OXFORD

Briefings in Bioinformatics, 2022, 1–12
https://doi.org/10.1093/bib/bbac346
Problem Solving Protocol

Identifying the kind behind SMILES—anatomical therapeutic chemical classification using structure-only representations

Yi Cao†, Zhen-Qun Yang†, Xu-Lu Zhang, Wenqi Fan, Yaowei Wang, Jiajun Shen, Dong-Qing Wei, Qing Li and Xiao-Yong Wei
Corresponding author: Xiao-Yong Wei, Department of Computer Science, Sichuan University, Chengdu 610065, China, and Department of Computing, Hong Kong Polytechnic University, Kowloon, Hong Kong. E-mails: cswei@scu.edu.cn, x1wei@polyu.edu.hk
†Yi Cao and Zhen-Qun Yang contributed equally to this work and share first authorship.

Abstract

Anatomical Therapeutic Chemical (ATC) classification for compounds/drugs plays an important role in drug development and basic research. However, previous methods depend on interactions extracted from STITCH dataset which may make it depend on lab experiments. We present a pilot study to explore the possibility of conducting the ATC prediction solely based on the molecular structures. The motivation is to eliminate the reliance on the costly lab experiments so that the characteristics of a drug can be pre-assessed for better decision-making and effort-saving before the actual development. To this end, we construct a new benchmark consisting of 4545 compounds which is with larger scale than the one used in previous study. A light-weight prediction model is proposed. The model is with better explainability in the sense that it is consists of a straightforward tokenization that extracts and embeds statistically and physicochemically meaningful tokens, and a deep network backed by a set of pyramid kernels to capture multi-resolution chemical structural characteristics. Its efficacy has been validated in the experiments where it outperforms the state-of-the-art methods by 15.53% in accuracy and by 69.66% in terms of efficiency. We make the benchmark dataset, source code and web server open to ease the reproduction of this study.

Keywords: Anatomical Therapeutic Chemical, ATC Classification, Drug Development, Deep Learning.

Introduction

To identify a given compound into Anatomical Therapeutic Chemical (ATC) system for studying its possible active ingredients, as well as its therapeutic, pharmacological and chemical properties, is of great significance to both drug development and basic research. A commonly adopted ATC system (https://www.whocc.no/atc/structure_and_principles/) is the one developed by the World Health Organization (WHO). It is a hierarchical classification system that contains five levels of categories, in which the first level consisting of 14 groups (as shown in Table 1) has been widely employed to develop methods for ATC classification in the recent decade.

ATC classification is modeled as a multi-label classification problem, where a given compound is assigned to one or several labels indicating its belongingness to the 14 groups. The study dates back to 2008 when Dunkel *et al.* [1] proposed the first method that predicts a single label of a compound using its physicochemical properties and molecular fingerprints. A great effort from researchers has been made since then by extending the problem from the single-label prediction [1–3] to multi-label

Yi Cao is a postgraduate student with the Dept. of Computer Science, Sichuan University, China. His research interests include drug representation learning and drug properties predication.
Zhen-Qun Yang is a project fellow with the Dept. of Computing, Hong Kong Polytechnic University, Kowloon, Hong Kong. She was a postdoctoral fellow with the Dept. of Biomedical Engineering, Chinese University of Hong Kong, Kowloon, Hong Kong when the study was initiated. Her research interests include health computing, multimedia retrieval, image processing, and machine learning.
Xu-Lu Zhang is a Ph.D student with the Dept. of Computing, Hong Kong Polytechnic University, Kowloon, Hong Kong. He was a postgraduate student with the Dept. of Computer Science, Sichuan University, China, when the study was initiated. He works on interpretable deep learning and biological signal processing.
Wenqi Fan is a research assistant professor with the Dept. of Computing, Kowloon, Hong Kong Polytechnic University. His research interests are in the broad areas of machine learning and data mining, with a particular focus on Recommender Systems, Graph Neural Networks, and Trustworthy AI.
Yaowei Wang is a tenured associate professor with Peng Cheng Laboratory, Shenzhen, China. His research interests include machine learning, multimedia content analysis, and understanding. He was the recipient of the second prize of the National Technology Invention in 2017 and the first prize of the CIE Technology Invention in 2015.
Jiajun Shen obtained his PhD. in Computer Science from the University of Chicago in 2018. Currently he holds the position of Chief AI Scientist at TCL Research and research associate at the University of Hong Kong. His research focuses on large scale representation learning and neural machine translation.
Dong-Qing Wei is a full professor at the School of Life Sciences and Biotechnology, Shanghai Jiao Tong University. He made many groundbreaking contributions to the development of bioinformatics techniques and their interdisciplinary applications to systems of ever-increasing complexity.
Qing Li is a chair professor with and the department head of the Department of Computing, Hong Kong Polytechnic University, Hong Kong. His current research interests include multimodal data management, conceptual data modeling, social media, Web services, and e-learning systems.
Xiao-Yong Wei is a professor with and the head of the Dept. of Computer Science, Sichuan University of China, and a visiting professor of the Dept. of Computing, Hong Kong Polytechnic University. Kowloon, Hong Kong. His research interests include multimedia computing, health computing, computer vision, and machine learning.

