensable component of workflows to re-discover information about chemicals and to make it available in open-access databases. Here we report preliminary findings in our development of Deep lEarning for Chemical ImagE Recognition (DECIMER), a deep learning method based on existing show-and-tell deep neural networks, which makes very few assumptions about the structure of the underlying problem. It translates a bitmap image of a molecule, as found in publications, into a SMILES. The training state reported here does not yet rival the performance of existing traditional approaches, but we present evidence that our method will reach a comparable detection power with sufficient training time. Training success of DECIMER depends on the input data representation: DeepSMILES are superior over SMILES and we have a preliminary indication that the recently reported SELFIES outperform DeepSMILES. An extrapolation of our results towards larger training data sizes suggests that we might be able to achieve near-accurate prediction with 50 to 100 million training structures. This work is entirely based on open-source software and open data and is available to the general public for any purpose.

Keywords: Optical chemical entity recognition, Chemical structure, Deep learning, Deep neural networks, Autoencoder/decoder

Main text

The automatic recognition of chemical structure diagrams from the chemical literature (herein termed Optical Chemical Entity Recognition, OCER) is an indispensable component of workflows to re-discover information about chemicals and to make it available in open-access databases. While the chemical structure is often at the heart of the findings reported in chemical articles, further information about the structure is present either in textual form or in other types of diagrams such as titration curves, spectra, etc. (Fig. 1).

Previous software systems for OCER have been described and were both incorporated into commercial and open-source systems. These software systems include Kekulé [1, 2], the Contreras system [3], the IBM system [4], CLIDE [5] as well as the open-source approaches chemOCR [6–8], ChemReader [9], OSRA [10] and ChemRobot described in a patent [11].

All of these software packages share a general approach to the problem, comprising the steps (a) scanning, (b) vectorization, (c) searching for dashed lines and dashed wedges, (d) character recognition, (e) graph compilation, (f) post-processing, (g) display and editing.

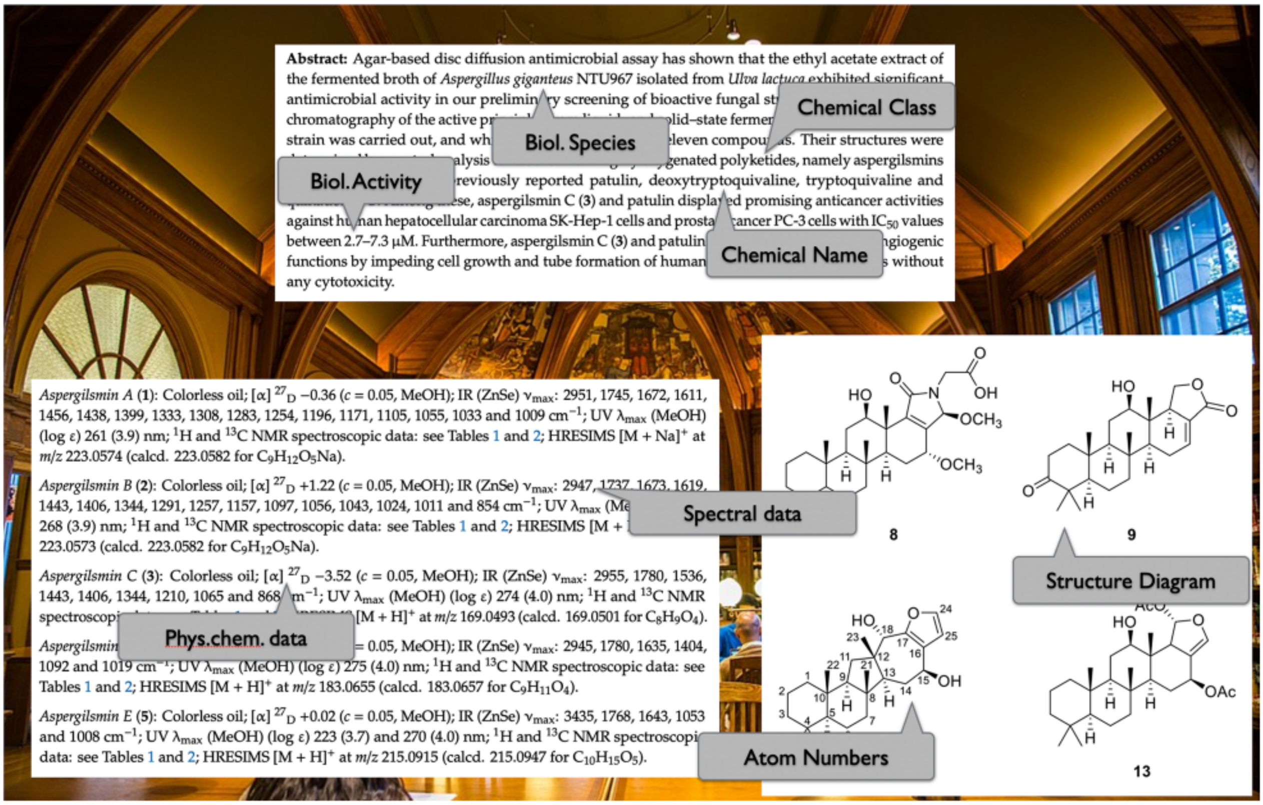
Each of the steps in such systems needs to be carefully hand-tuned both individually as well as for its interplay with the other steps. The incorporation of new image features to be detected is a laborious process.

We were recently inspired by the stunning success of AlphaGo Zero [13], a deep neural network (NN) based approach that enabled AlphaGo Zero to reach superhuman strength in the Game of Go by playing a potentially unlimited number of games against itself, starting with

---

*Correspondence: christoph.steinbeck@uni-jena.de
[1] Institute for Inorganic and Analytical Chemistry, Friedrich-Schiller-University Jena, Lessingstr. 8, 07743 Jena, Germany
Full list of author information is available at the end of the article

© The Author(s) 2020. This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.



Abstract: Agar-based disc diffusion antimicrobial assay has shown that the ethyl acetate extract of the fermented broth of *Aspergillus giganteus* NTU967 isolated from *Ulva lactuca* exhibited significant antimicrobial activity in our prelimin ry screening of bioactive fungal st chromatography of the active pr lid–state fermen strain was carried out, and wh leven compou s. Their structures were alysis genated polyketides, namely aspergilsmins reviously reported patulin, deoxytryptoquivaline, tryptoquivaline and se, aspergilsmin C (3) and patulin displa ed promising anticancer activities against hu nan hepatocellular carcinoma SK-Hep-1 cells and prosta ancer PC-3 cells with IC_{50} values between 2.7–7.3 µM. Furthermore, aspergilsmin C (3) and patulin ngiogenic functions by impeding cell growth and tube formation of human s without any cytotoxicity.

Aspergilsmin A (**1**): Colorless oil; $[\alpha]^{27}_{D}$ −0.36 (*c* = 0.05, MeOH); IR (ZnSe) ν_{max}: 2951, 1745, 1672, 1611, 1456, 1438, 1399, 1333, 1308, 1283, 1254, 1196, 1171, 1105, 1055, 1033 and 1009 cm^{-1}; UV λ_{max} (MeOH) (log ε) 261 (3.9) nm; ^{1}H and ^{13}C NMR spectroscopic data: see Tables 1 and 2; HRESIMS $[M + Na]^{+}$ at *m/z* 223.0574 (calcd. 223.0582 for $C_9H_{12}O_5Na$).

Aspergilsmin B (**2**): Colorless oil; $[\alpha]^{27}_{D}$ +1.22 (*c* = 0.05, MeOH); IR (ZnSe) ν_{max}: 2947, 1737, 1673, 1619, 1443, 1406, 1344, 1291, 1257, 1157, 1097, 1056, 1043, 1024, 1011 and 854 cm^{-1}; UV λ_{max} (Me 268 (3.9) nm; ^{1}H and ^{13}C NMR spectroscopic data: see Tables 1 and 2; HRESIMS [M + 223.0573 (calcd. 223.0582 for $C_9H_{12}O_5Na$).

Aspergilsmin C (**3**): Colorless oil; $[\alpha]^{27}_{D}$ −3.52 (*c* = 0.05, MeOH); IR (ZnSe) ν_{max}: 2955, 1780, 1536, 1443, 1406, 1344, 1210, 1065 and 868 m^{-1}; UV λ_{max} (MeOH) (log ε) 274 (4.0) nm; ^{1}H and ^{13}C NMR spectroscopi $[M + H]^{+}$ at *m/z* 169.0493 (calcd. 169.0501 for $C_8H_9O_4$).

Aspergilsmin = 0.05, MeOH); IR (ZnSe) ν_{max}: 2945, 1780, 1635, 1404, 1092 and 1019 cm^{-1}; UV λ_{max} (MeOH) (log ε) 275 (4.0) nm; ^{1}H and ^{13}C NMR spectroscopic data: see Tables 1 and 2; HRESIMS $[M + H]^{+}$ at *m/z* 183.0655 (calcd. 183.0657 for $C_9H_{11}O_4$).

Aspergilsmin E (**5**): Colorless oil; $[\alpha]^{27}_{D}$ +0.02 (*c* = 0.05, MeOH); IR (ZnSe) ν_{max}: 3435, 1768, 1643, 1053 and 1008 cm^{-1}; UV λ_{max} (MeOH) (log ε) 223 (3.7) and 270 (4.0) nm; ^{1}H and ^{13}C NMR spectroscopi data: see Tables 1 and 2; HRESIMS $[M + H]^{+}$ at *m/z* 215.0915 (calcd. 215.0947 for $C_{10}H_{15}O_5$).

Fig. 1 Information about a natural product is scattered across the various sections of an individual scientific article. Grouped around a structure and a chemical name, further information such as chemical classes, species, and organism parts from which the compound was isolated, spectral and other data are listed. Background image © Alina Chan, distributed under https://creativecommons.org/licenses/by-sa/4.0/deed.en, figures and text from Kwon et al. [12]

no more knowledge than the basic rules of the game. In this example, as well as in other prominent examples of successful deep learning, the key to success was the availability of a potentially unlimited or very large amount of training data.

The example of AlphaGo Zero made us realize that we are in a similar situation for the visual computing challenge described above. Instead of working with a necessarily small corpus of human-annotated examples from the printed literature, as has been common in the text mining and machine learning applications in chemistry in the past, we realised that we could generate training data from a practically unlimited source of structures generated by structure generators or by using the largest collections of open chemical data available to mankind.

After we started our work presented here, other attempts to use deep learning for OCER were reported. Work by the Schrödinger group [14] reports the successful extraction of machine-readable chemical structures from bitmaps but no software system available for the general public to replicate the reported results. A method called Chemgrapher [15] suggests to deal with the problem in a modular fashion by using a segmentation algorithm to segment the images containing chemical graphs to detect atoms locations, bonds and charges separately, and employ a graph building algorithm to re-generate the chemical graph.

Here we report preliminary findings of our development of Deep lEarning for Chemical ImagE Recognition (DECIMER), a deep learning method based on existing show-and-tell deep neural networks, which translates a pure bitmap image of a molecule, as found in publications, into a SMILES (Fig. 2). Unlike for example Chemgrapher, it makes no prior assumptions, such as the existence of bonds or element symbols in the graphic, about the structure of the underlying problem.

The training state reported here does not yet rival the performance of existing traditional approaches, but we present evidence that, given sufficient training data, our method will reach a comparable detection power without

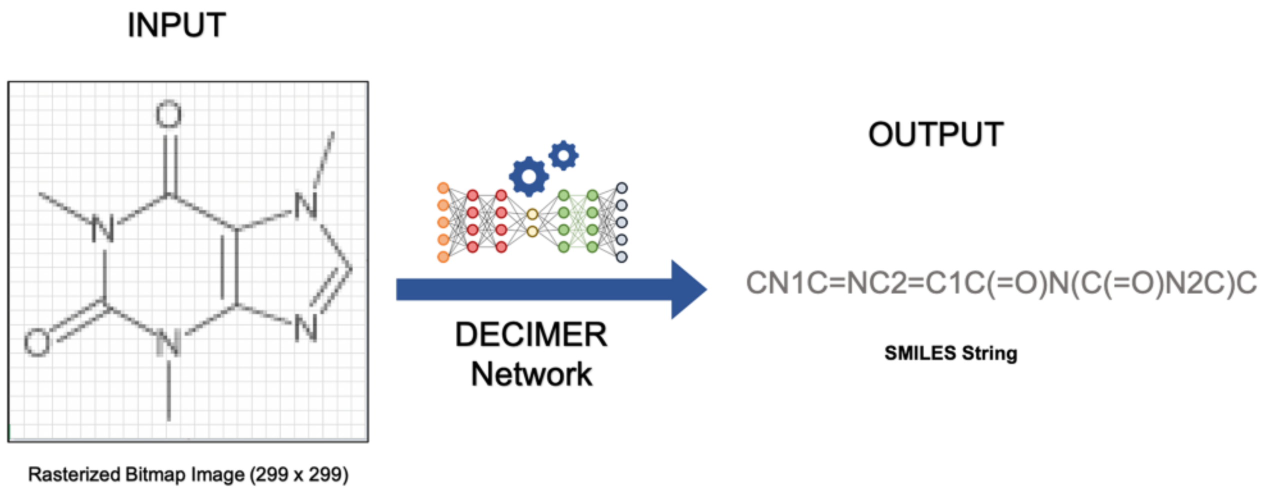

INPUT
OUTPUT
DECIMER Network
CN1C=NC2=C1C(=O)N(C(=O)N2C)C
SMILES String
Rasterized Bitmap Image (299 x 299)

Fig. 2 Chemical image to SMILES translation using DECIMER

the need of the sophisticated engineering steps of an OCER workflow.

The principal idea reported here is to repurpose a show-and-tell deep NN designed for general photo annotation earlier and train it to report a series of SMILES tokens when presented with the bitmap of a chemical structure image. The original NN reported sentences like "A giraffe standing in a forest with trees in the background" when presented with a corresponding photo.

Instead of abstracting chemical diagrams from the chemical literature to generate training data, we decided to use structure diagram generators (SDG) like the one found in the Chemistry Development Kit (CDK) [16] to generate a potentially unlimited amount of training data. This type of training data can be accommodated to become more realistic and comparable to the varying picture quality in the chemical literature by using image manipulation such as blurring, adding noise, etc. As a source of input structures for the CDK SDG, we turned to PubChem [17], one of the largest databases of organic molecules. The following rules were used to curate the Pubchem data for our work presented here (in future versions of this deep NN, these rules might be relaxed):

- Molecules must,
- Have a molecular weight of fewer than 1500 Daltons,
- Not possess counterions,
- Only contain the elements C, H, O, N, P, S, F, Cl, Br, I, Se and B,
- Not contain isotopes of Hydrogens (D, T),
- Have 5–40 bonds,
- Not contain any charged groups,
- Only contain implicit hydrogens, except in functional groups,
- Have less than 40 SMILES characters.

The generation of molecular bitmap images from chemical graphs was performed using the CDK SDG, which generates production quality 2D depictions to feed the deep learning algorithm. One random rotation for each molecule was used. No further modifications, such as the addition of noise, were applied. These types of modifications will be explored once a mature model has been reached.

The text data used here were SMILES [18] strings, which were encoded into different formats, regular SMILES, DeepSMILES [19] and SELFIES [20], to test the dependency of the learning success on the data representation. These datasets were used in different training models in order to evaluate their performance for our use case. With two dataset sizes, we confirmed the superiority of the DeepSMILES over the SMILES representation and continued to use DeepSMILES exclusively.

For our model (Fig. 3), we employed an autoencoder-based network with TensorFlow 2.0 [21] at the back-end. This kind of network refers to the model designed by Xu et al. [22], in their work on Show, Attend and Tell, where they demonstrate a higher accuracy for an Image caption generation system with the attention mechanism. The TensorFlow team used these results for their

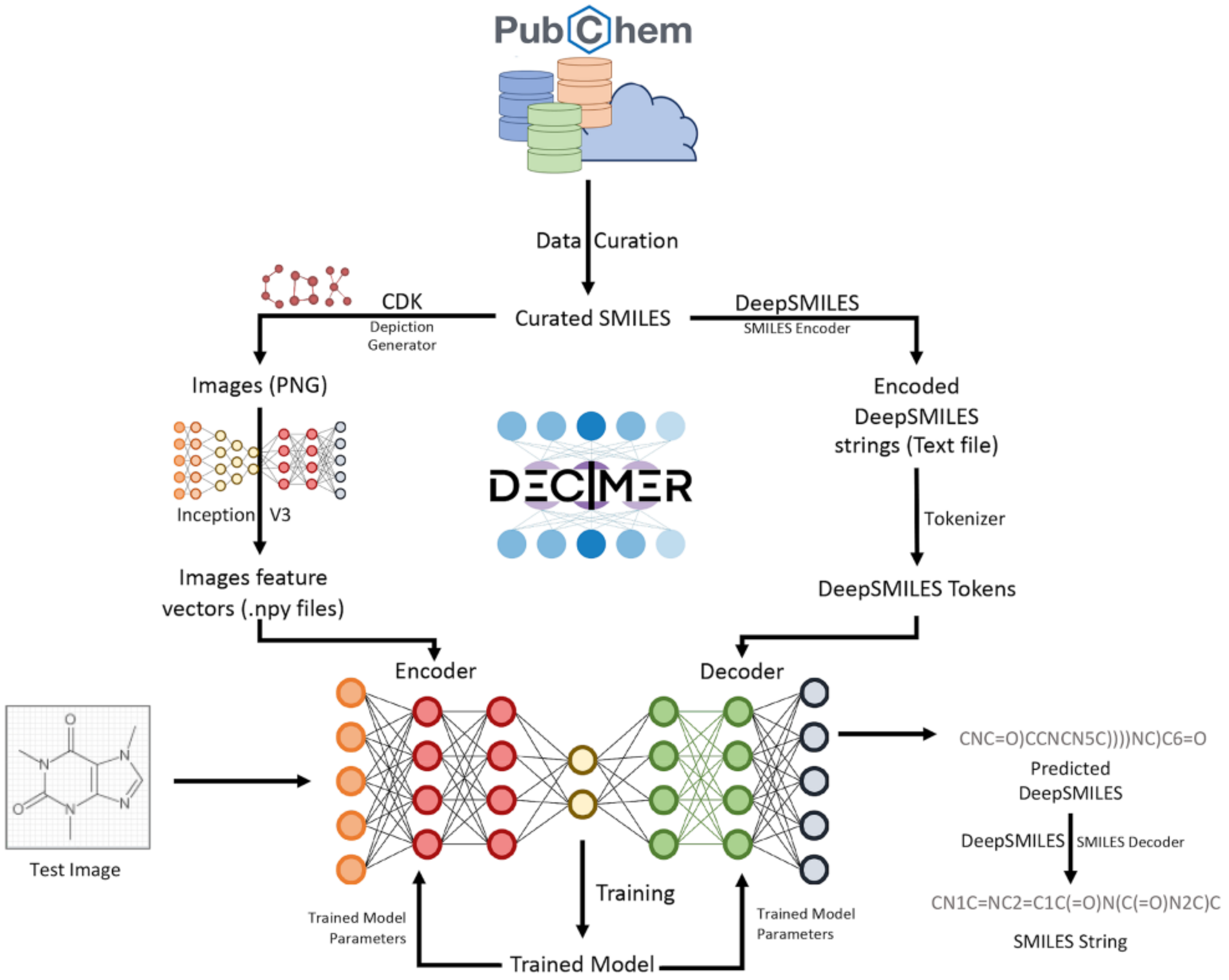


Fig. 3 Schema of the DECIMER workflow

implementation of Show, Attend and Tell, published at [23], which is used unaltered by us. Their encoder network is a convolutional NN (CNN), which consists of a single fully connected layer and a RELU activation function. Their decoder network is a recurrent NN (RNN), consisting of a gated recurrent unit (GRU) and two fully connected layers. The soft attention mechanism used in [23] was introduced by Bahdanau et al. [24].

We trained the model with DeepSMILES textual data and the corresponding bitmap of the chemical diagram. The text file is read by the model, the DeepSMILES is tokenized by the tokenizer, and the unique tokens are stored. The images are converted into feature vectors by using the unaltered Inception V3 [25] model and saved as NumPy arrays.

The model accuracy is determined by the average of all the calculated Tanimoto similarity scores as well as the number of Tanimoto 1.0 hits. The Tanimoto coefficient is superior to simple structure isomorphism because it yields the improvement of the recognition even when identity is not (yet) reached, and with a low degeneracy fingerprint such as the Pubchem fingerprint used here, the Tanimoto 1.0 is almost identical to the more rigorous structure isomorphism.

Initially, we trained multiple models with small training datasets to obtain the best hyper-parameters for our network. Exploration of the hyperparameter space led to 640 images per batch size, with embedding dimension size of 600 for the images, which we depicted on a 299 × 299 canvas size to match the Inception V3 model.

Table 1 Dataset sizes used in this work with corresponding computing times

Dataset index	Train data size	Test data size	Avrg. time/epoch (s)	Time for 25 epochs (s)
1	54,000	6000	94.32	2358
2	90,000	10,000	159.88	3997
3	450,000	50,000	880.6	22,015
4	900,000	100,000	2831.8	70,795
5	1,800,000	200,000	7239.28	180,982
6	2,700,000	300,000	11,964.72	299,118
7	4,050,000	450,000	17,495.12	437,378
8	5,850,000	650,000	25,702	642,550
9	7,200,000	800,000	32,926.8	823,170
10	8,969,751	996,639	41,652.24	1,041,306
11	12,600,000	1,400,000	64,909.28	1,622,732
12	15,102,000	1,678,000	91,880.84	2,297,021

The time for training the model with 15 million structures corresponds to approximately a month on a single Tesla V100 GPU

We used an Adam optimizer with a learning rate 0.0005 and Sparse Categorical Cross entropy to calculate the loss. We trained all the models for 25 epochs, which typically led to convergence. Once the models converged, we started the evaluation of the test set.

The models were trained on an inhouse server equipped with an nVidia Tesla V100 Graphics Card, 384 GB of RAM and two Intel(R) Xeon(R) Gold 6230 CPUs. Even though the training entirely happens on the GPU, the initial dataset preparation was CPU-based.

Training time obviously scales with data size (Table 1, Fig. 4). Model success was evaluated with an independent test data set. During the preparation of this manuscript, initial experiments with parallel training indicated that scaling was not satisfactory beyond 2 or 3 GPUs.

Here we report our results for training data sizes between 54,000 and 15,000,000 structures, with the largest training data set taking 27 days to converge on the hardware reported above (Table 1, Fig. 4). Figures 5 and 6 show the growth of the accuracy of predictions with increasing train data size.

Training success was determined with a number of indicators (Figs. 5 and 6), such as the percentage of Tanimoto 1.0 predictions, the average Tanimoto similarity of all predictions, and the percentage of invalid SMILES produced by the model. Figure 5a demonstrates that the model's ability to produce valid SMILES and avoid invalid ones steeply increases with larger training datasets. The same can be observed for the two key parameters of this application, the average Tanimoto similarity and the Tanimoto 1.0 percentage, which indicate the fitness of the model to accurately generate a machine-readable structure from a bitmap of a chemical diagram. We show here that the similarity of predicted chemical graphs to the correct chemical graph becomes constantly better with more training data. While we regularly operate with the chemical paradigm that similar structures have similar properties and therefore increasing structure similarity would convey increasing similarity of inherent properties, we chose to confirm that this would hold in our application case. We therefore further evaluated the models' success with additional descriptors such as LogP or ring count between original and the predicted SMILES, which indicates that the model consistently produces better and better machine representations with growing training data size. The improvements do not seem to converge prematurely.

In order to assess the promise of these preliminary results, we performed an idealistic linear extrapolation of our data toward larger training data sizes, which indicate that close-to-perfect detection of chemical structures would require training data sizes with 50 to 100 million structures. Such a training data volume will likely require a training time of 4 months with our setup with a single GPU. We are currently experimenting with the distributed learning solution currently available in the Tensorflow 2.0 API to reduce this training time significantly, also evaluating Google's Tensor Processing Units (TPU).

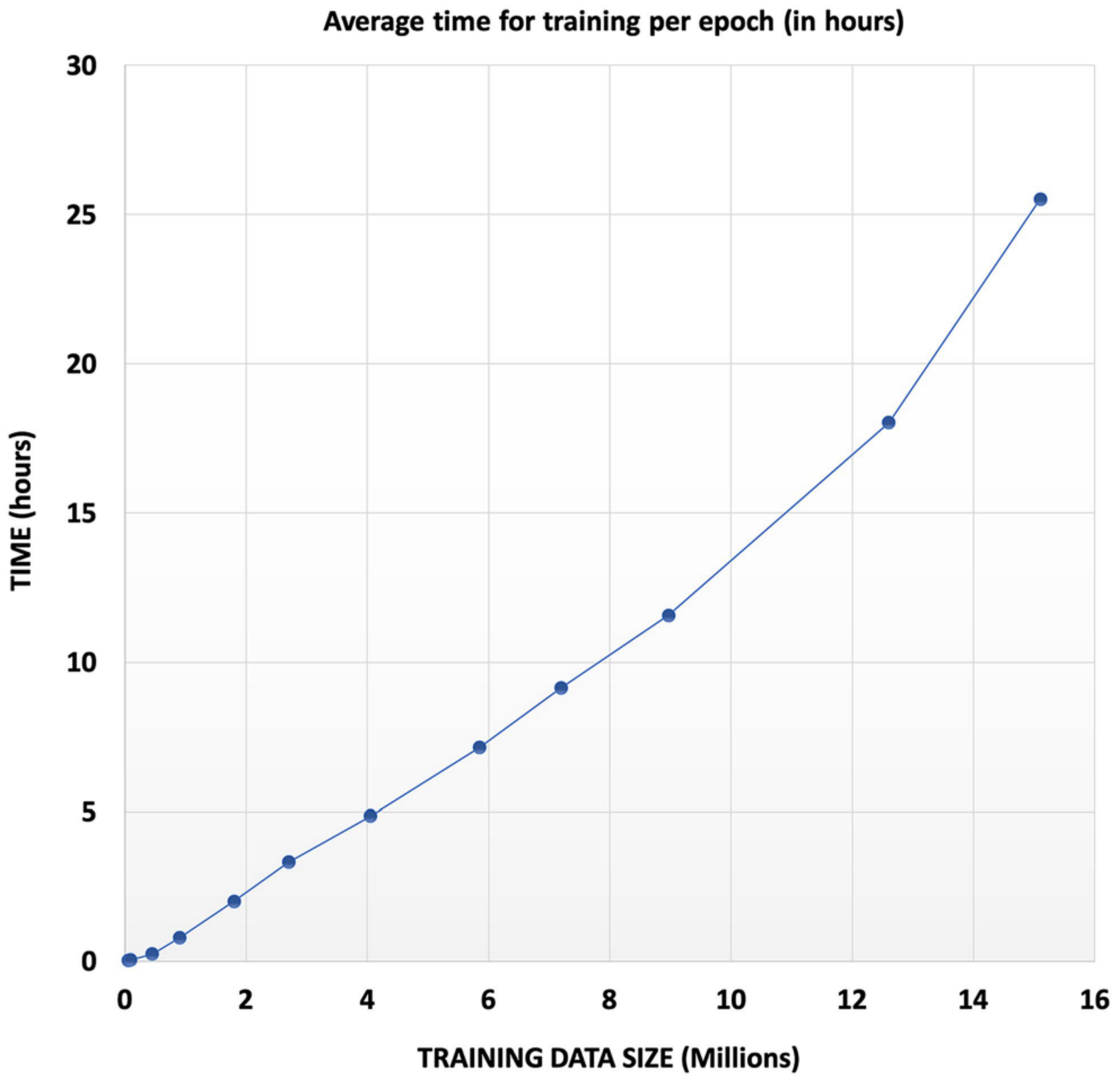
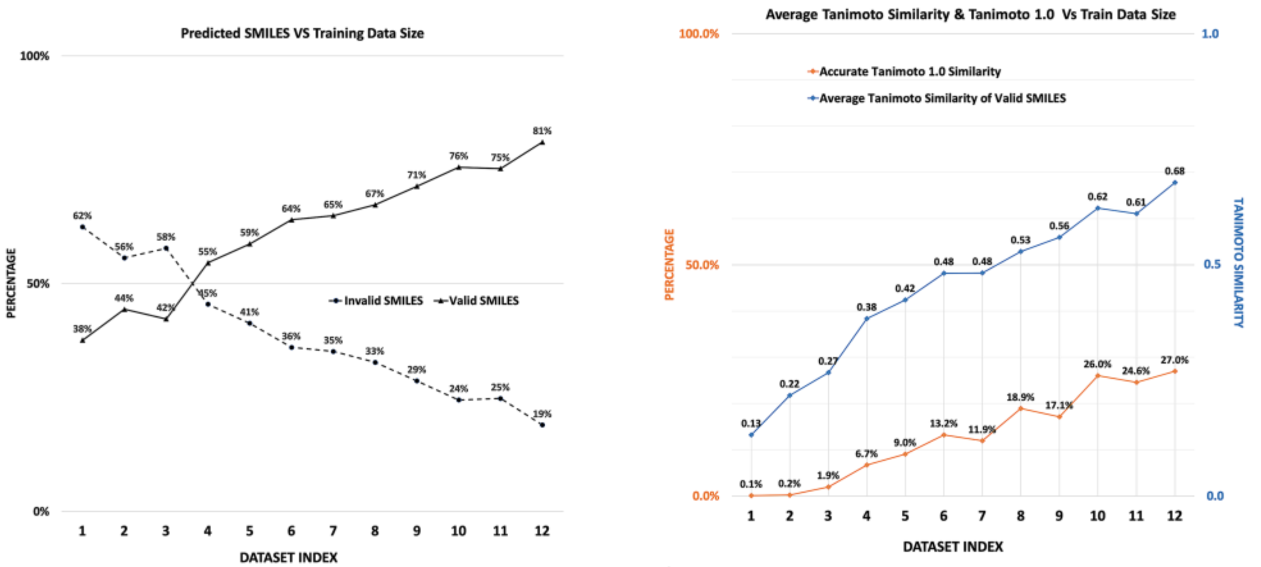


Fig. 4 Average time spent on training each epoch with increasing dataset size

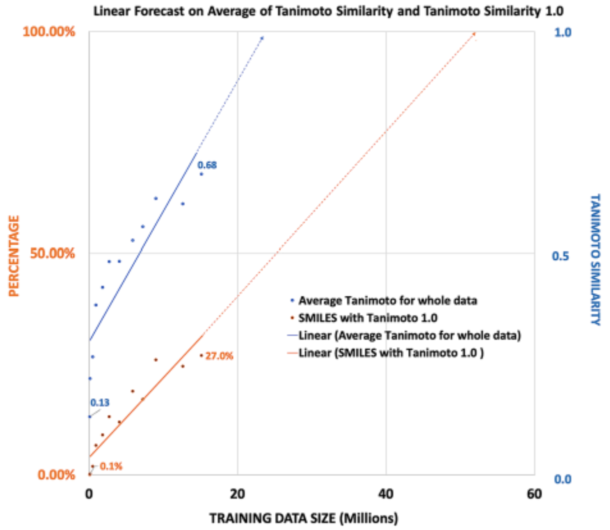
Conclusions

Here we have presented preliminary results indicating that a show-and-tell deep neural network setup has the potential to successfully extract a machine-readable structure representation when trained with tens of millions of examples. The training setup makes minimal assumptions about the problem. Training success depended on the input data representation. DeepSMILES were superior over SMILES and we have the preliminary indication that the recently reported SELFIES outperform DeepSMILES. For example, for a training data size of 6 Mio images, we obtained an average Tanimoto similarity of 0.53 with DeepSMILES and 0.78 with SELFIES. An extrapolation of our results towards larger training data sizes suggests that we might be able to achieve near-accurate prediction with 50 to 100 million training structures. Such training can be completed in uncomfortable but feasible time spans of several months on a single GPU.



a *Improved learning of the SMILES syntax with growing training data size. The percentages of valid and invalid SMILES add up to 100%. The dataset index refers to Table 1.*

b *Average Tanimoto similarity (right, orange) and percentage of structures with Tanimoto 1.0 similarity (left, blue) of valid SMILES predictions for the training data. The dataset index refers to Table 1.*

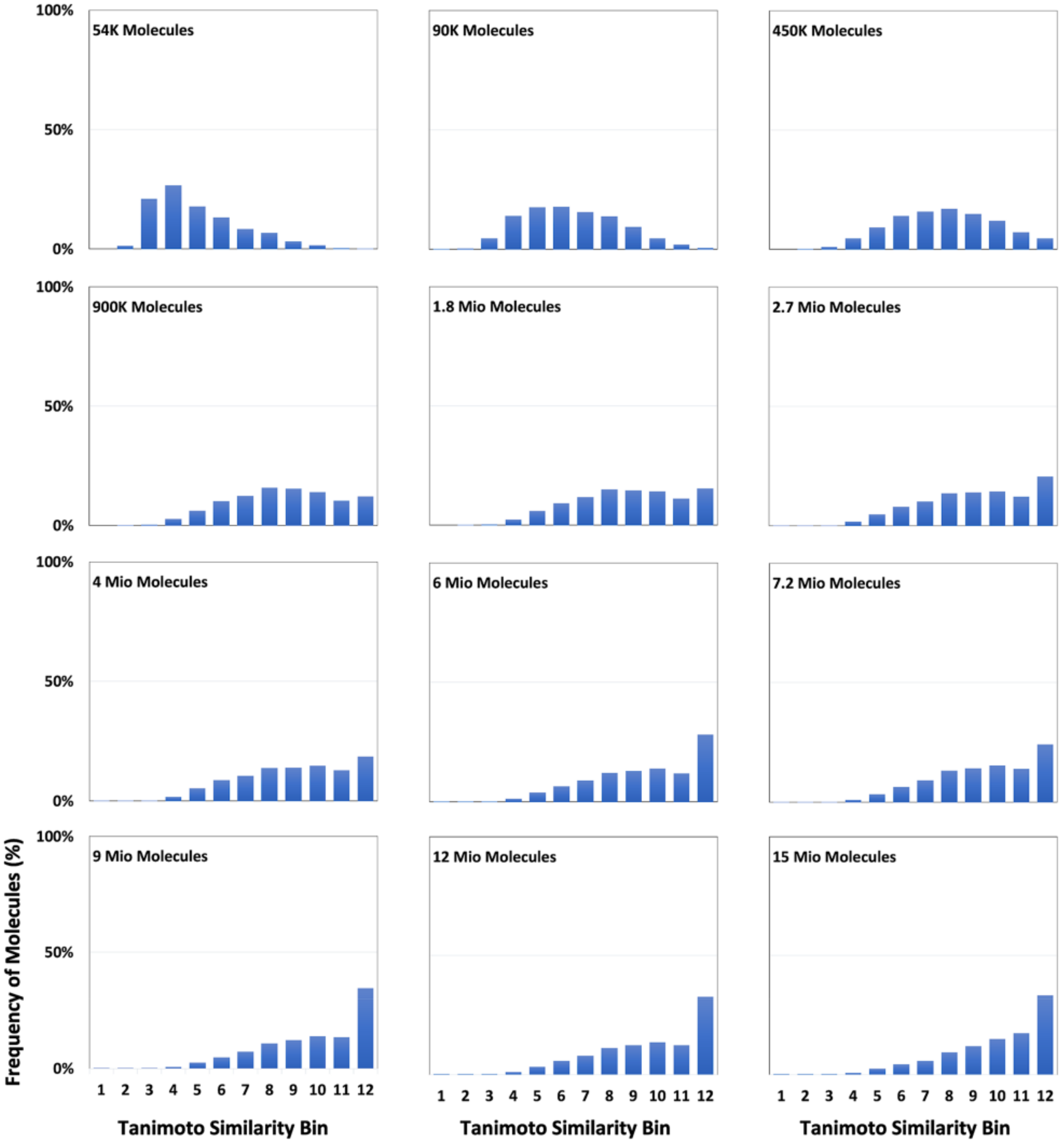


c *Linear extrapolation on the predicted results forecasting the achievable accuracy with more data. The linear trend is only used to indicate the order of magnitude of training data, which would be necessary for a successful structure prediction near perfection - the sketched linear growth will, of course, inevitably crossover into a saturation curve with increasing training set size.*

Fig. 5 Development of training success indicators with increasing train data size. **a** Improved learning of the SMILES syntax with growing training data size. The percentages of valid and invalid SMILES add up to 100%. The dataset index refers to Table 1. **b** Average Tanimoto similarity (right, orange) and percentage of structures with Tanimoto 1.0 similarity (left, blue) of valid SMILES predictions for the training data. The dataset index refers to Table 1. **c** Linear extrapolation on the predicted results forecasting the achievable accuracy with more data. The linear trend is only used to indicate the order of magnitude of training data, which would be necessary for a successful structure prediction near perfection—the sketched linear growth will, of course, inevitably crossover into a saturation curve with increasing training set size

Our work is entirely based on open-source software and open data and is available to the general public for any purpose.

We are currently moving towards larger training sets with the use of parallelization and more powerful hardware and hope to report the results in a full paper on this work in due time.



Tanimoto similarity bin values on x axis												
Bin	1	2	3	4	5	6	7	8	9	10	11	12
Value	0.0	0.0 - 0.1	0.1 – 0.2	0.2 – 0.3	0.3 – 0.4	0.4 – 0.5	0.5 – 0.6	0.6 – 0.7	0.7 – 0.8	0.8 – 0.9	0.9 – 1.0	1.0

Fig. 6 Distribution of Tanimoto-Similarity between the training structures and the structure recognised by DECIMER. Y-axis: frequency of molecules in percentage, x-axis: Tanimoto similarity range in bins

Acknowledgements
Not applicable.

Authors' contributions
KR developed the software and performed the data analysis. CS and AZ conceived the project and supervised the work. All authors contributed to, read and approved the final manuscript.

Funding
Open Access funding enabled and organized by Projekt DEAL. The authors acknowledge funding by the Carl-Zeiss-Foundation.

Availability of data and materials
The code for DECIMER is available at https://github.com/Kohulan/DECIMER, https://github.com/Kohulan/DECIMER-Image-to-SMILES.

Competing interests
The authors declare that they have no competing interests.

Author details
[1] Institute for Inorganic and Analytical Chemistry, Friedrich-Schiller-University Jena, Lessingstr. 8, 07743 Jena, Germany. [2] Institute for Bioinformatics and Chemoinformatics, Westphalian University of Applied Sciences, August-Schmidt-Ring 10, 45665 Recklinghausen, Germany.

Received: 11 June 2020 Accepted: 13 October 2020
Published online: 27 October 2020

References

1. McDaniel JR, Balmuth JR (1992) Kekule: OCR-optical chemical (structure) recognition. J Chem Inf Model 32(4):373–378. https://doi.org/10.1021/ci00008a018
2. Borman S (1992) New computer program reads, interprets chemical structures. Chem Eng News 70(12):17–19. https://doi.org/10.1021/cen-v070n012.p017
3. Contreras ML, Allendes C, Alvarez LT, Rozas R (1990) Computational perception and recognition of digitized molecular structures. J Chem Inf Model 30(3):302–307. https://doi.org/10.1021/ci00067a014
4. Casey R, Boyer S, Healey P, Miller A, Oudot B, Zilles K (1993) Optical recognition of chemical graphics. In: Proceedings of 2nd international conference on document analysis and recognition (ICDAR '93). IEEE Computer Society Press, Washington, DC, pp 627–631. https://ieeexplore.ieee.org/document/395658/
5. Ibison P, Jacquot M, Kam F, Neville AG, Simpson RW, Tonnelier C et al (1993) Chemical literature data extraction: the CLiDE Project. J Chem Inf Model 33(3):338–344. https://doi.org/10.1021/ci00013a010
6. Zimmermann M, Bui Thi LT, Hofmann M (2005) Combating illiteracy in chemistry: towards computer-based chemical structure reconstruction. ERCIM News 60(60):40–41. https://www.ercim.eu/publication/Ercim_News/enw60/zimmermann.html, https://www.researchgate.net/publication/228766116_Combating_illiteracy_in_chemistry_towards_computer-based_chemical_structure_reconstruction
7. Algorri M-E, Zimmermann M, Friedrich CM, Akle S, Hofmann-Apitius M (2007) Reconstruction of chemical molecules from images. In: 2007 29th annual international conference of the IEEE engineering in medicine and biology society. IEEE, New York, pp 4609–4612. https://ieeexplore.ieee.org/document/4353366/
8. Algorri M-E, Zimmermann M, Hofmann-Apitius M (2007) Automatic recognition of chemical images. In: Eighth Mexican international conference on current trends in computer science (ENC 2007). IEEE, New York, pp 41–46. https://ieeexplore.ieee.org/document/4351423/
9. Park J, Rosania GR, Shedden KA, Nguyen M, Lyu N, Saitou K (2009) Automated extraction of chemical structure information from digital raster images. Chem Cent J 3(1):4. https://doi.org/10.1186/1752-153X-3-4
10. Filippov IV, Nicklaus MC (2009) Optical structure recognition software to recover chemical information: OSRA, an open source solution. J Chem Inf Model 49(3):740–743. https://doi.org/10.1021/ci800067r
11. Karthikeyan M (2017) Chemical structure recognition tool. US Patent 9,558,403 B2
12. Kwon O-S, Kim D, Kim C-K, Sun J, Sim CJ, Oh D-C et al (2020) Cytotoxic scalarane sesterterpenes from the sponge *Hyrtios erectus*. Mar Drugs 18(5):253
13. Silver D, Schrittwieser J, Simonyan K, Antonoglou I, Huang A, Guez A et al (2017) Mastering the game of go without human knowledge. Nature 550(7676):354–359. https://doi.org/10.1038/nature24270
14. Staker J, Marshall K, Abel R, McQuaw CM (2019) Molecular structure extraction from documents using deep learning. J Chem Inf Model 59(3):1017–1029
15. Oldenhof M, Arany A, Moreau Y, Simm J (2020) ChemGrapher: optical graph recognition of chemical compounds by deep learning. https://arxiv.org/abs/2002.09914
16. Willighagen EL, Mayfield JW, Alvarsson J, Berg A, Carlsson L, Jeliazkova N et al (2017) The Chemistry Development Kit (CDK) v2.0: atom typing, depiction, molecular formulas, and substructure searching. J Cheminform 9(1):33. https://doi.org/10.1186/s13321-017-0220-4
17. Kim S, Chen J, Cheng T, Gindulyte A, He J, He S et al (2019) PubChem 2019 update: improved access to chemical data. Nucleic Acids Res 47(D1):D1102–D1109
18. Weininger D (1988) SMILES, a chemical language and information system: 1: introduction to methodology and encoding rules. J Chem Inf Comput Sci 28(1):31–36
19. O'Boyle N, Dalke A (2018) DeepSMILES: an adaptation of SMILES for use in machine-learning of chemical structures. chemRxiv: 1026434, pp 1–9. https://github.com/nextmovesoftware/deepsmiles
20. Krenn M, Häse F, Nigam A, Friederich P, Aspuru-Guzik A (2020) Self-referencing embedded strings (SELFIES): a 100% robust molecular string representation. https://github.com/aspuru-guzik-group/selfies. Accessed 2 June 2020
21. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C et al (2016) TensorFlow: large-scale machine learning on heterogeneous distributed systems. https://arxiv.org/abs/1603.04467
22. Xu K, Ba JL, Kiros R, Cho K, Courville A, Salakhutdinov R et al (2015) Show, attend and tell: neural image caption generation with visual attention. In: 32nd International conference on machine learning, ICML 2015, vol 3, pp 2048–2057
23. tensorflow. tensorflow/docs. https://github.com/tensorflow/docs/blob/master/site/en/tutorials/text/image_captioning.ipynb. Accessed 18 Aug 2020
24. Bahdanau D, Cho KH, Bengio Y (2015) Neural machine translation by jointly learning to align and translate. In: 3rd international conference on learning representations, ICLR 2015—Conf Track Proc, pp 1–15
25. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition, pp 2818–2826

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

BMC

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Rajan *et al. J Cheminform (2021) 13:5*
https://doi.org/10.1186/s13321-021-00485-4

Journal of Cheminformatics

PRELIMINARY COMMUNICATION

Open Access

Molecule Set Comparator (MSC): a CDK-based open rich-client tool for molecule set similarity evaluations

Kohulan Rajan[1], Jan-Mathis Hein[2], Christoph Steinbeck[1] and Achim Zielesny[2*]

Abstract

The open rich-client Molecule Set Comparator (MSC) application enables a versatile and fast comparison of large molecule sets with a unique inter-set molecule-to-molecule mapping obtained e.g. by molecular-recognition-oriented machine learning approaches. The molecule-to-molecule comparison is based on chemical descriptors obtained with the Chemistry Development Kit (CDK), such as Tanimoto similarities, atom/bond/ring counts or physicochemical properties like logP. The results are summarized and presented graphically by interactive histogram charts that can be examined in detail and exported in publication quality.