Received: May 6, 2022. **Revised:** July 11, 2022. **Accepted:** July 26, 2022

Table 1. Comparison of the two benchmark datasets over the level-1 ATC codes.

Code	Anatomical/Pharmacological Group	ATC-SMILES (#drugs)	Chen-2012 [4] (#drugs)	Overlapped (#drugs)
A	Alimentary tract and metabolism	618	540	517
B	Blood and blood-forming organs	158	133	126
C	Cardiovascular system	625	591	581
D	Dermatologicals	455	421	402
G	Genito urinary system and sex hormones	285	248	245
H	Systemic hormonal preparations, excl. sex hormones and insulins	143	126	123
J	Antiinfectives for systemic use	621	521	501
L	Antineoplastic and immunomodulating agents	402	232	223
M	Musculo-skeletal system	227	208	204
N	Nervous system	826	737	724
P	Antiparasitic products, insecticides and repellents	138	127	122
R	Respiratory system	462	427	419
S	Sensory organs	415	390	376
V	Various	252	211	204
Total #drugs (counted by virtual drugs defined in [4])		5627	4912	4767
Total #drugs (counted by the number of identical SMILES sequences)		4545	3883	3785

prediction [4–18], introducing unified benchmark ATC datasets and evaluation methodology [4, 19], enriching the features with additional information such as chemical–chemical interactions [4–11, 13–18, 20, 21], structural similarities [2, 4–7, 9–11, 13–18, 20, 21] and chemical ontology [6, 9, 14, 20], and increasing the availability with web servers [1, 3, 5, 6, 12, 13].

In Figure 1, we summarize 18 representative works proposed in recent 10 years to study the evolution. Two trends can be observed. One is the introduction of deep learning (DL) (e.g. GCN in [15], CNN in [14, 15]) to take the place of traditional machine learning methods (e.g. SVM in [2, 20], ML-GKR in [5, 6]). This is not surprising because DL is considered as a game changer in a wide range of disciplinary for its power of modeling complex relations. The other trend is the integration of more and more data sources for richer representations. For example, the compound descriptions from Wikipedia are used to construct the word embedding in [15], and the chemical subgraph similarities are employed in [5, 9, 13, 15, 16]. The inclusion of new resources can enrich the representations but may introduce additional issues at the same time. For example, it increases the complexity of the representations and models, which requires additional computing power for reproduction and deployment, not to mention the fact that some methods even require additional efforts to query external tools like Rdkit [22] or web services like SIMCOMP (https://www.genome.jp/tools/simcomp/) and SUBCOMP (https://www.genome.jp/tools/subcomp/). More importantly, most of these new resources are interactions extracted from STITCH [23] which is a dataset collected from previous clinical trials, physicochemical experiments and meta analysis. The reliance on STITCH makes the ATC prediction depend on lab experiments and thus less feasible and practical for new/unseen drugs/compounds.

In this paper, we present a pilot study to explore the feasibility of conducting ATC classification with a single resource, which simplifies both the data acquisition process and the model complexity. More specifically, we generate the representations based only on molecular structures. We argue that the (conventional) structural models such as molecular fingerprints and graphs are incomplete and suffer from information loss. For example, the molecular fingerprints are generated by simplifying the structure into binary vectors (e.g. Morgan fingerprint [24], MACCSKeys [25]). The molecular graphs are generated by transforming the structures into low-dimensional manifolds in which only the neighboring information to adjacent atoms are preserved [16]. The high-order or continuous information like the functional groups or branches are either simplified or ignored in these models. Therefore, a majority of previous methods are using these structural information as a supplementary source to the trail-dependent sources (e.g. STITCH [23]). In this paper, we propose to model the structure directly to avoid information loss. Specifically, we use the Simplified Molecular Input Line Entry System (SMILES) [26] as the data source, and leverage the power of deep learning to model the complex relations behind. More importantly, SMILES is used as the sole data source so that the reliance on lab experiments is eliminated in this study. We demonstrate that this can achieve comparable (or even superior) performance to those of the state-of-the-art ATC prediction methods which are using multiple data sources. The structure-only nature gives the proposed method the potential to save a significant amount of expensive resources for drug development or basic research (at least for that of pre-research steps). Other contributions of this study include: (1) we collect a new ATC dataset of 4545 compounds/drugs, which can be used either to develop the structure-only methods in the future, or as the supplementary to existing benchmarks; (2) the dataset, source code and web server will be publicly available; and (3) we propose a light-weight DL model called ATC-CNN which outperforms the state-of-the-art methods significantly in terms of both effectiveness and efficiency. The representations are straightforward and with better explainability.

This paper is organized following the five-step guideline in [27] which is widely adopted in ATC studies [4, 5, 7, 13, 20] as (1) benchmark dataset, (2) sample formulation, (3) operation algorithm, (4) anticipated accuracy and (5) web-server.

Materials and Methods

Benchmark Dataset

We construct a new benchmark ATC-SMILES for ATC classification in this study. ATC-SMILES consists of 4545 compounds/drugs and their SMILES sequences. The benchmark is with the maximum coverage (81.34%) of KEGG dataset [28] which contains all 5588 known drugs/compounds used for ATC analysis. Prior to this benchmark, the most widely adopted one is Chen-2012 [4] which covers 3883 (69.49%) drugs in KEGG and is mainly used for generating inter-drug correlations (e.g. STITCH [23]). The two benchmarks are compared in Table 1. ATC-SMILES is designed to be inclusive to Chen-2012, but there are 2.16% misalignment due to the mismatching of drug IDs that we will explain soon. ATC-SMILES can be extended with new drugs much easier than