Keywords: Molecule set comparison, Chemistry Development Kit, CDK, Descriptor, Machine learning

The comparison of molecules lies at the heart of cheminformatics from its beginnings with molecular comparative studies addressing a wide range of research activities [1]. A variety of molecular comparisons may be computationally performed with open cheminformatics libraries like RDKit [2], Indigo [3] or CDK [4–8] driven by appropriate scripting solutions (which require programming skills) or with open rich-client applications like DataWarrior [9, 10] or Scaffold Hunter [11–13] (which are accessible to scientific end-users). Halfway between scripting solutions and rich-clients there are pipelining-workflow systems like the open analytics platform KNIME [14, 15] that offer specific worker nodes—which themselves may be based on open cheminformatics libraries like the RDKit [16], Indigo [17] or CDK [18] nodes for KNIME—that can be flexibly connected to construct automated molecule comparison workflows where the node composition is comfortably supported by a graphical editor.

Besides the frequent use cases, which are already covered by available solutions, current machine learning tasks make demands on dedicated molecule-to-molecule comparisons, which have to be addressed by new specific applications to effectively support corresponding research activities.

"Intelligent" molecular recognition systems based on new deep learning approaches in cheminformatics try to predict a molecule in question (the system's output) from a specific molecular representation (the system's input), where the input representation of the desired molecule may be a set of its molecular features, a pixel image of the molecule's 2D structure or any other encoding that relates to the original molecule. To assess the predictive power of a molecular recognition system, the predicted molecules have to be comprehensively compared with their corresponding original molecules that were used to create the molecular representation for the system's input. For these comparisons a fingerprint based Tanimoto similarity between original and predicted molecule may be used or the difference of their atom/ring counts may be calculated. Also differences regarding their physico-chemical properties like logP may be of interest. For

---

*Correspondence: achim.zielesny@w-hs.de
[2] Institute for Bioinformatics and Chemoinformatics, Westphalian University of Applied Sciences, August-Schmidt-Ring 10, 45665 Recklinghausen, Germany
Full list of author information is available at the end of the article

© The Author(s) 2021. This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

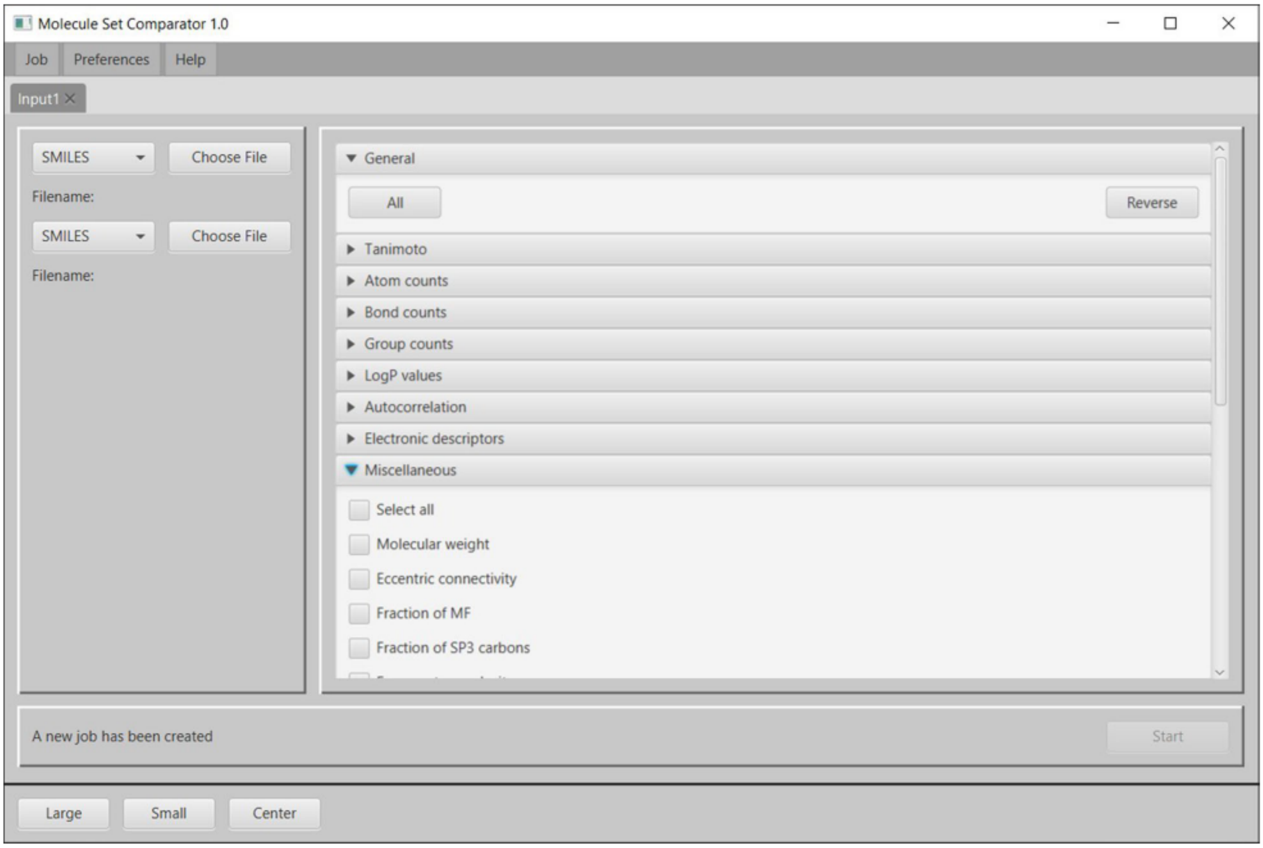
large sets of original and corresponding predicted molecules these similarity or difference values may be neatly summarized by frequency histograms which then allow for a versatile and fast assessment of the recognition abilities of the machine with regard to the selected comparator. The new Molecule Set Comparator (MSC) application focuses on these comparisons and aims to alleviate them.

MSC is a Java rich-client end-user application which architecturally follows a Model-View-Controller (MVC) pattern [19] and utilizes JavaFX [20] for graphical user interface (GUI) design and charting. All molecular operations are performed with the Chemistry Development Kit (CDK) [4–8]. Graphical image generation is realized with the PDFBox library [21] and the Batik SVG toolkit [22].

Figure 1 shows the MSC input view with molecule sets and comparative chemical descriptor selection. Supported molecule set formats are SMILES or SDF text files. The first set of molecules (e.g. a text file with a single SMILES string in each line) should contain the original molecules from which specific molecular representations have been derived to be used as input for the molecular recognition system. The second molecule set should contain the molecules predicted by the molecular recognition system at a corresponding position (i.e. the SMILES string of the predicted molecule must be on the same line as its original molecule in the first set of molecules). It should be noted that the order of the two molecule sets to be specified does not affect the subsequent comparative evaluations since these rely on absolute molecule-pair properties only, i.e. the molecule sets could be mutually interchanged without any effect.

The available descriptors for original/predicted molecule comparison are summarized in Table 1. The Tanimoto similarity directly refers to an original/predicted molecule pair, for all other numerical descriptors the absolute difference between the descriptor value of the original and the predicted molecule is calculated. The resulting Tanimoto similarities and absolute descriptor value differences of the original/predicted molecule pairs are then used for the particular histogram binnings in the following.

The MSC input view allows the concurrent selection of multiple descriptors for original/predicted molecule comparisons. A comparative histogram chart is then generated for each selected descriptor (see Fig. 2): each



A new job has been created

1 MSC input view with molecule sets selection and comparator choice for subsequent molecule comparisons [illegible] 1 [illegible]

[illegible]	[illegible]
Tanimoto similarity	Basic, LINGO, Extended, E-State, PubChem, Shortest Path, Substructure
Atom counts	All atoms, Carbon, Oxygen, Sulphur, Nitrogen, Phosphor, Aromatic atoms, Spiro atoms, C1SP1, C2SP1, C1SP2, C2SP2, C3SP2, C1SP3, C2SP3, C3SP3, C4SP3
Bond counts	All bonds, Aromatic bonds, Single, Double, Triple, Quadruple, Rotatable
Group counts	Acidic groups, Basic groups, All small rings, Aromatic rings, Rings of size 3–9, All rings, H-bond acceptors, H-Bond donors
LogP values	Mannhold LogP, JPLogP, XLogP, ALogP, ALogP2
Autocorrelation	ATS charge, ATS mass, ATS polarizability
Electronic descriptors	Atomic polarizability, Bond polarizability, Fractional PSA, Topological PSA, Molar refractivity
Miscellaneous	Molecular weight, Eccentric connectivity, FMF, SP3 fraction, Fragment complexity, 1–3. kappa shape index, Largest pi system, Largest chain, Longest aliphatic chain, Petitjean number, Petitjean shape index, VdW volume, Vertex adjacency, Weighted path descriptor, Wiener path number, Wiener polarity number, Zagreb index, Equality

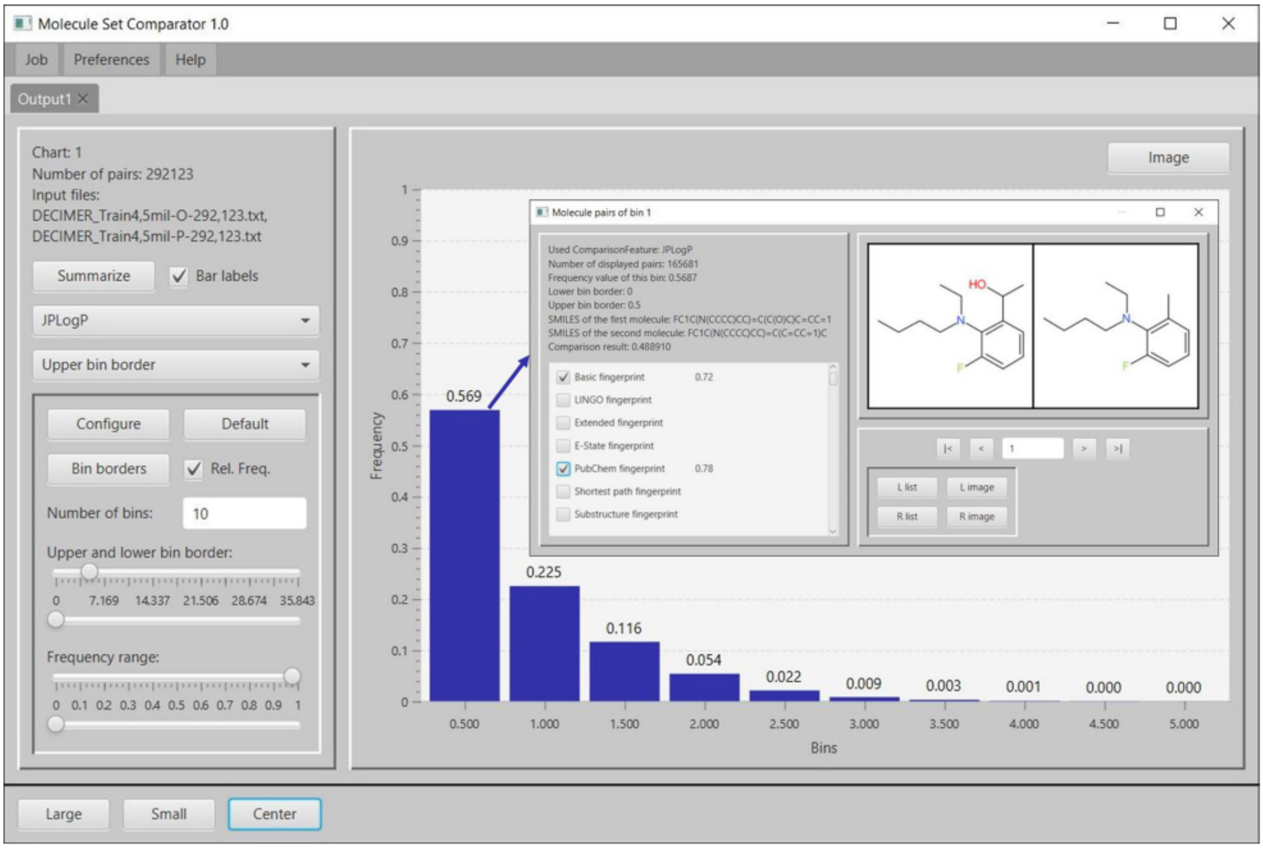


Chart: 1
Number of pairs: 292123
Input files:
DECIMER_Train4,5mil-O-292,123.txt,
DECIMER_Train4,5mil-P-292,123.txt

☑ Bar labels

JPLogP

Upper bin border

☑ Rel. Freq.

Number of bins: 10

Upper and lower bin border:
0 7.169 14.337 21.506 28.674 35.843

Frequency range:
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

Used ComparisonFeature: JPLogP
Number of displayed pairs: 165681
Frequency value of this bin: 0.5687
Lower bin border: 0
Upper bin border: 0.5
SMILES of the first molecule: FC1C(N(CCCC)CC)=C(C(O)C)C=CC=1
SMILES of the second molecule: FC1C(N(CCCC)CC)=C(C=CC=1)C
Comparison result: 0.488910

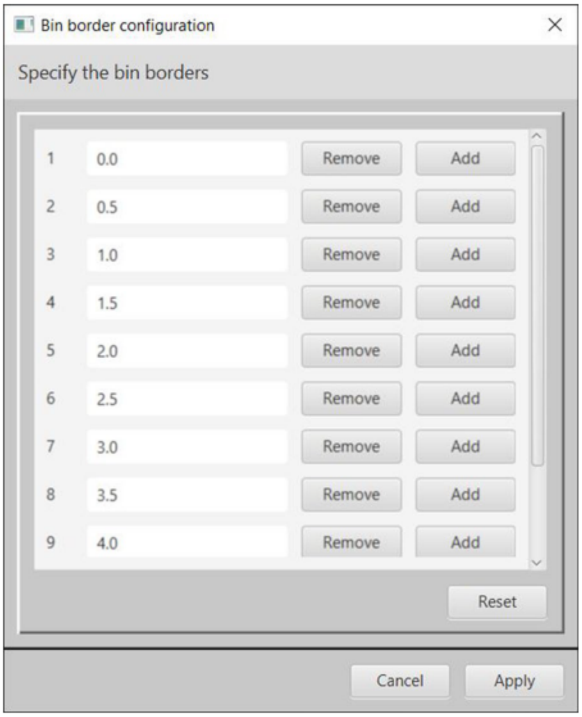
☑ Basic fingerprint 0.72
☐ LINGO fingerprint
☐ Extended fingerprint
☐ E-State fingerprint
☑ PubChem fingerprint 0.78
☐ Shortest path fingerprint
☐ Substructure fingerprint

[illegible] 2 Output view with an interactive comparative histogram chart. A histogram-bar related modal window (see arrow) provides detailed information about the corresponding original/predicted molecule pairs

histogram consists of a number of bars where each bar comprises a specific range of evaluated similarity or absolute descriptor difference values: The height of a bar corresponds to the number of molecule pairs whose similarity/absolute descriptor difference value lies within the bar's value range. The default number of bars is 10 and the default value for the lower border of the first bar and the upper border of the last bar are set to the minimal and maximal similarity/absolute descriptor difference values respectively.

Figure 2 depicts the calculated output view for a JPlogP-descriptor-based comparison as an example:

The numerical differences between the JPlogP values of all original/predicted molecule pairs are evaluated and binned according to their absolute difference values in order to match the specified number of bars of the histogram. The highest left bar contains 56.9 % of all original/predicted molecule pairs which have an absolute JPlogP difference value between 0 and 0.5, the second next left bar contains 22.5 % of all molecule pairs with an absolute JPlogP difference between 0.5 and 1 etc. (note, that the bin labels describe the upper bin border value in this case). For documentation purposes, the *Summarize* button (see Fig. 2 top left) can be used to retrieve a textual summary containing the calculated results together with some of their statistical characteristics (mean, minimum and maximum value). Each histogram chart can be comfortably configured with sliders for lower/upper bar borders or an input field for definition of the desired number of bars. In addition, bar borders may be arbitrarily adjusted via a separate dialog (see Fig. 3). Bar labels or the y-axis range may also be changed on-the-fly and bars may be labelled with their frequencies. Charts can be exported in arbitrarily high quality to different graphics formats (PNG, JPEG, PDF, or SVG). For an interactive exploration of the original/predicted molecule pairs behind a specific bar, this bar may be clicked to open a modal window which allows for navigation through all the corresponding original/predicted molecule pairs that sum up to the bar's height/frequency: For the displayed molecule pair in Fig. 2 the comparison result, (i.e. the calculated absolute JPlogP difference value) is 0.48891 where additional descriptors like *Basic* or *PubChem fingerprint* may be selected and calculated on-the-fly to inspect further similarities and differences. The molecular images may also be saved as PNG, JPEG, PDF, or SVG files. A calculated output view can be permanently saved and reloaded for later use.


Bin border configuration

Specify the bin borders

1 0.0
2 0.5
3 1.0
4 1.5
5 2.0
6 2.5
7 3.0
8 3.5
9 4.0

Fig. 3 Configuration dialog for arbitrary bar border definitions

MSC offers several preference settings: Default output directories for images, molecule lists, summary reports, calculated results, or a default input directory for molecule sets. Also, the number of parallel threads for comparative molecule calculations, the number of molecule pairs for MSC output, the default number of histogram bins, or the image quality can be specified. Preferences are permanently saved as an XML file in the *MSC_Files* directory of the MSC start directory.

MSC supports concurrent calculations via the *Parallel threads* preference which may considerably reduce computing times. For up to eight concurrent calculation threads MSC performs nearly inversely proportional to the number of threads (acceleration factor of 7.4 for an Intel Xeon Gold 6254 18-core processor, on a workstation running with Windows 10 Pro for Workstations, using 256 gigabytes of RAM). Using more than 8 threads still yields improvement up to 16 threads (factor of 11.9 on the same machine). On average, a (basic CDK fingerprint) Tanimoto similarity comparison of one million original/predicted molecule pairs using 8 calculation threads and 16 gigabytes of RAM takes less than 3 min.

The MSC GitHub repository contains the complete source code, all used libraries, installation instructions for all major platforms, a Gradle project for Netbeans as well as supplementary tutorials for installation, overview and application.

MSC is a versatile and fast end-user tool for comparing large molecule sets containing millions of chemical structures. As a rich client it does not require any programming skills and runs on all major platforms (Windows, Linux, and MacOS). A major application area is the support of molecular-recognition oriented machine learning tasks that require a large-scale and thorough comparative analysis of molecular features: The MSC tool allows to replace tedious scripting approaches with cumbersome manual PDF views by fast, flexible and comprehensive graphical point-and-click inspections. In addition to saving time, the new open tool may provide insights that might have been overlooked otherwise.

Project name: MSC.

Project home page: MSC repository at https://github.com/zielesny/MSC.

Operating system(s): Platform independent.

Programming language: Java.

Other requirements: JavaFX 14 [20], CDK 2.3 [4], PDFBox 2.0.17 [21], Batik SVG Toolkit 1.13 [22], Apache Commons Logging 1.2 [23].

License: GNU General Public License version 3.

The authors like to thank the communities that created the open libraries utilized by the MSC tool. The support of the Open Access Publication Fund of the Westphalian University of Applied Sciences is gratefully acknowledged.

KR, AZ and CS initiated the MSC project. KR and JMH designed, tested, applied and validated the application features. JMH developed the Java code. KR, AZ and CS lead the project development. All authors read and approved the final manuscript.

Open Access funding enabled and organized by Projekt DEAL.

MSC repository at https://github.com/zielesny/MSC.

Not applicable.

AZ is co-founder of GNWI - Gesellschaft für naturwissenschaftliche Informatik mbH, Dortmund, Germany.

[1] Institute for Inorganic and Analytical Chemistry, Friedrich-Schiller-University Jena, Lessingstr. 8, Jena 07745, Germany. [2] Institute for Bioinformatics and Chemoinformatics, Westphalian University of Applied Sciences, August-Schmidt-Ring 10, 45665 Recklinghausen, Germany.

Published online: 01 February 2021

1. Gasteiger J, Engel T (2018) Chemoinformatics. Basic concepts and methods. Wiley-VCH, Weinheim
2. RDKit: Open-source cheminformatics software. http://rdkit.org/. Accessed 10 Sept 2020.
3. Indigo Toolkit. https://lifescience.opensource.epam.com/indigo/. Accessed 17 Dec 2020.
4. Chemistry Development Kit (CDK). https://cdk.github.io/. Accessed 01 June 2020
5. Willighagen EL, Mayfield JW, Alvarsson J, Berg A, Carlsson L, Jeliazkova N, Kuhn S, Pluskal T, Rojas-Chertó M, Spjuth O, Torrance G, Evelo CT, Guha R, Steinbeck C (2017) The Chemistry Development Kit (CDK) v2.0: atom typing, depiction, molecular formulas, and substructure searching. J Cheminform. 9:33
6. May JW, Steinbeck C (2014) Efficient ring perception for the Chemistry Development Kit. J Cheminform. 6:3
7. Steinbeck C, Hoppe C, Kuhn S, Floris M, Guha R, Willighagen EL (2006) Recent Developments of the Chemistry Development Kit (CDK)—an open-source java library for chemo- and bioinformatics. Curr Pharm Des 12(17):2111–2120
8. Steinbeck C, Han Y, Kuhn S, Horlacher O, Luttmann E, Willighagen EL (2003) The Chemistry Development Kit (CDK): An Open-Source Java Library for Chemo- and Bioinformatics. J Chem Inf Comput Sci 43(2):493–500
9. DataWarrior. http://openmolecules.org/datawarrior/. Accessed 10 Sept 2020.
10. Sander T, Freyss J, von Korff M, Rufener C (2015) DataWarrior: an open-source program for chemistry aware data visualization and analysis. J Chem Inf Model 55(2):460–473
11. Wetzel S, Klein K, Renner S, Rauh D, Oprea TI, Mutzel P, Waldmann H (2009) Interactive exploration of chemical space with Scaffold Hunter. Nat Chem Biol 5:581–583
12. Schäfer T, Kriege N, Humbeck L, Klein K, Koch O, Mutzel P (2017) Scaffold Hunter: a comprehensive visual analytics framework for drug discovery. J Cheminform 9(1):1758–2946
13. Scaffold Hunter. http://scaffoldhunter.sourceforge.net/. Accessed 17 Dec 2020.
14. Berthold MR, Cebron N, Dill F, Gabriel TR, Koetter T, Meinl T, Ohl P, Sieb C, Thiel K, Wiswedel B (2008) KNIME: the konstanz information miner. In: Preisach C, Burkhardt H, Schmidt-Thieme L, Decker R (eds) Data analysis, machine learning and applications. studies in classification, data analysis, and knowledge organization. Springer, Berlin
15. KNIME Analytics Platform. https://www.knime.com/knime-analytics-platform. Accessed 17 Dec 2020.
16. RDKit Nodes for KNIME. https://www.knime.com/rdkit. Accessed 17 Dec 2020.
17. Indigo Nodes for KNIME. https://www.knime.com/community/indigo. Accessed 17 Dec 2020.
18. CDK Nodes for KNIME. https://www.knime.com/community/cdk. Accessed 17 Dec 2020.
19. Reenskaug TMH, Xerox MVC PARC (1978-79) http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html. Accessed 01 June 2020.
20. JavaFX. https://openjfx.io/. Accessed 01 June 2020.
21. Apache PDFBox | A Java PDF Library. https://pdfbox.apache.org/. Accessed 01 June 2020.
22. Apache Batik—Apache XML Graphics—Apache Software. https://xmlgraphics.apache.org/batik/. Accessed 01 June 2020.
23. Apache Commons Logging—Overview. https://commons.apache.org/proper/commons-logging/. Accessed 01 June 2020.

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]					
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Rajan *et al. J Cheminform (2021) 13:20*
https://doi.org/10.1186/s13321-021-00496-1

Journal of Cheminformatics

SOFTWARE

Open Access

DECIMER-Segmentation: Automated extraction of chemical structure depictions from scientific literature

Kohulan Rajan[1†], Henning Otto Brinkhaus[1†], Maria Sorokina[1], Achim Zielesny[2] and Christoph Steinbeck[1*]

Abstract

Chemistry looks back at many decades of publications on chemical compounds, their structures and properties, in scientific articles. Liberating this knowledge (semi-)automatically and making it available to the world in open-access databases is a current challenge. Apart from mining textual information, Optical Chemical Structure Recognition (OCSR), the translation of an image of a chemical structure into a machine-readable representation, is part of this workflow. As the OCSR process requires an image containing a chemical structure, there is a need for a publicly available tool that automatically recognizes and segments chemical structure depictions from scientific publications. This is especially important for older documents which are only available as scanned pages. Here, we present *DECIMER (Deep lEarning for Chemical IMagE Recognition) Segmentation*, the first open-source, deep learning-based tool for automated recognition and segmentation of chemical structures from the scientific literature. The workflow is divided into two main stages. During the detection step, a deep learning model recognizes chemical structure depictions and creates masks which define their positions on the input page. Subsequently, potentially incomplete masks are expanded in a post-processing workflow. The performance of DECIMER Segmentation has been manually evaluated on three sets of publications from different publishers. The approach operates on bitmap images of journal pages to be applicable also to older articles before the introduction of vector images in PDFs. By making the source code and the trained model publicly available, we hope to contribute to the development of comprehensive chemical data extraction workflows. In order to facilitate access to DECIMER Segmentation, we also developed a web application. The web application, available at https://decimer.ai, lets the user upload a pdf file and retrieve the segmented structure depictions.

Keywords: Deep learning, Image Segmentation, Optical Chemical Structure Recognition, Neural Networks, Chemical data extraction

Introduction

Chemical information is communicated as text and images in scientific publications [1]. These data formats are not intrinsically machine-readable and the manual extraction of chemical information from the literature is a time-consuming and error-prone procedure [2]. Hence, the increasing amount of chemical information being published creates a demand for automated chemical information extraction methods [3].

Over the course of the last three decades, there has been an active development in the field of Optical Chemical Structure Recognition (OCSR). OCSR is the translation of an image of a chemical structure into a machine-readable representation [4]. Most OCSR tools

---

*Correspondence: christoph.steinbeck@uni-jena.de

†Kohulan Rajan and Henning Otto Brinkhaus contributed equally to the work

[1] Institute for Inorganic and Analytical Chemistry, Friedrich-Schiller-University Jena, Lessingstr. 8, 07743 Jena, Germany

Full list of author information is available at the end of the article

© The Author(s) 2021. This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

are only capable of processing images with pure chemical structure depictions. Consequently, an automated segmentation of chemical structures from surrounding document information (text, tables etc.) is desirable. Previous approaches for this task are briefly described in the following paragraphs.

The open-source OCSR tool OSRA was published with a rule-based page segmentation algorithm. This mechanism identifies a chemical structure depiction based on the dimensions of a rectangular bounding box around a region of interest and the ratio of black and white pixels within the bounding box [5].

The open-source tool ChemSchematicResolver (CSR) is capable of segmenting images which only contain labels and chemical structure depictions. The classification of objects as labels or structure depictions is done using k-means clustering based on a custom feature density metric. If the publication is available as a markup document, these images can be extracted automatically, so that CSR is capable of processing whole documents [6]. Nevertheless, CSR is incapable of handling scanned pages or images which contain other objects than labels and structure depictions.

In 2019, Staker et al. reported a deep-learning-based OCSR tool which contains a segmentation procedure [7]. Opposed to the previously mentioned feature density-based approaches used by OSRA and CSR, they trained a convolutional neural network based on the U-Net architecture [8] to address the segmentation problem. Every image is processed multiple times at different resolutions and the masks generated by the model are averaged. The model was trained on a semi-synthetic dataset: OSRA was used to identify bounding boxes of potential chemical structure depictions in an unspecified amount of publications and patents. These areas were then cut out of the original documents and replaced with structures from publicly available datasets. During training, the images were randomly modified (with e.g. binarization, brightness adjustments) for data augmentation purposes. The segmentation accuracy has not been reported independently and the accuracy for the whole process of segmentation and structure resolution on different training datasets has been reported to be between 41 and 83% [7]. Unfortunately, the authors have not made their code and the trained models openly available.

With the DECIMER [9] project, we are currently working on the development of an open-source platform for the automated chemical structure extraction from printed literature. It aims at segmenting all chemical structure depictions from a given scanned document from the printed scientific literature and resolving their identity to yield a machine-readable presentation of the molecule. Here, we present *DECIMER Segmentation,* the first step of the DECIMER project and the first openly available deep learning tool for the segmentation of chemical structure depictions from scanned whole-page documents. Perspectively, the segmented chemical structure depictions will be used as input for the DECIMER algorithm, an OCSR method which predicts the SMILES string of the depicted chemical structure.

The algorithm consists of two main stages: First, during the detection step, a deep learning-based model generates masks that define the positions of the chemical structures in a given image. This is followed by a mask expansion procedure during which potentially incomplete masks are expanded until they cover the depictions completely (Fig. 1).

We did not attempt to extract vector graphics from modern PDF articles since this approach would fail for older articles before the early 1990 s, which are mostly scanned pages from printed versions of the journal. Instead, our approach operates on bitmap images of journal pages to be widely applicable also to older articles before the introduction of vector images in PDFs.

The source code of the application described herein as well as the trained model are publically available. Additionally, we created a web application accessible at *decimer.ai* to ensure that the segmentation algorithm becomes widely usable.

Implementation

The DECIMER Segmentation backend mechanism was built using Python 3 with Tensorflow 2.3.0 [10]. It mainly consists of the recognition of chemical structure diagrams using a deep learning model and the subsequent expansion of the resulting masks. The web application is developed in Python 3 using the Django version 3.1.3 framework and React.js for the front-end. The implementation details of the key elements as well as the complete workflow which accepts a pdf document as an input and returns the segmented chemical structure diagrams as an output are described below.

Deep learning algorithm

For the chemical structure detection, a model utilizing the Mask R-CNN network [11] was trained where the Mask R-CNN implementation published by the Matterport team [12] was used with some modifications to work on Tensorflow 2.3.0 with Keras at the backend.

The dataset used for training the model is based on 994 articles from the *Journal of Natural Products* which were chosen arbitrarily from all available issues. We converted the pages of these articles into JPEG images (96 dpi) using the Python pdf2image package [13] and deleted all images that did not contain any chemical structure diagram. After deleting pages which did not contain any

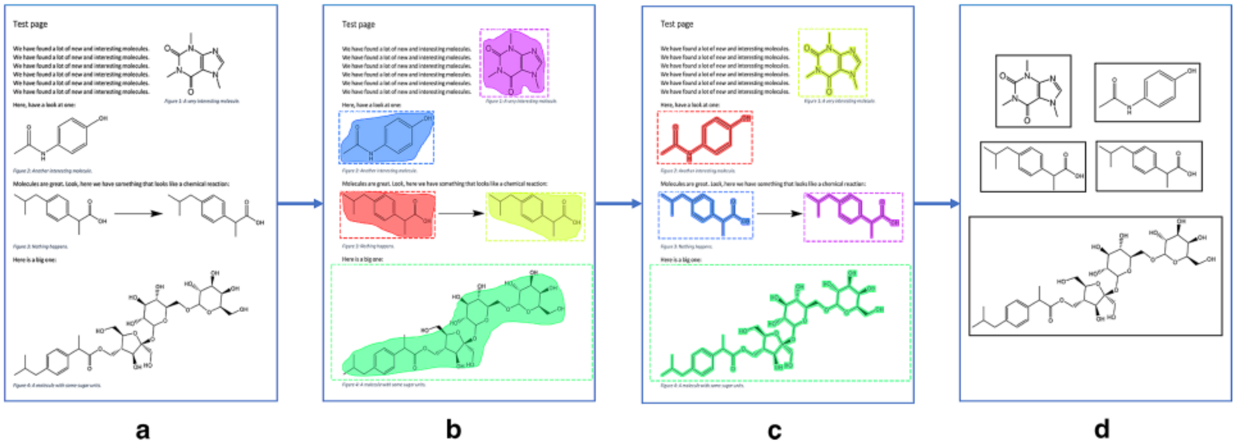


Fig. 1 Graphical summary of the DECIMER Segmentation workflow. The input is an image of a page with chemical structure depictions (**a**). Then, the chemical structure depictions are detected using the Mask-RCNN model (**b**). Subsequently, the masks that define the positions of the depictions are refined and expanded (**c**). Finally, the regions defined by the masks are segmented to yield individual images of chemical structure depictions (**d**)

chemical structure diagrams, there were a total of 1820 pages. The VGG image annotator tool [14] was used to manually annotate the chemical structure diagrams present in each image. Each depiction of a chemical structure was annotated by defining a polygon around it. If there were mechanism arrows or numbers within the structure, these were also included. Other objects like reaction arrows or labels around the chemical structures were not included. This resulted in 9992 annotated regions in the images which each contained one structure diagram (approximately 5.5 annotated structures per image). This dataset was split randomly into a training and validation subset of 90 and 10% respectively.

The model used the hyperparameters pre-defined by the Matterport team, furthermore, we used a batch size of two images per batch, learning rate of 0.001, learning momentum of 0.9, 500 steps per epoch and 50 steps for validation. The model was trained on a compute-server equipped with an Nvidia 1080Ti GPU, 64 GB of RAM and two Intel(R) Xeon(R) Silver 4114 CPUs. The training started from the pre-trained COCO weights provided by the Matterport team. The layers that could not be imported from the pre-trained weights of the model due to different amounts of classes (network heads) were trained for an initial 100 epochs, then the complete model was fine-tuned for another 100 epochs. During the whole training process, the parameters remained the same. This took approximately 26 h in total.

When applying the resulting model to an image of an article page, it returns masks which indicate whether or not a pixel in the original image belongs to a chemical structure diagram. These masks are binary matrices with the first two dimensions of the input image which can contain the values *True* or *False*. This means that every pixel in the original image has a corresponding value in the mask that defines whether or not this pixel is part of a chemical structure depiction. The positional information given in the masks can then be used for the segmentation of the chemical structures.

Mask expansion algorithm

A common problem with the masks generated by the Mask R-CNN model is an unwanted partial coverage of chemical structures only: The model did correctly recognize the chemical structure diagrams on a given page but did not cover them completely (Fig. 2, top row). Therefore, a custom mask expansion algorithm was developed which takes an image and a mask and creates a mask that covers the previously partially detected objects completely.

The expansion workflow begins with the binarization of the input image using a high threshold as recommended by the developers of CSR [6]. The binarization ensures that a non-white background or relicts from low-quality scans are filtered. Then, a binary dilation is applied to turn chemical structure depictions into connected objects, closing, for example, the gaps between an element symbol and its adjacent bonds with non-white pixels. The kernel object used for the dilation is a square with a resolution-dependent size.

Then, the initial seeds for the expansion are determined. For this, the center of the mask is defined as the position in the middle between the highest and the lowest x- and y- coordinates of *True* values of the mask. If the resulting center point is not covered by the mask due to its asymmetric shape, the center point is defined as a

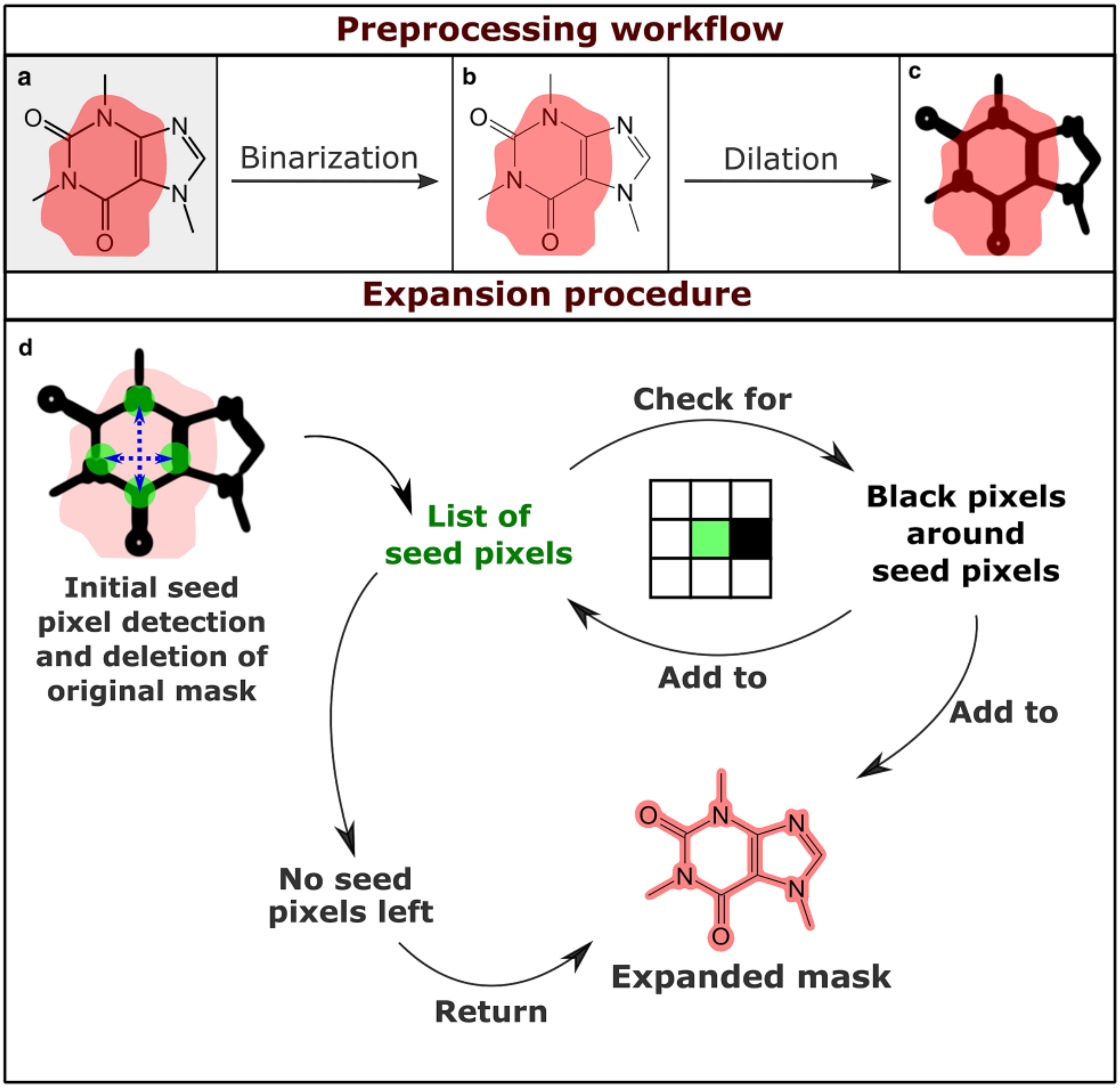


Fig. 2 Mask expansion workflow: During the preprocessing workflow, the original image (**a**) is binarized. The overlaying red patch represents an incomplete mask which is returned by the model. The resulting binary image (**b**) is then dilated to fill gaps within the structure (**c**). This is followed by the expansion procedure (**d**) where the mask is reconstructed by tracing the connected set of non-white pixels starting from a list of seed pixels until no further connected non-white pixels can be found in any direction. This ensures the segmentation of complete chemical structure depictions

random point between the highest and lowest x-coordinates which is covered by the mask. Based on the center point position, the algorithm attempts to determine four black pixels which are covered by the mask in four different directions. If at least one seed pixel is found, the original mask is replaced by a matrix of the same shape which only contains zeros and the expansion is initiated. If no seed pixels have been determined, objects on the contours of the mask are detected as seed pixels. In this case, the original mask is kept and only expanded based on the seed pixels.

The resulting list of seed pixel coordinates is used in the expansion procedure. The eight surrounding pixels of every seed pixel are examined. If one of them is black and not already covered by the mask, the mask is edited to cover it and it is added to the list of seeds. This recursive procedure leads to the inclusion of a complete object in the mask even if the original mask had not covered it completely. This outlined procedure is illustrated in Fig. 2.

The complete tool

DECIMER-Segmentation accepts PDF documents as input and returns grayscale images which contain the segmented chemical structure diagrams. Figure 3 illustrates the workflow.

All pages of the given input PDF document are converted to separate PNG images. All the images are stored in a folder with the name of the input PDF file. During the following procedure, the processing of each image can be parallelized. The structure detection model is initialized for each thread and generates the masks which define the positions of the chemical structure diagrams in the given image. Subsequently, these masks are processed by the expansion algorithm.

The final masks and images are then processed in a segmentation procedure. First, each segment is converted into a grayscale image. Then the maximal width and height of every mask are determined. With this information, an empty image with the dimensions of the resulting segment is created and the chemical structure diagram is placed in it. After all the segments are generated, they are resized into separate square images. These segments are displayed to the user at the end in the web application or saved locally.

Decimer.ai web application

The single-page web application (SPA) is freely available at https://decimer.ai and allows DECIMER usage without any local installation. It is implemented with the Django framework version 3.1.3 to manage the back-end and the API and with the JavaScript React.js library for the front-end. The SPA allows the user to upload a PDF file of a research article, performs image segmentation on it, and returns the extracted molecular images. The latter can be downloaded. The user can also click on the "I'm Feeling Lucky" button, to randomly select a recent article from the Open Access journal *MDPI Molecules* and run the segmentation on it.

Validation

Methods

In order to evaluate the performance of DECIMER Segmentation, we processed 25 articles from the *Journal of Natural Products*, 25 articles from *Phytochemistry* and 25 articles from *Molecules*. None of these journal articles were included in the training dataset. The 75 articles contained a total of 777 pages (365 in *Molecules*, 228 in *Phytochemistry*, 184 in *Journal of Natural Products*) and contained 887 segmented images (398 in *Molecules* 183 in *Phytochemistry*, 306 in *Journal of Natural Products*). We then manually inspected all segmented images to determine if they contain a complete chemical structure diagram or additional objects such as labels or reaction arrows. Furthermore, we determined the number of additional missed structure diagrams on the pages where structures had been determined.

Results and discussion

Without the application of the mask expansion, 81.6% of the segmented images contained complete chemical structure depictions (80.7% in *Molecules*, 83.5% in *Phytochemistry*, 81.7% in *Journal of Natural Products*). Here, a *complete detection* means that the structure was completely covered by the mask. It is necessary to mention that 9.4% (11.1% in *Molecules*, 5.5% in *Phytochemistry*, 9.4% in *Journal of Natural Products*) of these segments contained additional surrounding objects like labels or reaction arrows. Mechanism arrows or atom numbering were not counted as additional objects here as they are often positioned within the structure itself.