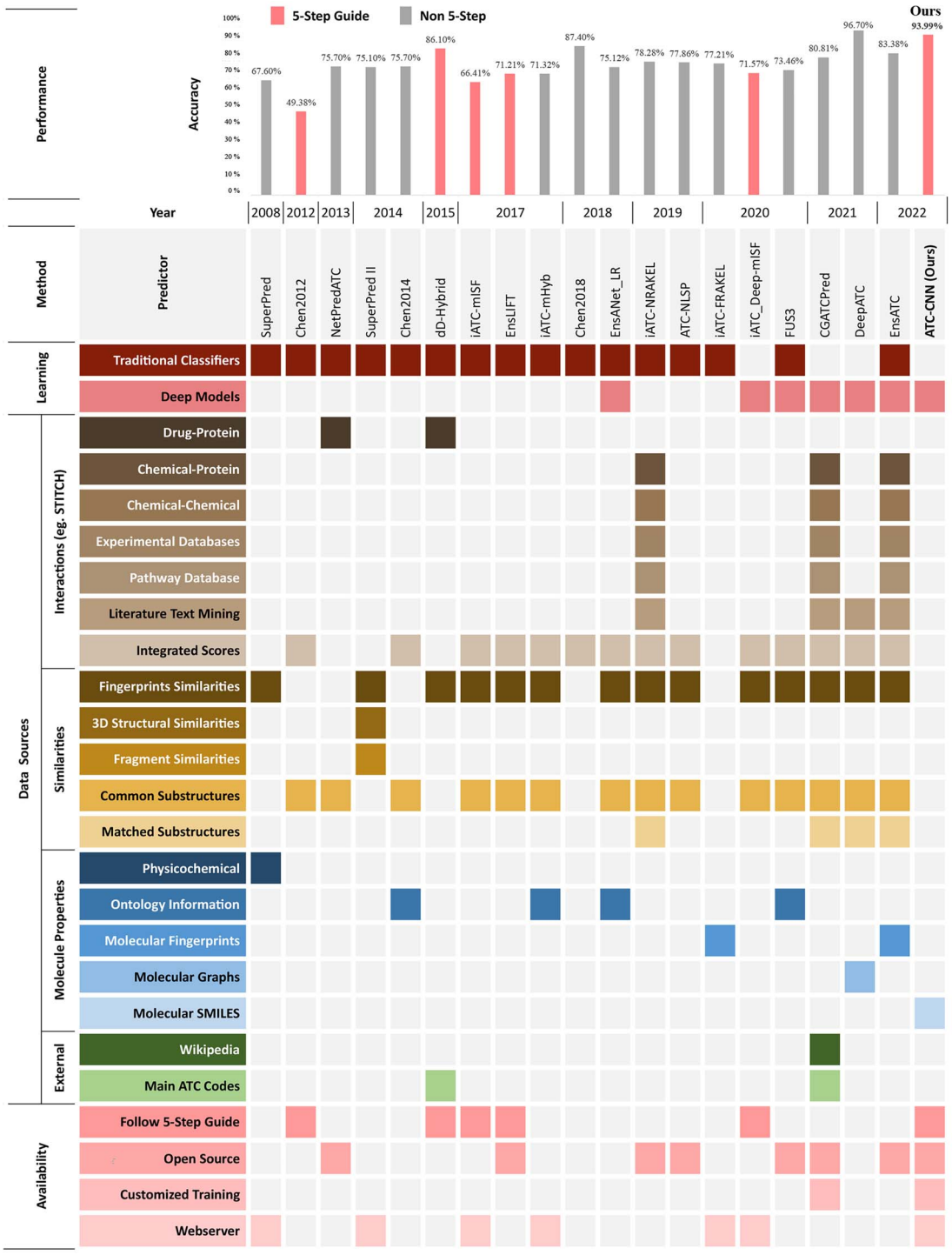

Figure 1. Summary of ATC classification methods in the last 10 years.

previous benchmarks as long as the SMILES sequences are available. Trails/experiments are not a must.

The process for generating ATC-SMILES starts by collecting all the 5588 drug IDs from KEGG using BioPython [29]. With the IDs as primary keys, we search the CIDs over PubChem dataset [30] for SMILES representations of these drugs. We skip drugs when there are no matched items or corresponding SMILES representations in PubChem (e.g. KEGG D11856: Empagliflozin, linagliptin and metformin hydrochloride). This results in the exclusion of 712 drugs.

Note that we generate the class labels using the same merging process that has been adopted in previous studies. The process merges two ATC codes as a single label if they are the same at level-1, otherwise, considers them as two separated labels. For example, Ketoprofen is with two ATC codes M01AE03 and M02AA10, but its level-1 code is consistent as M (i.e. MUSCULOSKELETAL SYSTEM). Therefore, Ketoprofen is assigned with a single label M. Ciprofloxacin is with four ATC codes J01MA02, S01AE03, S02AA15 and S03AA07. It is assigned with two labels of J (from J01MA02 for ANTIINFECTIVES FOR SYSTEMIC USE) and S (by combining S01AE03, S02AA15 and S03AA07 for SENSORY ORGANS).

Problem Formulation

We follow the common practice of modeling the ATC classification as a multi-label learning problem. To ease the introduction, we formulate it as a general framework before going into details.

Given a drug/compound representation $\mathbf{x} \in \mathbb{R}^d$ in a d-dimensional feature space, the goal is to learn a function $f : \mathbb{R}^d \rightarrow \{0,1\}^c$ that predicts the labels of $\mathbf{x}$ as

$$\hat{\mathbf{y}} = f(\mathbf{x}; \boldsymbol{\theta}), \tag{1}$$

where $\hat{\mathbf{y}} \in \{0,1\}^c$ is a multi-hot binary vector with the i^{th} element being 1/0 indicating the membership of $\mathbf{x}$ to the i^{th} ATC class, and the c is the number of classes which is a constant of 14 if the first-level ATC system is adopted. The f is parameterized by a set of p parameters ($\boldsymbol{\theta} \in \mathbb{R}^p$).

Given a ground-truth label vector $\mathbf{y} \in \{0,1\}^c$ formulated with the same way as the $\hat{\mathbf{y}}$, the learning is conducted with the objective of minimizing the loss $L(\hat{\mathbf{y}}, \mathbf{y})$ between the prediction and the ground-truth by finding the optimal $\boldsymbol{\theta}$ as

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} L(\hat{\mathbf{y}}, \mathbf{y}) \tag{2}$$

$$= \arg\min_{\boldsymbol{\theta}} L(f(\mathbf{x}; \boldsymbol{\theta}), \mathbf{y}). \tag{3}$$

Most of the previous studies use the same forms of $\hat{\mathbf{y}}$ and $\mathbf{y}$, but are different from each other regarding the $\mathbf{x}$, f, θ and $L(\hat{\mathbf{y}}, \mathbf{y})$ adopted.