When the mask expansion was added to the procedure, the proportion of completely segmented structures increased to 99.8% (99.5% in *Molecules*, 100% in *Phytochemistry*, 100% in *Journal of Natural Products*). Among the validation results there were only two segments which did not contain a chemical structure diagram at all. Unfortunately, the proportion of segments that also contained additional objects rose to 11.2% (12.6% in *Molecules*, 11.5% in *Phytochemistry*, 9.5% in *Journal of Natural Products*). On average, 91.3% of the chemical structures were detected by the model (92.8% in *Molecules*, 86.3% in *Phytochemistry*, 92.7% in *Journal of Natural Products*). These results which represent the final output of DECIMER Segmentation are illustrated in Fig. 4.

Throughout the data used for validation, 885 of the 887 segments contained a complete chemical structure depiction. Given the fact that the model was only trained on articles from *Journal of Natural Products*, it is interesting to note that DECIMER Segmentation performs comparably well on the subsets of *Molecules* and *Phytochemistry* articles. This elucidates the general applicability of DECIMER Segmentation—it is capable of detecting chemical structures in the printed scientific literature in general, independent of specific publisher formats.

The inclusion of additional objects in approximately 11% of the segments is, in many cases, caused by surrounding labels or arrows, which were placed closely to the actual chemical structure diagram by the human creator of the graphic (see Fig. 5). It is worth

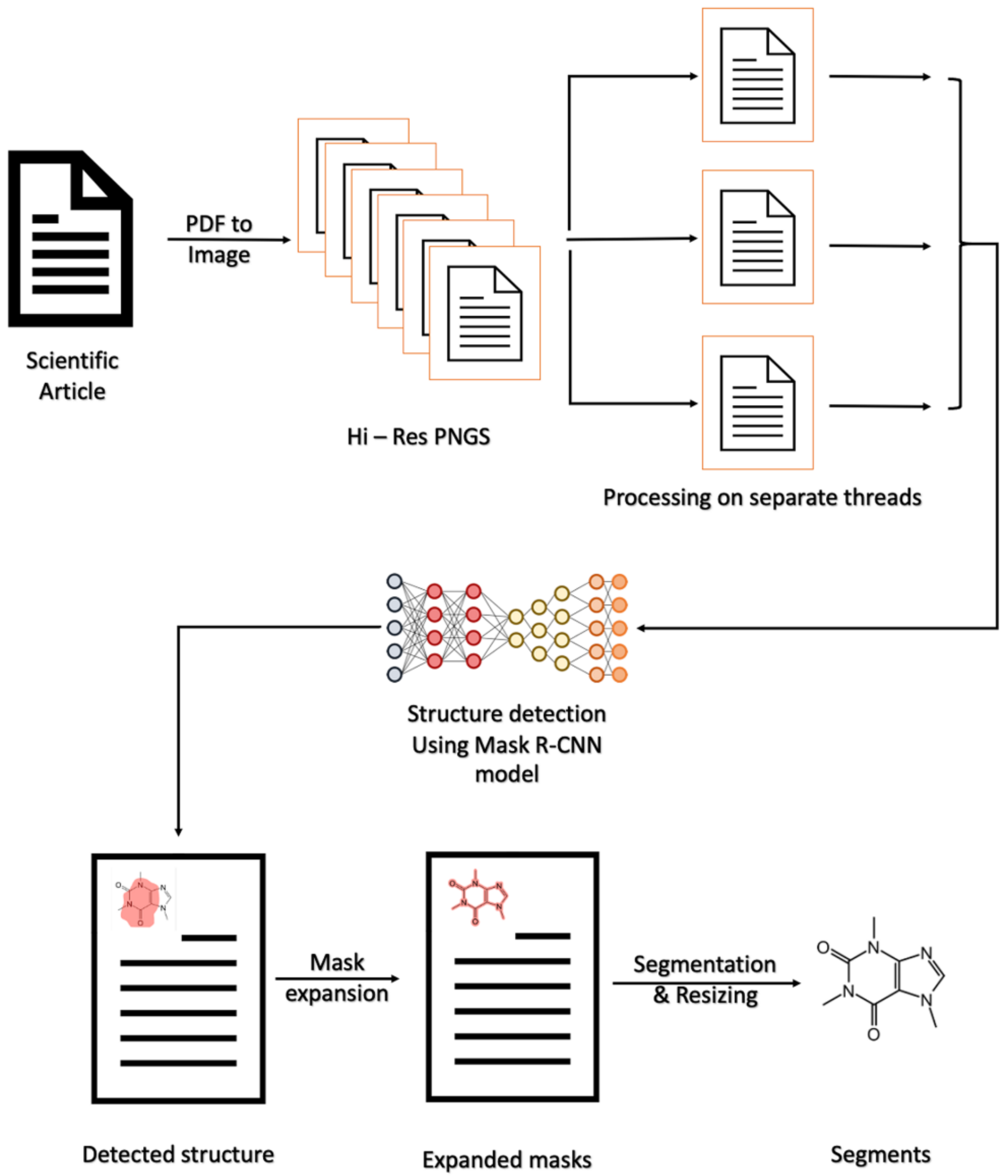


Fig. 3 Visual representation of the complete architecture of the tool

mentioning that the mask expansion sometimes aggravates the problem. For example, in some cases, the tip of a reaction arrow is covered by the mask which is returned by the model. If the arrow is close enough to the structure, the mask expansion leads to its complete inclusion. In other cases, an initially included reaction

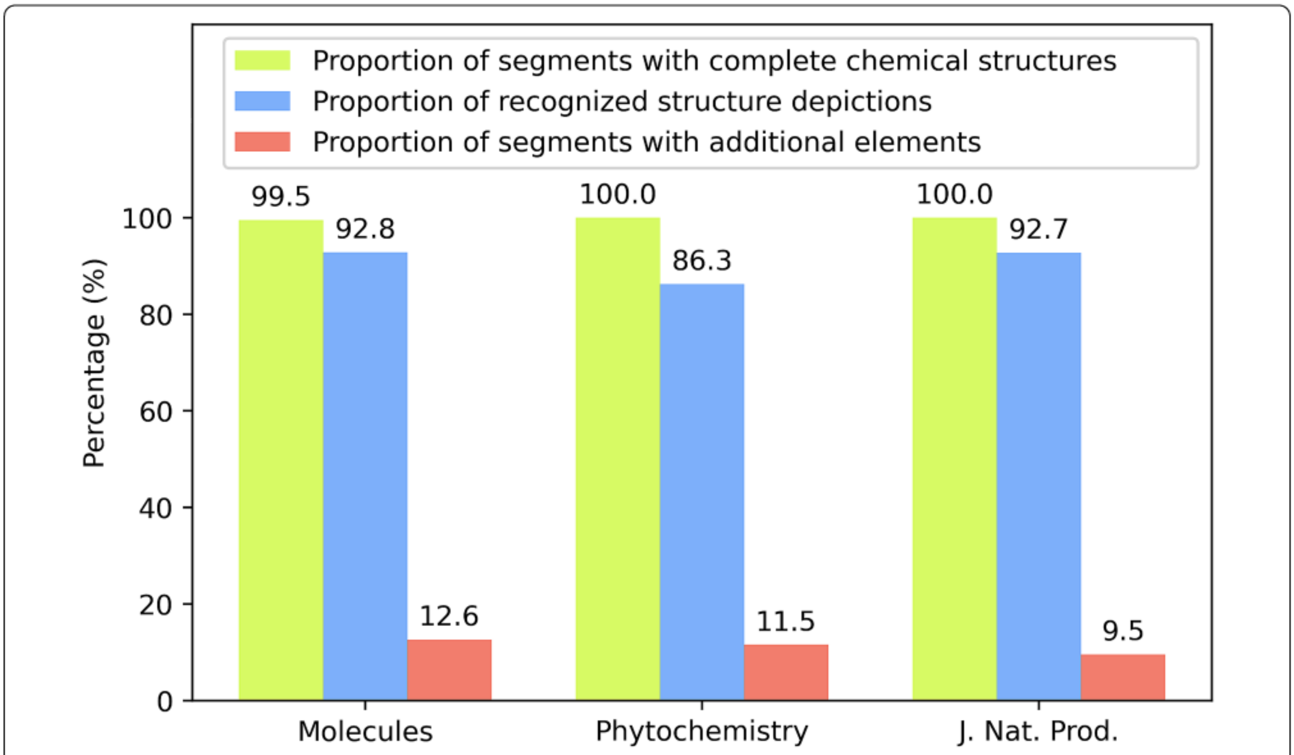


Fig. 4 Overview of the validation results of DECIMER Segmentation

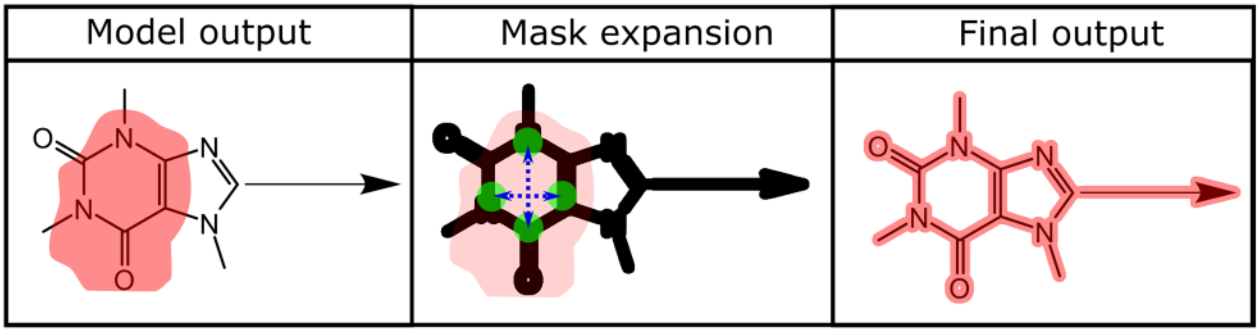


Fig. 5 Exemplary illustration of the wrong inclusion of a reaction arrow in the mask output

arrow may be excluded after the mask expansion if it is not too close to the structure. This is the advantage of choosing seed pixels in the center of the mask which is returned by the model. In an earlier version of DECIMER Segmentation, every pixel of the mask in the model output was included in the final output and the seeds for the expansion were set on the contours of the original mask. This led to wrongly included objects in above 20% of the segments because all objects close to the structure *and* all objects that were wrongly included in the model output were included in the final output. Hence, the mask expansion from seeds in the mask center led to significantly improved results.

During the mask expansion, the application of a binary dilation is necessary to turn the chemical structure depictions into connected objects. This can lead to nearby objects being connected with the structures. This could be addressed by using a smaller kernel for the dilation. The dilemma is that a smaller kernel leads to more cases where the structure is not one connected object which leads to incomplete expansion results. Hence, reducing the amount of wrongly included surrounding objects would necessarily lead to a reduction of the complete segments.

When processing pages parallelly on four threads, on average, the tool took about 1.89 min to process a single article with an average amount of 10.4 pages per article. The time required for processing depends on the number of pages and the number of chemical structures on each page. The numbers mentioned above correspond to an average processing time of 10.9 s per page.

Conclusion

The DECIMER Segmentation tool and the web implementation on decimer.ai for chemical image segmentation are a complete open-source implementation for the segmentation of chemical structure depictions from the published scientific literature.

With the help of deep learning, our method is capable of distinguishing between chemical structures and other content on a page. By applying the system to images, we can mine information from scanned documents which are not available in markup file formats. This allows us to extract information even from the old articles which are only available as scanned files. With the implemented mask expansion process, we are able to segment chemical structure diagrams from the publications completely in high quality.

Although the model was only trained on articles from the *Journal of Natural Products*, we were able to see that the application works well on publications from three different publishers. In future, the detection accuracy of the model can be improved further by training it on an increased amount of publications. In its current state, DECIMER Segmentation can reduce the workload for those who are responsible for the manual creation and curation of chemical databases immensely and could eventually contribute to the full automation of this task.

Abbreviations

API: Application Programming Interface; CNN: Convolutional Neural Network; CPU: Central Processing Unit; CSR: ChemSchematicResolver; DECIMER: Deep lEarning for Chemical IMagE Recognition; GB: Gigabyte; GPU: Graphics Processing Unit; JSON: JavaScript Object Notation; JPEG: Joint Photographic Experts Group; MDPI: Multidisciplinary Digital Publishing Institute; OCSR: Optical Chemical Structure Recognition; OSRA: Optical Structure Recognition Application; PDF: Portable Document Format; PNG: Portable Network Graphics; RAM: Random Access Memory; R-CNN: Region-Based Convolutional Neural Networks; VGG: Visual Geometry Group.

Acknowledgements

The authors would like to thank Ann-Kathrin Brüggemann and Lisa Sophie Kersten for their help in annotating the data.

Authors' contributions

KR and HOB developed the software and performed the analysis, MS developed the web application. CS and AZ conceived the project and supervised the work. All authors contributed to and approved the manuscript.

Funding

Open Access funding enabled and organized by Projekt DEAL. This work was supported by the Carl-Zeiss-Foundation and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 239748522 – SFB 1127 ChemBioSys.

Availability of data and materials

The DECIMER—Segmentation web app can be accessed at https://decimer.ai The code for DECIMER—Segmentation is available at https://github.com/Kohulan/DECIMER-Image-Segmentation. Project name: DECIMER. Project home page: https://decimer.ai. Operating system(s): Linux, MacOS and Windows 10. Programming language: Python 3. Other requirements: Python packages: Tensorflow 2.0 or above, pillow, opencv-python, matplotlib, scikit-image, imantics, IPython and pdf2image. License: MIT. Any restrictions to use by non-academics: Not applicable.

Competing interests

AZ is co-founder of GNWI—Gesellschaft für naturwissenschaftliche Informatik mbH, Dortmund, Germany.

Author details

[1] Institute for Inorganic and Analytical Chemistry, Friedrich-Schiller-University Jena, Lessingstr. 8, 07743 Jena, Germany. [2] Institute for Bioinformatics and Chemoinformatics, Westphalian University of Applied Sciences, August-Schmidt-Ring 10, 45665 Recklinghausen, Germany.

Received: 8 January 2021 Accepted: 22 February 2021
Published online: 08 March 2021

References

1. O'Boyle NM, Guha R, Willighagen EL et al (2011) Open data, open source and open standards in chemistry: the Blue Obelisk five years on. J Cheminform 3:1–15
2. Swain MC, Cole JM (2016) ChemDataExtractor: a toolkit for automated extraction of chemical information from the scientific literature. J Chem Inf Model 56:1894–1904
3. Krallinger M, Rabal O, Lourenço A, Oyarzabal J, Valencia A (2017) Information retrieval and text mining technologies for chemistry. Chem Rev 117:7673–7761
4. Rajan K, Brinkhaus HO, Zielesny A, Steinbeck C (2020) A review of optical chemical structure recognition tools. J Cheminform. https://doi.org/10.1186/s13321-020-00465-0
5. Filippov IV, Nicklaus MC (2009) Optical structure recognition software to recover chemical information: OSRA, an open source solution. J Chem Inf Model 49:740–743
6. Beard EJ, Cole JM (2020) ChemSchematicResolver: a toolkit to decode 2d chemical diagrams with labels and R-groups into annotated chemical named entities. J Chem Inf Model 60:2059–2072
7. Staker J, Marshall K, Abel R, McQuaw CM (2019) Molecular Structure extraction from documents using deep learning. J Chem Inf Model 59:1017–1029
8. Ronneberger O, Fischer P, Brox T (2015) U-net: convolutional networks for biomedical image segmentation. Lecture notes in computer science, p 234–241
9. Rajan K, Zielesny A, Steinbeck C (2020) DECIMER: towards deep learning for chemical image recognition. J Cheminform 12:65
10. Abadi M, Agarwal A, Barham P, et al (2016) TensorFlow: large-scale machine learning on heterogeneous distributed systems. arXiv:1603.04467 [cs.DC]
11. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, p 2961–2969
12. Abdulla W (2017) Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow. https://github.com/matterport/Mask_RCNN. Accessed 1 Dec 2020
13. Belval E. pdf2image. https://github.com/Belval/pdf2image. Accessed 9 Dec 2020
14. Dutta A, Zisserman A (2019) The VIA annotation software for images, audio and video. In: Proceedings of the 27th ACM international conference on multimedia. Association for Computing Machinery, New York, NY, p 2276–2279

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

BMC

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible] [illegible]					
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Sorokina *et al. J Cheminform* (2021) 13:2
https://doi.org/10.1186/s13321-020-00478-9

Journal of Cheminformatics

DATABASE

Open Access

COCONUT online: Collection of Open Natural Products database

Maria Sorokina*, Peter Merseburger, Kohulan Rajan, Mehmet Aziz Yirik and Christoph Steinbeck

Abstract

Natural products (NPs) are small molecules produced by living organisms with potential applications in pharmacology and other industries as many of them are bioactive. This potential raised great interest in NP research around the world and in different application fields, therefore, over the years a multiplication of generalistic and thematic NP databases has been observed. However, there is, at this moment, no online resource regrouping all known NPs in just one place, which would greatly simplify NPs research and allow computational screening and other *in silico* applications. In this manuscript we present the online version of the COlleCtion of Open Natural prodUcTs (COCONUT): an aggregated dataset of elucidated and predicted NPs collected from open sources and a web interface to browse, search and easily and quickly download NPs. COCONUT web is freely available at https://coconut.naturalproducts.net.

Keywords: Natural products, Database, NoSQL, MongoDB, Molecular similarity search, Molecular substructure search

Introduction

Natural products (NPs) have received constant attention from the scientific community due to their relevance in drug discovery, chemical ecology and molecular biology in general. In a recently published review on NPs databases [1] we inventoried over 120 natural products databases that have been published and used in the last 20 years. However, 16% of these are not available online anymore, 40% are commercial and their content cannot be easily accessed. The open resources are generally either specialized on a particular type of NPs, either lack annotations. For instance, the catalog of NPs from the ZINC database [2] is composed of over 80,000 entries, some of which can be purchased, but apart from their structure and that they are from natural origin, no additional information is provided. Super Natural II [3] is considered as the largest among all the NP databases, is accessible online in 2020, but it seems not to be maintained anymore and is mainly composed of compounds that can be purchased. Another recent database, NPAtlas [4], is constantly growing and extremely well annotated, but it is focusing on microbial NPs only. Another major NPs category, plant-produced compounds, also called phytochemicals, is available in several popular and well maintained databases, such as NuBBEDB [5], KnapSack [6], CMAUP [7] and TCM@Taiwan [8]. In addition to these relatively big databases, there is a plethora of smaller, more specialized NPs collections, such as FooDB [9], a user-friendly database hosting a relatively large number of NPs that are found in food. There is, therefore, a need for a generalistic NPs database, that will efficiently aggregate NPs information from various sources, improve its annotation and offer a pleasant user experience. With this ultimate goal in mind, we first assembled the most complete up-to-date COlleCtion of Open Natural ProdUcTs (COCONUT) that we have been continuously curating and annotating. Studies [10, 11] showed that fragments from NPs present in COCONUT have high diversity and structural complexity, which makes it, among other possible applications, a suitable source for drug discovery and can be included in drug design pipelines. Our next step was to make this data available to the scientific community as a full-fledged online natural

---

*Correspondence: maria.sorokina@uni-jena.de
Institute for Inorganic and Analytical Chemistry, University Friedrich-Schiller, Lessing Strasse 8, 07743 Jena, Germany

© The Author(s) 2021. This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

products database, maintained at https://coconut.naturalproducts.net.

The COCONUT database is free and open to all users and there is no login required to access it. Its web interface allows diverse simple searches (e.g. by molecule name, InChI, InChI key, SMILES, drawn structure, molecular formula), advanced search by molecular features, together with substructure and similarity searches. Users can also download the whole dataset or search results in different formats. The database can be queried programmatically via a REST API, which facilitates COCONUT integration in workflows. The web interface, the back-end and the database are deployed as Docker containers, making it easily portable for hosting other sets of NPs and to be deployed on local installations.

Construction and content

COCONUT is assembled from a large number of chemical data sources (Table 1), from which NPs have been thoroughly extracted, curated, processed and annotated. The resulting NPs collection is presented within the full-fledged chemical database developed specially for this purpose (Fig. 1). Data curation and processing steps, together with the construction of the web interface and the description of available features are described below.

Data provenance, model and content

COCONUT data has been extracted from 53 various data sources and several manually collected from literature sets, as shown in Table 1. In the current COCONUT release (October 2020), there are 406,076 unique "flat" (with no stereochemistry) NPs, and a total of 730,441 NPs where stereochemistry has been preserved when available.

Every molecule collected from external sources passed a quality control and a registration procedure, where its structure is checked for size (between 5 and 210 heavy atoms), connectivity (only the biggest connected structure is kept), presence of pseudo-atoms, if implicit and explicit hydrogens are correct, and if the bonds are correct and the valences are conserved. The Kekulé representation is also assigned to the aromatic systems of each compound. At this step, tautomers and ionisation states are standardized following the ChEMBL chemical structure curation pipeline [12].

Then, NPs from different provenance are unified based on the identity of their InChI keys without stereochemistry. This unification step is performed without stereochemistry, as in different data sources stereochemistry is not systematically present and can also be represented differently. When available, the original molecular structure with stereochemistry is preserved and can be accessed for each NP entry.

The authors are well aware that different stereoisomers of a compound can have very different biological activity. The procedure described above was a necessary step to create a unified resource out of distributed databases of varying quality. Further curation will gradually improve stereochemical assignments and linkage to original source articles.

Each unique NP is then assigned a unique identifier, composed of the "CNP" prefix and 7 digits. An automatic curation for NP metadata is performed, comprising the retrieval of its official name, synonyms, cross-references

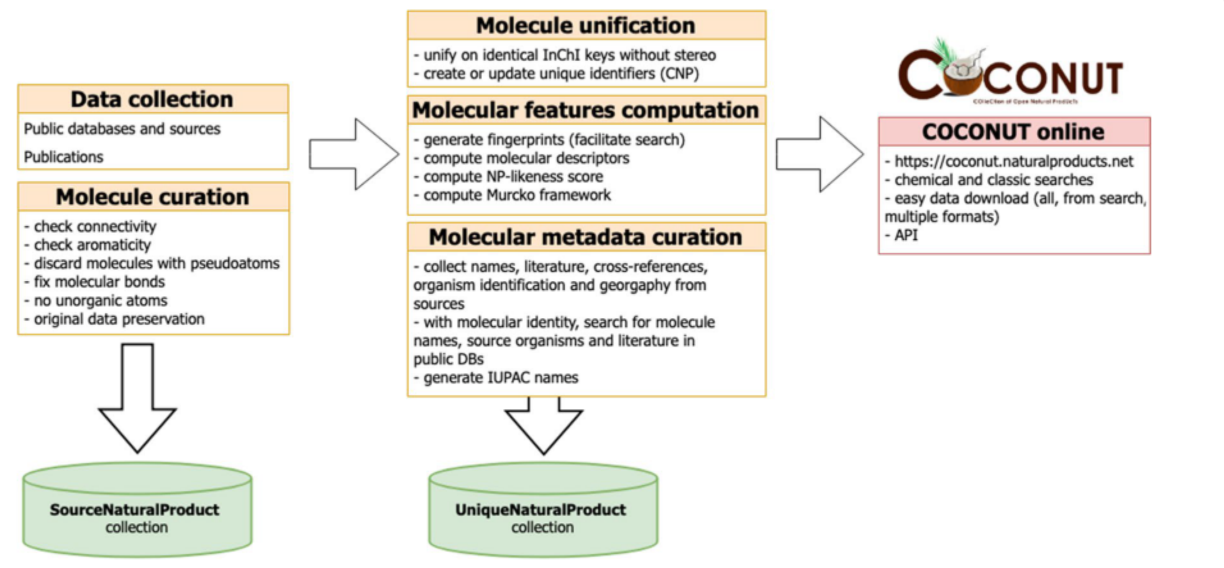

Data collection
Public databases and sources
Publications
Molecule curation
- check connectivity
- check aromaticity
- discard molecules with pseudoatoms
- fix molecular bonds
- no unorganic atoms
- original data preservation
SourceNaturalProduct collection
Molecule unification
- unify on identical InChI keys without stereo
- create or update unique identifiers (CNP)
Molecular features computation
- generate fingerprints (facilitate search)
- compute molecular descriptors
- compute NP-likeness score
- compute Murcko framework
Molecular metadata curation
- collect names, literature, cross-references, organism identification and georgaphy from sources
- with molecular identity, search for molecule names, source organisms and literature in public DBs
- generate IUPAC names
UniqueNaturalProduct collection
COCONUT
COCONUT online
- https://coconut.naturalproducts.net
- chemical and classic searches
- easy data download (all, from search, multiple formats)
- API

Fig. 1 Construction and curation of COCONUT online

Table 1 Public databases and datasets from which COCONUT was assembled

Database name ("NP" suffix is added to database name when only a subset of it contains natural products)	Number of entries integrated in COCONUT from the source	Most resent publication or resource URL
AfroCancer	365	[33]
AfroDB	874	[34]
AfroMalariaDB	252	[35]
AnalytiCon Discovery NPs	4908	[36]
BIOFACQUIM	400	[37]
BitterDB	625	[38]
Carotenoids Database	986	[39]
ChEBI NPs	14603	[20]
ChEMBL NPs	1585	[21]
ChemSpider NPs	9027	[40]
CMAUP (cCollective molecular activities of useful plants)	20868	[7]
ConMedNP	2504	[41]
ETM (Ethiopian Traditional Medicine) DB	1633	[42]
Exposome-explorer	478	[43]
FooDB	22123	[9]
GNPS (Global Natural Products Social Molecular Networking)	6740	[44]
HIM (Herbal Ingredients in-vivo Metabolism database)	962	[45]
HIT (Herbal Ingredients Targets)	470	[46]
Indofine Chemical Company	46	[47]
InflamNat	536	[48]
InPACdb	122	[49]
InterBioScreen Ltd	67291	[50]
KNApSaCK	44422	[6]
Lichen Database	1453	[51]
Marine Natural Products	11880	[52]
Mitishamba database	1010	[53]
NANPDB (Natural Products from Northern African Sources)	3914	[54]
NCI DTP data	404	[55]
NPACT	1453	[56]
NPASS	27424	[57]
NPAtlas	23914	[4]
NPCARE	1362	[58]
NPEdia	16166	[59]
NuBBEDB	2022	[5]
p-ANAPL	467	[60]
Phenol-explorer	681	[61]
PubChem NPs	2828	[27]
ReSpect	699	[62]
SANCDB	592	[63]
Seaweed Metabolite Database (SWMD)	348	[64]
Specs Natural Products	745	[65]
Spektraris NMR	242	[66]
StreptomeDB	6058	[67]
Super Natural II	214420	[3]
TCMDB@Taiwan (Traditional Chinese Medicine database)	50862	[8]
TCMID (Traditional Chinese Medicine Integrated Database)	10552	[68]
TIPdb (database of Taiwan indigenous plants)	7742	[69]

Table 1 (continued)

Database name ("NP" suffix is added to database name when only a subset of it contains natural products)	Number of entries integrated in COCONUT from the source	Most resent publication or resource URL
TPPT (Toxic Plants–PhytoToxins)	1483	[70]
UEFS (Natural Products Databse of the UEFS)	481	[71]
UNPD (Universal Natural Products Database)	156865	[72]
VietHerb	4759	[73]
ZINC NP	67327	[74]
Manually selected molecules	61	x

to other major chemical databases. Then, a range of molecular properties, descriptors and fingerprints (full list in Table 2) are computed using the in-build CDK [13] libraries. As the number of the computed properties is quite big (73 fields in each document corresponding to one unique NP), only a selected fraction of them is displayed on the COCONUT web interface. Finally, the first round of automatic curation of NP metadata, in particular the molecular name synonyms, cross-references with other major chemical databases, correction of the literature references (PubMed identifiers and DOIs) and taxonomy is performed. All original data, unified NPs and the derived and calculated information are stored in MongoDB. The chemical classification of all NPs in COCONUT is performed with ClassyFire [14], and, when available, is displayed in the corresponding section of the compound page. ClassyFire provides a hierarchical chemical classification of chemical compounds and enables grouping NPs by their chemical class. Additionally, frameworks facilitating NPs analyses for their chemical and therapeutic properties are computed for NPs, such as Murcko frameworks [15], Ertl Functional Groups [16] and deepSMILES [17]. DeepSMILES is an adaptation of SMILES for use in deep machine learning of chemical structures. Due to the increased usage of deep learning in chemistry, it is indeed interesting to provide this new chemical representation type pre-computed for NPs.

Last, the annotation level of each NP in COCONUT is computed. It is a 5-star-based system, where 1 star is the lowest annotation quality (no verified common name, no taxonomic provenance annotation, no literature reference and no trusted data source) and 5 stars is the highest quality, with all the intermediate annotation qualities reflected by 2, 3 and 4 stars. Only ChEBI [18], KNApSAcK [6], ChEMBL [19], CMAUP [7], NPAtlas [4] and, of course, the manually picked data are considered as trusted data provenances. For example, caryolivine (CNP0235854) has a 5-stars annotation because it has a verified common name, is known to be produced by *Caryomene olivascens*, a plant, is associated to a scientific publication and is present in KNApSAcK. The COCONUT NP CNP0330764 has no verified common name, only a computed IUPAC one, but is present in CMAUP and is known to be produced by a range of plants, therefore it's annotation level is 3. The annotation level is represented with stars on the NP page.

Natural product naming

NPs common names in COCONUT have been retrieved, when available, from their databases of origin. The remaining NPs have been searched by InChI in major chemical databases (PubChem, ChEMBL and ChEBI) and common names and synonyms were retrieved when occurences the compound were found. Additionally to this, IUPAC names were computed for all COCONUT NPs using ChemAxon's MolCovert [20], to add nomenclature homogeneity to the dataset. Furthemore, the IUPAC names are used as the main NP name when no official chemical name has been found nor in the original sources, nor by searching big chemical databases. Therefore, all NPs in COCONUT have at least one molecular name.

Computed molecular features

Figure 2 demonstrates the distributions and relationships of a small selection of computed molecular features within COCONUT. Sugar moieties occur frequently in NPs and have an important impact on their bioactivities and physicochemical properties. However, they are often redundant and therefore obstruct the study of the aglycon. For this reason, NPs in COCONUT have been analysed for sugar moieties presence, and a deglycosylated structure representation was made available in the database. Sugar moieties manipulations were performed using the Sugar Removal Utility [21]. To track their influence on other features, their absence and presence are colour-mapped (no sugar moiety in the molecular structure in blue, and the presence of at least one sugar moiety in orange). The wide molecular weight range is typical for NPs; it is, however, interesting to notice its

Table 2 Molecular features present in COCONUT and in their disponibility in the web interface

Natural product feature	Field name in MongoDB—uniqueNaturalProduct collection	Displayed on the website
COCONUT identifier	coconut_id	x
List of SMILES with stereochemistry and their provenance	absolute_smiles	x
AlogP (Ghose-Crippen LogKow)	alogp	x
AlogP2	alogp2	x
AMR—molar refractivity	amralogp	
Annotation level of the NP (from 1 to 5)	annotationLevel	x
BCUT decriptor (Eigenvalue based)	bcutDescriptor	
Bond number in the NP	bond_count	x
BPol descriptor	bpol	x
CAS number	cas	x
List of literature DOIs mentioning the NP	citationDOI	x
Boolean—if the molecule contains linear sugars	contains_linear_sugars	
Boolean—if the molecule contains circular sugars	contains_ring_sugars	
Boolean—if the molecule contains sugar moieties	contains_sugar	
deepSMILES	deep_smiles	x
Eccentric Connectivity Index Descriptor	eccentricConnectivityIndexDescriptor	x
List of tl Functional Groups	ertlFuntionalFragments	
List of Ertl Functional Groups in pseudo SMILES	ertlFunctionalFragmentsPseudoSmiles	
FMF descriptor	fmfDescriptor	x
List of data sources containing the NP	found_in_databases	
Fragment complexity descriptor	fragmentComplexityDescriptor	x
List of circular fragments (molecular signatures) of the deglycosylated NP	fragments	
List of circular fragments (molecular signatures) of the whole NP	fragmentsWithSugar	
Fractional CSP3 Descriptor (non-flatness of a molecule)	fsp3	x
List of continents and regions where the organism producing the NP is found	geoLocation	
Gravitational index descriptor (heavy atoms only)	gravitationalIndexHeavyAtoms	
Hydrogen bond acceptor count	hBondAcceptorCount	
Hydrogen bond donor count	hBondDonorCount	
Number of heavy atoms in the NP	heavy_atom_number	x
Hybridization Ratio Descriptor (fraction of sp3 carbons to sp2 carbons)	hybridizationRatioDescriptor	
InChI (without stereochemistry)	inchi	x
InChI key	inchikey	x
IUPAC name	iupac_name	x
First kappa shape index	kappaShapeIndex1	
Second kappa shape index	kappaShapeIndex2	
Third kappa shape index	kappaShapeIndex3	
Number of failures in the Lipinski rule of 5	lipinskiRuleOf5Failures	x
LogP descriptor (Mannhold version)	manholdlogp	
Maximal number of rings in the NP	max_number_of_rings	x
Minimal number of rings in the NP	min_number_of_rings	x
Molecular formula	molecular_formula	x
Molecular weight	molecular_weight	x
Murcko Framework	murcko_framework	x
Official name (when available)	name	x
NP-likeness score	npl_score	x
NP-likeness score computed on the glycosylated molecule	npl_sugar_score	
Total number of carbons	number_of_carbons	x
Total number of nitrogens	number_of_nitrogens	
Total number of oxygens	number_of_oxygens	

Table 2 (continued)

Natural product feature	Field name in MongoDB—uniqueNaturalProduct collection	Displayed on the website
Number of sporo atoms	numberSpiroAtoms	
Petitjean Number Descriptor	petitjeanNumber	x
Petitjean geometrical shape index	petitjeanShapeGeom	
Petitjean geometrical shape index	petitjeanShapeTopo	
PubChem fingerprint in MongoDB BinData format	pubchemBits	
PubChem fingerprint as list of booleans	pubchemFingerprint	
SMILES with all hydrogen explicit	smiles	x
Number of heavy atoms of the deglycosylated moiety	sugar_free_heavy_atom_number	
SILES of the deglycosylated moiety	sugar_free_smiles	
Total atom number of the deglycosylated moiety	sugar_free_total_atom_number	
List of synonym names of the NP	synonyms	x
List of NCBI taxonomy identifiers of organisms producing the NP	taxid	
List of organisms producing the NP in text form	textTaxa	
Topological polar surface area descriptor	topoPSA	
Total atom count in the NP (incuding hydrogens)	total_atom_number	
Fractional polar surface area descriptor	tpsaEfficiency	
Unique SMILES (CDK)	unique_smiles	x
Volume descriptor	vabcDescriptor	
Vertex adjacency information	vertexAdjMagnitude	
Wiener Path Number	wienerPathNumber	x
Wiener Polarity Number	wienerPolarityNumber	
XLogP descriptor	xlogp	x
Cross-references to toher chemical ressources	xrefs	x
Zagreb Index	zagrebIndex	x
Chemical superclass of the NP computed with ClassyFire	chemicalSuperClass	x
Chemical class of the NP computed with ClassyFire	chemicalClass	x
Chemical subclass of the NP computed with ClassyFire	chemicalSubClass	x
Direct parent in the chemical ontology of the NP computed withClassyFire	directParentClassification	x

correlation with the number of oxygen atoms in the molecule, regardless of the presence and absence of sugar. Another interesting correlation to be noted is between the molecular weight and the nitrogen atom number in sugar-free molecules. The NP-likeness score [22], trained on high-quality NPs dataset and computed with NaPLeS [23], which was trained on high-quality NPs dataset, has a typical distribution for an NPs set, where most molecules have a positive score.

At this point, an additional NPs curation step has been performed, due to the possible inconsistency in genuine NPs of one of the used sources, SuperNatural II. NPs that are not occuring in other datasets used to assemble COCONUT, but only in SuperNatural II, have been thoroughly tested. To be kept in COCONUT and be considered as a genuine or predicted NP, such a molecule has to have a strictly positive NP-likeness score, be classified as a NP by NPclassifier [24], a deep neural network-based structural classification tool specialised in NPs or have a sugar moiety in its structure. The 24,880 molecules from SuperNatural II that didn't pass this additional quality control have been removed from COCONUT, until further proof of their natural provenance.

Counting rings in a molecule can become a complex task, as the outer perimeter of two fused rings can be counted as one big ring. With more condensed rings, the number of fused ring perimeters (aka as the set of all rings) can grow steeply. In Fig. 2, only the minimal ring count (the minimal cycle base) is represented.

Natural product annotation

The particularity of NPs, opposed to synthetic molecules, which constitute the biggest part of chemical databases, resides in their production by living organisms. Therefore, in addition to their structure and computable structural properties, NPs need to be annotated with at least one literature reference, mentioning where, when and from which organism the NP was isolated. As

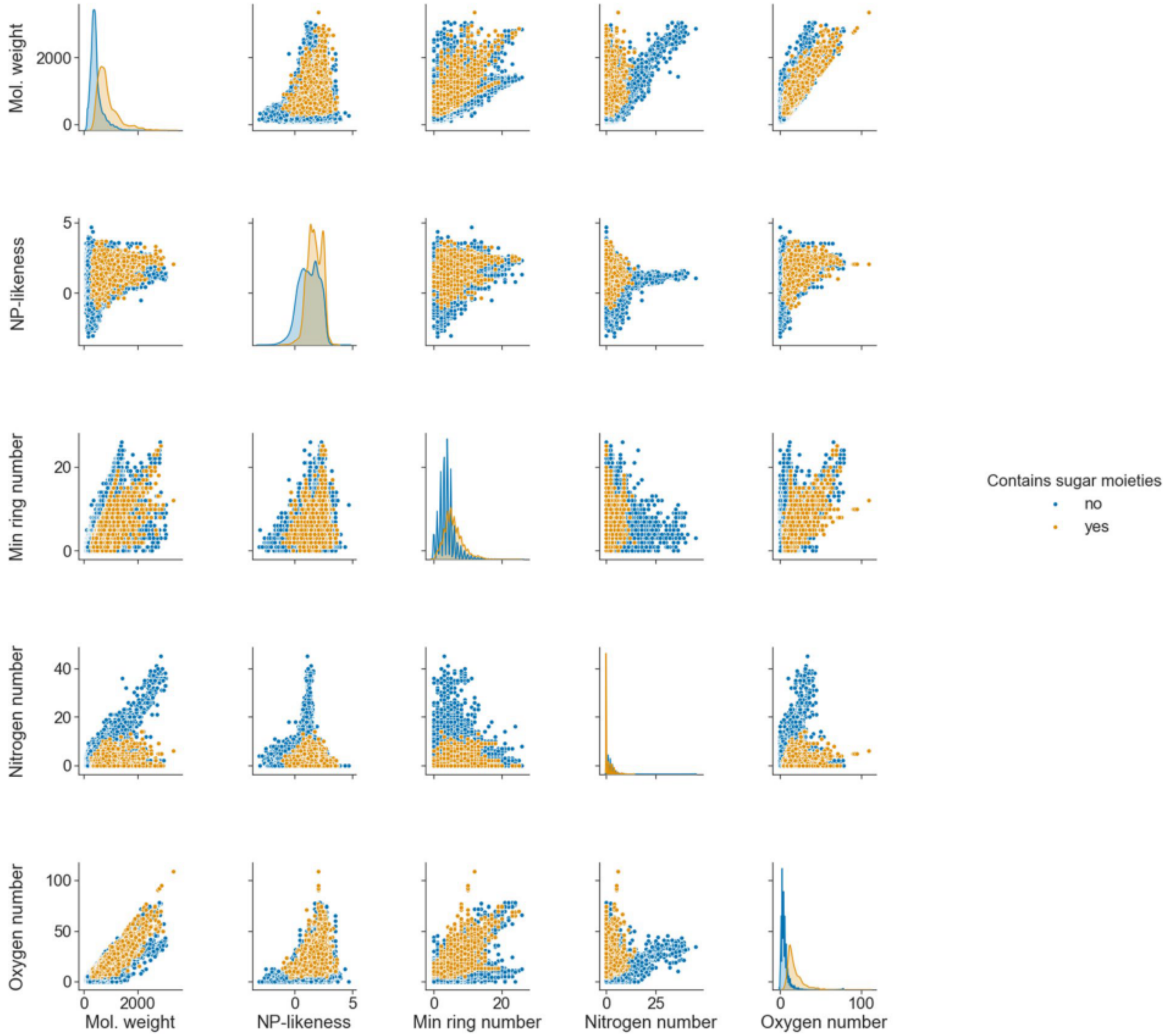


Fig. 2 Pair plot of a selection of five of the molecular features available in COCONUT. Colour mapping corresponds to the presence (yellow) and absence (blue) of glycosidic moieties in the molecular structures of the NPs in COCONUT

a direct consequence, an NP entry should be associated with at least one organism, preferentially with an NCBI taxonomy identifier and the geographic location where the organism is naturally occuring or has been collected. Regrettably, this metadata is often omitted in public databases from which COCONUT was assembled. Therefore, only 31.5% (134,379) of NPs in COCONUT are annotated with at least one organism taxa, for 15.4% (66,068) of NPs the geographic location (on the continent level) of the organism occurence or collection is known and only 16.6% (70,730) of NPs have at least one literature reference. These numbers combine both the retrieval of the original NP annotations from their sources and our efforts to retrieve more extensive information from major trusted chemical databases, PubChem [25], ChEMBL [19], ChEBI [18], CMAUP [7] and KnapSacK [6]. Despite our efforts, most of the links between the original publication of the structure elucidation of an NP and its reference, source organism and its geographical location are still missing. A possible solution to fill these gaps is manual curation, but the amount of data in COCONUT is redhibitory for even considering this approach. Another solution is to use unsupervised machine learning for optical recognition approaches, to parse modern peer-reviewed literature and books to re-establish links between NPs structures and their provenance.

We analysed the taxonomic classification of known NPs producers together with overlaps in NPs production between superkingdom for the 31% of the NPs in COCONUT for which the provenance organism is known (Fig. 3). Here are distinguished five taxonomic categories: plants, bacteria, fungi, animals and marine. The last one is not a proper monoclade classification, but rather reflects a group of organisms that are found only in marine and oceanic environments, and therefore can overlap in terms of its species and NP content with other categories, which are more stringent taxonomically. A large part (65%) of these annotated NPs are produced only by plants, and only very few (0.5%) are from animal origin. Main overlaps in terms of NP production between the taxonomic kingdoms are between plants and marine organisms (which is unsurprising, as there can be real plants among the marine entities) and surprisingly between plants and fungi. The other overlaps between taxonomic kingdoms are not as significant. It needs to be pointed out here that multicellular organisms, such as plants, animals and some of the fungi are most of the time in symbiosis with microorganisms, in particular bacteria. Therefore, NPs isolated from such a multicellular organism can be synthesized and secreted by their symbionts or microbiomes, and therefore mistakenly assigned to an incorrect organism.

The geographic location of the collection or the natural presence of the NP-producing organism is a piece of information that is even more difficult to obtain. Nowadays, a range of organisms, and in particular plants, can be found in different parts of the planet due to globalisation and their success in human consumption (e.g. garlic, tomatoes, curcuma or ginger). It is, therefore, difficult, if not impossible, to determine their original provenance. Also, the geographical information is often omitted in literature and most NPs databases. When available, the geographical provenance is stored in the MongoDB dump of COCONUT, but not displayed on the website.

For NPs where geographical information is available, it appears that most of them are produced by

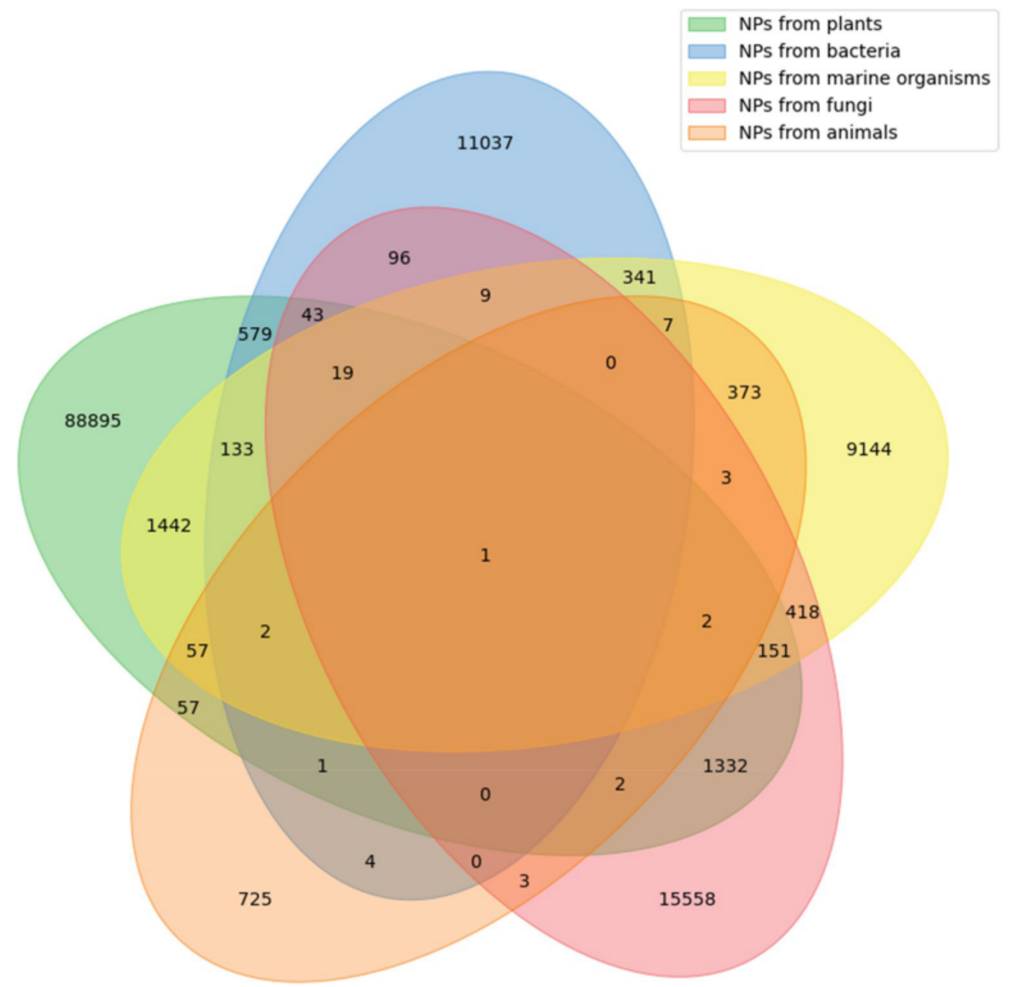


Fig. 3 Overlap of NPs taxonomic provenance in COCONUT

organisms that have been isolated in Asia (Fig. 4). This bias is introduced by the intensive study by scientists of the traditional Chinese and Indian medicines and by the big efforts in isolation and elucidation of NPs from medicinal plants. NPs from the African continent are also well represented in COCONUT (Fig. 4), mainly due to the scientific interest in African traditional medicines and African biodiversity. There is, for now, no data from the biodiversity of the Australian continent, and only very little data for NPs isolated from endemic European organisms. NPs from the Americas are mainly extracted and solved while Brazilian and Mexican biodiversity exploration. Only a few NPs are present in more than one continent, mainly in Asia and Africa, and the overlap values are biased by the very different NPs set sizes between the different continents.

Web interface and technical specificities

All COCONUT data is stored with MongoDB, a cross-platform document-oriented NoSQL database program. The smallest unit in MongoDB is a document, composed of key and value pairs that are similar to JSON objects. Documents of the same nature are organized in collections, which are the equivalent of the SQL-based databases tables. MongoDB is particularly adapted to big and complex data, supports multiple indexing, including text indexing allowing enhanced text search in text-indexed fields and contains a wide range of in-build search and analysis functions.

Two major collections are present in the COCONUT database: SourceNaturalProduct, which contains the original NPs data collected from the open sources, and UniqueNaturalProduct, the unified and curated collection of NPs. The full version of COCONUT with all the calculated features can be accessed as a MongoDB dump in the Downloads section of the website. Requests for displaying additional crucial features in the web interface and making them searchable through the advanced search interface are welcome via the COCONUT GitHub tracker (see below).

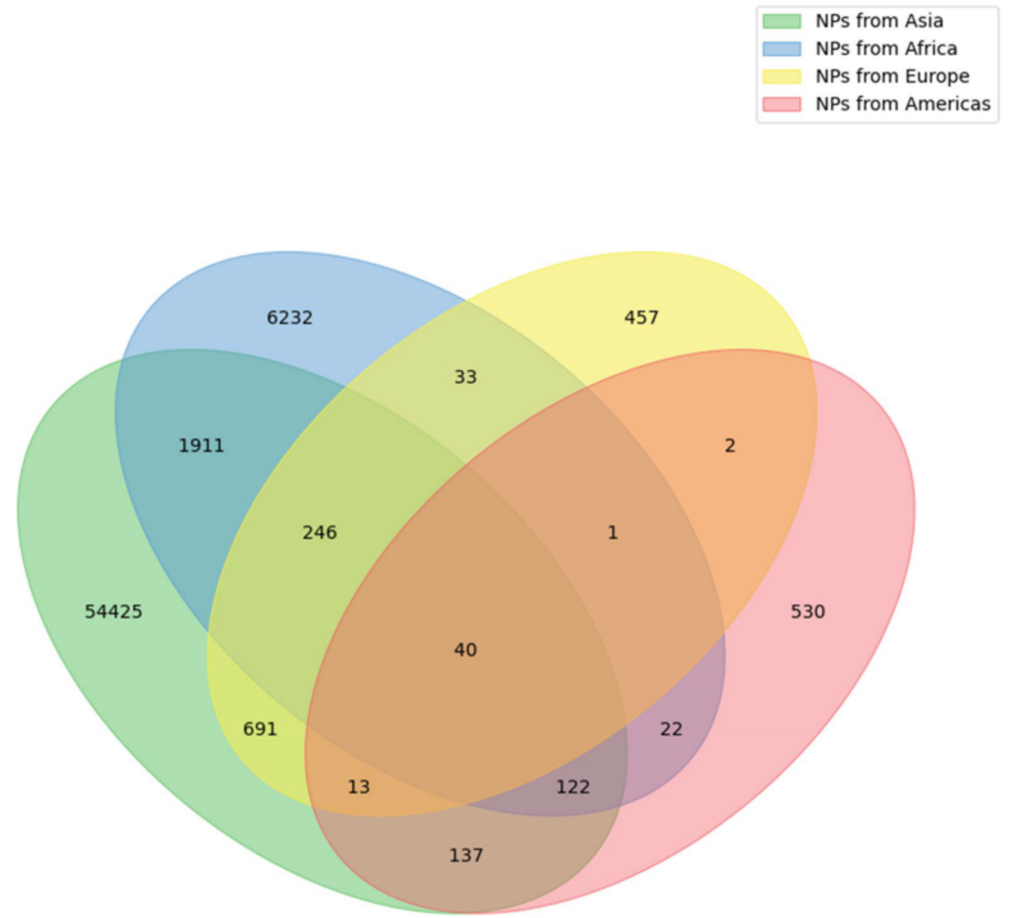


Fig. 4 Overlap of NPs geographic provenance in COCONUT

The COCONUT online front-end is developed entirely with React.js [26], a JavaScript library to build responsive and efficient user interfaces. The OpenChemLib library [27] is used to handle the chemical editor for the search functions. The COCONUT back-end, allowing to process the front-end requests and to communicate with the database is written in Kotlin and Java 11 using the Spring framework. The CDK [13] library is used to process chemical information and formats.

COCONUT web interface, back-end and database are entirely Dockerised, allowing a quick and easy deployment on local servers and cloud. All the code, for both front-end and back-end, is available on GitHub (https://github.com/mSorok/NaturalProductsOnline).

Searching the database

COCONUT online has been developed to be a full-fledged chemical database and in particular to fit the NPs structural and annotational particularities, with all the subsequent functions. At the moment, the chemical search is uncommon with MongoDB, therefore several approaches have been implemented to run molecular substructure and similarity searches.

Simple search

The so-called "simple" search can be performed using the header search bar. The users can enter there molecule names (e.g. "curcumin"), SMILES, InChI, InChi key, COCONUT ids and molecular formulas. Name search uses native MongoDB text indexing, allowing fuzzy, flexible search in the "name" and "synonyms" fields. The input string type is first identified using regular expressions, then the DB is queried against the appropriate fields, and the result, when exists, is returned to the front-end.

Substructure search implementation

Searching for an exact substructure in a MongoDB database of molecules appears to be surprisingly easy. Each molecule in the database needs to have their fingerprints of choice (in COCONUT are used the PubChem fingerprints) to be precomputed and stored as a list of bytes (BinData type in MongoDB). The query molecule (substructure) then needs to have its fingerprint to be also computed and to be matched against the database using the $allBitsSet function [28]. This native to MongoDB function allows to select documents in a collection where a BinData field has all the query bits set to "on" (but can have bits set to "on" that are not present in the query). To confirm the substructure match, the user can select between the default Ullmann [29], the Vento-Foggia [30] and the depth-first (DF) [31] pattern matching approaches, all performed using the CDK in-build algorithms. These three pattern matching techniques tend to, generally, return very similar results, the difference between them lying rather in their approach to matching substructures, therefore the usage of the default, Ullmann, method is to be privileged by users unfamiliar with the intricacies of such matching.

Similarity search implementation

Similarity search with MongoDB was implemented following the excellent ChEBML blog post on LSH-based similarity search in MongoDB [32] and adapted to Java, Kotlin and Spring data. In this approach, the MongoDB aggregation framework is used to perform inverted indexing search against PubChem fingerprints stored in a separate table and referencing COCONUT identifiers that contain the molecular features encoded by each bit.

Advanced search

The advanced search supports searching for NPs in COCONUT according to a range of parameters, such as molecular formula, molecular descriptor values, number of rings, type of sugar moieties present in them, etc.

Querying COCONUT through the API

A REST API has been developed for COCONUT online in order to permit programmatic querying and facilitate its integration in workflows. It relies on Kotlin API functionalities and it's usage, together with some examples, is described in detail in the documentation section of the website (https://coconut.naturalproducts.net/documentation).

Documentation

Complete documentation describing COCONUT, its data and functionalities are available at the documentation section of the website https://coconut.naturalproducts.net/documentation.

Utility and discussion

The online COCONUT database is an open tool for researchers in the natural products community. COCONUT is the biggest collection of NPs in 2020 and the data it contains already benefits researchers in NPs with various aims, such as biodiversity research and drug discovery. The web interface allows querying and parsing the data collection in various, chemically relevant ways with adequate performance. It is also the first big chemical database using MongoDB as a storage management system.

A wide range of molecular descriptors are pre-computed and literature, producer taxonomy and their geography are as much annotated as currently possible without extensive manual curation. The web database can be searched in multiple ways, by molecular structure, by compound name and by molecular features, making this repository a complete chemical database. The user interface is modern and easy to use. Besides, the whole content of COCONUT is available for download in multiple formats.

In the close future, COCONUT will support user registration to enable user-driven NPs curation and submission and will undergo a better data annotation, in particular regarding the organisms that are producing the NPs, their geography and the corresponding literature, using deep learning approaches.

Feedback

Bugs, annotation issues and requests of new COCONUT entries or re-annotation of existing ones can be reported at the project issues tracker (https://github.com/mSorok/NaturalProductsOnline/issues). Suggestions for new features are also welcome.

Availability

All COCONUT data, code to process raw NPs data, data quality control and annotation, and the code for the font- and the back-end of the COCONUT online website are freely available without any restriction. The latest COCONUT data, as MongoDB full dump can be downloaded at https://coconut.naturalproducts.net/download. Code for data assembly, processing and quality control process codes is available on GitHub at https://github.com/mSorok/COCONUT. The code for the front-end and back-end is also available on GitHub at https://github.com/mSorok/NaturalProductsOnline.

Conclusions

COCONUT is the largest open collection of elucidated and predicted NPs at this time. It has a great potential of being of particular importance for the NPs research community as it gathers most of open NPs knowledge in one single place, and makes it easily accessible and queryable.

The final aim of COCONUT is to provide to the scientific community NPs structures and their provenance, i.e.. organisms that synthesize them and geographic location of the latter. However, a lot of data curation, in particular using new generation deep learning-based methods of extracting information from publications and books, together with website functionalities developments are still need to be done for COCONUT, but the database as it is now is already an important tool to facilitate NPs and medicinal chemistry research.

Abbreviations

NPs: Natural products; COCONUT: COlleCtion of Open Natural ProdUcTs; CDK: Chemistry Development Kit; IUPAC: International Union of Pure and Applied Chemistry; API: Application Programming Interface; REST: REpresentational State Transfer; DOI: Digital Object Identifier; JSON: JavaScript Object Notation; (No)SQL: (non-/not only) Structured Query Language.

Acknowledgements

Not applicable.

Authors' contributions

MS coordinated the study, participated in the implementation of the front-end and the backend, collected; processed and curated the COCONUT data and wrote the manuscript. PM participated in the implementation of the front-end and the back-end of COCONUT online. KR and MAY helped with COCONUT data curation. KR designed the COCONUT logo. CS designed and supervised the study. All authors read and approved the final manuscript.

Funding

Open Access funding enabled and organized by Projekt DEAL. This work was supported by the German Research Foundation within the framework CRC1127 ChemBioSys.

Availability of data and materials

the source code of the web interface and the back-end is available on GitHub at https://github.com/mSorok/NaturalProductsOnline. The data was curated and processed using the COCONUT code suite available on GitHub at https://github.com/mSorok/COCONUT. All COCONUT data can be accessed on the website at https://coconut.naturalproducts.net/ and downloaded entirely or partially in several formats (MongoDB dump, SDF and SMI (SMILES)).

Competing interests

The authors declare no competing interests.

Received: 9 September 2020 Accepted: 23 November 2020
Published online: 10 January 2021

References

1. Sorokina M, Steinbeck C (2020) Review on natural products databases: where to find data in 2020. J Cheminform 12:20. https://doi.org/10.1186/s13321-020-00424-9 **[cito:citesForInformation]**
2. ZINC natural products subset. https://zinc15.docking.org/substances/subsets/natural-products/. Accessed 19 Nov 2020 **[cito:usesDataFrom]**
3. Banerjee P, Erehman J, Gohlke B-O, Wilhelm T, Preissner R, Dunkel M (2015) Super Natural II—a database of natural products. Nucleic Acids Res 43:D935–D939. https://doi.org/10.1093/nar/gku886 **[cito:usesDataFrom]**
4. van Santen JA, Jacob G, Singh AL, Aniebok V, Balunas MJ, Bunsko D et al (2019) The natural products atlas: an open access knowledge base for microbial natural products discovery. ACS Cent Sci 5:1824–1833. https://doi.org/10.1021/acscentsci.9b00806 **[cito:usesDataFrom]**
5. Pilon AC, Valli M, Dametto AC, Pinto MEF, Freire RT, Castro-Gamboa I (2017) NuBBEDB: an updated database to uncover chemical and biological information from Brazilian biodiversity. Sci Rep 7:7215. https://doi.org/10.1038/s41598-017-07451-x **[cito:usesDataFrom]**
6. Nakamura K, Shimura N, Otabe Y, Hirai-Morita A, Nakamura Y, Ono N (2013) KNApSAcK-3D: a three-dimensional structure database of plant metabolites. Plant Cell Physiol 54:e4–e4. https://doi.org/10.1093/pcp/pcs186 **[cito:usesDataFrom]**
7. Zeng X, Zhang P, Wang Y, Qin C, Chen S, He W (2019) CMAUP: a database of collective molecular activities of useful plants. Nucleic Acids Res 47:D1118–27 **[cito:usesDataFrom]**
8. Chen CY-C (2011) TCM Database: the World's Largest Traditional Chinese Medicine Database for Drug Screening in silico. PLOS ONE 6:e15939. https://doi.org/10.1371/journal.pone.0015939 **[cito:usesDataFrom]**
9. FooDB. http://foodb.ca/. Accessed 3 Oct 2019 **[cito:usesDataFrom]**
10. Chávez-Hernández AL, Sánchez-Cruz N, Medina-Franco JL (2020) A fragment library of natural products and its comparative chemoinformatic

characterization. Mol Inform 39:2000050. https://doi.org/10.1002/minf.202000050 **[cito:citesForInformation]**

11. Chávez-Hernández AL, Sánchez-Cruz N, Medina-Franco JL (2020) Fragment library of natural products and compound databases for drug discovery. Biomolecules 10:1518. https://doi.org/10.3390/biom10111518 **[cito:citesForInformation]**
12. Bento AP, Hersey A, Félix E, Landrum G, Gaulton A, Atkinson F (2020) An open source chemical structure curation pipeline using RDKit. J Cheminform 12:51. https://doi.org/10.1186/s13321-020-00456-1 **[cito:usesMethodIn]**
13. Willighagen EL, Mayfield JW, Alvarsson J, Berg A, Carlsson L, Jeliazkova N (2017) The Chemistry Development Kit (CDK) v2.0: atom typing, depiction, molecular formulas, and substructure searching. J Cheminform 9:33. https://doi.org/10.1186/s13321-017-0220-4 **[cito:usesMethodIn]**
14. Djoumbou Feunang Y, Eisner R, Knox C, Chepelev L, Hastings J, Owen G (2016) ClassyFire: automated chemical classification with a comprehensive, computable taxonomy. J Cheminform 8:61. https://doi.org/10.1186/s13321-016-0174-y **[cito:usesMethodIn]**
15. Bemis GW, Murcko MA (1996) The properties of known drugs. 1. Molecular frameworks. J Med Chem 39:2887–2893. https://doi.org/10.1021/jm9602928 **[cito:usesMethodIn]**
16. Fritsch S, Neumann S, Schaub J, Steinbeck C, Zielesny A (2019) ErtlFunctionalGroupsFinder: automated rule-based functional group detection with the Chemistry Development Kit (CDK). J Cheminform 11:37. https://doi.org/10.1186/s13321-019-0361-8 **[cito:usesMethodIn]**
17. O'Boyle N, Dalke A (2018) DeepSMILES: an adaptation of SMILES for use in machine-learning of chemical structures. https://doi.org/10.26434/chemrxiv.7097960.v1 **[cito:usesMethodIn]**
18. Hastings J, de Matos P, Dekker A, Ennis M, Harsha B, Kale N (2013) The ChEBI reference database and ontology for biologically relevant chemistry: enhancements for 2013. Nucleic Acids Res 41:D456–D463. https://doi.org/10.1093/nar/gks1146
19. Gaulton A, Hersey A, Nowotka M, Bento AP, Chambers J, Mendez D (2017) The ChEMBL database in 2017. Nucleic Acids Res 45:D945–D954. https://doi.org/10.1093/nar/gkw1074 **[cito:usesDataFrom]**
20. ChemAxon (2012) JChem Base was used for structure searching and chemical database access and management. http://www.chemaxon.com.
21. Schaub J, Zielesny A, Steinbeck C, Sorokina M (2020) Too sweet: cheminformatics for deglycosylation in natural products. J Cheminform 12:67. https://doi.org/10.1186/s13321-020-00467-y **[cito:usesMethodIn]**
22. Ertl P, Roggo S, Schuffenhauer A (2008) Natural product-likeness score and its application for prioritization of compound libraries. J Chem Inf Model 48:68–74. https://doi.org/10.1021/ci700286x **[cito:usesMethodIn]**
23. Sorokina M, Steinbeck C (2019) NaPleS: a natural products likeness scorer—web application and database. J Cheminformatics. https://doi.org/10.1186/s13321-019-0378-z **[cito:usesMethodIn]**
24. Kim H, Wang M, Leber C, Nothias L-F, Reher R, Kang KB, et al. (2020) NPClassifier: a Deep Neural Network-Based Structural Classification Tool for Natural Products. https://doi.org/10.26434/chemrxiv.12885494.v1 **[cito:usesMethodIn]**
25. Kim S, Thiessen PA, Bolton EE, Chen J, Fu G, Gindulyte A (2016) PubChem Substance and Compound databases. Nucleic Acids Res 44:D1202–D1213. https://doi.org/10.1093/nar/gkv951 **[cito:usesMethodIn]**
26. React – A JavaScript library for building user interfaces. https://reactjs.org/. Accessed 21 Aug 2020 **[cito:usesMethodIn]**
27. OpenChemLib (https://github.com/cheminfo/openchemlib-js). JavaScript (2020) https://github.com/cheminfo/openchemlib-js. Accessed 21 Aug 2020 **[cito:usesMethodIn]**
28. $bitsAllSet — MongoDB Manual. https://github.com/mongodb/docs/blob/master/source/reference/operator/query/bitsAllSet.txt. https://docs.mongodb.com/manual/reference/operator/query/bitsAllSet. Accessed 21 Aug 2020 **[cito:usesMethodIn]**
29. Ullmann (cdk 2.3 API). http://cdk.github.io/cdk/latest/docs/api/index.html. Accessed 21 Aug 2020 **[cito:usesMethodIn]**
30. Cordella LP, Foggia P, Sansone C, Vento M (2004) A (sub)graph isomorphism algorithm for matching large graphs IEEE Trans Pattern Anal Mach Intell 26:1367–1372. **[cito:usesMethodIn]**
31. DfPattern (cdk 2.3 API). http://cdk.github.io/cdk/latest/docs/api/index.html. Accessed 28 Sep 2020 **[cito:usesMethodIn]**
32. Michał LSH-based similarity search in MongoDB is faster than postgres cartridge. THE CHEMBL-OG The Organization of Drug Discovery Data. http://chembl.blogspot.com/2015/08/lsh-based-similarity-search-in-mongodb.html. Accessed 21 Aug 2020 **[cito:usesMethodIn]**
33. Ntie-Kang F, Nwodo JN, Ibezim A, Simoben CV, Karaman B, Ngwa VF (2014) Molecular modeling of potential anticancer agents from African medicinal plants. J Chem Inf Model 54:2433–2450. https://doi.org/10.1021/ci5003697
34. Ntie-Kang F, Zofou D, Babiaka SB, Meudom R, Scharfe M, Lifongo LL (2013) AfroDb: a select highly potent and diverse natural product library from African medicinal plants. PLoS ONE 8:e78085
35. Onguéné PA, Ntie-Kang F, Mbah JA, Lifongo LL, Ndom JC, Sippl W (2014) The potential of anti-malarial compounds derived from African medicinal plants, part III: an in silico evaluation of drug metabolism and pharmacokinetics profiling. Org Med Chem Lett 4:6. https://doi.org/10.1186/s13588-014-0006-x
36. AnalytiCon Discovery, Screening Libraries. In: AnalytiCon Discovery. https://ac-discovery.com/screening-libraries/. Accessed 16 Oct 2020
37. Pilón-Jiménez BA, Saldívar-González FI, Díaz-Eufracio BI, Medina-Franco JL (2019) BIOFACQUIM: a Mexican compound database of natural products. Biomolecules 9:31. https://doi.org/10.3390/biom9010031
38. Dagan-Wiener A, Di Pizio A, Nissim I, Bahia MS, Dubovski N, Margulis E (2019) BitterDB: taste ligands and receptors database in 2019. Nucleic Acids Res 47:D1179–D1185. https://doi.org/10.1093/nar/gky974
39. Yabuzaki J (2017) Carotenoids Database: structures, chemical fingerprints and distribution among organisms. Database J Biol Databases Curation. https://doi.org/10.1093/database/bax004
40. Pence HE, Williams A (2010) ChemSpider: an online chemical information resource. J Chem Educ 87:1123–1124. https://doi.org/10.1021/ed100697w
41. Ntie-Kang F, Amoa Onguéné P, Scharfe M, Owono LCO, Megnassan E, Meva'a Mbaze L (2014) ConMedNP: a natural product library from Central African medicinal plants for drug discovery. RSC Adv 4:409–419. https://doi.org/10.1039/c3ra43754j
42. Bultum LE, Woyessa AM, Lee D (2019) ETM-DB: integrated Ethiopian traditional herbal medicine and phytochemicals database. BMC Complement Altern Med 19:212. https://doi.org/10.1186/s12906-019-2634-1
43. Neveu V, Moussy A, Rouaix H, Wedekind R, Pon A, Knox C (2017) Exposome-Explorer: a manually-curated database on biomarkers of exposure to dietary and environmental factors. Nucleic Acids Res 45:D979–D984. https://doi.org/10.1093/nar/gkw980
44. Wang M, Carver JJ, Phelan VV, Sanchez LM, Garg N, Peng Y (2016) Sharing and community curation of mass spectrometry data with Global Natural Products Social Molecular Networking. Nat Biotechnol 34:828. https://doi.org/10.1038/nbt.3597
45. Kang H, Tang K, Liu Q, Sun Y, Huang Q, Zhu R (2013) HIM-herbal ingredients in vivo metabolism database. J Cheminform 5:28. https://doi.org/10.1186/1758-2946-5-28
46. Ye H, Ye L, Kang H, Zhang D, Tao L, Tang K (2011) HIT: linking herbal active ingredients to targets. Nucleic Acids Res 39:D1055–D1059 https://doi.org/10.1093/nar/gkq1165
47. NDOFINE Chemical Company. http://www.indofinechemical.com/Media/sdf/sdf_files.aspx. Accessed 16 Oct 2019
48. Zhang R, Lin J, Zou Y, Zhang X-J, Xiao W-L (2019) Chemical space and biological target network of anti-inflammatory natural products, J Chem Inf Model 59:66–73. https://doi.org/10.1021/acs.jcim.8b00560
49. Vetrivel U, Subramanian N, Pilla K (2009) InPACdb—Indian plant anticancer compounds database. Bioinformation 4:71–74
50. InterBioScreen | Natural Compounds. https://www.ibscreen.com/natural-compounds. Accessed 9 Oct 2019
51. Lichen Database. In: MTBLS999: A database of high-resolution MS/MS spectra for lichen metabolites. https://www.ebi.ac.uk/metabolights/MTBLS999. Accessed 16 Oct 2019
52. Gentile D, Patamia V, Scala A, Sciortino MT, Piperno A, Rescifina A (2020) Putative inhibitors of SARS-CoV-2 main protease from a library of marine natural products: a virtual screening and molecular modeling study. Marine Drugs 18:225. https://doi.org/10.3390/md18040225
53. Derese S, Oyim J, Rogo M, Ndakala A (2015) Mitishamba database: a web based in silico database of natural products from Kenya plants. Nairobi, University of Nairobi

54. Ntie-Kang F, Telukunta KK, Döring K, Simoben CV, Moumbock AF, Malange YI (2017) NANPDB: a resource for natural products from Northern African sources. J Nat Prod 80:2067–2076. https://doi.org/10.1021/acs.jnatprod.7b00283
55. Compound Sets—NCI DTP Data—National Cancer Institute—Confluence Wiki. https://wiki.nci.nih.gov/display/NCIDTPdata/Compound+Sets. Accessed 18 Oct 2019
56. Mangal M, Sagar P, Singh H, Raghava GPS, Agarwal SM (2013) NPACT: naturally occurring plant-based anti-cancer compound-activity-target database. Nucleic Acids Res 41:D1124–D1129. https://doi.org/10.1093/nar/gks1047
57. Zeng X, Zhang P, He W, Qin C, Chen S, Tao L (2018) NPASS: natural product activity and species source database for natural product research, discovery and tool development. Nucleic Acids Res 46:D1217–D1222. https://doi.org/10.1093/nar/gkx1026
58. Choi H, Cho SY, Pak HJ, Kim Y, Choi J, Lee YJ (2017) NPCARE: database of natural products and fractional extracts for cancer regulation. J Cheminformatics 9:2. https://doi.org/10.1186/s13321-016-0188-5
59. Tomiki T, Saito T, Ueki M, Konno H, Asaoka T, Suzuki R (2006) RIKEN natural products encyclopedia (RIKEN NPEdia), a chemical database of RIKEN natural products depository (RIKEN NPDepo). J Comput Aid Chem 7:157–162
60. Ntie-Kang F, Onguéné PA, Fotso GW, Andrae-Marobela K, Bezabih M, Ndom JC (2014) Virtualizing the p-ANAPL library: a step towards drug discovery from African medicinal plants. PLoS ONE 9:e90655. https://doi.org/10.1371/journal.pone.0090655
61. Rothwell JA, Perez-Jimenez J, Neveu V, Medina-Remón A, M'Hiri N, García-Lobato P (2013) Phenol-Explorer 3.0: a major update of the Phenol-Explorer database to incorporate data on the effects of food processing on polyphenol content. Database. https://doi.org/10.1093/database/bat070
62. Sawada Y, Nakabayashi R, Yamada Y, Suzuki M, Sato M, Sakata A (2012) RIKEN tandem mass spectral database (ReSpect) for phytochemicals: a plant-specific MS/MS-based data resource and database. Phytochemistry 82:38–45. https://doi.org/10.1016/j.phytochem.2012.07.007
63. Hatherley R, Brown DK, Musyoka TM, Penkler DL, Faya N, Lobb KA (2015) SANCDB: a South African natural compound database. J Cheminformatics 7:29. https://doi.org/10.1186/s13321-015-0080-8
64. Davis GDJ, Vasanthi AHR (2011) Seaweed metabolite database (SWMD): a database of natural compounds from marine algae. Bioinformation 5:361–364.
65. Specs. Compound management services and research compounds for the life science industry. https://www.specs.net/index.php. Accessed 16 Oct 2019
66. Fischedick JT, Johnson SR, Ketchum REB, Croteau RB, Lange BM (2015) NMR spectroscopic search module for Spektraris, an online resource for plant natural product identification—Taxane diterpenoids from Taxus × media cell suspension cultures as a case study. Phytochemistry 113:87–95. https://doi.org/10.1016/j.phytochem.2014.11.020
67. Moumbock AFA, Gao M, Qaseem A, Li J, Kirchner PA, Ndingkokhar B (2020) StreptomeDB 3.0: an updated compendium of streptomycetes natural products. Nucleic Acids Res. https://doi.org/10.1093/nar/gkaa868
68. TCMID: traditional Chinese medicine integrative database for herb molecular mechanism analysis. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3531123/. Accessed 29 Apr 2019
69. Tung C-W, Lin Y-C, Chang H-S, Wang C-C, Chen I-S, Jheng J-L (2014) TIPdb-3D: the three-dimensional structure database of phytochemicals from Taiwan indigenous plants. Database. https://doi.org/10.1093/database/bau055
70. ünthardt BF, Hollender J, Hungerbühler K, Scheringer M, Bucheli TD (2018) Comprehensive toxic plants-phytotoxins database and its application in assessing aquatic micropollution potential. J Agric Food Chem 66:7577–7588. https://doi.org/10.1021/acs.jafc.8b01639
71. UEFS Natural Products. http://zinc12.docking.org/catalogs/uefsnp. Accessed 6 Nov 2019
72. Gu J, Gui Y, Chen L, Yuan G, Lu H-Z, Xu X (2013) Use of natural products as chemical library for drug discovery and network pharmacology. PLoS ONE 8:e62839. https://doi.org/10.1371/journal.pone.0062839
73. Nguyen-Vo T-H, Le T, Pham D, Nguyen T, Le P, Nguyen A (2019) VIETHERB: a database for Vietnamese herbal species. J Chem Inf Model 59:1–9. https://doi.org/10.1021/acs.jcim.8b00399
74. Sterling T, Irwin JJ (2015) ZINC 15—ligand discovery for everyone. J Chem Inf Model 55:2324–2337. https://doi.org/10.1021/acs.jcim.5b00559

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

BMC

Rajan *et al. J Cheminform (2021) 13:34*
https://doi.org/10.1186/s13321-021-00512-4

Journal of Cheminformatics

RESEARCH ARTICLE

Open Access

STOUT: SMILES to IUPAC names using neural machine translation

Kohulan Rajan[1], Achim Zielesny[2] and Christoph Steinbeck[1*]

Abstract

Chemical compounds can be identified through a graphical depiction, a suitable string representation, or a chemical name. A universally accepted naming scheme for chemistry was established by the International Union of Pure and Applied Chemistry (IUPAC) based on a set of rules. Due to the complexity of this ruleset a correct chemical name assignment remains challenging for human beings and there are only a few rule-based cheminformatics toolkits available that support this task in an automated manner. Here we present STOUT (**S**MILES-**TO**-**I**UPAC-name **t**ranslator), a deep-learning neural machine translation approach to generate the IUPAC name for a given molecule from its SMILES string as well as the reverse translation, i.e. predicting the SMILES string from the IUPAC name. In both cases, the system is able to predict with an average BLEU score of about 90% and a Tanimoto similarity index of more than 0.9. Also incorrect predictions show a remarkable similarity between true and predicted compounds.

Keywords: Neural machine translation, Chemical language, IUPAC names, SMILES, DeepSMILES, SELFIES, Deep neural network, Attention mechanism, Recurrent neural network

Introduction

Assigning names to chemical compounds so that an author can refer to them in the text of a scientific article, book or patent has a long history. In the early days and even still today, such names were often chosen based on physicochemical or perceptible properties, but also named after species, people, named after fictional characters, related to sex, bodily functions, death and decay, religion or legend, or other [1]. Usually, this makes it impossible to conclude from the name to the chemical structure of the compound. To overcome this dilemma, the International Union of Pure and Applied Chemistry (IUPAC) established a set of rules and guidelines for chemical nomenclature [2–5] so that a systematic name can be generated from the structure and substructures of a chemical compound and vice versa. Often, more than one systematic IUPAC name can be generated for the same compound: Therefore, the IUPAC introduced the IUPAC preferred name in their current edition of the Blue Book, preferring one of the possible names over all others.

Other types of string representations of molecules, such as SMILES [6], InChI [7], SYBYL line notation [8], Wiswesser line notation [9], and SMARTS [10] are more concise forms of line representations. While in principle being human-readable, these representations are primarily designed to be understood by machines. Thus, they are not commonly used in text to denominate chemical compounds for recognition by human readers, but have been incorporated into many major open-source and proprietary cheminformatics toolkits.

IUPAC name generation, due to its algorithmic complexity and the large set of rules, is missing in many cheminformatics toolkits in general. For a human, IUPAC name generation for more than a handful of molecules is cumbersome. People, therefore, resort to the few available automatic tools for IUPAC name generation.

Among the available and reliable solutions are the “molconvert” software, a command-line program in

---

*Correspondence: christoph.steinbeck@uni-jena.de
[1] Institute for Inorganic and Analytical Chemistry, Friedrich-Schiller-University Jena, Lessingstr. 8, 07743 Jena, Germany
Full list of author information is available at the end of the article

© The Author(s) 2021. This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

Marvin Suite 20.15 from ChemAxon (https://www.chemaxon.com) [11]. It is available for researchers under an academic license. Open-source programs such as the Chemistry Development Kit (CDK) [12], RDKit [13], or Open Babel [14] do not (yet) provide any algorithms that can automate the process of IUPAC naming for molecules.

With this work, we report a proof-of-concept application of Neural Machine Translation (NMT) for the conversion of machine-readable chemical line notations into IUPAC names and vice versa. A large training set was generated with ChemAxon's molconvert software and we would like to emphasise that this work would not have been possible without the generous offer by ChemAxon for the academic scientific community to use their software for free. We also like to point out that the purpose of this work is not to make ChemAxon's tool obsolete. As a deterministic tool, it will continue to be the first choice for practical naming tasks in databases.

For the work presented here, we were inspired by Google's multiple NMT models and came up with the idea to build a **S**MILES-**TO**-IUPAC-name **t**ranslator called STOUT. STOUT was developed based on language translation and language understanding. We treated the two chemical representations as two different languages—each SMILES string and corresponding IUPAC name was treated as two different sentences that have the same meaning in reality.

All these language models can only achieve greater than 90% accuracy with sufficient data to train them on. The majority of state-of-the-art language translation models are trained on millions of words and sentences to achieve such high levels of accuracy. Moreover, to train such large models in an adequate amount of time dedicated and powerful machine learning hardware is required. In this work, we report substantially shortened training times for our models using Google's Tensor Processing Units (TPU).

Methods

Using deep machine learning methods such as NMT for SMILES-to-IUPAC-name translation is a completely data-driven task so that high-quality data from a reliable source is mandatory. In this work, datasets were created for SMILES-to-IUPAC-name translation as well as for IUPAC-name-to-SMILES translation respectively.

Data

All molecules were obtained from PubChem [15], one of the openly available large small molecule databases, where the entire PubChem database was downloaded from its FTP site in SDF format. Using the CDK, explicit hydrogens were removed from the molecules and their topological structures were converted to canonical SMILES strings. The obtained 111 million molecules were filtered according to the ruleset of our previous DECIMER work [16], i.e. molecules must

- have a molecular weight of fewer than 1500 Da,
- not possess any counter ions,
- contain only C, H, O, N, P, S, F, Cl, Br, I, Se and B,
- not contain any hydrogen isotopes (D, T),
- have between 3 and 40 bonds,
- not contain any charged group,
- contain implicit hydrogens only, except in functional groups,

to arrive at a dataset of 81 million molecules. These selected SMILES strings were converted into IUPAC names using Chemaxon's molconvert software, a command-line program in Marvin Suite 20.15 from ChemAxon (https://www.chemaxon.com).

Using SMILES strings directly for training Neural Networks (NN) may cause various problems due to their intricate structure which is difficult to split into separate meaningful tokens necessary for the machine input. To tackle this problem, two other representations are available, DeepSMILES [17] and SELFIES [18]. For a discussion of the problems of string tokenization for deep learning, we refer our readers to those two publications. Our results confirm the superiority of SELFIES for the task discussed here and in our work on Optical Chemical Entity Recognition [16]. Thus, for this work all SMILES strings were converted into SELFIES using a custom python script (Fig.).

Two datasets were constructed, a 30 million and 60 million molecule set with SELFIES and corresponding IUPAC names, where the 60 million sets contained all 30 million molecule entries of the former. Every SELFIES string and IUPAC name was split into separate tokens using the space character as a delimiter. SELFIES were split according to a closed square bracket "]" and an open square bracket "[". For IUPAC names a small set of rules was applied to split them uniformly: After every,

- open bracket "(", "{" and "[",
- close bracket ")", "}" and "]",
- dash symbol "-",
- full stop ".",
- comma ","

and after every word in the following list,

- mono,di,tri,tetra,penta,hexa,hepta,octa,nona
- deca,oxo,methyl,hydroxy,benzene,oxy,chloro,cyclo,amino,bromo,hydro,fluoro

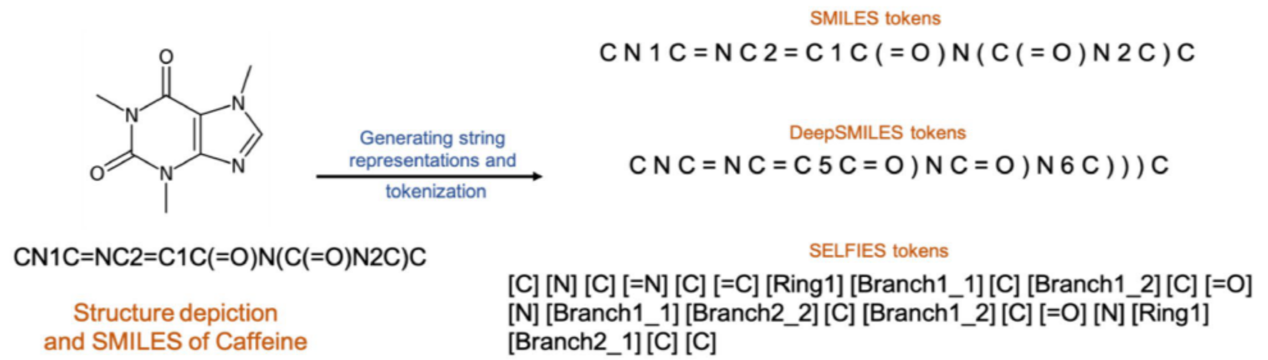

SMILES tokens
C N 1 C = N C 2 = C 1 C (= O) N (C (= O) N 2 C) C
Generating string representations and tokenization
DeepSMILES tokens
C N C = N C = C 5 C = O) N C = O) N 6 C))) C
CN1C=NC2=C1C(=O)N(C(=O)N2C)C
SELFIES tokens
[C] [N] [C] [=N] [C] [=C] [Ring1] [Branch1_1] [C] [Branch1_2] [C] [=O] [N] [Branch1_1] [Branch2_2] [C] [Branch1_2] [C] [=O] [N] [Ring1] [Branch2_1] [C] [C]
Structure depiction and SMILES of Caffeine

Fig.1 SMILES, DeepSMILES and SELFIES split into tokens which are separated by a space character

- methane,cyano,amido,ethene,phospho,amide,butane,carbono,hydro,sulfane,butane,sulfino
- iodo,ethane,ethyne,bi,imino,nitro,butan,idene,sulfo,carbon,propane,ethen,acetaldehyde,benzo,oxa,nitroso,hydra,iso

a space character was added as a delimiter. After adding the delimiter, the dataset was padded to fit the maximum length of 48 characters for SELFIES strings and 78 characters for IUPAC name strings, a "start" token was added to each string to indicate its beginning, and an "end" token was added at the end of the string. The strings were tokenized and saved into small TFRecord files for training with GPUs or TPUs. Finally, two SELFIES-to-IUPAC-name datasets and two IUPAC-name-to-SELFIES datasets—with 30 million (exactly 30,000,128) and 60 million (exactly 60,000,256) molecules each - were generated.

Network

The NMT network follows the implementation reported by Google for their language translation models, which itself is built on the network designed by Luong et al. [19] for neural machine translation, using a soft attention mechanism developed by Bahdanau et al. [20]. It is based on an autoencoder–decoder architecture and is written on Python 3 with Tensorflow 2.3.0 [21] at the backend. The encoder network and the decoder network use Recurrent Neural Networks (RNNs) with Gated Recurrent Units (GRU). The input strings are passed to the encoder and the output strings to the decoder. The encoder network generates the encoder output and the encoder hidden state. The attention weight is calculated by the attention mechanism implemented in the network. Encoder output with attention weights then creates the context vector. Meanwhile, the decoder output is passed through an embedding layer. The output generated by the embedding layer and the context vector is concatenated and passed on to the GRUs of the decoder. An Adam optimizer with a learning rate of 0.0005 is applied and sparse categorical cross-entropy is used to calculate the loss with a modified loss function. A batch size of 256 Strings is used for a GPU and a global batch size of 1024 Strings for a TPU where the global batch size is divided between the nodes.

For SELFIES-to-IUPAC-name and IUPAC-name-to-SELFIES translation the same network architecture is used with the input/output datasets simply being swapped. Figure 2 shows the STOUT architecture for SMILES-to-IUPAC-name translation.

Model training

For large datasets, training a neural network efficiently is a challenging task. As an initial test, the network was trained with 15 million molecules on a server with an nVidia Tesla V100 GPU, 384GB of RAM, and two Intel(R) Xeon(R) Gold 6230 processors. The average training epoch was evaluated to be about 27 h so that training of larger datasets appeared to be prohibitive. With more than 100 epochs of training time used in our training described below, those 27 h per epoch translate into almost 4 months of training time, with multiples of that for training with 30 million or 60 million structures. Thus, the training scripts were modified to use Tensor Processing Units (TPUs) available on the Google cloud using the Tensorflow distributed training API. A corresponding training with TPU V3-8 units (with 8 nodes each) reduced the average training epoch to about 2 h.

Model testing

To evaluate the models' performance, a test set of 2.2 Million molecules was used, which was not present in the 30

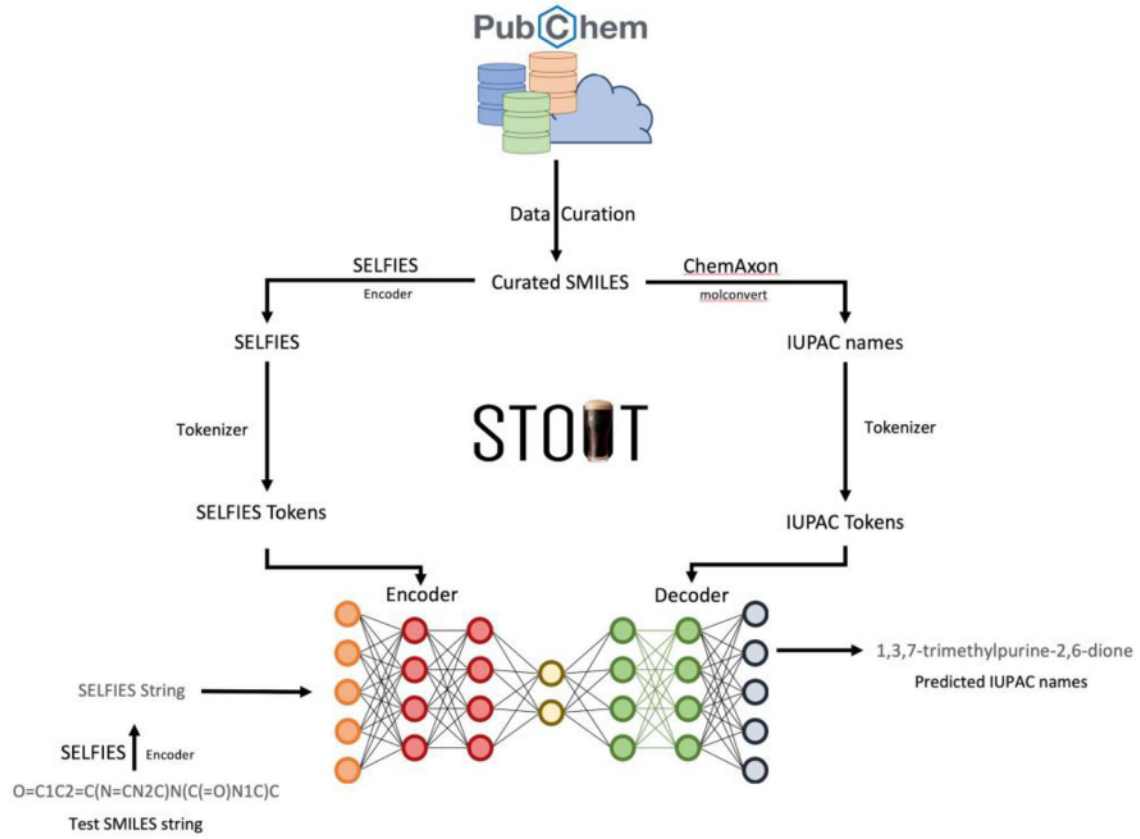


Fig. 2 STOUT architecture for SMILES-to-IUPAC-name translation

million and the 60 million molecules training sets. A uniform and highly similar frequency distribution of unique SELFIES tokens in training and test data were ensured by corresponding test molecule selection. The SELFIES-to-IUPAC-name translation and the reverse IUPAC-name-to-SELFIES translation were tested with the same set.

To assess the predictive accuracy BLEU scoring [22] was used (see Appendix for details). Also, Tanimoto similarities were calculated between original and predicted strings using PubChem fingerprints. For the predictions of IUPAC names as an output, the IUPAC names were reconverted to SMILES using OPSIN 2.5 [23] and canonicalised using the CDK, with the resulting SMILES being utilized for Tanimoto similarity calculations.

Results and discussion

Computational considerations

Table 1 shows the number of unique SELFIES/IUPAC-name tokens for both data sets. Note that the 30 million and the larger 60 million molecules datasets have the same number of tokens. To keep the same number of tokens we removed the least occurring tokens from both sets using a cutoff. In contrast, the SELFIES token set size is smaller than that of the IUPAC name tokens because the IUPAC names cover a far greater language space.

Table 1 Number of unique SELFIES and IUPAC-name tokens for each dataset

Dataset size	Number of SELFIES tokens	Number of IUPAC tokens
30 Million	27	1190
60 Million	27	1190

We used a 15 Mio training dataset to compare the training speed between a GPU and TPUs. Training 15 Million molecules on a TPU V3-8 requires 2 h per epoch which is 13 times faster than using a GPU V100. Using a TPU V3-32 allows for an additional 4 times faster performance in comparison to a TPU V3-8 and is 54 times faster compared to a GPU V100, see Fig. 3.

Figure 4 shows the different training times per epoch of the different datasets on TPU V3-8 units where all

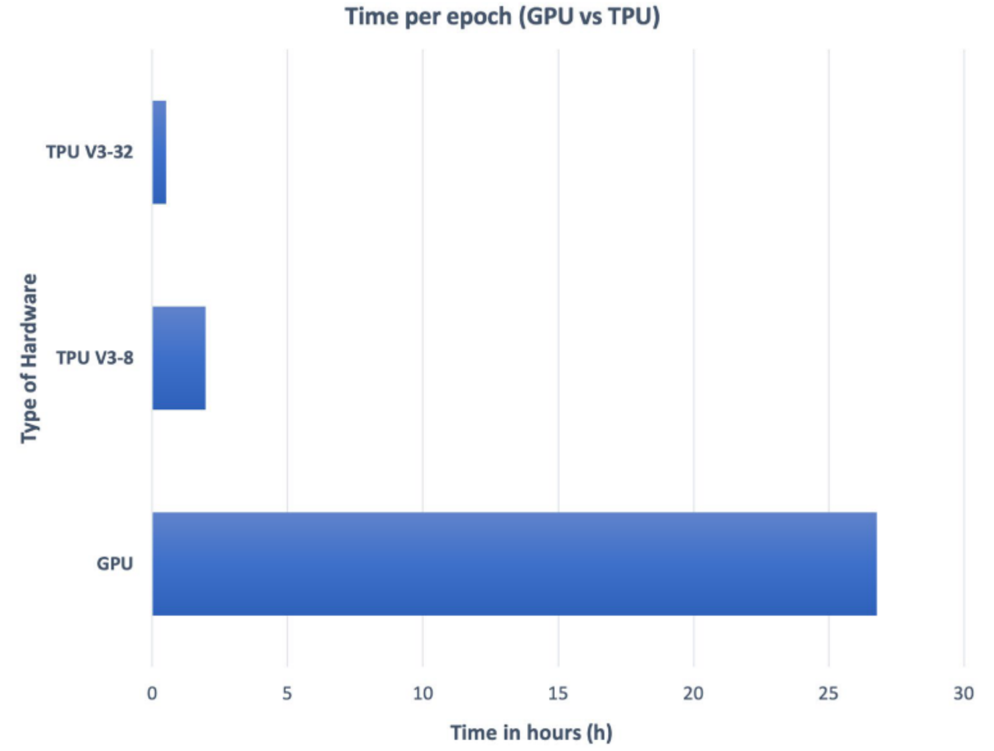


Fig. 3 Average training time per epoch on different hardware (lower is better)

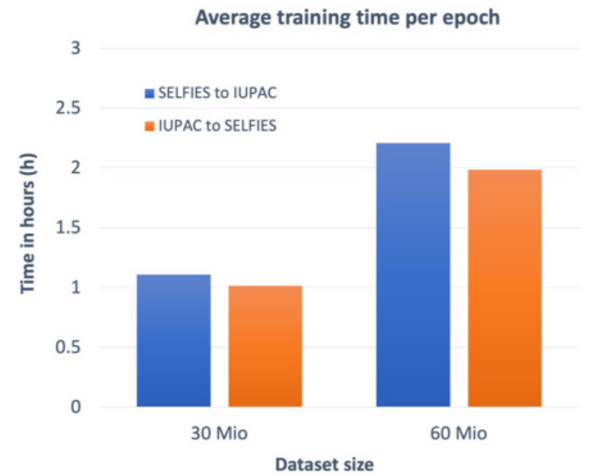


Fig. 4 Average training time per epoch for different datasets using TPU V3-8

Table 2 BLEU scores analysis

Training dataset size	30 Mio	60 Mio
Average BLEU score	0.89	0.94
Total number of strings with BLEU 1.0	52.48%	66.65%
BLEU-1	0.92	0.95
BLEU-2	0.90	0.94
BLEU-3	0.88	0.93
BLEU-4	0.86	0.92

models were trained for more than 100 epochs until convergence. The difference between the SELFIES-to-IUPAC-name and IUPAC-name-to-SELFIES training is caused by the different number of I/O tokens of each dataset: For the SELFIES-to-IUPAC-name translation, the output tokens are derived from the IUPAC names whereas for the IUPAC-name-to-SELFIES translation the output tokens are taken from SELFIES strings. Since SELFIES strings are smaller and less complex than IUPAC name strings the IUPAC-name-to-SELFIES translation is faster.

Test results

SELFIES-to-IUPAC-nametranslation

Table 2 summarizes the average and individual BLEU scores for the 30 million and the 60 million molecules dataset. A predicted string with a BLEU score of 1.0 means a score of 1.0 using the NLTK sentence BLEU scoring function[24] and they are mostly identical strings (see Appendix).

Compared to the 30 million molecules dataset, a model trained with 60 million molecules makes better predictions, as demonstrated by all BLEU score types.

To assess the network's ability to learn "chemistry" we calculated the Tanimoto similarities between the predicted and the original molecules by translating the original and the predicted IUPAC names back to SMILES strings using OPSIN and canonicalised the retranslated SMILES using the CDK. We used the CDK with Pubchem fingerprints to calculate the Tanimoto similarity indices. The IUPAC names that OPSIN was able to translate back to SMILES strings were counted as valid IUPAC names while the others were counted as invalid. Only the valid IUPAC-name-to-SMILES translations were used for the Tanimoto similarity calculations. The average Tanimoto similarity was measured on valid IUPAC-name-to-SMILES translations. Additionally, both Tanimoto similarity calculations were readjusted to the number of data points present on the test dataset (see Table 3). We also computed full isomorphism matches using InChIs and found that 98% of all Tanimoto similarity 1.0 cases were full graph isomorphisms.

The invalid IUPAC names are the ones that were rejected by OPSIN and could not be converted into SMILES. This inability is the result of errors of the IUPAC names being predicted. In most cases, the IUPAC-name-to-SMILES translation failed because

- they did not contain a comma,
- some of them were missing a close bracket symbol corresponding to the open bracket symbol,
- the valence of an atom was wrong,
- a certain block of text was uninterpretable,
- they failed to assign all bonds correctly,
- of a disagreement between lengths of bridges and alkyl chain length
- of long names with repeating words.

Table 3 Tanimoto similarities

Training dataset size	30 Mio	60 Mio
Invalid IUPAC names	21.41%	14.50%
Valid IUPAC names	78.59%	85.50%
Tanimoto 1.0 count on the total test dataset	58.36%	72.33%
Tanimoto 1.0 count on valid IUPAC names	74.26%	84.59%
Average Tanimoto (measured for total test dataset)	0.75	0.83
Average Tanimoto (measured for valid IUPAC names)	0.96	0.98

Table 4 presents a few examples of IUPAC names that could not be converted to SMILES strings with an explanation of the failure.

The Tanimoto similarity index 1.0 count with 72% (60 million molecules set) of the test data is already remarkable but the average Tanimoto similarity of 0.83 (60 million molecules set) suggests that an "understanding" of the "language of chemistry" emerged. Also, it becomes obvious that the number of predictions with a Tanimoto similarity of 1.0 is greater than the number of predictions with a BLEU score of 1.0, see Table 5: Although there are different IUPAC names, using OPSIN to re-translate these names led to SMILES representations with similar or even identical chemical graphs, see Figure 5. This also illustrates the extent to which the model is capable to successfully generalise the information of the training data. We found that only five predictions had a Tanimoto similarity index less than 1.0 but a BLEU score of 1.0, see Table 6 and Fig. 6.

IUPAC-name-to-SELFIESTranslation

The IUPAC-name-to-SELFIES translation was tested with the same 2.2 million test molecules as the SELFIES-to-IUPAC-name model before, but in reverse order. To use OPSIN as a performance measure, we analyzed our test set using OPSIN. It was able to convert 98.31% of IUPAC names generated by the molconvert algorithm back to SMILES and 96.24% were found to show a Tanimoto 1.0 similarity, see Table 7 for details. Table 8 summarizes the average BLEU score, the calculated BLEU

Table 4 Failed IUPAC-name-to-SMILES translations

IUPAC names	Reason for failure (OPSIN error messages)
1. *N*-[6-(2,3-diaminopropylidene)-1-methyl-1,2,4a,5,6,8a-hexahydroquinolin-6-yl]-*N*-methylpropanamide	Atoms are in an unphysical valency state. Element: C valency: 5
2. 2-[({[(3-ethoxypropyl)amino]({[2-(2-fluorophenyl)ethyl]amino})methylidene}amino)-*N*,*N*-dimethylacetamide	Unmatched opening bracket found
3. 3′-(propan-2-yl)-2′,3′,4′,5′,6′,7′,8′,8′a-octahydro-2′H-spiro[imidazole-4,1′-indolizin]-2-amine	The following being uninterpretable: 2′,3′,4′,5′,6′,7′,8′,8′
4. ({2′,6′-difluoro-2′,6′-dimethyl-[1,1′-biphenyl]-4-yl}methyl)(propyl)amine	Failed to assign all double bonds
5. 1,4,5-trimethyl-1-[1,2-dimethylpropyl)-2-methyl-1-propylbicyclo[12.2.1]tetradeca-1,5-diene	Disagreement between lengths of bridges and alkyl chain length

Table 5 Predicted IUPAC name strings with a Tanimoto similarity index of 1.0 but a low BLEU score

No.	IUPAC names		BLEU Score	IUPAC names translated into SMILES using OPSIN		Tanimoto similarity Index
	Original	Predicted		Original	Predicted	
1	butyl3-methyl-12-methylidene-2,4,7,10-tetraoxatridecan-13-oate	butyl2-({2-[2-(1-methoxyethoxy)ethoxy]ethoxy}methyl)prop-2-enoate	0.00	O C(OCCCC)C(C)COCCOCCOC(OC)C	O C(OCCCC)C(C)COCCOCCOC(OC)C	1.0
2	ethyl3-[1,10-diiodo-9-(iodosulfanyl)-1,10-dithia-2,9-diazadecan-2-yl]propanoate	ethyl3-({6-[bis(iodosulfanyl)amino]hexyl}(iodosulfanyl)amino}propanoate	0.10	O C(OCC)CCN(SI)CCCCCCN(SI)SI	O C(OCC)CCN(SI)CCCCCCN(SI)SI	1.0
3	*N*,*N*dimethyl-1-(prop-2-enamido)cyclopentane-1-carboxamide	*N*-[1-(dimethylcarbamoyl)cyclopentyl]prop-2-enamide	0.24	O C(C C)NC1(C(O)N(C)C)CCCC1	O C(C C)NC1(C(O)N(C)C)CCCC1	1.0
4	6-[4-(2-cyanoethyl)phenyl]-*N*-[1-(hydroxycarbamoyl)ethyl]hexanamide	2-{6-[4-(3-cyanopropyl)phenyl]hexanamido}-*N*-hydroxypropanamide	0.32	N#CCCC1 CC C(C C1)CCCCCC(O)NC(C(O)NO)C	N#CCCCC1 CC C(C C1)CCCCCC(O)NC(C(O)NO)C	1.0
5	12-aminochrysene-6-carboxylicacid	6-aminotetraphene-11-carboxylicacid	0.41	O C(O)C1 CC 2C 3C CC CC3C(N) CC2C 4C CC CC41	O C(O)C1 CC CC2 CC 3C(N) CC 4C CC CC4C3C C21	1.0
6	1,3-bis[(6-bromopyridin-2-yl)methyl]-1,3-diazinane	2-bromo-6-({3-[(6-bromopyridin-2-yl)methyl]-1,3-diazinan-1-yl}methyl)pyridine	0.50	BrC 1N C(C CC1) CN2CN(CC3 NC(Br) CC C3)CCC2	BrC 1N C(C CC1) CN2CN(CC3 NC(Br) CC C3)CCC2	1.0
7	2,3,7-trifluoro-5-methylocta-1,3,5-triene	2,5-difluoro-3-methylocta-1,3,5-triene	0.61	FC(C)C(F) CC(CC(F)C)C	FC(C)C(CC(F) CCC)C	1.0
8	tert-butyl4-acetyl-2-[(acetyloxy)methyl]piperazine-1-carboxylate	tert-butyl2-[(acetyloxy)methyl]-4-acetylpiperazine-1-carboxylate	0.72	O C(OC(C)(C)C)N1CCN(C(O)C)CC1COC(O)C	O C(OC(C)(C)C)N1CCN(C(O)C)CC1COC(O)C	1.0
9	*N*-[(3-cyanophenyl)methyl]pyrimido[4,5-b]indolizine-10-carboxamide	*N*-[(3-cyanophenyl)methyl]-5H-pyrimido[4,5-b]indolizine-10-carboxamide	0.83	N#CC1 CC CC(C1)CNC(O)C 2C 3N CN CC3N4C CC CC24	N#CC1 CC CC(C1)CNC(O)C 2C 3N CN CC3N4C CC CC24	1.0
10	(5-benzylhexa-3,5-dien-2-ylidene)aminomethanesulfonate	(6-benzylhexa-3,5-dien-2-ylidene)aminomethanesulfonate	0.92	O S(O)([O-])CN C(C CC(C)CC 1C CC CC1)C	O S(O)([O-])CN C(C CC CCC 1C CC CC1)C	1.0

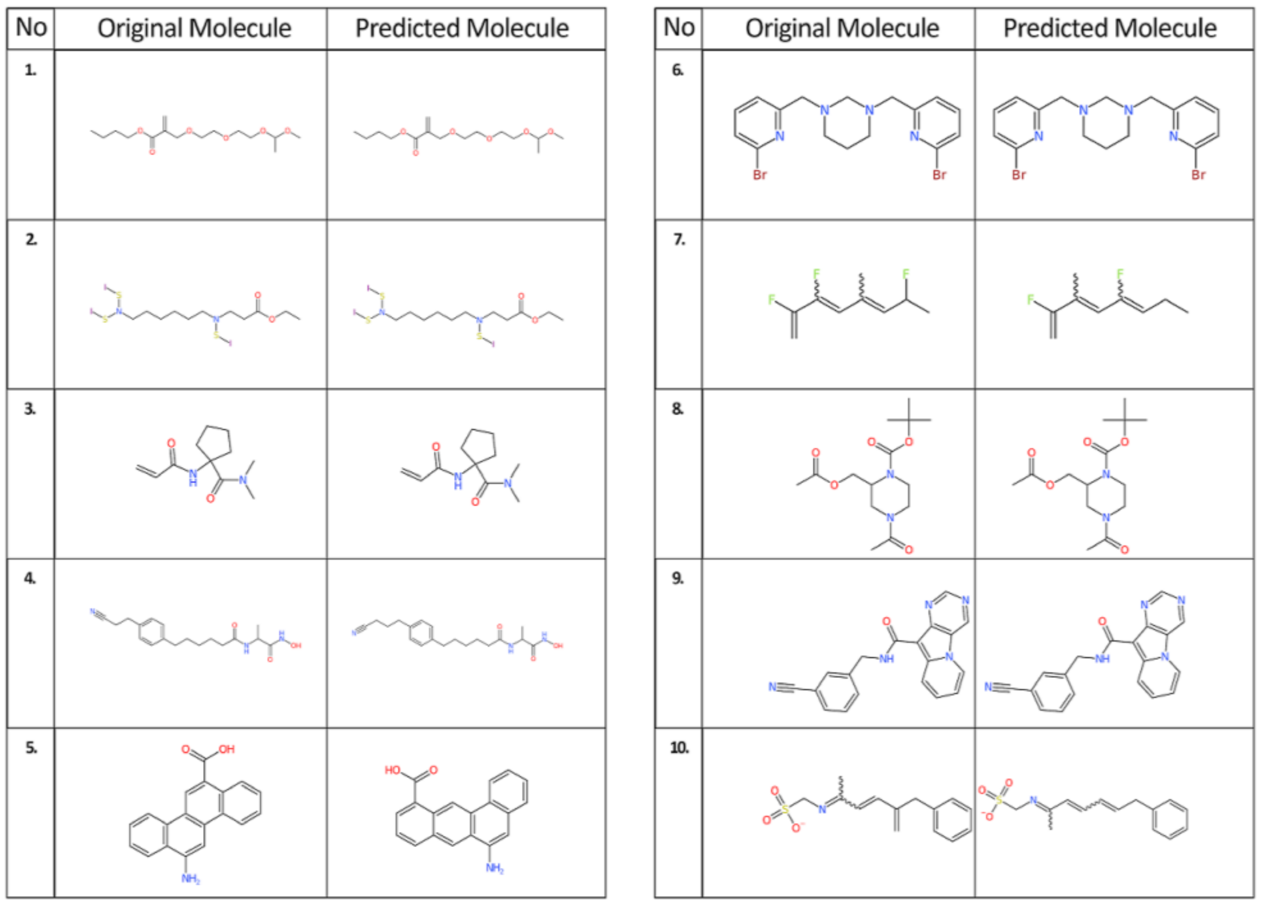


Fig. 5 Chemical structures depicted with the CDK depiction generator for predictions with Tanimoto similarity 1.0 but low BLEU score

Table 6 Predicted IUPAC name strings with a BLEU score of 1.0 but a low Tanimoto similarity index

No.	IUPAC names		BLEU Score	IUPAC names translated into SMILES using OPSIN		Tanimoto similarity Index
	Original	Predicted		Original	Predicted	
1	4-[(4-amino-2,3,6-trimethylphenyl)methyl]-2,3,5-trimethylaniline	4-[(4-amino-2,3,5-trimethylphenyl)methyl]-2,3,6-trimethylaniline	1.0	NC 1C C(C(C(C1C)C) CC 2C(CC(N) C(C2C) C)C)C	NC1 C(C C(C(C1C)C) CC2 CC(C(N)C(C2C) C)C)C	0.97
2	3-[(3-amino-2,6-diethylphenyl)methyl]-2,4-diethylaniline	3-[(3-amino-2,4-diethylphenyl)methyl]-2,6-diethylaniline	1.0	NC1 CC C(C(C1CC) CC 2C(CC C(N)C2CC) CC)CC	NC 1C(CC C(C1CC) CC2 CC C(C(N) C2CC) CC)CC	0.92
3	2-{4-[(dimethylamino)methyl]-6-[(2,6-dimethylphenoxy)methyl]-6-hydroxycyclohexa-2,4-dien-1-yl}acetonitrile	2-{4-[(2,6-dimethylphenoxy)methyl]-6-[(dimethylamino)methyl]-6-hydroxycyclohexa-2,4-dien-1-yl}acetonitrile	1.0	N#CCC1C CC(CC1(O) COC 2C(CC CC2C)C) CN(C)C	N#CCC1C CC(CC1(O) CN(C)C) COC 2C(CC CC2C)C	0.93
4	4-[4-(3-hydroxycyclohepta-1,3,6-trien-1-yl)phenyl]-*N*-(7-methylcyclohepta-1,4,6-trien-1-yl)butanamide	4-[4-(3-hydroxycyclohepta-1,4,6-trien-1-yl)phenyl]-*N*-(7-methylcyclohepta-1,3,6-trien-1-yl)butanamide	1.0	O C(NC1 CCC CC C1C) CCCC 2C CC(CC2) C 3C CCC C(O)C3	O C(NC1 CC CCC C1C) CCCC 2C CC(CC2) C 3C CC CC(O)C3	0.95
5	(but-1-en-2-yl)(prop-1-en-1-yl)amine	(but-1-en-1-yl)(prop-1-en-2-yl)amine	1.0	C C(NC CC)CC	C C(NC CCC)C	0.97

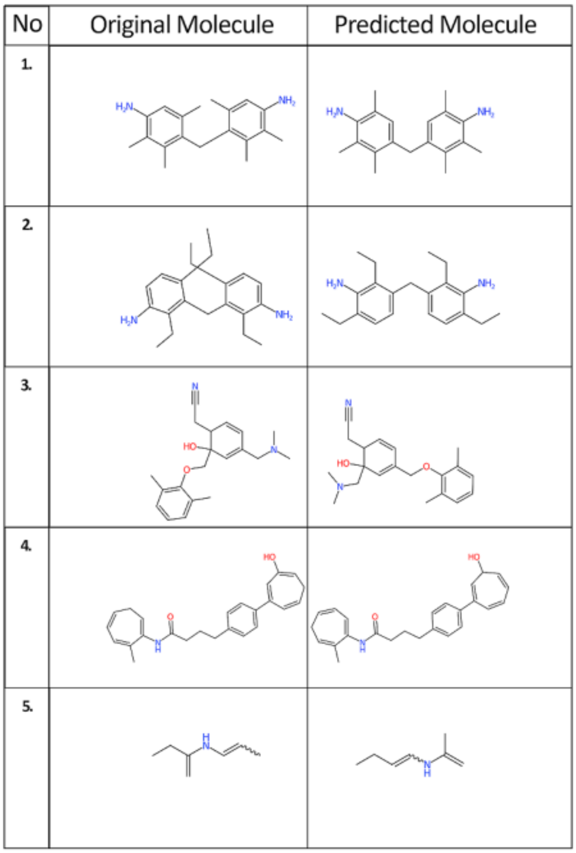


Fig. 6 Chemical structures depicted with the CDK depiction generator for predictions with BLEU score 1.0 but Tanimoto similarity less than 1.0

Table 7 Analysis on test set using OPSIN

OPSIN analysis on test set	Values
Invalid IUPAC names	1.69%
Valid IUPAC names	98.31%
Tanimoto 1.0 count on the total test dataset	97.89%
Tanimoto 1.0 count on valid IUPAC names	96.24%
Average Tanimoto (measured for total test dataset)	0.99
Average Tanimoto (measured for valid IUPAC names)	0.98

Table 8 Average BLEU scores, BLEU Scores, and Tanimoto similarity calculations

	30 Mio	60 Mio
Average BLEU score	0.90	0.94
Total number of predicted strings with BLEU 1.0	46.78%	68.47%
BLEU-1	0.94	0.97
BLEU-2	0.91	0.95
BLEU-3	0.89	0.94
BLEU-4	0.85	0.92
Tanimoto calculations		
Average Tanimoto similarity index	0.89	0.94
Number of predicted strings with Tanimoto 1.0	52.27%	73.26%

scores, and the Tanimoto similarities that were carried out on the test molecules for IUPAC-name-to-SELFIES translation.

The larger 60 million molecules dataset again performs better than the 30 million molecules dataset. Invalid SELFIES do not occur because all the predicted SELFIES were retranslated into SMILES without any error. Again, the predictions with Tanimoto similarity index 1.0 exceed those with BLEU scores 1.0. The reason for this is that BLEU is calculated by mapping word to word for an original and predicted SELFIES string while Tanimoto similarity is calculated according to the corresponding chemical structure, see Table 9 and Figure 7. To improve these results, more molecules with the same set of unique tokens would be needed. We also saw that 860 out of 2.2 million molecules (0.0003%) had BLEU 1.0 but a slightly lower Tanimoto similarity index because of slight differences in the chemical structures.

Conclusion

With this work, purely data-driven deep learning models for translation between different chemical entity representations are reported. We show that deep learning models are able to capture the essence of SMILES to IUPAC name string conversion (and vice versa) with reaching the 90% accuracy threshold. Despite this promising finding, any large scale and uncurated application should be currently handled with care.

With more data and additional training epochs STOUT is expected to further improve its prediction accuracy in the future. At best, it may finally play in the ballpark of the rule-based systems which further on define the possible top performance. Using the TPU platform will enable the models to be trained in an acceptable amount of time in the order of a few weeks. In addition, STOUT may be extended to alternative sophisticated models used in language translation and understanding, such as BERT [25].

During our revisions, there were two similar preprints, Struct2IUPAC [26] and Translating the Molecules [27], which has been published, reflecting an increase of interest in the translation of SMILES into IUPAC names and vice versa.

Table 9 Predicted SELFIES with low BLEU scores and Tanimoto similarity 1.0

No.	SELFIES		BLEU Score	SELFIES decoded back into SMILES		Tanimoto similarity Index
	Original	Predicted		Original	Predicted	
1.	[I][C][C][Branch1_2][Branch1_3][C][N][C][Expl Ring1][Branch1_1][C][C]	[I][C][C][Branch1_1][Branch1_3][N][C][C][Ring1][Branch1_1][C][C]:	0.00	IC 1C(CNC1C)C	IC 1C(CNC1C)C	1.0
2.	[O][C][C][C][C][C][Branch1_1][Ring2][C][Expl Ring1][Branch1_2][C][N][N][C][C][Branch1_1][Ring2][C][Expl Ring1][Branch1_2][C]	[O][C][C][C][C][C][Branch1_2][Ring2][C][Ring1][Branch1_2][C][N][N][C][C][Branch1_2][Ring2][C][Ring1][Branch1_2][C]:	0.18	OC 1C CC C(C1)C 2N NC C(C2) C	OC 1C CC C(C1)C 2N NC C(C2) C	1.0
3.	[C][Branch1_2][C][C][C][C][Branch1_2][Branch2_1][C][C][Branch1_2][Ring1][C][C][C][C][C]	[C][Branch1_1][N][C][C][Branch1_1][Branch1_3][C][C][Branch1_1][C][C][C][C][C][C][C]:	0.21	C(CCC(CC(CC)C)C)C	C(C C(C C(C)C)CC) CC	1.0
4.	[N][C][C][Branch1_2][N][C][C][C][Ring1][Branch1_2][O][C][Branch1_1][C][C][C][C][N][N][C][C][C][C][Branch1_1][Ring2][N][C][N][C][C][Ring1][N][Expl Ring1][Branch2_2]	[N][Branch1_2][Ring1][C][N][C][C][C][C][N][N][C][Branch1_1][P][C][C][N][C][Branch1_1][Branch1_3][O][C][Branch1_1][C][C][C][C][C][Expl Ring1][Branch2_3][C][Expl Ring1][#C][C][Expl Ring2][Ring1][Ring1]:	0.32	N1 CC(CC C1OC(C)C) C2 NNC 3C CC(N CN) CC23	N1 CC(CC C1OC(C)C) C2 NNC 3C CC(N CN) CC23	1.0
5.	[O][C][N][C][C][Branch1_1][Branch1_2][N][C][Ring1][Branch1_2][C][C][C][C][C][C][Ring1][Branch2_3]	[O][C][N][C][C][C][C][C][C][C][Expl Ring1][Branch1_3][N][C][Ring1][O][C]:	0.45	O C1NC2 C(N C1C)C CC CC2	O C1NC 2C CC CCC2N C1C	1.0
6.	[O][N][C][Branch1_2][C][O][C][C][C][C][C][C][Branch1_2][Branch2_2][C][C][C][Ring1][Branch1_2][C][Expl Ring1][Branch2_3][C]	[O][N][C][Branch1_2][C][O][C][C][C][C][C][C][Branch1_1][Branch2_2][C][C][C][Ring1][Branch1_2][C][Ring1][Branch2_3][C]:	0.53	O NC(O) C 1C CC2 CC(CC C2C1)C	O NC(O) C 1C CC2 CC(CC C2C1)C	1.0
7.	[O][B][Branch1_1][C][O][C][C][C][Branch1_2][C][C][C][C][Ring1][Branch1_2][C][C][C][C][C][C][Ring1][Branch1_2][C][C][N][C][C][C][Ring1][Branch1_2]	[O][B][Branch1_1][C][O][C][C][C][Branch1_1][C][C][C][C][Expl Ring1][Branch1_2][C][C][C][C][C][C][Expl Ring1][Branch1_2][C][C][N][C][C][C][Ring1][Branch1_2]:	0.60	OB(O) C1 CC(CC C1C2 CC CC C2) C3 CN CC C3	OB(O) C1 CC(CC C1C2 CC CC C2) C3 CN CC C3	1.0
8.	[O][C][N][C][Branch2_1][Ring1][C][C][C][C][C][Branch1_1][Ring1][O][C][C][C][Expl Ring1][Branch2_1][N][Branch1_1][C][C][C][C][Branch1_1][Branch1_2][C][C][Ring1][P][C][C][C]	[O][C][N][C][Branch2_1][Ring1][C][C][C][C][C][Branch1_1][Ring1][O][C][C][C][Ring1][Branch2_1][N][Branch1_1][C][C][C][C][Branch1_1][Branch1_3][C][C][Ring1][P][C][C][C]:	0.71	O C1NC(C 2C CC(OC) CC2N(C) C) C(C C1C)CC	O C1NC(C 2C CC(OC) CC2N(C) C) C(C C1CC)C	1.0

Table 9 (continued)

No.	SELFIES		BLEU Score	SELFIES decoded back into SMILES		Tanimoto similarity Index
	Original	Predicted		Original	Predicted	
9.	[O][P][Branch2_1][Ring1][Branch1_2][C][N][N][C][Branch1_2][Ring2][C][Ring1][Branch1_1][C][Branch1_1][C][F][Branch1_1][C][F][C][Branch1_1][C][F][F][Branch1_1][Branch2_2][C][C][C][C][C][C][Expl Ring1][Branch1_2][C][C][C][C][C][C][Expl Ring1][Branch1_2]	[O][P][Branch1_1][Branch2_2][C][C][C][C][C][C][Expl Ring1][Branch1_2][Branch1_1][Branch2_2][C][C][C][C][C][C][Expl Ring1][Branch1_2][C][N][N][C][Branch1_2][Ring2][C][Ring1][Branch1_1][C][Branch1_1][C][F][Branch1_1][C][F][C][Branch1_1][C][F][F]:	0.86	O P(C1 NNC(C1)C(F)(F)C(F)F)(C 2C CC CC2)C 3C CC CC3	O P(C1 NNC(C1)C(F)(F)C(F)F)(C 2C CC CC2)C 3C CC CC3	1.0
10.	[O][C][Branch2_1][Ring1][N][O][C][C][C][C][C][Branch1_2][N][C][Ring1][Branch1_2][O][C][Branch1_2][C][O][C][C][C][C][C][C][C][C][C][C][C][C][C][C][C]	[O][C][Branch2_1][Ring1][C][O][C][C][C][C][C][Branch1_2][N][C][Ring1][Branch1_2][O][C][Branch1_2][C][O][C][C][C][C][C][C][C][C][C][C][C][C][C][C][C]:	0.93	O C(OC1 CC CC(C1OC(O)CCC)CCCCCCCCC)CCC	O C(OC1 CC CC(C1OC(O)CCC)CCCCCCCCCC)CC	1.0

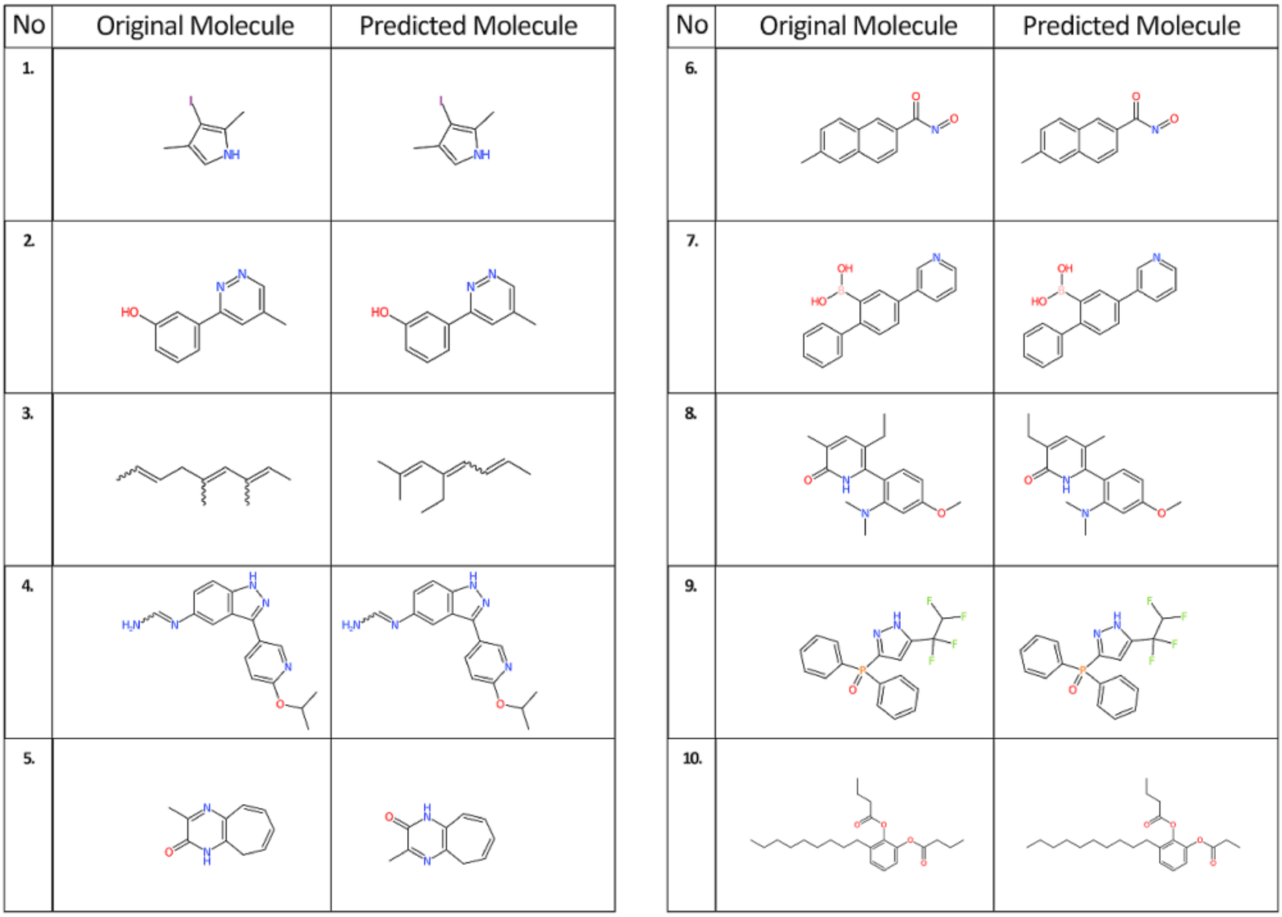


Fig. 7 Chemical structures depicted with the CDK depiction generator for predictions with Tanimoto similarity 1.0 and low BLEU score

Appendix

BLEU scoring for machine translations is a scoring metric introduced in 2002 used to compare a predicted sentence with the original sentence. Each predicted word is compared with the original, and each word is called an unigram or a 1-gram. In longer sentences we can also compare word pairs or bigrams. Here, we calculated BLEU-1 for unigram comparison, BLEU-2 for the bigram comparison, BLEU-3 for 3-gram comparison and BLEU-4 for 4-gram comparison.

In order to compare the predicted IUPAC name with the original IUPAC name a sentence to sentence comparison should be done, so we used the sentence BLEU scoring function inbuilt in Python Natural Language Toolkit [28]. We use the original IUPAC name as the reference string and the predicted IUPAC name as the candidate string to calculate the BLEU scores.

For all BLEU calculations we used the NLTK sentence BLEU scoring function [24].

Weight distributions for different BLEU scores,

BLEU-1: weights	(1.0, 0, 0, 0)
BLEU-2: weights	(0.5, 0.5, 0, 0)
BLEU-3: weights	(0.3, 0.3, 0.3, 0)
BLEU-4: weights	(0.25, 0.25, 0.25, 0.25).

BLEU score can reduce according to the following,

- each wrong word match
- each wrong n-gram matches
- length of the candidate string is longer/shorter than reference string
- order of the predicted words are wrong.

For these a penalty will be awarded so the overall score will decrease. A few examples are given below.

Reference: 1,3,7-trimethylpurine-2,6-dione
Candidate: 1,3,7-trimethylpurine-2,6-dione
BLEU score: **1.0**
BLEU-1: 1.00

BLEU-2: 1.00
BLEU-3: 1.00
BLEU-4: 1.00

Wrong word

Reference: 1,3,7-tri methyl purine-2,6-di one
Candidate: 1,3,7-tri methyl purine-2,6-tri one
BLEU score: **0.87**
BLEU-1: 0.94
BLEU-2: 0.90
BLEU-3: 0.90
BLEU-4: 0.88

Wrong word pair

Reference: 1,3,7-tri methyl purine-2,6-di one
Candidate: 1,3,7-tri methyl purine-2,6,tri one
BLEU score: **0.81**
BLEU-1: 0.88
BLEU-2: 0.84
BLEU-3: 0.84
BLEU-4: 0.81

Shorter prediction

Reference: 1,3,7-tri methyl purine-2,6-di one
Candidate: 1,3,7-tri methyl purine-2
BLEU score: **0.63**
BLEU-1: 0.63
BLEU-2: 0.63
BLEU-3: 0.63
BLEU-4: 0.63

Longer prediction

Reference: 1,3,7-tri methyl purine-2,6-di one
Candidate: 1,3,7-tri methyl purine-2,6-di one, 6-di one, 6-di one
BLEU score: **0.52**
BLEU-1: 0.63
BLEU-2: 0.59
BLEU-3: 0.59
BLEU-4: 0.52

Wrong order of predictions

Reference: 1,3,7-tri methyl purine-2,6-di one
Candidate: 1,3,7-tri methyl purine-6,2-di one
BLEU score: **0.71**
BLEU-1: 1.00
BLEU-2: 0.86
BLEU-3: 0.80
BLEU-4: 0.71

For the BLEU score calculation, we are using the default settings of sentence BLEU. This corresponds to a four-gram comparison. The weights are distributed evenly. In very few cases as reported in the Results section, we encountered the predictions with BLEU 1.0 where the strings were not identical. The problem can be rectified using more N-gram comparisons with different weight distributions. In our results these cases were very low in number so we used the default settings.

Reference: 4-[(4-amino-2,3,**6**-tri methyl phenyl) methyl]-2,3,**5**-tri methyl aniline
Candidate: 4-[(4-amino-2,3,**5**-tri methyl phenyl) methyl]-2,3,**6**-tri methyl aniline
With sentence BLEU, 4-gram (weights (0.25,0.25,0.25,0.25))
BLEU score: 1.00
With sentence BLEU, 5-gram (weights (0.2,0.2,0.2,0.2,0.2))
BLEU score: 0.98
With sentence BLEU, 8-gram (weights (0.125,0.125, 0.125,0.125,0.125,0.125,0.125,0.125))
BLEU score: 0.88.

Abbreviations

BLEU: BilinguaL Evaluation Understudy; BERT: Bidirectional encoder representations from transformers; CDK: Chemistry development kit; DECIMER: Deep lEarning for Chemical Image Recognition; FTP: File transfer protocol; GPU: Graphics processing unit; IUPAC: International Union of Pure and Applied Chemistry; InChI: International chemical identifier; NMT: Neural machine translation; OPSIN: Open parser for systematic IUPAC nomenclature; RAM: Random access memory; RNN: Recurrent neural network; SDF: Structure data file; SELFIES: Self-referencing embedded strings; SMARTS: SMILES arbitrary target specification; SMILES: Simplified molecular-input line-entry system; STOUT: Smiles TO iUpac Translator; TPU: Tensor processing units; TFRecord: TensorFlow Record file; VRAM: Video random access memory.

Acknowledgements

We cordially acknowledge the company ChemAxon for making their deterministic IUPAC name generator available for free for academic purposes, without which this project would not have been possible. We are also grateful for the company Google making free computing time on their Tensorflow Research Cloud infrastructure available to us.

Authors' contributions

KR developed the software and performed the data analysis. CS and AZ conceived the project and supervised the work. All authors contributed to the manuscript. All authors read and approved the final manuscript.

Funding

Open Access funding enabled and organized by Projekt DEAL. The authors acknowledge funding by the Carl-Zeiss-Foundation.

Availability of data and materials

The code for STOUT and the trained models are available at https://github.com/Kohulan/Smiles-TO-iUpac-Translator.

Declarations

Competing interests

AZ is co-founder of GNWI-Gesellschaft für naturwissenschaftliche Informatik mbH, Dortmund, Germany.

Author details

[1]Institute for Inorganic and Analytical Chemistry, Friedrich-Schiller-University Jena, Lessingstr. 8, 07743 Jena, Germany. [2]Institute for Bioinformatics and Chemoinformatics, Westphalian University of Applied Sciences, August-Schmidt-Ring 10, 45665 Recklinghausen, Germany.

Received: 21 December 2020 Accepted: 19 April 2021
Published online: 27 April 2021

References

1. Contributors to Wikimedia projects (2004) List of chemical compounds with unusual names. https://en.wikipedia.org/wiki/List_of_chemical_compounds_with_unusual_names. Accessed 1 Dec 2020
2. Favre HA, Powell WH (2013) Nomenclature of Organic Chemistry: IUPAC Recommendations and Preferred Names 2013. Royal Society of Chemistry, London
3. Nomenclature of Inorganic Chemistry – IUPAC Recommendations 2005. Chem Int 27:25–26
4. Inczedy J, Lengyel T, Ure AM, Gelencsér A, Hulanicki A, Others, (1998) Compendium of analytical nomenclature. Blackwell Science, Hoboken
5. Union internationale de chimie pure et appliquée. Physical, International Union of Pure and Applied Chemistry. Physical and Biophysical Chemistry Division (2007) Quantities, Units and Symbols in Physical Chemistry. Royal Society of Chemistry
6. Weininger D (1988) SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. J Chem Inf Comput Sci 28:31–36
7. Heller SR, McNaught A, Pletnev I, Stein S, Tchekhovskoi D (2015) InChI, the IUPAC international chemical identifier. J Cheminform 7:23
8. Homer RW, Swanson J, Jilek RJ, Hurst T, Clark RD (2008) SYBYL line notation (SLN): a single notation to represent chemical structures, queries, reactions, and virtual libraries. J ChemInf Model 48:2294–2307
9. Wiswesser WJ (1954) A line-formula chemical notation. Thomas Crowell Company publishers, Washington
10. Website. Daylight Inc. 4. SMARTS—a language for describing molecular patterns. http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html. Accessed 16 Dec 2020
11. ChemAxon - Software Solutions and Services for Chemistry & Biology. https://www.chemaxon.com. Accessed 23 Nov 2020
12. Steinbeck C, Han Y, Kuhn S, Horlacher O, Luttmann E, Willighagen E (2003) The chemistry development kit (CDK): an open-source Java library for chemo- and bioinformatics. J Chem Inf Comput Sci 43:493–500
13. Website. RDKit: open-source cheminformatics. https://www.rdkit.org. Accessed 26 Nov 2020
14. O'Boyle NM, Banck M, James CA, Morley C, Vandermeersch T, Hutchison GR (2011) Open Babel: an open chemical toolbox. J Cheminform 3:33
15. Kim S, Chen J, Cheng T et al (2019) PubChem 2019 update: improved access to chemical data. Nucleic Acids Res 47:D1102–D1109
16. Rajan K, Zielesny A, Steinbeck C (2020) DECIMER: towards deep learning for chemical image recognition. J Cheminform 12:65. https://doi.org/10.1186/s13321-020-00469-w
17. O'Boyle N, Dalke A DeepSMILES: An Adaptation of SMILES for Use in Machine-Learning of Chemical Structures. Doi: https://doi.org/10.26434/chemrxiv.7097960
18. Krenn M, Häse F, Nigam A, Friederich P, Aspuru-Guzik A (2020) Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. Mach Learn: Sci Technol 1:045024
19. Luong M-T, Pham H, Manning CD (2015) Effective Approaches to Attention-based Neural Machine Translation. arXiv:1508.04025[cs.CL]
20. Bahdanau D, Cho K, Bengio Y (2014) Neural Machine Translation by Jointly Learning to Align and Translate. arXiv:1409.0473[cs.CL]
21. Abadi M, Agarwal A, Barham P, et al (2016) TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. arXiv:1603.04467[cs.DC]
22. Papineni K, Roukos S, Ward T, Zhu W-J (2002) BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the 40th annual meeting of the Association for Computational Linguistics. pp 311–318
23. Lowe DM, Corbett PT, Murray-Rust P, Glen RC (2011) Chemical name to structure: OPSIN, an open source solution. J ChemInf Model 51:739–753
24. nltk.translate package — NLTK 3.5 documentation. https://www.nltk.org/api/nltk.translate.html. Accessed 18 Mar 2021
25. Devlin J, Chang M-W, Lee K, Toutanova K (2018) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805v2[cs.CL]
26 Krasnov L, Khokhlov I, Fedorov M, Sosnin S (2021) Struct2IUPAC – transformer-based artificial neural network for the conversion between chemical notations. ChemRxiv. https://doi.org/10.26434/chemrxiv.13274732.v2
27 Handsel J, Matthews B, Knight N, Coles S (2021) Translating the molecules: adapting neural machine translation to predict IUPAC names from a chemical identifier. ChemRxiv. https://doi.org/10.26434/chemrxiv.14170472.v1
28. Bird S, Klein E, Loper E (2009) Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. O'Reilly Media Inc, Newton

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

BMC

3.7 Publication G

DECIMER 1.0: Deep Learning for Chemical Image Recognition using Transformers

Rajan, K., Zielesny, A. & Steinbeck, C.

Submitted on: 29.05.2021 to J Cheminform
Preprint published Chemrxiv: 29.05.2021
DOI: 10.26434/chemrxiv.14479287.v1

[illegible] [illegible]			
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	x	x	x
[illegible]	x	x	x
[illegible]	x		
[illegible]	x		
[illegible]	x		
[illegible]	x	x	x
[illegible]	1.0		

Journal of Cheminformatics

DECIMER 1.0: Deep Learning for Chemical Image Recognition using Transformers

--Manuscript Draft--

Manuscript Number:	CHIN-D-21-00055	
Full Title:	DECIMER 1.0: Deep Learning for Chemical Image Recognition using Transformers	
Article Type:	Research article	
Funding Information:	Carl-Zeiss-Foundation	Mr Kohulan Rajan Prof. Dr. Christoph Steinbeck
Abstract:	The amount of data available on chemical structures and their properties has increased exponentially over the past decades. In particular, articles published before the mid-1990 are available only in printed or scanned form. The extraction and storage of data from those articles in a publicly accessible database are desirable, but doing this manually is a slow and error-prone process. In order to extract chemical structure depictions and convert them into a computer-readable format, optical chemical structure recognition (OCSR) tools were developed where the best performing OCSR tools are mostly rule-based. The DECIMER (Deep lEarning for Chemical ImagE Recognition) project was launched to address the OCSR problem with the latest computational intelligence methods to provide an automated open-source software solution. Various current deep learning approaches were explored to seek a best-fitting solution to the problem. In a preliminary communication, we outlined the prospect of being able to predict SMILES encodings of chemical structure depictions with about 90% accuracy using a dataset of 50-100 million molecules. In this article, the new DECIMER model is presented, a transformer-based network, which can predict SMILES with above 96% accuracy from depictions of chemical structures without stereochemical information and above 89% accuracy for depictions with stereochemical information.	
Corresponding Author:	Christoph Steinbeck Friedrich-Schiller-Universitat Jena Jena, State GERMANY	
Corresponding Author E-Mail:	christoph.steinbeck@uni-jena.de	
Corresponding Author Secondary Information:		
Corresponding Author's Institution:	Friedrich-Schiller-Universitat Jena	
Corresponding Author's Secondary Institution:		
First Author:	Kohulan Rajan	
First Author Secondary Information:		
Order of Authors:	Kohulan Rajan	
	Achim Zielesny	
	Christoph Steinbeck	
Order of Authors Secondary Information:		
Suggested Reviewers:		
Opposed Reviewers:		
Additional Information:		
Question	**Response**	

Powered by Editorial Manager® and ProduXion Manager® from Aries Systems Corporation

Manuscript Click here to access/download;Manuscript;DECIMER.pdf

Click here to view linked References

DECIMER 1.0: Deep Learning for Chemical Image Recognition using Transformers

Kohulan Rajan[1], Achim Zielesny[2] & Christoph Steinbeck[1*]

[1]Institute for Inorganic and Analytical Chemistry, Friedrich-Schiller-University Jena, Lessingstr. 8, 07743 Jena, Germany

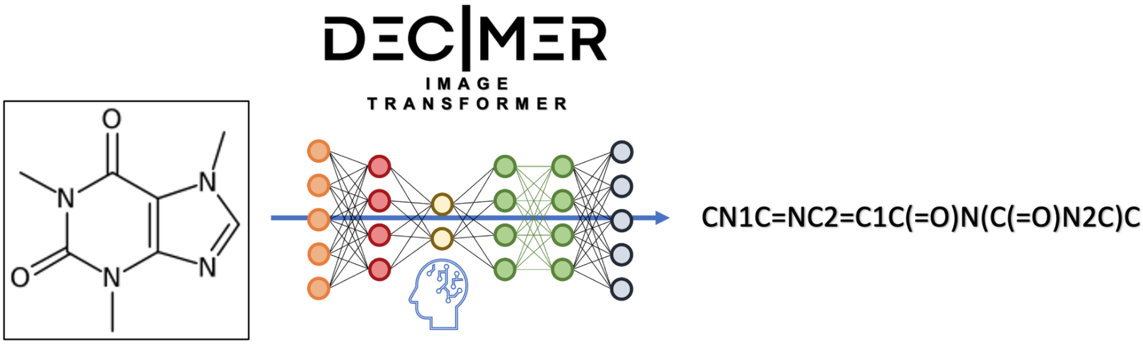
[2]Institute for Bioinformatics and Chemoinformatics, Westphalian University of Applied Sciences, August-Schmidt-Ring 10, D-45665 Recklinghausen, Germany

*Corresponding author email: christoph.steinbeck@uni-jena.de

Abstract

The amount of data available on chemical structures and their properties has increased exponentially over the past decades. In particular, articles published before the mid-1990 are available only in printed or scanned form. The extraction and storage of data from those articles in a publicly accessible database are desirable but doing this manually is a slow and error-prone process. In order to extract chemical structure depictions and convert them into a computer-readable format, optical chemical structure recognition (OCSR) tools were developed where the best performing OCSR tools are mostly rule-based.

The DECIMER (Deep lEarning for Chemical ImagE Recognition) project was launched to address the OCSR problem with the latest computational intelligence methods to provide an automated open-source software solution. Various current deep learning approaches were explored to seek a best-fitting solution to the problem. In a preliminary communication, we outlined the prospect of being able to predict SMILES encodings of chemical structure depictions with about 90% accuracy using a dataset of 50-100 million molecules. In this article, the new DECIMER model is presented, a transformer-based network, which can predict SMILES with above 96% accuracy from depictions of chemical structures without stereochemical information and above 89% accuracy for depictions with stereochemical information.



Keywords

Chemical data extraction, Deep learning, Neural networks, Optical chemical structure recognition

Introduction

Scientists build on the results of their peers. Knowledge and data arising from previous research is shared through scientific publications and increasingly through the deposition of data in repositories. To enable progress in core areas of chemistry, the availability of open data has a beneficial impact [1]. Most of the chemical data is published in the form of text and images in scientific publications [2]. Retrieving and storing published information into open-access databases will facilitate the reuse as well as the development of new methods and products [3]. But most of the data published is non-machine readable and manual curation is still the standard. This manual work is tedious and error-prone [4]. The increase of publications with valuable chemical information [5] does encourage the developments of tools for automated data retrieval. Information retrieval with corresponding database storage is an ongoing task and multiple projects are working towards this. The CHEMDNER [6] challenge is one good example of it.

There has been a significant amount of development in the field of chemical data mining [5] with a couple of open source solutions including ChemDataExtracter [4] and ChemSchematicResolver (CSR) [7], building upon each other. CSR can't process scanned pages of the publications and doesn't work on all types of publications. Although most publishers offer documents in markup format, many of the older publications are stored in scanned PDF files. For example, the Journal of Natural Products did publish scientific articles since 1978, one of their issues even dates back to 1949; however, these publications were not formatted in markup format. So, retrieving this information is a difficult process.

Image mining methods for chemical structure depictions and their conversion into a machine-readable file format is a comparatively small research area [8]. The automatic recognition of chemical structure depictions and their conversion into machine-readable formats such as SMILES [9] or InChI [10], however, is an important task for creating corresponding databases.

The publications include chemical structure depictions along with other information in textual format and contain some information presented as tables, graphs, spectra, etc.

Optical Chemical Structure Recognition (OCSR) software was built to parse chemical structure depictions. However, most of these softwares\tools are unable to handle whole page articles or scanned ones. In order to use these tools, it is necessary to segment the chemical structure depictions into separate images from printed literature and then use these segmented images as inputs. Also, the user should ensure that the image does not contain any other elements or artefacts other than a representation of a chemical structure in a segmented image. All of the available systems vary in their accuracy, OSRA [11] and MolVec [11, 12] can resolve a chemical structure with 80-90% accuracy [8].

With the advancements in computer vision, a few deep learning-based OCSR tools have been developed, e.g., by Staker et al. [13], the first machine learning-based system for segmentation of images and resolution into a computer-readable format. Another deep learning-based work is Chemgrapher [14], where multiple neural networks are combined for the recognition of molecules. Recently, there was a new publication called ChemPix [15], a deep learning-based method that was developed to recognize hand drawn hydrocarbon chemical structures. Another recent publication describes SMILES generation from images [16] where an encoder-decoder method with a pre-trained decoder is used from previous work [17]. These contributions demonstrate an increasing interest in this field of research. Even though they all claim to provide enhanced accuracy, none of them is accessible to the general public to date.

The DECIMER (Deep lEarning for Chemical IMagE Recognition) project [18] is an end-to-end open-source system that can perform chemical structure segmentation on scanned scientific literature and use the segmented structure depictions to convert them into a computer-readable molecular file format.

In our work on DECIMER-Segmentation [19], the segmentation workflow was specifically addressed. Here we now present a transformer-based algorithm that converts the bitmap of a chemical structure depiction into a computer-readable format. The system does not inherit any rules or make any assumptions; thus, it solely relies on the chemical structure depiction to perform its task.

The DECIMER algorithm was primarily inspired by the successful AlphaGo Zero algorithm [20] developed by Google’s DeepMind. The success of AlphaGo Zero allowed us to realize that very challenging problems could be adequately tackled by having a sufficient amount of data and using an adequate neural network architecture. With dozens of millions of molecules available in the databases like PubChem [21], Zinc20 [22], and GDB-17 [23], we have shown in our preliminary communication that our goal to have a system that can work with about 90% accuracy, could be achieved by training the network on a dataset of 50-100 million molecules.

Materials and Methods

DECIMER is a completely data-driven solution to chemical image recognition. Recent impressive applications of deep learning, such as the AlphaGo Zero example, all relied on the availability of very large to unlimited amounts of training data. In our case, one of the largest chemical databases on the planet, PubChem [21], was used.

Data Preparation

The latest version of PubChem was downloaded from their FTP site. All explicit hydrogens were removed using the CDK [24] and isomeric SMILES [9] were generated, which inherit the canonicalisation and retain the stereochemistry information. After generating the SMILES, the following set of rules were used to filter the dataset for a balanced dataset. The molecules in both training and test set should,

- have a molecular weight of fewer than 1500 Daltons,
- not possess counter ions,
- only contain the elements C, H, O, N, P, S, F, Cl, Br, I, Se and B,
- not contain isotopes of Hydrogens (D, T),
- have 3 - 40 bonds,
- not contain any charged groups,
- only contain implicit hydrogens, except in functional groups,
- have less than 40 SMILES characters,
- no stereochemistry is allowed.

The resulting main dataset contains 39 million molecules. The same rule set was used to generate a second dataset, but the molecules with charged groups and stereochemistry were retained. Furthermore, the molecules containing tokens that were rare in the dataset were removed (see section on tokenization), resulting in a dataset that contains approximately 37 million molecules.

Molecular bitmap images were generated using the CDK Structure Diagram Generator (SDG). The CDK depiction generator enables the generation of production-quality 2D images. In this work, every molecule was randomly rotated and depicted as 8 Bit PNG images with a 299x299 resolution. It was made sure that each image contains only one structure.

Using the set of images from the second dataset and introducing image augmentations the third dataset was generated. The image augmentations were applied using the imgaug [25] python package. One of the following augmentations were randomly applied to the images.

- Gaussian Blur
- Average Blur
- Additive Gaussian Noise
- Salt And Pepper
- Salt
- Pepper
- Coarse Dropout
- Gamma Contrast
- Sharpen

- Enhance Brightness

Often, deep learning in chemistry is using SMILES as a textual representation of structures. Training Neural Networks (NNs) directly with SMILES, however, has pitfalls: In order to generate tokens, a set of rules has to be set up on how and where to split long strings of SMILES into smaller words. After training, invalid SMILES are often encountered in the predictions, which results in overall significantly reduced accuracy. To tackle this problem there are two new text representations named DeepSMILES [26] and SELFIES [27]. DeepSMILES exhibited better results in comparison to standard SMILES, but again invalid DeepSMILES cause similar problems. In the end, SELFIES were used, since they can be split easily into tokens and they translate into a SMILES string without any errors. All SMILES strings in our 3 datasets were converted into SELFIES using Python.

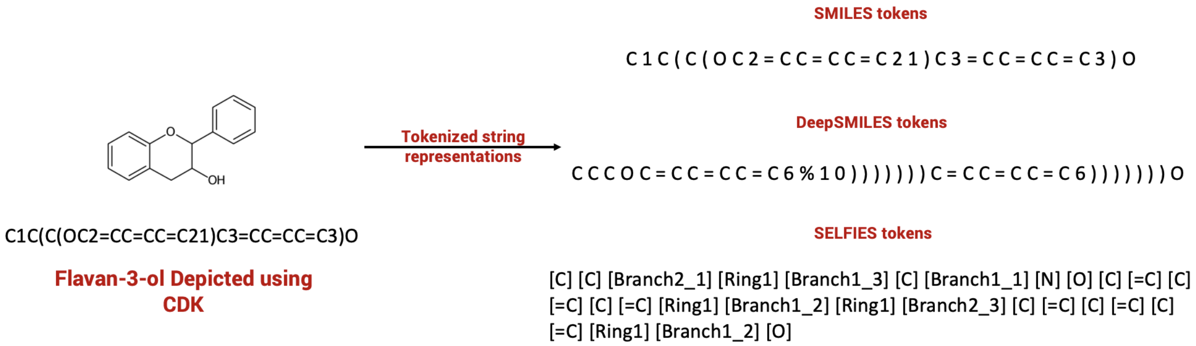

SMILES tokens
C 1 C (C (O C 2 = C C = C C = C 2 1) C 3 = C C = C C = C 3) O
DeepSMILES tokens
C C C O C = C C = C C = C 6 % 1 0))))))) C = C C = C C = C 6))))))) O
SELFIES tokens
[C] [C] [Branch2_1] [Ring1] [Branch1_3] [C] [Branch1_1] [N] [O] [C] [=C] [C] [=C] [C] [=C] [Ring1] [Branch1_2] [Ring1] [Branch2_3] [C] [=C] [C] [=C] [C] [=C] [Ring1] [Branch1_2] [O]
Tokenized string representations
O
OH
C1C(C(OC2=CC=CC=C21)C3=CC=CC=C3)O
Flavan-3-ol Depicted using CDK

Figure 1: SMILES, DeepSMILES and SELFIES represented as tokens with a space character as a delimiter.

To summarize, the datasets used in this work are:

1. Dataset 1: PNG images of chemical structure depictions plus corresponding canonical SMILES converted into SELFIES, without stereochemical information and charged groups.
2. Dataset 2: PNG images of chemical structure depictions plus corresponding canonical SMILES converted into SELFIES, with stereochemical information and charged groups.
3. Dataset 3: Augmented PNG images of chemical structure depictions plus corresponding canonical SMILES converted into SELFIES, with stereochemical information and charged groups.

Image feature extraction

A Convolutional Neural Network (CNN) is used to parse the images, where the second last layer retains the features to be extracted for calculations. For training our model, we evaluated InceptionV3 [28] and EfficientNet-B3 [29], see Figure 2. The EfficientNet base model for an image size of 299 x 299 outperforms InceptionV3 in our task at hand [29].

Every image was scaled to a resolution of 299x299 pixels, and the pixel values were normalized to interval -1 to 1, which corresponds to the format used on InceptionV3 and EfficientNet-B3. Then the features were extracted into a vector format using the pre-trained weights of ImageNet [30] on InceptionV3 and pre-trained weights of Noisy-student [31] training on EfficientNet-B3.

For Inception V3 a feature vector size of 8x8x2048 was used and for EfficientNet-B3 a feature vector size of 10x10x1536 was obtained. These extracted feature vectors can be saved into NumPy arrays.

Tokenization

SELFIES were tokenized into a unique set of tokens and padded to fit the maximum length of SELFIES strings. Here the Keras [32] tokenizer in Tensorflow 2.3 [33] was used. Table 1 summarizes the details regarding the tokens present in each dataset.

Table 1: SELFIES tokens and maximum length found on each dataset.

Datasets	**Number of SELFIES Tokens**	**The maximum length of the SELFIES strings**
Dataset 1	27	47
Dataset 2	61	47
Dataset 3	61	47

Tokens in Dataset 1: [C], [=C], [Branch1_1], [N], [Ring1], [O], [Branch1_2], [Expl=Ring1], [=N], [Branch2_1], [Branch1_3], [Ring2], [S], [F], [=O], [Branch2_2], [Cl], [Branch2_3], [#C], [Br], [P], [=S], [I], [=P], [Expl=Ring2], [B], [#N]

Tokens in Dataset 2 & 3: [C], [=C], [Branch1_1], [Branch1_2], [Ring1], [N], [O], [=O], [=N], [Ring2], [Branch2_1], [S], [Branch1_3], [F], [Branch2_2], [Cl], [Branch2_3], [Br], [#C], [/C], [#N], [P], [C@Hexpl], [C@@Hexpl], [=N+expl], [=S], [=N-expl], [I], [O-expl], [N+expl], [\C], [/N], [/O], [C@expl], [B], [C@@expl], [\N], [Expl/Ring1], [\O], [NH+expl], [I-expl], [Expl\Ring1], [P+expl], [NH2+expl], [/Cl], [/S], [NH3+expl], [Cl-expl], [/F], [#N+expl], [C-expl], [\S], [N-expl], [=NH+expl], [=I], [S-expl], [\Cl], [S+expl], [#C-expl], [B-expl], [/Br]

Generating TFRecords

Extracted feature vectors and tokenized data must be converted into TFRecords before training the models on Tensor Processing Units (TPU) [34]. TFRecords stores the data in binary format which allows training the models faster using GPUs and TPUs. The TPUs are currently available through the Google Cloud Platform. TFRecords are stored in a Google cloud bucket for training. This reduces the training time significantly and reduces the overhead of loading the data and performing the calculations on a TPU.

Using a custom python script all the datasets were converted into 75 MB chunks of TFRecords. Each TFRecord contains 128 Datapoints (128 image vectors + 128 tokenized strings).
After generating the TFRecords locally, they were moved to a Google cloud storage bucket.

Networks

In this work, two different types of networks were evaluated. Initially, an encoder-decoder model was tested, which is based on the work by Google on their *Show, Attend and Tell* [35] publication. The network eventually selected is a transformer-based model based on the *Attention is all you need [36]* publication by Google. The models are written using Python and Tensorflow 2.3 as a backend.

Encoder-decoder network

The encoder-decoder network used is an unaltered implementation by the TensorFlow team [37]. The model uses a CNN-based encoder with a ReLU activation function, a soft attention mechanism introduced by Bahdanau et al. [38] and the RNN based decoder uses Gated Recurrent Units (GRU) and two fully connected layers. The decoder consists of 1024 units and an embedding size of 512.

The network is trained using an Adam optimizer [39] with a learning rate of 0.0005 throughout all learning epochs. The loss is calculated using sparse categorical cross-entropy between real and predicted SELFIES.

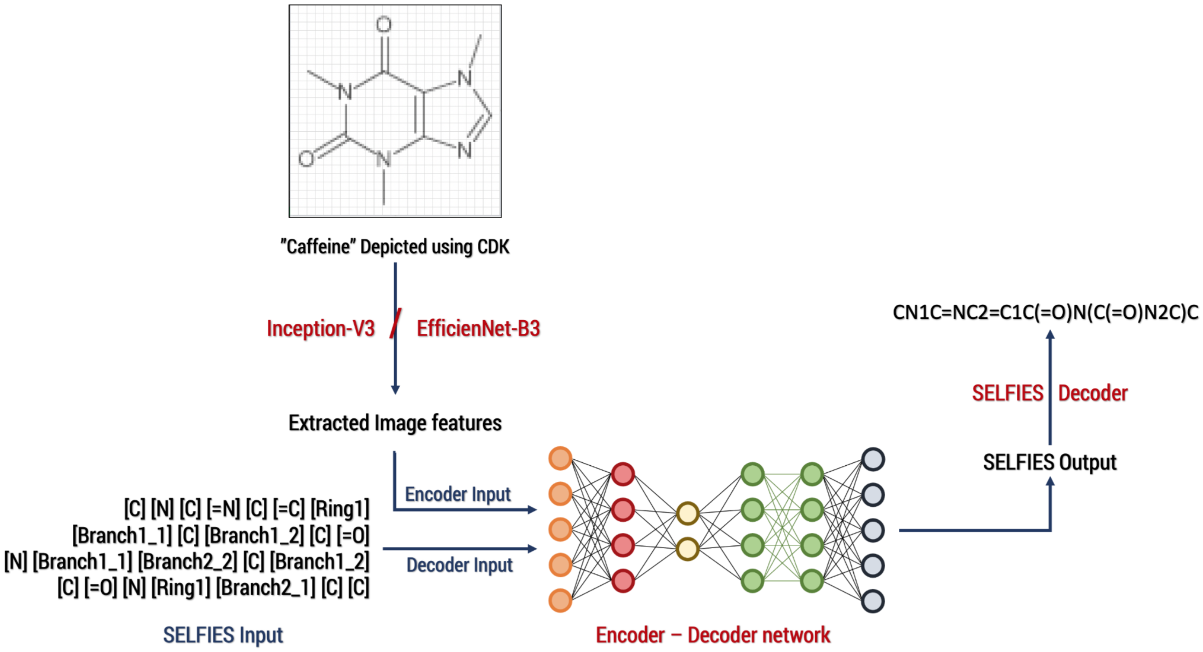

"Caffeine" Depicted using CDK
Inception-V3 / EfficienNet-B3
Extracted Image features
Encoder Input
[C] [N] [C] [=N] [C] [=C] [Ring1] [Branch1_1] [C] [Branch1_2] [C] [=O] [N] [Branch1_1] [Branch2_2] [C] [Branch1_2] [C] [=O] [N] [Ring1] [Branch2_1] [C] [C]
Decoder Input
SELFIES Input
Encoder – Decoder network
CN1C=NC2=C1C(=O)N(C(=O)N2C)C
SELFIES Decoder
SELFIES Output

Figure 2: Schema of the Encoder-Decoder network used in DECIMER for training.

Transformer network

The transformer model (Figure 3) used in this work is the model from the 2017 publication *Attention is all you need*. It uses four encoder-decoder layers and eight parallel attention heads. The attention has a dimension size of 512 and the feed-forward networks have a dimension size of 2048. Here the number of rows and columns corresponds to our image feature vectors, so for the InceptionV3, the feature vector size is eight and for the EfficientNet-B3 it is ten. A drop out of 0.1 is used to avoid overfitting.

The network is trained using an Adam optimizer with a custom learning rate scheduler according to [36]. The loss is calculated using sparse categorical cross-entropy between real and predicted SELFIES.

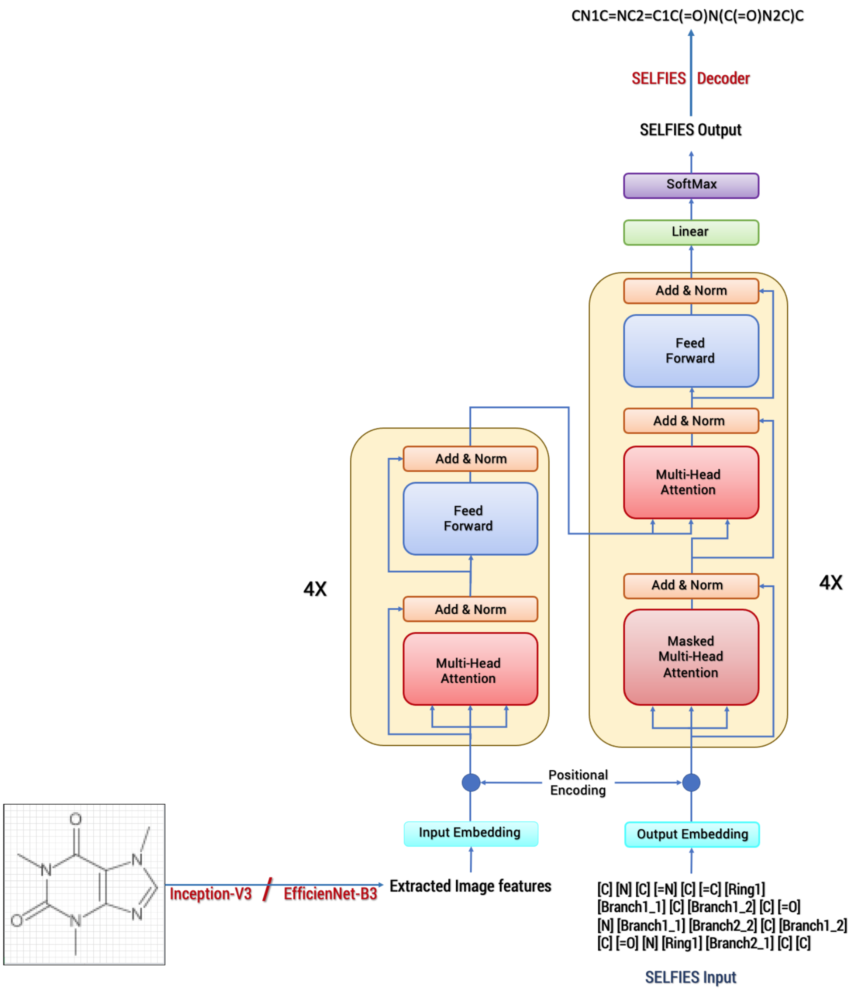


Figure 3: Schema of the Transformer network used in DECIMER for training.

Training the models

Initially, all the models were trained using an in-house server equipped with an Nvidia V100 Tesla with 32GB GPU, 384 GB of RAM and two Intel(R) Xeon(R) Gold 6230 CPUs. The details regarding the scaling & performance were explained in our previous publication [18]. For this work, a model with a dataset of one million molecules is initially trained using the same GPU equipped server. A batch size of 512 images is used to train the model, resulting in an epoch time of 29 minutes and 48 seconds, on average. For a complete convergence of the model, it took about 1 day, 5 hours and 48 minutes on the hardware mentioned above.

On a TPU v3-8 (TPU version 3 with 8 nodes) the same model was trained with a batch size of 1024 which is distributed between 8 nodes, and it took on average 8 minutes and 41 seconds per epoch and for a complete convergence of the model, it took 8 hours 41 minutes and 4 seconds.

This is a reduction of 71.9% in computation time and we, therefore, decided to train all models with the TensorFlow distributed training API using the Tensor Processing Units v3-8.

Testing the models

All the models were trained until their training loss converged, then each model was tested with a test data size of 10% of the training data. Throughout the process of selecting molecules for the test dataset, the RDKit [40] MaxMin algorithm is used to select a diverse test dataset covering the same chemical space as the training dataset.

Test dataset evaluations were performed on the GPUs. Predicted SELFIES were decoded back to SMILES and then the Tanimoto Similarity Index was calculated for the original and predicted SMILES using PubChem fingerprints, included in the CDK.

For the predictions with the Tanimoto similarity index of 1.0, we additionally generated InChIs using the CDK to perform an isomorphism check and determined, whether Tanimoto 1.0 predictions are a good proxy for structure identity.

Models trained with augmentations were tested with augmented images and with images without any augmentation.

Results and Discussion

Computational considerations

Training large datasets such as the ones used here on deep neural networks take months even on GPUs, let alone regular CPUs. For performance measure, a dataset with one million molecules was trained for 50 epochs on an Nvidia Tesla V100 GPU and the same model was also trained on a TPU V3-8 (version 3 TPU with 8 nodes) and TPU V3-32 (version 3 TPU with 32 nodes).

Training a model on a V3-8 TPU helped by increasing training speed up to 4 times compared to a V100 GPU and by using a V3-32 TPU a 16 times faster training speed was achieved, see figure 4. Concerning these results and considering the costs of V3-32 TPUs, it was decided to train all the models on a V3-8 TPU.

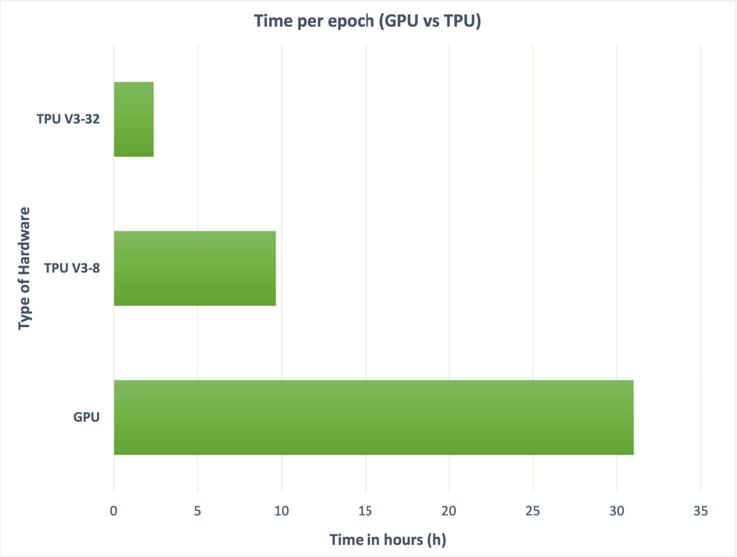


Figure 4: Training time comparison between a GPU and TPUs (lower is better)

To evaluate if testing accuracy could be improved by increasing the training dataset size, different subsets generated using dataset 1 were trained on TPU V3-8. The maximum length of SELFIES strings stayed the same throughout the training. As shown in Figure 5, training time increases with the increase in datasets.

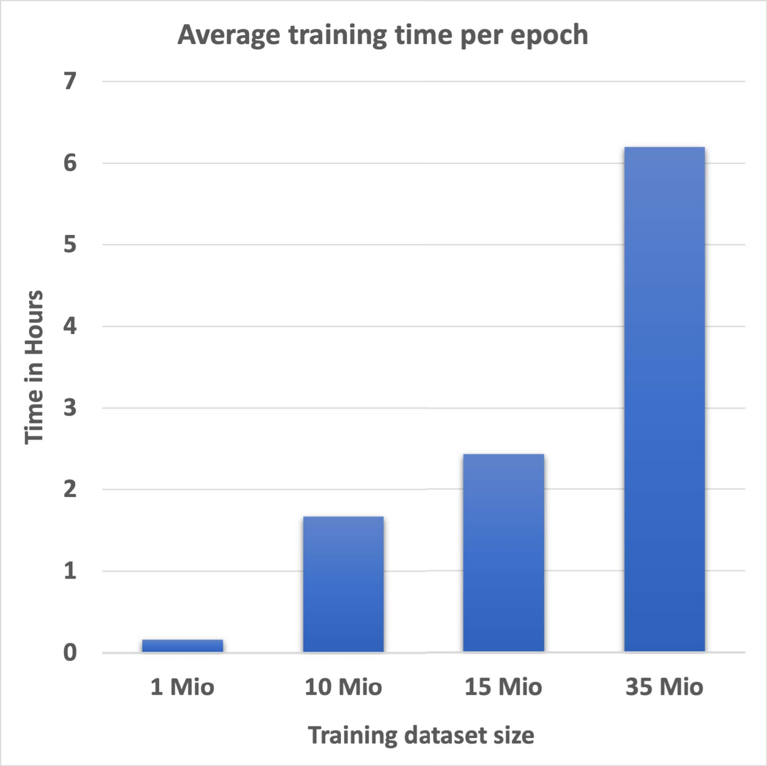


Figure 5: Average training time per epoch with increasing training dataset size.

It would take a considerable amount of time to examine the performance of the network using a bigger dataset. For the initial tests, a subset of 1 million was used, which was derived from Dataset 1. We split the dataset into 90% training data (921600) and 10% test data (102400) using the RDKIT MaxMin algorithm to ensure that the test data picked are diverse and resemble the training dataset.

Image feature extraction test

Correct extraction of the image features will result in an improved overall model at the end. In our previous work, the InceptionV3 model was used for image feature extraction. InceptionV3 is a state-of-the-art image classification network. A newer network, called EfficientNet, was created to enable better classification accuracy, and the results of noisy-student training using EfficientNet [31] were better than the InceptionV3 network. The EfficientNet-B3 model was then compared to InceptionV3 while still using the same image size (299x299) to test whether EfficientNet-based image feature extraction would improve our models' accuracy.

To compare the InceptionV3 feature extraction with EfficientNet-B3 feature extraction a subset of 1 million molecules was used. Using these models, the features were extracted and then used to train encoder-decoder based networks for 60 epochs until the training loss converged. The training time for the network that uses the features extracted using the InceptionV3 model was found to be shorter than the network which uses the EfficientNet-B3 model.

After training, the models were tested with a test dataset. The predicted SELFIES were retranslated into SMILES strings and the Tanimoto similarity index was calculated between original SMILES and the retranslated SMILES. Here, no errors have occurred in translating SELFIES to SMILES. Table 2 summarizes the evaluation.

Table 2: 1 million molecules model testing results for comparing InceptionV3 and EfficientNet-B3 feature extraction.

Metrics	**InceptionV3**	**EfficientNet-B3**
Average Training time per epoch	7mins 34secs	8mins 57Secs
Tanimoto	0.5459	0.6345
Tanimoto 1.0	1.41%	7.03%

The Tanimoto 1.0 count indicated that the EfficientNet-B3 model led to a remarkable overall performance increase, so it was used for the entire work.

Encoder-Decoder Model vs. Transformer Model

In our previous work [18], the encoder-decoder network was extensively explored. Meanwhile, great progress was made in transformer-based networks and the results seemed promising, so we decided to implement a transformer-based network in this work as well.

First, the transformer network was tested with InceptionV3 based image feature extraction, then it was tested using the EfficientNet-B3 based image feature extraction. The extracted image features with tokenized SELFIES were used as inputs for the transformer. For this work, the same 1 million molecules subset was used with a 90:10 split for training and testing.

The models were trained on TPU V3-8 until the training loss converged. The average time for transformer-based models was higher than the other, and the highest average training time was recorded for the EfficientNet-B3 Transformer network. Once the training was completed, the models were tested using the same test set. Table 3 summarizes the final evaluation.

Table 3: Comparing the encoder-decoder- and transformer-based approach with a 1 million images test dataset

	encoder-decoder		**Transformer**	
Metrics	**InceptionV3**	**EfficientNet-B3**	**InceptionV3**	**EfficientNet-B3**
Average Training time per epoch	7mins 34secs	8mins 57Secs	8mins 33secs	9Mins 27 secs
Tanimoto	0.5459	0.6345	0.8764	0.9318
Tanimoto 1.0	1.41%	7.03%	55.29%	72.29%

By comparing the Tanimoto 1.0 count, the transformer-based models clearly outperformed the encoder-decoder based models.

With these results, it was decided to train all the other datasets using transformers with image features extracted using EfficientNet-B3 based image feature extraction.

Image feature extraction comparison using EfficientNet-B3 and B7

The work described in [29] indicated that EfficientNet-B7 outperforms EfficientNet-B3 marginally by 2.7%. We, therefore, implemented EfficientNet-B7 image feature extraction and training on the extracted features. The number of parameters to train using EfficientNet-B7 (66 million parameters) compared to B3 (12 million parameters) is almost 5.5 times larger, however, which makes the network rather big and complex. Furthermore, images had to be rescaled to 600x600 for B7, in which the chemical structure depictions had to be magnified twice the normal scale. For B3, it is easy to use the images with a scale of 299x299 without any alterations.

To test these two image feature extraction methods and to see how well this helps us to achieve our main goal, a 1 million molecules image subset was used to train the transformer networks and the final models were evaluated using respective Images generated using the same test set. Table 4 summarizes the results.

Table 4: Comparison of evaluation of using EfficientNet-B3 and B7 for image feature extraction.

Metrics	**EfficientNet-B3**	**EfficientNet-B7**
Train Data Size	921,600	921,600
Test Data Size	102,400	102,400
Train Data size	0.46 TB	2.8 TB
Average training time	9Mins 27 secs	11mins 42secs
Tanimoto	0.9371	0.9669
Tanimoto 1.0	74.57%	84.82%

It is evident that the Image feature extraction using EfficientNet-B7 outperforms B3. We found, however, that most of the chemical structure depictions found on printed literature can easily fit the scale of 299x299, so to use the 600x600 scale the images should be upscaled. Upscaling will result in losing information which will be a major downside for this approach since the models majorly rely on the image features.

Chemical structure depictions larger than 299x299 square pixels can be downscaled easily to be used in our models without losing any pixel information. Thus, the size of the image was decided to be 299x299 and the feature extraction performed using EfficientNet-B3.

In future for chemical image depictions with higher resolutions using the EfficientNet-B7 image feature extraction could be applied.

The performance measure with increasing dataset size

The performance measure with increasing dataset size was done initially using the 1 million molecules subset. The data was split into different sizes (see Table 5) of train and test sets using RDKit MaxMin algorithm, and then each model was trained separately and evaluated. Table 5 summarizes the results.

Table 5: Results of training the 1 million molecules dataset with different train and test dataset sizes.

No	Train Data Size	Test Data Size	Split	Average Time per epoch	Average Tanimoto	Tanimoto 1.0
1	102,400	921,600	10|90	42.22	0.86	45.05%
2	204,800	819,200	20|80	69.95	0.91	63.59%
3	307,200	716,800	30|70	199.52	0.93	71.63%
4	409,600	614,400	40|60	276.09	0.94	73.93%
5	512,000	512,000	50|50	320.25	0.95	77.37%

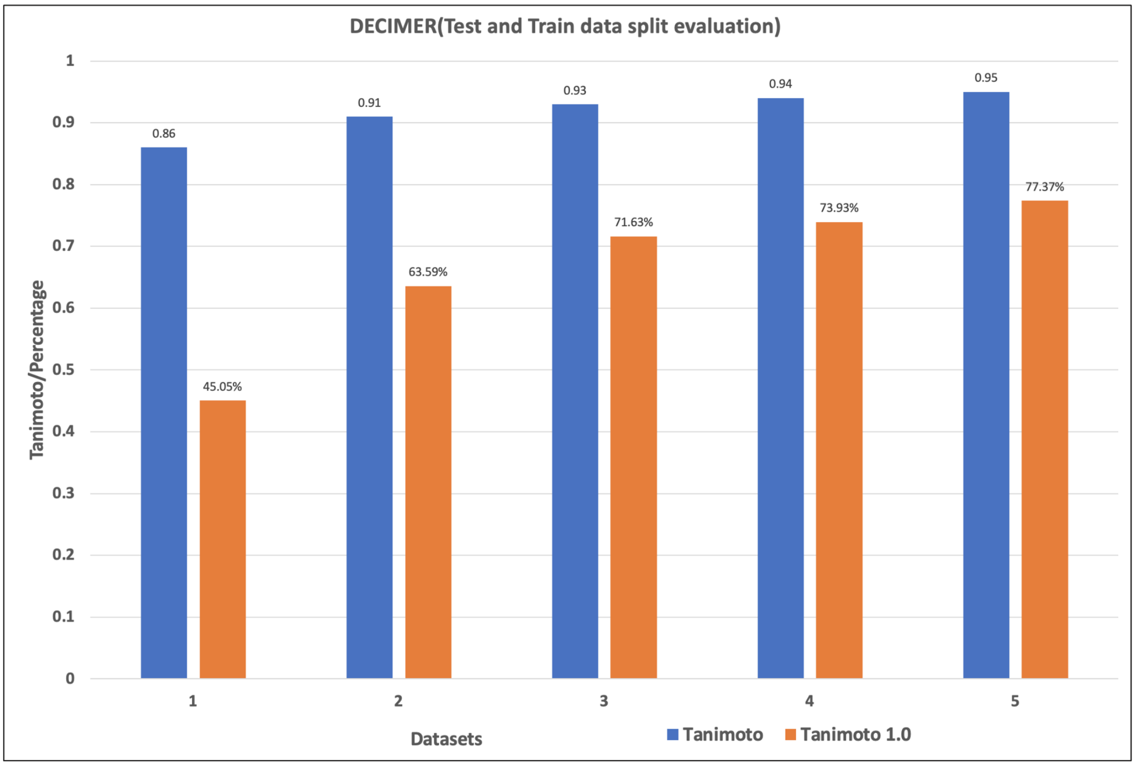


Figure 6: Average Tanimoto similarity indices and Tanimoto similarity 1.0 count with dataset number.

Figure 6 shows that model performance increases with training dataset size. To see how well the transformer performs with an increased number of data another subset of 10 million molecules images which was derived from the Dataset 1 was utilized. The image features were extracted using the InceptionV3 based network and the EfficientNet-B3 based network. Every dataset was converted into TFRecords and moved to the Google cloud. Two different models based on these two different image feature extractions were trained. After the model completed the training, they were tested using a test dataset size of 1 million molecule images of chemical structure depictions. Table 6 summarizes the results.

Table 6: Testing results of the models trained on 10 million molecule images of chemical structure depictions

Metrics	**InceptionV3**	**EfficientNet-B3**
Train Data Size	10,240,000	10,240,000
Test Data Size	1,024,000	1,024,000
Tanimoto	0.9310	0.9695
Tanimoto 1.0	74.52%	87.85%

Looking at the Tanimoto similarity average and the Tanimoto 1.0 count one can see that the dataset trained with EfficientNet-B3 based image feature extraction method outperforms the InceptionV3 based method. This also was evident in the previous training with 1 million molecule images. With these results, the next set of training included only the EfficientNet-B3 based image feature extraction.

A total of four subsets were now extracted from Dataset 1, the train and test datasets were created using the RDKit MaxMin algorithm. All four datasets included the same number of tokens. All four datasets were converted into TFRecords and stored on Google Cloud Storage Buckets and used to train the models. Table 7 summarizes the overall results for different subsets.

1. Dataset 1 : 0.9 Mio training images + 102400 test images.
2. Dataset 2: 10 Mio training images + 1 Mio test images.
3. Dataset 3: 15 Mio training images + 1.5 Mio test images.
4. Dataset 4: 35 Mio training images + 3.9 Mio test images.

Table 7: Test data results for subsets.

Metrics	**Dataset 1**	**Dataset 2**	**Dataset 3**	**Dataset 4**
Train Data Size	921,600	1,024,0000	15,360,000	35,002,240
Test Data Size	102,400	1,024,000	1,536,000	3,929,093
Tanimoto	0.9311	0.9691	0.9779	0.9923
Tanimoto 1.0	72.44%	87.88%	91.02%	96.47%

These results demonstrate an increasing trend of accurate predictions due to increasing data in the training datasets. In addition, with 35 million molecules training, we reached an average Tanimoto similarity of 0.99, along with a 96.47% Tanimoto 1.0 count. Because of using SELFIES as the input textual data, all of the predictions were successfully retranslated into valid molecules. An isomorphism check using InChIs was carried out in order to find out how many molecules in Tanimoto 1.0 are full isomorphic.

InChI strings were generated using the CDK for all the predictions with a Tanimoto similarity index of 1.0 and then checked whether they are isomorphic or not by string matching.

Table 8: Results of isomorphism calculations for the subsets of Dataset 1

Metrics	**Subset 1**	**Subset 2**	**Subset 3**	**Subset 4**
Train Data Size	921,600	10,240,000	15,360,000	35,002,240
Test Data Size	102,400	1,024,000	1,536,000	3,929,093
Predictions with Tanimoto 1.0	74,176	899,941	1,398,028	3,790,273
Isomorphic Predictions	98.63%	99.45%	99.59%	99.75%
Non-Isomorphic Predictions	1.37%	0.55%	0.41%	0.25%

Table 8 shows that 99% of all predictions which have Tanimoto 1.0 are structurally identical to the depicted molecule. Also with the increasing Training dataset size, the isomorphic structure count kept increasing slightly.

Analysis of the predictions with low Tanimoto similarity indices.

The model trained with the largest dataset 4 extract machine-readable representations of molecules depicted in the test dataset with near 100% accuracy. In order to understand why predictions with low Tanimoto scores were not predicted correctly, the following analysis was performed.

Table 9: Predicted SMILEs with lower Tanimoto similarity indices compared with the Original SMILES.

No	Original SMILES	Predicted SMILES	Tanimoto Similarity Index
1	P#CP=PP=PP=PP=PP=PP=PP=PP=PP=PP=PP=P	N#CC=NSSSSSSSSSSSSSSSSC=N	0
2	N1=NOO1	C=1=NOC1	0.14
3	OC1OC(C=2C(F)=C(F)C(F)=C(F)C21)C)C	O=C(OCOCCC(F)=C(F)C(F)=C(F)C)NC	0.35
4	OCC(C)(CO)C12CCC(C1)C3SSSC32	OCC(C)(CO)C1C=2SSSC2CCC1C	0.59
5	O=C1N=CC2=CC(=O)C=CC2=N1	O=C1N=CC2=NC(=O)C=CC2=N1	0.81

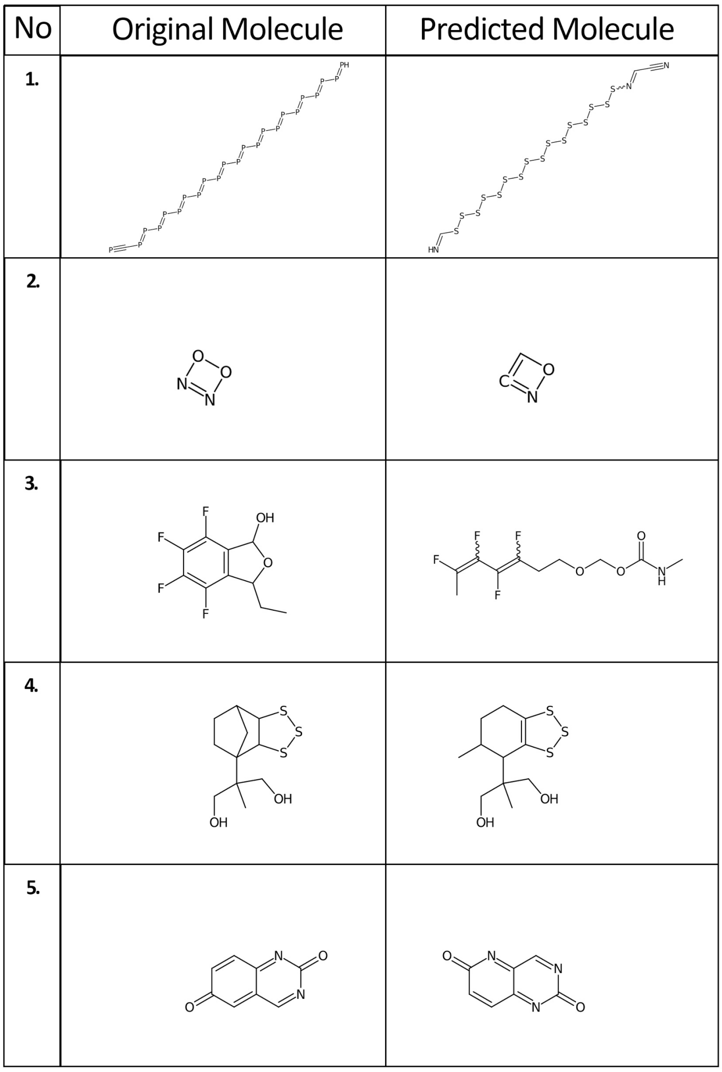


Figure 7: Depictions of chemical structures with lower Tanimoto similarity indices.

In most cases the network was able to interpret the skeleton of the chemical structure well. Semantically small errors such as the miss of a ring closure will lead to seemingly large errors in the eyes of a chemist, as can be seen in case 3.

In the majority of cases, the Tanimoto similarity was low due to the predicted SMILES,

- having one or more wrong atoms.
- missing a bond.
- having a wrong bond.
- missing an aromatic ring.

A strategy to overcome such issues could be to use multiple depictions of the same chemical structure in the training set with different rotations so that the network sees more examples of the same set of input data. Also implementing different and more image augmentation methods and training the augmented images along with the non-augmented images might enable the network to see the chemical structures clearer.

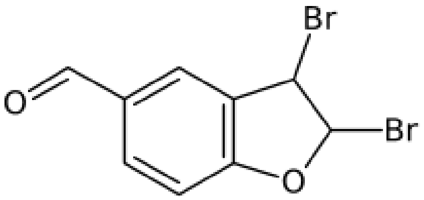
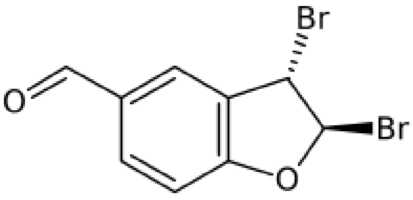
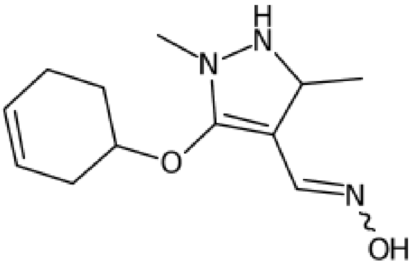
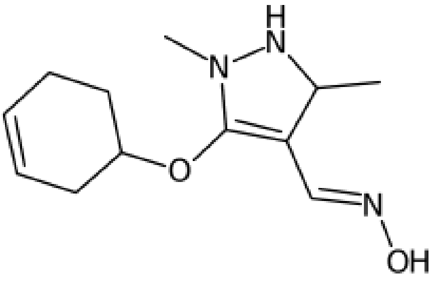
Performance of the network with training data using stereochemistry information - Dataset 2

To assess the performance of the transformer network on chemical structure depictions with stereochemistry and ions, the same dataset was used but stereochemistry and ion information were included. By including this information, the unique number of tokens increased, and the molecules with the least number of tokens were removed after the calculation of the token distribution. A new dataset with 37 Mio molecules was created and split into training and test datasets using the RDKit MaxMin algorithm. This whole dataset is called Dataset-2 from now on.

By adding stereochemical information and ions, the number of unique SELFIES tokens increased from 27 to 61, almost twice the number of the tokens found on Dataset 1. From Table 10 one could see the same molecule with and without stereochemistry and how it affects the number of tokens present in the SELFIES and the depicted structure.

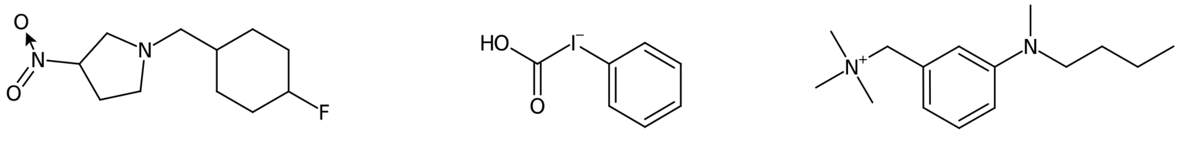
Table 10: Analysis of a molecule for with and without stereochemical information

		Molecules with stereochemical information	Molecules without stereochemical information
01.	SMILES (Canonical/Isomeric)	C1=CC2=C(C=C1C=O)C(C(O2)Br)Br	C1=CC2=C(C=C1C=O)[C@@H]([C@H](O2)Br)Br
	SELFIES	[C][=C][C][=C][Branch1_1][Branch1_3][C][=C][Ring1][Branch1_2][C][=O][C][Branch1_1][Branch2_1][C][Branch1_1][Ring2][O][Ring1][Branch2_2][Br][Br]	[C][=C][C][=C][Branch1_1][Branch1_3][C][=C][Ring1][Branch1_2][C][=O][C@@Hexpl][Branch1_1][Branch2_1][C@Hexpl][Branch1_1][Ring2][O][Ring1][Branch2_2][Br][Br]
	Number of Unique SELFIES tokens	12	14

	Depicted Structure		
02.	SMILES (Canonical/Isomeric)	CC1C(=C(N(N1)C)OC2CCC=CC2)C=NO	CC1C(=C(N(N1)C)OC2CCC=CC2)/C=N/O
	SELFIES	[C][C][C][Branch2_2][Ring1][Ring2][=C][Branch1_1][Branch2_1][N][Branch1_1][Ring2][N][Ring1][Branch1_1][C][O][C][C][C][C][=C][C][Ring1][Branch1_2][C][=N][O]	[C][C][C][Branch2_2][Ring1][Ring2][=C][Branch1_1][Branch2_1][N][Branch1_1][Ring2][N][Ring1][Branch1_1][C][O][C][C][C][C][=C][C][Ring1][Branch1_2][/C][=N][/O]
	Number of Unique SELFIES tokens	11	13
	Depicted Structure		

Inclusion of stereochemistry increased the number of tokens, but also introduced new artifacts in chemical structure depictions such as wedged and dashed bonds. Including the cis/trans information reduced the amount of curly bonds in the new dataset.

Including the information about the ions also increased the number of tokens, also this introduced new artifacts to the chemical structure depictions such as the “+,-” signs and arrows, see figure 8.



C1CC(CCC1CN2CCC(C2)[N+](=O)[O-])F

C1=CC=C(C=C1)[I-]C(=O)O

CCCCN(C)C1=CC(=CC=C1)C[N+](C)(C)C

Figure 8: Chemical structure depictions with ions

Two subsets of Data Set 2 were generated, one with the 15 million training molecules plus 1.5 million test molecules and another with 33 million training molecules plus 3.7 million test molecules. TFRecords were generated from the chemical structure depictions using these datasets and moved into Google cloud storage buckets. Finally, two models were trained using these two datasets. Table 11 summarizes the results.

Table 11: Results on for the subsets of Dataset 2.

Metrics	**Subset 1**	**Subset 2**
Train Data Size	15,360,000	33,304,320
Test Data Size	1,536,000	3,700,480
Tanimoto	0.9372	0.9761
Tanimoto 1.0	75.23%	89.87%

It can be seen from the results shown in Table 11 that the average Tanimoto is lower compared to Dataset 1 that was seen in Table 7. The Tanimoto 1.0 count is also lower. This is mainly due to the new artefacts included in the new dataset and now the number of tokens in use also doubled. Increasing the data for the newly introduced tokens can improve the results significantly. To check how many of the predicted structures are isomorphic the InChIs were generated for the original and predicted structures and a string matching was performed as explained before, see Table 12.

Table 12: Results of isomorphism calculations for the subsets of Dataset 2.

Metrics	**Subset 1**	**Subset 2**
Train Data Size	15,360,000	33,304,320
Test Data Size	1,536,000	3,700,480
Predictions with Tanimoto 1.0	1,155,483	3,325,656
Isomorphic Predictions	96.42%	98.50%
Non-Isomorphic Predictions	3.58%	1.50%

Table 12 shows that more than 96% of the predicted SMILES are isomorphic. By increasing the training dataset, the number of isomorphic structures did increase, which is similar to the results for dataset 1.

Increasing the training data points will likely increase isomorphic structure predictions in general. Due to the applied ruleset, only a limited amount of data is available to work with. Therefore, the next step will be to train these models on augmented images to assess whether or not they improve overall accuracy.

Performance of the network with training data using stereochemistry and image augmentation - Dataset 3

By applying image augmentation to Dataset-2 we generated Dataset-3. The resulting images look similar to Figure 9.

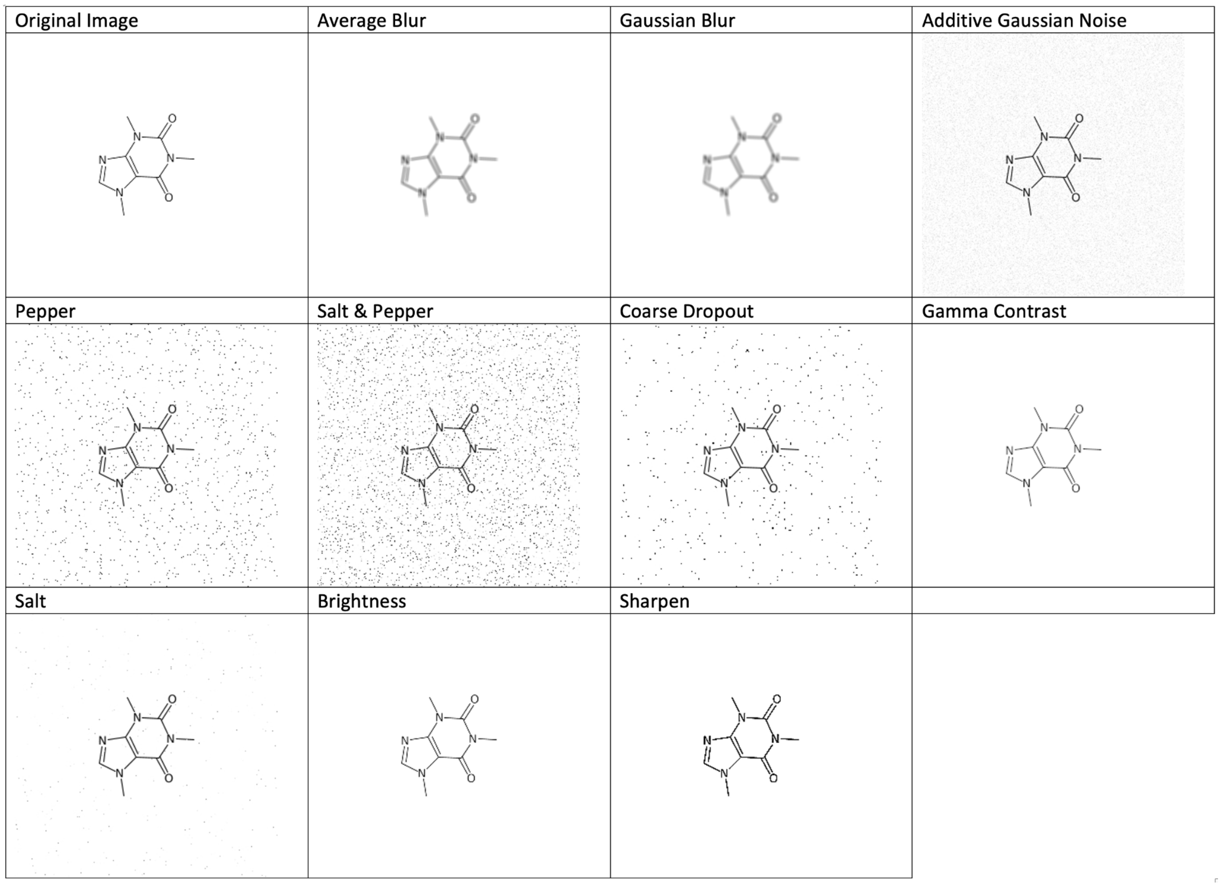


Figure 9: Images augmented with parameters within a given range.

The parameters were restricted to reflect the real-world images, not to add extreme augmentations. The parameter is shown in Table 13 during augmentations. Here the list of parameters provided is the ones that were implemented to augment the images, for more details about the parameters and how they are implemented, we refer our readers to the imgaug documentation [41].

Table 13: Image augmentations and their parameters.

Image augmentations	**Parameters (imgaug)**
Gaussian Blur	0-1.8
Average Blur	k=0-3
Additive Gaussian Noise	scale=(0, 0.1*255)
Salt and Pepper	0-0.05

Salt	0-0.05
Pepper	0-0.05
Coarse Dropout	0-0.01, size percent = 0.9
Gamma Contrast	0.5-2.0
Sharpen	alpha=(0.0, 1.0), lightness=1.0
Enhance Brightness	factor=(0.95, 1.5)

The generated dataset was then used to train two models. One model was trained from scratch using augmented images and another model was trained with dataset 2 and refitted with the augmented images. Both of them were tested on a dataset size of 4 million images, which includes 2 million images with augmentations and 2 million images without any augmentations. Table 14 summarizes the results.

The first two columns of the table explain the performance of the model trained only on augmented images and tested on augmented and non-augmented images. The last two columns summarize the evaluation of the model which was previously trained on non-augmented images and refitted with dataset 2.

In refitting, we used weights from the best model previously trained on non-augmented images instead of random weights as a starting point for training. This was done to see whether using the weights from a previously trained model would improve the performance of the newly trained model trained using a similar type of data.

Table 14: Results on Dataset 3 and Dataset 2+3

	Augmented Dataset (3)		**Non-augmented + Augmented Dataset (2+3)**	
Metrics	**Non augmented test set**	**Augmented test set**	**Non augmented test set**	**Augmented test set**
Train Data Size	33,304,320	33,304,320	33,304,320	33,304,320
Test Data Size	2,000,000	2,000,000	2,000,000	2,000,000
Tanimoto	0.9663	0.9501	0.9708	0.9521

Tanimoto 1.0	86.43%	80.26%	88.04%	80.87%
Isomorphic Predictions	97.89%	97.46%	98.15%	97.61%
Non-Isomorphic Predictions	2.11%	2.54%	1.85%	2.39%

The above results clearly show that our models were able to retain the Tanimoto average of above 0.95 and Tanimoto 1.0 of above 80%. Also, the isomorphic results are high in all cases, and this was similar to the earlier results. The overall accuracy of these models could be improved by increasing the number of Augmented and Non-Augmented training images.
Very likely, training with more data will improve the outcome.

Conclusion and Future work

In our preliminary communication [18], we claimed that with data around 50-100 million molecule images will help us obtain a model that can predict SMILES with about 90% accuracy. Here, we have now presented a solution based on a transformer network that delivers this promise.

Using the improved EfficientNet-B3 method rather than Inception-V3 for image feature extraction helped in extracting relevant features required for network training. Through the implementation of the new transformer-based models, we've been able to improve the accuracy of our Image-to-SMILES models overall.

We have achieved an accuracy level of about 96% for chemical structure depictions using DECIMER's new algorithm without stereochemistry training the network using 30-35 Million molecules.

When the models were extended to include stereochemical information and ions, a near 90% accuracy was achieved, despite increasing the number of tokens twofold. This can be further improved by increasing the data on stereochemical information and ions. This also applies to the models trained using image augmentations. In order to improve these models, more data should be incorporated into training.

With TPUs, the models could be trained within days, and the largest model took less than 14 days to train. That means even bigger models could be trained within a month using TPUs rather than training on GPUs, which may take several months to complete. It is also cost-effective as well as energy-efficient to implement the TPU solution on the Google cloud platform rather than relying on the local hardware setup.

Our results showed that DECIMER was achieving the intended objective with synthetic data. Further steps in future will include training with more data, refining models using a variety of real-world examples and image datasets with more augmentations. Additionally, training images created by using a variety of tools will contribute to the model's improved accuracy. Ultimately, the DECIMER project aims to provide an open-source tool that is capable of performing optical chemical structure recognition (OCSR) reliably on segmented images from the scanned literature.

The DECIMER software is fully open-source and hosted on GitHub. All data and trained models are openly available.

Availability of data and materials

The code for DECIMER and the trained models are available at https://github.com/Kohulan/DECIMER-TPU

Abbreviations

API - Application Programming Interface.

CDK - Chemistry Development Kit.

CNN - Convolutional Neural Network.

CSR - ChemSchematicResolver.

FTP - File Transfer Protocol.

GB - GigaByte.

GPU - Graphical Processing Unit.

GRU - Gated Recurrent Unit.

InChI - International Chemical Identifier.

DECIMER - Deep lEarning for Chemical ImagE Recognition.

MB - MegaByte.

NNs - Neural Networks.

OCSR - Optical Chemical Structure Recognition.

OSRA - Optical Structure Recognition Application.

PDF - Portable Document Format.

PNG - Portable Network Graphics.

RNN - Recurrent Neural Network.

SDG - Structure Diagram Generator.

SELFIES - Self-Referencing Embedded Strings.

SMILES - Simplified Molecular-Input Line-Entry System.

TFRecord - TensorFlow Record.

TPU - Tensor Processing Unit.

Declarations

Competing interests

AZ is co-founder of GNWI - Gesellschaft für naturwissenschaftliche Informatik mbH, Dortmund, Germany.

Funding

The authors acknowledge funding by the Carl-Zeiss-Foundation. Open Access funding enabled and organized by Projekt DEAL.

Authors' contributions

KR developed the software and performed the data analysis. CS and AZ conceived the project and supervised the work. All authors contributed to and approved the manuscript.

Acknowledgments

We are grateful for the company Google making free computing time on their TensorFlow Research Cloud infrastructure available to us.

1. Gaulton A, Overington JP (2010) Role of open chemical data in aiding drug discovery and design. Future Med Chem 2:903–907
2. O'Boyle NM, Guha R, Willighagen EL, et al (2011) Open Data, Open Source and Open Standards in chemistry: The Blue Obelisk five years on. J Cheminform 3:1–15
3. Zhou JZ (2011) Chemoinformatics and Library Design. In: Zhou JZ (ed) Chemical Library Design. Humana Press, Totowa, NJ, pp 27–52
4. Swain MC, Cole JM (2016) ChemDataExtractor: A Toolkit for Automated Extraction of Chemical Information from the Scientific Literature. J Chem Inf Model 56:1894–1904
5. Krallinger M, Rabal O, Lourenço A, Oyarzabal J, Valencia A (2017) Information Retrieval and Text Mining Technologies for Chemistry. Chem Rev 117:7673–7761
6. Krallinger M, Leitner F, Rabal O, Vazquez M, Oyarzabal J, Valencia A (2015) CHEMDNER:

The drugs and chemical names extraction challenge. J Cheminform 7:S1

7. Beard EJ, Cole JM (2020) ChemSchematicResolver: A Toolkit to Decode 2D Chemical Diagrams with Labels and R-Groups into Annotated Chemical Named Entities. J Chem Inf Model 60:2059–2072

8. Rajan K, Brinkhaus HO, Zielesny A, Steinbeck C (2020) A review of optical chemical structure recognition tools. J Cheminform 12:60

9. Weininger D (1988) SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. J Chem Inf Comput Sci 28:31–36

10. Heller SR, McNaught A, Pletnev I, Stein S, Tchekhovskoi D (2015) InChI, the IUPAC International Chemical Identifier. J Cheminform 7:23

11. Filippov IV, Nicklaus MC (2009) Optical structure recognition software to recover chemical information: OSRA, an open source solution. J Chem Inf Model 49:740–743

12. Peryea T, Katzel D, Zhao T, Southall N, Nguyen D-T (2019) MOLVEC: Open source library for chemical structure recognition. ABSTRACTS OF PAPERS OF THE AMERICAN CHEMICAL SOCIETY 258:

13. Staker J, Marshall K, Abel R, McQuaw CM (2019) Molecular Structure Extraction from Documents Using Deep Learning. J Chem Inf Model 59:1017–1029

14. Oldenhof M, Arany A, Moreau Y, Simm J (2020) ChemGrapher: Optical Graph Recognition of Chemical Compounds by Deep Learning. J Chem Inf Model 60:4506–4517

15. Weir H, Thompson K, Choi B, Woodward A, Braun A, Martínez TJ (2021) ChemPix: Automated Recognition of Hand-drawn Hydrocarbon Structures Using Deep Learning. ChemRxiv. https://doi.org/10.26434/chemrxiv.14156957.v1

16. Clevert D-A, Le T, Winter R, Montanari F (2021) Img2Mol - Accurate SMILES Recognition from Molecular Graphical Depictions. ChemRxiv. https://doi.org/10.26434/chemrxiv.14320907.v1

17. Le T, Winter R, Noé F, Clevert D-A (2020) Neuraldecipher – reverse-engineering extended-connectivity fingerprints (ECFPs) to their molecular structures. Chem Sci 11:10378–10389

18. Rajan K, Zielesny A, Steinbeck C (2020) DECIMER: towards deep learning for chemical image recognition. J Cheminform 12:65

19. Rajan K, Brinkhaus HO, Sorokina M, Zielesny A, Steinbeck C (2021) DECIMER-Segmentation: Automated extraction of chemical structure depictions from scientific literature. J Cheminform 13:1–9

20. Silver D, Schrittwieser J, Simonyan K, et al (2017) Mastering the game of Go without human knowledge. Nature 550:354–359

21. Kim S, Chen J, Cheng T, et al (2019) PubChem 2019 update: improved access to chemical data. Nucleic Acids Res 47:D1102–D1109

22. Irwin JJ, Tang KG, Young J, Dandarchuluun C, Wong BR, Khurelbaatar M, Moroz YS,

Mayfield J, Sayle RA (2020) ZINC20-A Free Ultralarge-Scale Chemical Database for Ligand Discovery. J Chem Inf Model 60:6065–6073

23. Ruddigkeit L, van Deursen R, Blum LC, Reymond J-L (2012) Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17. J Chem Inf Model 52:2864–2875

24. Steinbeck C, Han Y, Kuhn S, Horlacher O, Luttmann E, Willighagen E (2003) The Chemistry Development Kit (CDK): an open-source Java library for Chemo- and Bioinformatics. J Chem Inf Comput Sci 43:493–500

25. Jung AB, Wada K, Crall J, et al (2020) Imgaug. GitHub: San Francisco, CA, USA

26. O'Boyle N, Dalke A (2018) DeepSMILES: An Adaptation of SMILES for Use in Machine-Learning of Chemical Structures. ChemRxiv. https://doi.org/10.26434/chemrxiv.7097960.v1

27. Krenn M, Häse F, Nigam A, Friederich P, Aspuru-Guzik A (2020) Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. Mach Learn: Sci Technol 1:045024

28. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 2818–2826

29. Tan M, Le Q (2019) Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning. PMLR, pp 6105–6114

30. Deng J, Dong W, Socher R, Li L, Kai Li, Li Fei-Fei (2009) ImageNet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp 248–255

31. Xie Q, Luong M-T, Hovy E, Le QV (2020) Self-training with noisy student improves imagenet classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp 10687–10698

32. Chollet F, Others (2015) Keras. https://keras.io.

33. Martín Abadi, Ashish Agarwal, Paul Barham, et al (2015) TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.

34. Norrie T, Patil N, Yoon DH, Kurian G, Li S, Laudon J, Young C, Jouppi N, Patterson D (2021) The Design Process for Google's Training Chips: TPUv2 and TPUv3. IEEE Micro 41:56–63

35. Xu K, Ba J, Kiros R, Cho K, Courville A, Salakhudinov R, Zemel R, Bengio Y (2015) Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In: Bach F, Blei D (eds) Proceedings of the 32nd International Conference on Machine Learning. PMLR, Lille, France, pp 2048–2057

36. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention Is All You Need. arXiv [cs.CL]

37. Image captioning with visual attention.

https://www.tensorflow.org/tutorials/text/image_captioning. Accessed 17 Mar 2021

38. Bahdanau D, Cho K, Bengio Y (2014) Neural Machine Translation by Jointly Learning to Align and Translate. arXiv [cs.CL]

39. Kingma DP, Ba J (2014) Adam: A Method for Stochastic Optimization. arXiv [cs.LG]

40. Landrum G, Others (2016) RDKit: Open-Source Cheminformatics Software.(2016). URL http://www. rdkit. org/, https://github. com/rdkit/rdkit

41. dtype support — imgaug 0.4.0 documentation. https://imgaug.readthedocs.io/en/latest/source/dtype_support.html. Accessed 15 Apr 2021

4. Discussion

The last three decades have shown a growing interest in Optical Chemical Structure Recognition (OCSR) methods (**Publication A**) for mining data published in the printed literature. The number of articles in this domain has increased over time [14], making the manual extraction of information much more challenging (**Publication A**). Even though many natural products databases are available, the information they hold about specific compounds is often insufficient [145] (**Publication E**). Some of the published databases are not accessible as well [145]. The availability of an open-access database for natural products (**Publication E**) and its continued expansion with more information and data will have a significant impact on natural product research.

In recent years, interest has grown in natural products [146] as potential drug leads [147], resulting in a greater need to mine data about already known chemical compounds. A renewed interest in the development of OCSR methods resulted from this.

4.1 Disadvantages of traditional tools and the need for a deep learning-based method

Most of the existing OCSR methods are rule-based approaches, and only three of them are open-source (**Publication A**). A user needs to know how to compile *Optical Structure Recognition Application* (OSRA) in their system in order to use it. MolVec is available as a command-line application deployed as an executable JAR file, while Imago is available as a GUI application and command-line application (**Publication A**). Moreover, most tools do not have the capability to extract a chemical structure depiction automatically from a printed document. OSRA and Imago can do this manually, but MolVec does not contain a component for image segmentation.

In most cases, the chemical structure depictions must be segmented separately before being given to the OCSR methods. The recently published open-source tool ChemSchematicResolver (CSR) can segment images containing labels and chemical structure depictions if the publication used as input is in a markup format. It cannot handle scanned pages or images containing other objects than labels and structure depictions [148].

To use the other existing OCSR methods, the chemical structure depictions from printed literature need to be segmented into individual images and then fed to the tools individually to get a computer-readable file.

To assess the performance of the open-source OCSR methods OSRA, Imago, and MolVec on a given set of images of chemical structures, a benchmark study was conducted (**Publication A**). As a result, it was found that all three tools performed relatively well on a given image of a chemical structure. Most of the time, MolVec worked faster than OSRA and Imago. In most cases comparing the accuracy, MolVec and OSRA seem to be on par with each other, Imago being second to them. In a recent publication, Clevert *et. al [149]* reported that when slight distortions are introduced to the images in the same datasets used for our benchmark, the performance of the rule-based tools drops significantly. The tools have embedded rules which are not adapted to handle the distorted images. These tools may fail on a chemical structure depiction taken out of an old book for testing real-world scenarios due to different depiction styles. Furthermore, adapting these rule-based systems for the before-mentioned applications would be a complex task. Here, deep learning-based methods would be a far better option since they could be trained on a large corpus of images with various depiction styles and with various levels of augmentations to enable them to handle even distorted images.

The process of segmenting certain objects from high-resolution images using deep learning has been applied in many other areas [150], and these systems perform much better than conventional approaches. Hence, the application of a deep learning-based segmentation algorithm for the recognition and segmentation of chemical structure depictions from scanned pages led to convincing results (**Publication D**). The DECIMER-Segmentation tool lifts the restriction of the segmentation algorithm to a particular file format and allows the automated extraction of chemical structures from the printed literature.

Additionally, an OCSR method based on deep learning is beneficial because there is no need to implement any rules for image pre-processing. The neural network here is trained to recognize chemical structures based only on pixel-wise information from images (**Publications B and G**). During the training, the introduction of distortions and augmentations to the training images increased the model's overall accuracy. Consequently, the model can understand and interpret chemical structures from an image, even if they have slight distortions. Also, the network can be trained with images similar to the images found in publications by introducing image augmentations so that the model's overall accuracy can be increased. Whenever a new set of images is received, a deep learning-based model does not have to be retrained from scratch; it can be fine-tuned with a new dataset for better performance [151]. Having a model that performs with a higher accuracy is possible in the near future with the ongoing developments of deep learning methods in computer vision.

4.2 Increasing interest in the field

The need for mining information regarding chemical compounds is increasing with the increasing demand for new compounds in computer-assisted drug design. Even though there are databases [127, 152] populated with chemical compounds, most of them lack information [145]. With an increased need for data in natural product-based drug discovery, mining information on existing chemical compounds already published in printed literature has become necessary.

The existing rule-based methods perform reasonably well, but the chemical structure recognition is unsatisfactory in many cases. Improving them using rules systematically is a challenging task (**Publication A**). Here deep learning based OCSR methods provide a promising alternative. OCSR methods are most useful on publications that were printed originally but then scanned and stored as PDF files by the publisher. The most convenient tool to work on these publications is one that can handle all types of publications.

There have been some deep learning-based methods [25, 149, 153–155] published recently. This indicates a renewed interest in this field. The major problem in all of these tools is that most are proprietary and not open-source. Therefore, it was decided to build an open-source deep learning-based tool to perform OCSR tasks. It is mentioned in the preliminary communication of this work (**Publication B**) that it is possible to obtain near-perfect OCSR results on synthetic datasets by using a complete data-driven, deep learning approach. After this publication, more research groups came up with somewhat similar neural network approaches to solve this OCSR problem. They announced that their code would soon become open-source, which is much appreciable.

In recent years, the pharmaceutical industry has been using machine learning-based tools to mine the proprietary databases and literature to identify candidate molecules effective on specific drug targets [156]. *Bristol-Myers Squibb*, a pharmaceutical company, launched a Kaggle competition [157] recently, which indicates that the drug industry is highly keen on mining chemical compounds and is searching for an accurate tool for this application.

4.3 DECIMER and STOUT: Implementation and Applications

The primary goal of the DECIMER project was to create an end-to-end method for segmenting chemical structure depictions from a printed publication and translating them into a computer-readable format, such as SMILES.

As interest in deep learning-based OCSR methods continues to grow, this project aspired to develop a deep learning-based tool that would be useful and future-proof. It was decided to work on separate tools to solve the segmentation and translation problems and then combine them in the end. The DECIMER project includes DECIMER-segmentation (**Publication D**) for segmentation and DECIMER-image transformer (**Publication G**) for translation.

4.3.1 DECIMER - Segmentation

The DECIMER-Segmentation tool works exclusively on images. This was a conscious choice due to the convertibility of other formats into high-resolution images using Python packages, such as pdf2image [158]. It is much easier to train a network that can recognise a chemical structure in a high-resolution image and segment it out. With the development in semantic segmentation using deep learning, it is possible to segment an object out of an image with near-perfect accuracy [150]. Additionally, this project was focused on segmenting chemical structure depictions from printed publications, which are available as PDF files of the scanned images of every page in the respective publication (**Publication D**). Hence, DECIMER-Segmentation provides an open-source method for segmenting chemical structures from any publication.

One challenge was the lack of freely available datasets for training the image segmentation algorithm. To create a dataset, the chemical structures depicted in printed literature had to be annotated manually. The Mask-RCNN model was trained on these images. Despite being trained on a very small dataset, the model was able to generalise and detect chemical structure depictions in multiple publisher formats. However, it can potentially be improved by training on more data in the future. Any time new data becomes available, the best model at this point can be fine-tuned on the new data to improve its accuracy. This does not affect the performance, and the computational requirements are also manageable.

Furthermore, the tool was made available as a web application so that anyone with no programming knowledge can take advantage of it. The source code can be found on GitHub, and an easy-to-implement Jupyter notebook is available for those who want to do segmentation on a much larger scale. By making this open-source, a contribution to further progress in this field of research has been made.

4.3.2 DECIMER - Image Transformer

The main achievement of this thesis is the DECIMER-Image Transformer. Multiple neural network architectures were experimented with to arrive at the currently implemented architecture.

The initial idea was to create a deep neural network that would learn to read a chemical structure from a picture and translate it into chemical descriptors. This would have been followed by using a separate algorithm to generate the molecule using the descriptor information. Initially, both the regular feed-forward networks and the more complex convolutional neural networks were trained on pixel values of an image to learn how to interpret atom counts. However, it was observed that these networks could not interpret the atom counts of the test data successfully. With more complex CNN setups, better results were achieved on a small number of descriptors, but overall, these approaches failed.

During this time, the idea of using image captioning methods was considered, in which one can train the encoder-decoder network to look at an image and attempt to describe it by recognizing the objects in the image and generating a meaningful sentence based on the ones identified by the network [51]. The same methodology was applied to create the DECIMER image-to-SMILES network, which was trained on images of chemical structures to translate them into meaningful SMILES strings (**Publication B**).

During the work, it was observed that training with more data would lead to an increased overall accuracy; it was estimated that using 50-100 Million images would lead to an accuracy of nearly 90%. Increased amounts of data were necessary to achieve this goal, but there was also a necessity to re-think image feature extraction and SMILES representation. For a start, the network needed to be trained to learn the exact SMILES representation of a chemical structure. One reason for the poor accuracy was due to many invalid SMILES strings generated by the network, which cannot be parsed by any chemistry toolkits [128, 159, 160]. To solve this, DeepSMILES [110] was used in the initial work (**Publication B**), a first-ever SMILES-like syntax explicitly intended for deep learning tasks. This method produced better results than using SMILES. Still, it had similar problems, resulting in some DeepSMILES strings generated by the network, which could not be decoded back into SMILES (invalid DeepSMILES). Some of the SMILES decoded back from DeepSMILES were also invalid. So finally, it was decided to work with another string representation, SELFIES (Self-Referencing Embedded Strings) [111]. Utilizing SELFIES for training resulted in the network producing exclusively valid SELFIES, which can be decoded to valid SMILES strings. By solving the invalid SMILES problem, the overall accuracy improved, and the overall error rate was minimized (**Publications B and G**).

In the next step, the focus was on extracting the relevant image features. This is crucial because the overall network relies on the extracted image features to learn the depicted chemical structure. Training a CNN from scratch to extract features from images can be time-consuming and might not result in a model capable of extracting only the valuable image features. Therefore, the state-of-the-art image recognition network, Inception-V3 [122], was used for image feature extraction at first. However, EfficientNet [59] has recently surpassed Inception-V3 in terms of overall accuracy, so the EfficientNet-B3 model for image feature extraction was implemented, which was originally trained using the 'Noisy-Student' [161] method (**Publication G**). However, the EfficientNet-B7 model performs better than the EfficientNet-B3. Despite that, the EfficientNet-B3 model was used in this work because the scale of input images (300x300) utilized by this model is more similar to the scale of images seen in the printed literature than the input image scale (600x600) of the EfficientNet-B7 model. Performance-wise a 6-fold performance increase was seen in the DECIMER image-to-SMILES network using the EfficientNet-B3 model for image feature extraction (**Publication G**).

The final aspect that needed to be improved was the encoding and decoding architecture of the entire network. With the advent of transformer-based neural networks, recent language translation models gained an exceptional level of accuracy. In the light of this, changes to the DECIMER image-to-SMILES network were made by replacing the encoder-decoder network with a transformer-based network, and the DECIMER-image transformer network was created. Utilizing the newly created DECIMER-image transformer network, the testing resulted in a 10-fold improvement in overall accuracy.

After implementing all these new components, training a model using 1 Million images and testing it on 100 thousand images, the average Tanimoto similarity index based on PubChem fingerprints [162] between the true and the predicted SMILES representations increased from 0.55 to 0.93. Nevertheless, this new model was more interesting because it resulted in almost 70 times more molecule pairs (original & predicted) with Tanimoto 1.0 **(Publication G)**.

The new DECIMER-image transformer models trained on more than 30 Million images can predict canonical SMILES with 96% accuracy, isomeric SMILES (including ions) with 89% accuracy (**Publication G**). Models and code from this project are available on GitHub and are open-source, encouraging further research in the field.

Compared to the other open-source OCSR tools, DECIMER does not require any handwritten rules. The model is entirely data driven. The more different types of chemical structure depictions generated by different toolkits incorporated in the training images, the higher the overall accuracy will be. Therefore, the tool will always keep improving to fit different types of depiction formats. Since the tool solely relies on image features, the model can improve by improving image feature extraction. Like for Chemgrapher [153], chemical structures do not need to be segmented into smaller parts to train the models. The models are trained on complete chemical structures (**Publication A, B, and G**). Similar to the work by Staker *et al.*

[25], DECIMER can also work on images with stereochemistry with higher accuracy of 89% (**Publication G**). DECIMER models will be trained in future with different depiction styles using various other toolkits, similar to the work done by Clevert *et al* [149].

DECIMER is trained exclusively on a v3-8 TPU. This enables computations to be performed much faster, allowing big models to use more data to be trained in the future. Training DECIMER with a bigger training data set is feasible in a limited amount of time since TPUs are also available as TPU pods. Their cost is lower than when using multiple GPUs. So, scaling up DECIMER to fit a larger training corpus is feasible.

4.3.3 COCONUT database and STOUT

When developing COCONUT, the largest open-access databases for natural products today, retrieving information from other open-access databases such as PubChem [127], ChEMBL [108], and ChEBI [10] and generating useful information such as IUPAC names [163–165] were important steps (**Publication E**).

It was discovered that there were no open-source methods for generating IUPAC names from SMILES representations. However, converting an IUPAC name to SMILES can be done using Open Parser for Systematic IUPAC Nomenclature (OPSIN) [166], a rule-based open-source Java program. So, a neural machine translation method, STOUT, was developed that takes advantage of deep learning to solve this problem (**Publication F**). Based on Google's paper 'Effective Approaches to Attention-based Neural Machine Translation' from 2015 [52], STOUT uses a TensorFlow implementation of the model presented in the paper. While STOUT is the first open-source implementation that solves the problem of generating IUPAC names from SMILES, several rule-based proprietary options are available. All of them are commercial.

During the review process of STOUT, a transformer-based implementation of SMILES to IUPAC was published, which is called STRUCT2IUPAC [167], built using Pytorch [168], which is available as open-source software. Simultaneously, a similar transformer-based approach [169] using OpenNMT [170] has also been published, which attempts to translate IUPAC names from InChIs. Both of these publications demonstrate a growing interest in solving problematic chemical translations with deep learning. Evidently, such complex tasks can be addressed with deep learning rather than using algorithms constructed based on handwritten rules.

Compared to the transformer-based approaches, even with an encoder-decoder network, STOUT can handle more than 90% of SMILES to IUPAC name translations correctly on average. Like DECIMER, STOUT was also trained on TPUs, so training a bigger model using more data will take relatively less time. The OpenNMT-based method uses InChIs, and the

STRUCT2IUPAC tools use SMILES in their training data. Due to the tokenization difficulty, specific rules had to be devised for how these representations should be split into meaningful tokens. Both systems can generate invalid SMILES strings and invalid InChIs that do not correspond to a valid chemical structure during predictions. In STOUT, this problem is non-existent due to the use of SELFIES. Also, these can be split into meaningful tokens, and the predicted SELFIES [111] converts back to a valid chemical structure. In the future, a benchmark study involving these chemical representations and the comparison of the available tools should be conducted to determine which performs better overall.

With an average BLEU score greater than 90% and a Tanimoto score greater than 0.9, STOUT can translate SMILES strings to IUPAC names and the other way around (**Publication F**). STOUT is a straightforward implementation of the existing encoder-decoder models and has plenty of room for improvement. The next stage in STOUT development would be to incorporate transformers instead of encoder-decoder networks. Furthermore, large language models such as BERT [75] and GPT-3 [76] greatly impacted the improvement of natural language processing. The next possible step could be to make STOUT use these models.

4.4 Future developments in software, hardware, and DECIMER

One of the reasons why the currently used software might fail or not work in the future is backward compatibility. At the same time, the open-source community and Linux-based open-source operating systems try to assure and standardize backward compatibility [171, 172].

When libraries used in the systems are no longer supported, there will be difficulties with backward compatibility. Additionally, using the tools on a more recent operating system can result in problems, too. Most rule-based systems and deep learning-based methods are susceptible to these problems. Therefore, these systems should be actively developed to support the newer version of the programming languages as well as to be able to use more recent libraries.

The new libraries usually have backward compatibility with the older versions, but sometimes there may be slight changes in the code that could break a system. This is especially true when it comes to rule-based systems, which need to be re-optimized to work with the newer updates. In deep learning-based methods, the already trained models do not have to undergo any changes to work on a more recent version of the programming language or utilize the newly developed libraries. Only a few code changes need to be made on networks previously developed with an older version of deep learning libraries is necessary (before TensorFlow

2.0). These changes can also be carried out automatically with the TensorFlow automatic conversion script [173].

Keeping this in mind, when DECIMER and STOUT were developed, the newest versions of Python (3.0<) and TensorFlow (2.0<) were used. Even though the programming language and libraries have undergone minor revisions, the overall code did not change. Also, DECIMER was designed in a modular fashion from the beginning, meaning that the networks and functions could be replaced if necessary. There might be the possibility to develop new implementations for image feature extraction or implement newer network architectures in the future. At that time, these new features could be added to DECIMER. Also, modifications to existing training scripts are not required for training purposes. As the training becomes more large-scale with more training data, the current training scripts can still be used. The scripts developed in this work currently work perfectly on the latest TensorFlow version 2.4.0.

The models are trained with the TPU strategy for distributed training on TPUs. To increase the training speed of the models, TPU pods [174] can be used (**Publications F and G**). Changes to the code are not required for this, except for adjusting the batch size, which is also done automatically by the current training scripts. However, one can optimize the hyperparameters according to their needs. For the DECIMER training, people who wish to use TPUs but do not have access to them can get free TPUs on Kaggle [175] or apply for TPUs through Google's TPU Research Cloud infrastructure [176].

There is no need to train models exclusively using TPUs. The models can also be trained using GPUs with the mirrored strategy in TensorFlow distributed learning[4]. This only involves changing a few lines of code related to the strategies. Evaluating the results can be done on both CPUs and GPUs.

In many fields where deep-learning approaches are widely adopted, the need for better hardware for deep learning is also increasing for training models faster without losing performance. Several companies, such as Nvidia [177], Google [178], and Apple [179], are investing in hardware development that can enable training the models faster.

In recent years, accessing hardware for machine learning projects has become much easier thanks to the development of cloud infrastructures, such as Google Cloud Platform, Amazon Web Services, and Microsoft Azure cloud. Users can access Virtual Machines (VMs) on these platforms to work on their machine learning projects. These VMs are much less expensive and easier to access because of the large competition in these markets. Many cloud companies also offer their services through regional servers due to data protection concerns.

---

[4] https://www.tensorflow.org/guide/distributed_training

These improvements will make the task of retraining or implementing deep learning-based methods much more straightforward. Each time new datasets are obtained, the DECIMER pre-trained weights can be used to fine-tune the model on those datasets. This will help in getting a better overall model. By making the code and models open-source, a contribution to both the ongoing research on this subject and the larger research community is intended. The models can be retrained by users so that they can apply them to their data.

To improve the prediction accuracy of DECIMER in future, the successive steps would be implementing and testing different image feature extraction methods and different text representation methods. Considering that using EfficientNet-B3 rather than Inception-V3 for image feature extraction resulted in a significant improvement in accuracy, the next step will be to apply the EfficientNet-V2 [180] models and see how much better the overall accuracy becomes. Additionally, the new and improved vision-based transformers for image captioning could be tested, such as the Meshed Memory Transformer [181], to see whether it improves overall performance.

The DECIMER network can be trained with more datasets in the next phase. The dataset filtering rules can be eliminated one by one and analysed how they affect the accuracy. Eliminating all the filters and having a model that works with more than 90% would be a significant advance. When this threshold is reached, the DECIMER network could be trained with real-world images (or images generated using image augmentations). The next big step of the DECIMER project would be to have a single model that can complete the OCSR task with more than 90% accuracy on real-world images. Next is incorporating this developed model to the existing decimer.ai web application. Thus, one can upload a printed publication to the web application, use DECIMER-Segmentation to segment out the chemical structure depictions, and then use DECIMER-image transformer to generate SMILES. Our ultimate goal is to have an AI solution that automatically mines data and adds it to open-access databases.

5. References

1. Zhou JZ (2011) Chemoinformatics and library design. Methods Mol Biol 685:27–52

2. Swain MC, Cole JM (2016) ChemDataExtractor: A Toolkit for Automated Extraction of Chemical Information from the Scientific Literature. J Chem Inf Model 56:1894–1904

3. O'Boyle NM, Guha R, Willighagen EL, et al (2011) Open Data, Open Source and Open Standards in chemistry: The Blue Obelisk five years on. J Cheminform 3:37

4. Weininger D (1988) SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. J Chem Inf Comput Sci 28:31–36

5. McNaught A (2006) The IUPAC international chemical identifier. Chem Int 12–14

6. Haug K, Salek RM, Steinbeck C (2017) Global open data management in metabolomics. Curr Opin Chem Biol 36:58–63

7. Brown PD, Lawrence AL (2017) The importance of asking "how and why?" in natural product structure elucidation. Nat Prod Rep 34:1193–1202

8. Romano JD, Tatonetti NP (2019) Informatics and Computational Methods in Natural Product Drug Discovery: A Review and Perspectives. Front Genet 10:368

9. Logan DC (2009) Known knowns, known unknowns, unknown unknowns and the propagation of scientific enquiry. J Exp Bot 60:712–714

10. Hastings J, de Matos P, Dekker A, et al (2013) The ChEBI reference database and ontology for biologically relevant chemistry: enhancements for 2013. Nucleic Acids Res 41:D456–63

11. Kim HK, Choi YH, Verpoorte R (2011) NMR-based plant metabolomics: where do we stand, where do we go? Trends Biotechnol 29:267–275

12. Bisson J, Simmler C, Chen S-N, Friesen JB, Lankin DC, McAlpine JB, Pauli GF (2016) Dissemination of original NMR data enhances reproducibility and integrity in chemical research. Nat Prod Rep 33:1028–1033

13. Vazquez M, Krallinger M, Leitner F, Valencia A (2011) Text Mining for Drugs and Chemical Compounds: Methods, Tools and Applications. Mol Inform 30:506–519

14. Krallinger M, Rabal O, Lourenço A, Oyarzabal J, Valencia A (2017) Information Retrieval and Text Mining Technologies for Chemistry. Chem Rev 117:7673–7761

15. Krallinger M, Rabal O, Leitner F, et al (2015) The CHEMDNER corpus of chemicals and drugs and its annotation principles. J Cheminform 7:S2

16. Bento AP, Hersey A, Félix E, Landrum G, Gaulton A, Atkinson F, Bellis LJ, De Veij M, Leach AR (2020) An open source chemical structure curation pipeline using RDKit. J Cheminform 12:51

17. Contreras ML, Allendes C, Alvarez LT, Rozas R (1990) Computational perception and recognition of digitized molecular structures. J Chem Inf Comput Sci 30:302–307

18. Rozas R, Fernandez H (1990) Automatic processing of graphics for image databases in science. J Chem Inf Comput Sci 30:7–12

19. McDaniel JR, Balmuth JR (1992) Kekule: OCR-optical chemical (structure) recognition. J Chem Inf Comput Sci 32:373–378

20. Casey R, Boyer S, Healey P, Miller A, Oudot B, Zilles K (2002) Optical recognition of chemical graphics. Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR ’93). https://doi.org/10.1109/icdar.1993.395658

21. Ibison P, Jacquot M, Kam F, Neville AG, Simpson RW, Tonnelier C, Venczel T, Johnson AP (1993) Chemical literature data extraction: The CLiDE Project. J Chem Inf Comput Sci 33:338–344

22. Filippov IV, Nicklaus MC (2009) Optical structure recognition software to recover chemical information: OSRA, an open source solution. J Chem Inf Model 49:740–743

23. Smolov V, Zentsev F, Rybalkin M (2011) Imago: Open-Source Toolkit for 2D Chemical Structure Image Recognition. TREC

24. Peryea T, Katzel D, Zhao T, Southall N, Nguyen D-T (2019) MOLVEC: Open source library for chemical structure recognition. ABSTRACTS OF PAPERS OF THE AMERICAN CHEMICAL SOCIETY 258:

25. Staker J, Marshall K, Abel R, McQuaw CM (2019) Molecular Structure Extraction from Documents Using Deep Learning. J Chem Inf Model 59:1017–1029

26. Website. ChemAxon - Software Solutions and Services for Chemistry &

Biology.https://www.chemaxon.com. Accessed 23 Nov 2020. Accessed 9 Apr 2021

27. Mitchell TM (1997) Machine Learning. McGraw-Hill, New York

28. Samuel AL (1959) Some Studies in Machine Learning Using the Game of Checkers. IBM J Res Dev 3:210–229

29. Mitchell T, Buchanan B, DeJong G, Dietterich T, Rosenbloom P, Waibel A (1990) Machine Learning. Annu Rev Comput Sci 4:417–433

30. Ongsulee P (2017) Artificial intelligence, machine learning and deep learning. In: 2017 15th International Conference on ICT and Knowledge Engineering (ICT KE). pp 1–6

31. Brewka G (1996) Artificial intelligence—a modern approach by Stuart Russell and Peter Norvig, Prentice Hall. Series in Artificial Intelligence, Englewood Cliffs, NJ. The Knowledge Engineering Review 11:78–79

32. (2021) Machine Learning. https://www.sas.com/en_us/insights/analytics/machine-learning.html. Accessed 6 Apr 2021

33. Talabis MRM, McPherson R, Miyamoto I, Martin JL, Kaye D (2015) Analytics Defined. In: Information Security Analytics. Elsevier, pp 1–12

34. Hinton GE, Sejnowski TJ, Howard Hughes Medical Institute Computational Neurobiology Laboratory Terrence J Sejnowski (1999) Unsupervised Learning: Foundations of Neural Computation. MIT Press

35. (2021) Machine Learning. https://www.sas.com/en_us/insights/analytics/machine-learning.html. Accessed 6 Apr 2021

36. IBM Cloud Education What is Unsupervised Learning? https://www.ibm.com/cloud/learn/unsupervised-learning. Accessed 6 Apr 2021

37. Zhu X, Goldberg AB (2009) Introduction to semi-supervised learning. Synth lect artif intell mach learn 3:1–130

38. Zhu X, Goldberg AB (2009) Introduction to Semi-Supervised Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning 3:1–130

39. Sutton RS, Barto AG (2018) Reinforcement Learning: An Introduction. A Bradford Book

40. Kaelbling LP, Littman ML, Moore AW (1996) Reinforcement Learning: A Survey. J Artif Intell Res 4:237–285

41. Alpaydin E (2020) Introduction to Machine Learning. MIT Press

42. Krogh A (2008) What are artificial neural networks? Nat Biotechnol 26:195–197

43. Sharma S (2017) Activation functions in neural networks. towards data science 6:

44. Abraham A (2005) Artificial Neural Networks. Handbook of Measuring System Design. https://doi.org/10.1002/0471497398.mm421

45. Rumelhart DE, Durbin R, Golden R (1995) Backpropagation: The basic theory. Backpropagation

46. Goodfellow I, Bengio Y, Courville A (2016) Deep Learning (Adaptive Computation and Machine Learning series).

47. Takase T, Oyama S, Kurihara M (2018) Effective neural network training with adaptive learning rate based on training loss. Neural Netw 101:68–78

48. Schmidhuber J (2015) Deep learning in neural networks: an overview. Neural Netw 61:85–117

49. Goodfellow I, Bengio Y, Courville A (2016) Deep Learning. MIT Press

50. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444

51. Xu K, Ba J, Kiros R, Cho K, Courville A, Salakhudinov R, Zemel R, Bengio Y (2015) Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In: Bach F, Blei D (eds) Proceedings of the 32nd International Conference on Machine Learning. PMLR, Lille, France, pp 2048–2057

52. Luong M-T, Pham H, Manning CD (2015) Effective Approaches to Attention-based Neural Machine Translation. arXiv:1508.04025 [cs.CL]

53. Pawley JB (2006) Points, pixels, and gray levels: Digitizing image data. In: Handbook Of Biological Confocal Microscopy. Springer US, Boston, MA, pp 59–79

54. Fu H, Niu Z, Zhang C, Ma J, Chen J (2016) Visual Cortex Inspired CNN Model for Feature Construction in Text Analysis. Front Comput Neurosci 10:64

55. O’Shea K, Nash R (2015) An Introduction to Convolutional Neural Networks. arXiv:1511.08458 [cs.NE]

56. Yamashita R, Nishio M, Do RKG, Togashi K (2018) Convolutional neural networks: an overview and application in radiology. Insights Imaging 9:611–629

57. Krizhevsky A, Sutskever I, Hinton GE (2017) ImageNet classification with deep convolutional neural networks. Communications of the ACM 60:84–90

58. Szegedy C, Ioffe S, Vanhoucke V, Alemi A (2016) Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv:1602.07261 [cs.CV]

59. Tan M, Le Q (2019) Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning. PMLR, pp 6105–6114

60. He K, Gkioxari G, Dollar P, Girshick R (2017) Mask R-CNN. 2017 IEEE International Conference on Computer Vision (ICCV). https://doi.org/10.1109/iccv.2017.322

61. Medsker LR, Jain LC (2001) Recurrent neural networks. Proc. Inst. Mech. Eng. Part L J. Mat. Des. Appl. 5:

62. Hiriyannaiah S, Srinivas A, Shetty GK, G. M. S, Srinivasa KG (2020) Chapter 4 - A computationally intelligent agent for detecting fake news using generative adversarial networks. In: Bhattacharyya S, Snášel V, Gupta D, Khanna A (eds) Hybrid Computational Intelligence. Academic Press, pp 69–96

63. Tealab A (2018) Time series forecasting using artificial neural networks methodologies: A systematic review. Future Computing and Informatics Journal 3:334–340

64. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9:1735–1780

65. DiPietro R, Hager GD (2020) Chapter 21 - Deep learning: RNNs and LSTM. In: Zhou SK, Rueckert D, Fichtinger G (eds) Handbook of Medical Image Computing and Computer Assisted Intervention. Academic Press, pp 503–519

66. Smagulova K, James AP (2019) A survey on LSTM memristive neural network architectures and applications. Eur Phys J Spec Top 228:2313–2324

67. Carbune V, Gonnet P, Deselaers T, Rowley HA, Daryin A, Calvo M, Wang L-L, Keysers D, Feuz S, Gervais P (2020) Fast multi-language LSTM-based online handwriting

recognition. Int J Doc Anal Recogn 23:89–102

68. Graves A, Eck D, Beringer N, Schmidhuber J (2004) Biologically Plausible Speech Recognition with LSTM Neural Nets. In: Biologically Inspired Approaches to Advanced Information Technology. Springer Berlin Heidelberg, pp 127–136

69. Hochreiter S (1998) The vanishing gradient problem during learning recurrent neural nets and problem solutions. Internat J Uncertain Fuzziness Knowledge-Based Systems 06:107–116

70. Cho K, van Merrienboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. arXiv:1406.1078 [cs.CL]

71. Vinyals O, Toshev A, Bengio S, Erhan D (2015) Show and tell: A neural image caption generator. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 3156–3164

72. Bahdanau D, Cho K, Bengio Y (2014) Neural Machine Translation by Jointly Learning to Align and Translate. arXiv:1409.0473 [cs.CL]

73. Popescu-Belis A (2019) Context in Neural Machine Translation: A Review of Models and Evaluations. arXiv:1901.09115 [cs.CL]

74. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention Is All You Need. arXiv:1706.03762 [cs.CL]

75. Devlin J, Chang M-W, Lee K, Toutanova K (2018) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

76. Brown TB, Mann B, Ryder N, et al (2020) Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL]

77. Young T, Hazarika D, Poria S, Cambria E (2018) Recent trends in deep learning based natural language processing [review article]. IEEE Comput Intell Mag 13:55–75

78. Voulodimos A, Doulamis N, Doulamis A, Protopapadakis E (2018) Deep Learning for Computer Vision: A Brief Review. Comput Intell Neurosci 2018:7068349

79. Chong E, Han C, Park FC (2017) Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. Expert Syst Appl

83:187–205

80. Fang H, Zhang M (2017) Creatism: A deep-learning photographer capable of creating professional work. arXiv:1707.03491 [cs.CV]

81. Chen H, Engkvist O, Wang Y, Olivecrona M, Blaschke T (2018) The rise of deep learning in drug discovery. Drug Discov Today 23:1241–1250

82. Min S, Lee B, Yoon S (2017) Deep learning in bioinformatics. Brief Bioinform 18:851–869

83. Garg S, Ramakrishnan G (2020) Advances in Quantum Deep Learning: An Overview. arXiv:2005.04316 [quant-ph]

84. Silver D, Schrittwieser J, Simonyan K, et al (2017) Mastering the game of Go without human knowledge. Nature 550:354–359

85. Silver D, Hubert T, Schrittwieser J, et al (2018) A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science 362:1140–1144

86. Rasley J, Rajbhandari S, Ruwase O, He Y (2020) DeepSpeed: System Optimizations Enable Training Deep Learning Models with Over 100 Billion Parameters. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. Association for Computing Machinery, New York, NY, USA, pp 3505–3506

87. Shams S, Platania R, Lee K, Park S-J (2017) Evaluation of Deep Learning Frameworks Over Different HPC Architectures. In: 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS). pp 1389–1396

88. Abadi M, Agarwal A, Barham P, et al (2016) TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. arXiv:1603.04467 [cs.DC]

89. Chien SWD, Markidis S, Olshevsky V, Bulatov Y, Laure E, Vetter J (2019) TensorFlow Doing HPC. In: 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). pp 509–518

90. Deng L (2012) Three classes of deep learning architectures and their applications: a tutorial survey. APSIPA transactions on signal and information processing

91. Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I, Fergus R (2014)

Intriguing properties of neural networks. arXiv:1312.6199 [cs.CV]

92. Ba LJ, Caruana R (2013) Do Deep Nets Really Need to be Deep? arXiv:1312.6184 [cs.LG]

93. Minar MR, Naher J (2018) Recent Advances in Deep Learning: An Overview. arXiv:1807.08169 [cs.LG]

94. Deng L, Liu Y (2018) Deep Learning in Natural Language Processing. Springer

95. Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Teh YW, Titterington M (eds) Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. PMLR, Chia Laguna Resort, Sardinia, Italy, pp 249–256

96. Bianchini M, Scarselli F (2014) On the complexity of neural network classifiers: a comparison between shallow and deep architectures. IEEE Trans Neural Netw Learn Syst 25:1553–1565

97. Bengio Y (2012) Practical recommendations for gradient-based training of deep architectures. arXiv:1206.5533 [cs.LG]

98. Tan HH, Lim KH (2019) Vanishing Gradient Mitigation with Deep Learning Neural Network Optimization. In: 2019 7th International Conference on Smart Computing Communications (ICSCC). pp 1–4

99. Lecun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86:2278–2324

100. Simard PY, Steinkraus D, Platt JC (2005) Best practices for convolutional neural networks applied to visual document analysis. Seventh International Conference on Document Analysis and Recognition, 2003 Proceedings. https://doi.org/10.1109/icdar.2003.1227801

101. He K, Zhang X, Ren S, Sun J (2015) Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In: 2015 IEEE International Conference on Computer Vision (ICCV). pp 1026–1034

102. Schemmel J, Brüderle D, Grübl A, Hock M, Meier K, Millner S (2010) A wafer-scale neuromorphic hardware system for large-scale neural modeling. In: 2010 IEEE

International Symposium on Circuits and Systems (ISCAS). pp 1947–1950

103. Neftci EO, Augustine C, Paul S, Detorakis G (2017) Event-Driven Random Back-Propagation: Enabling Neuromorphic Deep Learning Machines. Front Neurosci 11:324

104. Mater AC, Coote ML (2019) Deep Learning in Chemistry. J Chem Inf Model 59:2545–2559

105. Koutsoukas A, Monaghan KJ, Li X, Huan J (2017) Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data. J Cheminform 9:42

106. Cortés-Ciriano I, Bender A (2019) Deep Confidence: A Computationally Efficient Framework for Calculating Reliable Prediction Errors for Deep Neural Networks. J Chem Inf Model 59:1269–1281

107. Lenselink EB, Ten Dijke N, Bongers B, Papadatos G, van Vlijmen HWT, Kowalczyk W, IJzerman AP, van Westen GJP (2017) Beyond the hype: deep neural networks outperform established methods using a ChEMBL bioactivity benchmark set. J Cheminform 9:45

108. Gaulton A, Hersey A, Nowotka M, et al (2017) The ChEMBL database in 2017. Nucleic Acids Res 45:D945–D954

109. Olivecrona M, Blaschke T, Engkvist O, Chen H (2017) Molecular de-novo design through deep reinforcement learning. J Cheminform 9:48

110. O'Boyle N, Dalke A (2018) DeepSMILES: An Adaptation of SMILES for Use in Machine-Learning of Chemical Structures. ChemRxiv. https://doi.org/10.26434/chemrxiv.7097960.v1

111. Krenn M, Häse F, Nigam A, Friederich P, Aspuru-Guzik A (2020) Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. Mach Learn: Sci Technol 1:045024

112. Lim J, Ryu S, Kim JW, Kim WY (2018) Molecular generative model based on conditional variational autoencoder for de novo molecular design. J Cheminform 10:31

113. Segler MHS, Preuss M, Waller MP (2018) Planning chemical syntheses with deep neural networks and symbolic AI. Nature 555:604–610

114. Corbett P, Boyle J (2018) Chemlistem: chemical named entity recognition using recurrent neural networks. J Cheminform 10:59

115. Korvigo I, Holmatov M, Zaikovskii A, Skoblov M (2018) Putting hands to rest: efficient deep CNN-RNN architecture for chemical named entity recognition with no hand-crafted rules. J Cheminform 10:28

116. Goh GB, Siegel C, Vishnu A, Hodas NO, Baker N (2017) Chemception: A Deep Neural Network with Minimal Chemistry Knowledge Matches the Performance of Expert-developed QSAR/QSPR Models. arXiv:1706.06689 [stat.ML]

117. Fernandez M, Ban F, Woo G, Hsing M, Yamazaki T, LeBlanc E, Rennie PS, Welch WJ, Cherkasov A (2018) Toxic Colors: The Use of Deep Learning for Predicting Toxicity of Compounds Merely from Their Graphic Images. J Chem Inf Model 58:1533–1543

118. Yi Z (2018) Evaluation and Implementation of Convolutional Neural Networks in Image Recognition. J Phys Conf Ser 1087:062018

119. Liu X, Xu Q, Wang N (2019) A survey on deep neural network-based image captioning. Vis Comput 35:445–470

120. Hu X, Yin X, Lin K, Wang L, Zhang L, Gao J, Liu Z (2021) VIVO: Visual Vocabulary Pre-Training for Novel Object Captioning. arXiv:2009.13682 [cs.CV]

121. Vinyals O, Toshev A, Bengio S, Erhan D (2017) Show and Tell: Lessons Learned from the 2015 MSCOCO Image Captioning Challenge. IEEE Trans Pattern Anal Mach Intell 39:652–663

122. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 2818–2826

123. Canziani A, Paszke A, Culurciello E (2017) An Analysis of Deep Neural Network Models for Practical Applications. arXiv:1605.07678 [cs.CV]

124. Radford A, Narasimhan K (2018) Improving Language Understanding by Generative Pre-Training.

125. Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I (2019) Language models are unsupervised multitask learners. OpenAI blog 1:9

126. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp 2961–2969

127. Kim S, Chen J, Cheng T, et al (2019) PubChem 2019 update: improved access to chemical data. Nucleic Acids Res 47:D1102–D1109

128. Steinbeck C, Han Y, Kuhn S, Horlacher O, Luttmann E, Willighagen E (2003) The Chemistry Development Kit (CDK): an open-source Java library for Chemo- and Bioinformatics. J Chem Inf Comput Sci 43:493–500

129. Van Rossum G, Drake FL (2009) Python 3 References Manual. Scotts Valley CA: CreateSpace.

130. TFRecord and tf.train.Example. https://www.tensorflow.org/tutorials/load_data/tfrecord. Accessed 7 Apr 2021

131. Tanimoto TT (1958) An Elementary Mathematical Theory of Classification and Prediction. International Business Machines Corporation

132. Heller SR, McNaught A, Pletnev I, Stein S, Tchekhovskoi D (2015) InChI, the IUPAC International Chemical Identifier. J Cheminform 7:23

133. Dutta A, Zisserman A (2019) The VIA Annotation Software for Images, Audio and Video. In: Proceedings of the 27th ACM International Conference on Multimedia. Association for Computing Machinery, New York, NY, USA, pp 2276–2279

134. Luebke D (2008) CUDA: Scalable parallel programming for high-performance scientific computing. In: 2008 5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro. pp 836–838

135. Heryadi D, Hampton S (2019) Characterizing Performance Improvement of GPUs. In: Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning). Association for Computing Machinery, New York, NY, USA, pp 1–5

136. Justus D, Brennan J, Bonner S, McGough AS (2018) Predicting the Computational Cost of Deep Learning Models. In: 2018 IEEE International Conference on Big Data (Big Data). pp 3873–3882

137. Campos V, Sastre F, Yagües M, Torres J, Giró-I-Nieto X (2017) Scaling a

Convolutional Neural Network for Classification of Adjective Noun Pairs with TensorFlow on GPU Clusters. In: 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID). pp 677–682

138. Norrie T, Patil N, Yoon DH, Kurian G, Li S, Laudon J, Young C, Jouppi N, Patterson D (2021) The Design Process for Google’s Training Chips: TPUv2 and TPUv3. IEEE Micro 41:56–63

139. Cloud tensor processing units (TPUs). https://cloud.google.com/tpu/docs/tpus. Accessed 7 Apr 2021

140. You Y, Zhang Z, Hsieh C, Demmel J, Keutzer K (2019) Fast Deep Neural Network Training on Distributed Systems and Cloud TPUs. IEEE Trans Parallel Distrib Syst 30:2449–2462

141. Smith MJS (1997) Application-specific integrated circuits. Addison-Wesley Reading, MA

142. XLA: Optimizing Compiler for Machine Learning. https://www.tensorflow.org/xla. Accessed 22 Apr 2021

143. Wang YE, Wei G-Y, Brooks D (2019) Benchmarking TPU, GPU, and CPU Platforms for Deep Learning. arXiv:1907.10701 [cs.LG]

144. Pak M, Kim S (2017) A review of deep learning in image recognition. In: 2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT). pp 1–3

145. Sorokina M, Steinbeck C (2020) Review on natural products databases: where to find data in 2020. J Cheminform 12:20

146. Ngo LT, Okogun JI, Folk WR (2013) 21st century natural product research and drug development and traditional medicines. Nat Prod Rep 30:584–592

147. Butler MS (2004) The role of natural product chemistry in drug discovery. J Nat Prod 67:2141–2153

148. Beard EJ, Cole JM (2020) ChemSchematicResolver: A Toolkit to Decode 2D Chemical Diagrams with Labels and R-Groups into Annotated Chemical Named Entities. J Chem Inf Model 60:2059–2072

149. Clevert D-A, Le T, Winter R, Montanari F (2021) Img2Mol - Accurate SMILES Recognition from Molecular Graphical Depictions. ChemRxiv. https://doi.org/10.26434/chemrxiv.14320907.v1

150. Garcia-Garcia A, Orts-Escolano S, Oprea S, Villena-Martinez V, Garcia-Rodriguez J (2017) A Review on Deep Learning Techniques Applied to Semantic Segmentation. arXiv:1704.06857 [cs.CV]

151. Yanai K, Kawano Y (2015) Food image recognition using deep convolutional network with pre-training and fine-tuning. In: 2015 IEEE International Conference on Multimedia Expo Workshops (ICMEW). pp 1–6

152. Irwin JJ, Tang KG, Young J, Dandarchuluun C, Wong BR, Khurelbaatar M, Moroz YS, Mayfield J, Sayle RA (2020) ZINC20-A Free Ultralarge-Scale Chemical Database for Ligand Discovery. J Chem Inf Model 60:6065–6073

153. Oldenhof M, Arany A, Moreau Y, Simm J (2020) ChemGrapher: Optical Graph Recognition of Chemical Compounds by Deep Learning. J Chem Inf Model 60:4506–4517

154. Weir H, Thompson K, Choi B, Woodward A, Braun A, Martínez TJ (2021) ChemPix: Automated Recognition of Hand-drawn Hydrocarbon Structures Using Deep Learning. ChemRxiv. https://doi.org/10.26434/chemrxiv.14156957.v1

155. Liang X, Guo Q, Qian Y, Ding W, Zhang Q (2021) Evolutionary Deep Fusion Method and Its Application in Chemical Structure Recognition. IEEE Trans Evol Comput 1–1

156. Smalley E (2017) AI-powered drug discovery captures pharma interest. Nat Biotechnol 35:604–605

157. Bristol-Myers Squibb – molecular translation. https://www.kaggle.com/c/bms-molecular-translation/overview. Accessed 16 Apr 2021

158. Belval E (2020) pdf2image.

159. O'Boyle NM, Banck M, James CA, Morley C, Vandermeersch T, Hutchison GR (2011) Open Babel: An open chemical toolbox. J Cheminform 3:33

160. Landrum G, Others (2016) RDKit: Open-Source Cheminformatics Software.(2016). URL http://www. rdkit. org/, https://github. com/rdkit/rdkit

161. Xie Q, Luong M-T, Hovy E, Le QV (2020) Self-training with noisy student improves imagenet classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp 10687–10698

162. [No title]. ftp://ftp.ncbi.nlm.nih.gov/pubchem/specifications/pubchem_fingerprints.txt. Accessed 23 Apr 2021

163. Favre HA, Powell WH (2013) Nomenclature of Organic Chemistry: IUPAC Recommendations and Preferred Names 2013. Royal Society of Chemistry

164. Union internationale de chimie pure et appliquée. Physical, International Union of Pure and Applied Chemistry. Physical and Biophysical Chemistry Division (2007) Quantities, Units and Symbols in Physical Chemistry. Royal Society of Chemistry

165. Nomenclature of Inorganic Chemistry – IUPAC Recommendations 2005. Chem Int 27:25–26

166. Lowe DM, Corbett PT, Murray-Rust P, Glen RC (2011) Chemical name to structure: OPSIN, an open source solution. J Chem Inf Model 51:739–753

167. Krasnov L, Khokhlov I, Fedorov M, Sosnin S (2021) Struct2IUPAC -- Transformer-Based Artificial Neural Network for the Conversion Between Chemical Notations. ChemRxiv. https://doi.org/10.26434/chemrxiv.13274732.v2

168. Paszke A, Gross S, Massa F, et al (2019) PyTorch: An Imperative Style, High-Performance Deep Learning Library. arXiv:1912.01703 [cs.LG]

169. Handsel J, Matthews B, Knight N, Coles S (2021) Translating the Molecules: Adapting Neural Machine Translation to Predict IUPAC Names from a Chemical Identifier. ChemRxiv. https://doi.org/10.26434/chemrxiv.14170472.v1

170. Klein G, Hernandez F, Nguyen V, Senellart J (2020) The OpenNMT neural machine translation toolkit: 2020 edition. In: Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (AMTA 2020). pp 102–109

171. Wu M-W, Lin Y-D (2001) Open source software development: an overview. Computer 34:33–38

172. Wang H, Wang C (2001) Open source software adoption: a status report. IEEE Softw 18:90–95

173. Migrate your TensorFlow 1 code to TensorFlow 2. https://www.tensorflow.org/guide/migrate. Accessed 19 Apr 2021

174. System Architecture. https://cloud.google.com/tpu/docs/system-architecture. Accessed 19 Apr 2021

175. Tensor processing units (TPUs) documentation. https://www.kaggle.com/docs/tpu. Accessed 19 Apr 2021

176. TPU Research Cloud. https://sites.research.google/trc/. Accessed 19 Apr 2021

177. Krashinsky R, Giroux O, Jones S, Stam N, Ramaswamy S, View All Posts by, View All Posts by, View All Posts by, View All Posts by, View All Posts by (2020) NVIDIA Ampere Architecture In-Depth. https://developer.nvidia.com/blog/nvidia-ampere-architecture-in-depth/. Accessed 19 Apr 2021

178. Google breaks AI performance records in MLPerf with world's fastest training supercomputer. https://cloud.google.com/blog/products/ai-machine-learning/google-breaks-ai-performance-records-in-mlperf-with-worlds-fastest-training-supercomputer. Accessed 19 Apr 2021

179. Leveraging ML Compute for Accelerated Training on Mac. https://machinelearning.apple.com/updates/ml-compute-training-on-mac. Accessed 19 Apr 2021

180. Tan M, Le QV (2021) EfficientNetV2: Smaller Models and Faster Training. arXiv:2104.00298 [cs.CV]

181. Cornia M, Stefanini M, Baraldi L, Cucchiara R (2020) Meshed-memory transformer for image captioning. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr42600.2020.01059

7. Curriculum vitae

Personal data

Full Name: Kohulan Rajan
Date of birth: 05/12/1990
Email: kohulan.rajan@uni-jena.de / kohul5972@gmail.com
ResearchGate: https://www.researchgate.net/profile/Kohulan-Rajan
ORCID: http://orcid.org/0000-0003-1066-7792
GitHub: https://github.com/Kohulan

Education

07/2017 - today	Ph.D. Student, Institute of Inorganic and Analytical Chemistry, Cheminformatics and Chemometrics, Friedrich-Schiller-University, Jena, Germany. 1st Supervisor: Prof. Dr. Christoph Steinbeck, Friedrich-Schiller-University, Jena, Germany. 2nd Supervisor: Prof. Dr. Achim Zielesny, Westphalian University of Applied Sciences, Recklinghausen, Germany.
07/2015 - 06/2017	Master of Science in Bioinformatics, Savithribai Phule Pune University, Pune, India.
09/2011 - 11/2014	Bachelor of Science in Biotechnology, Bangalore University, Bangalore, India.

Publications

Rajan, K., Zielesny, A. & Steinbeck, C. DECIMER: towards deep learning for chemical image recognition. *J Cheminform* **12,** 65 (2020). https://doi.org/10.1186/s13321-020-00469-w

Rajan, K., Brinkhaus, H.O., Zielesny, A. *et al.* A review of optical chemical structure recognition tools. *J Cheminform* **12,** 60 (2020). https://doi.org/10.1186/s13321-020-00465-0

Rajan, K., Hein, JM., Steinbeck, C. *et al.* Molecule Set Comparator (MSC): a CDK-based open rich‐client tool for molecule set similarity evaluations. *J Cheminform* **13,** 5 (2021). https://doi.org/10.1186/s13321-021-00485-4

Sorokina, M., Merseburger, P., Rajan, K. *et al.* COCONUT online: Collection of Open Natural Products database. *J Cheminform* **13,** 2 (2021). https://doi.org/10.1186/s13321-020-00478-9

Rajan, K., Brinkhaus, H.O., Sorokina, M. *et al.* DECIMER-Segmentation: Automated extraction of chemical structure depictions from scientific literature. *J Cheminform* **13,** 20 (2021). https://doi.org/10.1186/s13321-021-00496-1

Rajan, K., Zielesny, A. & Steinbeck, C. STOUT: SMILES to IUPAC names using neural machine translation. J Cheminform 13, 34 (2021). https://doi.org/10.1186/s13321-021-00512-4

Rajan, K., Zielesny, A. & Steinbeck, C. DECIMER 1.0: deep learning for chemical image recognition using transformers. J Cheminform 13, 61 (2021). https://doi.org/10.1186/ https://doi.org/10.26434/chemrxiv.14479287.v1

Teaching and Supervision

- Assisted in conducting “Blockpraktikum” for the chemometrics course.
- Conducted lectures for cheminformatics and chemometrics lectures.
- Conducted Python course as a part of cheminformatics practical course.
- Supervised two masters’ students.

8. Acknowledgements

As a PhD student of Prof. Dr. Christoph Steinbeck's Cheminformatics and Computational Metabolomics(caffeine) group at Friedrich-Schiller University, Jena, I have had the privilege of working on some amazing machine learning projects as well as collaborating with some outstanding scientists. I would like to use this section of the thesis to thank all those who have inspired, motivated and actively supported me during the four years of my doctoral studies.

First of all, I am incredibly grateful to Prof. Dr. Christoph Steinbeck for selecting me as one of the first members of your newly formed group in Jena. I still remember our very initial conversations on why you selected me and in the first place and I would never forget your answer. Thank you for being an amazing mentor, allowing me to work completely independently, providing me with all the necessary hardware, and being able to rely on my abilities. It has always been great to work under your guidance, as well as discussing topics outside of science freely since we both enjoy the same interests. I am also thankful to you for introducing me to Prof. Dr. Achim Zielesny, who is my second supervisor and another great mentor.

I am incredibly pleased to have Prof. Dr. Achim Zielesny as my second supervisor whom I can always go to for help whenever I become lost entirely or stuck in work without any further clue. His wisdom regarding using data and machine learning was something unparalleled that helped me in many ways during this thesis. I loved our long relaxing conversations and the stories you shared with us about your adventures. Although we were unable to meet more often due to the last pandemic, our meetings and the good time we had in Jena and Sasbachwalden are memories I will treasure forever. I am delighted to have both of you as my PhD supervisors, to call you both as my PhD fathers and there is nothing more I could ask for.

Working in the caffeine group has always been fun. I have enjoyed discussing my scientific findings as well as those of others. Besides science, being part of such an international group helped me grow individually, for which I am always grateful.

I would like to thank Michael and Aziz, the initial members, along with me. Michael, you have been an excellent friend, you helped me understand German culture very well, we often chatted about technology and it was fun. Thanks, Aziz, for always being a nice friend and

encouraging us to do social events. Our conversations about food and solving issues in Java have always been helpful. I am happy that I started my PhD work with you both. Thanks to Maria for her help in delivering DECIMER web application in a short amount of time and for being a very nice co-author. Thanks for allowing me to design your project logos. Thanks to Mahnoor for being such a nice friend and a nice colleague, along with the insightful talks we always shared.

A special thanks go to Jonas, a wonderful friend and colleague. Your knowledge in cheminformatics was always helpful, and the extended discussions on the questions that I had related to cheminformatics always helped me solve the problems. Our similar interest in movies and food led us to numerous pleasant evenings of fun and thank you for proofreading this work. Another special thanks go to Otto, one of my previous master's students, now a nice colleague and a friend. You are an excellent co-author who worked tirelessly on finishing all tasks. Our discussions about machine learning are valuable to me and thank you for proofreading this work. Thank you both for the patience and time you spent proofreading this work.

I would like to thank Jan-Mathis at this point. It was a pleasure to have worked with you on MSC. It was a pleasure to have you as my master's student too. I would also like to thank Ann-Kathrin Brüggemann and Lisa Sophie Kersten for their help in annotating the data for one of my projects. I thank Sarah for her administrative support and for guiding me initially during settling in Jena. I would like to thank Adelene, Luiz and Christian, with whom I had a very good time in the office. I consider myself lucky to have made many new friends among my colleagues, with whom I have been able to share many great experiences, from watching movies, enjoying food to countless coffee breaks. Also, I would like to thank Amol and Sagar for their wonderful friendship and the countless hours we spent discussing science and other stuff. Also, for being wonderful travelling companions.

Lastly, I would like to thank the most important people in my life, my parents, my brother and my uncle Sivakkumaran for their unwavering support all these years. Without your support, this whole journey would not have been possible.