In the proposed method, we learn the representations $\mathbf{x}$ using SMILES embedding and implement the f using a light-weight convolutional network (CNN) with parameters θ. Binary Cross Entropy is adopted as the loss $L(\hat{\mathbf{y}}, \mathbf{y})$. Let us breakdown our introduction into these components.

Algorithm 1 Offline Learning for Dictionary Construction

Input: SMILES sequences $\{s\}$
Output: Token Dictionary $\mathcal{D}$, Token Group Set $\mathcal{G}$, Transition Matrix $\mathcal{T}$, Regular Expressions $\mathcal{R}$

Initialization: $\mathcal{D} \Leftarrow \varnothing, \mathcal{G} \Leftarrow \varnothing, \mathcal{T} \Leftarrow \mathbf{0}, \mathcal{R} \Leftarrow \varnothing, \bar{\mathcal{D}} \Leftarrow \varnothing$
$\mathcal{C} \Leftarrow \{s\}$
while $\mathcal{C} \neq \varnothing$ **do**
  $\mathcal{C} \Leftarrow \varnothing$
  for $\forall s \in \{s\}$ **do**
    $\mathcal{C} \Leftarrow \mathcal{C} \cup \{t \mid \forall t \subset s, t \notin \mathcal{D} \cup \bar{\mathcal{D}}, len(t) \in [1,6]\}$
  end for
  Calculate confidences $\phi(\mathcal{C}) = \{\phi(t) \mid \forall t \in \mathcal{C}\}$ (Eq. (4))
  Sort $\mathcal{C}$ in descending order according to $\phi(\mathcal{C})$
  Select the top-20 to form a refined candidate set $\mathcal{C}^*$
  Add tokens that matches any expression in $\mathcal{R}$ to $\mathcal{C}^*$
  for $\forall t \in \mathcal{C}^*$ **do**
    if t passes human validation **then**
      $\mathcal{D} \Leftarrow \mathcal{D} \cup t$
    else
      $\bar{\mathcal{D}} \Leftarrow \bar{\mathcal{D}} \cup t$
    end if
  end for
  Identify new groups $\{g\}$ from $\mathcal{D}$
  $\mathcal{G} \Leftarrow \mathcal{G} \cup \{g\}$
  Identify new regular expressions $\{e\}$ from $\mathcal{G}$
  $\mathcal{R} \Leftarrow \mathcal{R} \cup \{e\}$
end while

for pair $\forall (g_i, g_j) \in \{(g_i, g_j) \mid g_i \in \mathcal{G}, g_j \in \mathcal{G}, g_i \neq g_j\}$ **do**
  Calculate $\mathcal{T}_{ij}$ using Eq. (7)
end for

Algorithm 2 Online Learning for Token Partition

Input: A SMILES sequences s, Regular Expressions $\mathcal{R}$, Probability Automaton $\boldsymbol{A}$
Output: The Partition of s into a List of Tokens $\mathcal{L}$

Initialization: $\mathcal{L} \Leftarrow \varnothing$, Position $i \Leftarrow 0$, Stack $\Xi \Leftarrow \varnothing$, Candidates $\mathcal{C} \Leftarrow \varnothing$, Current State (Group) of the Automaton $g \Leftarrow \boldsymbol{A}_0$
while $i < len(s)$ **do**
  if $\mathcal{C} = \varnothing$ **then**
    $\mathcal{C} \Leftarrow \{t \mid \forall t = s[i : i+n], \forall n \in [1,6], \exists \mathcal{R}(t) = True\}$
    Calculate $\Pr(\mathcal{C} \mid g) = \{\Pr(\mathcal{G}(t) \mid g) \mid \forall t \in \mathcal{C}\}$
    Sort $\mathcal{C}$ in deceding order regarding $\Pr(\mathcal{C} \mid g)$
  end if
  if $\mathcal{C} \neq \varnothing$ **then** *#candiates found*
    PUSH $(\mathcal{C}, i)$ into Ξ
    Update current state $g \Leftarrow \mathcal{G}(\mathcal{C}[0])$
    Update current position $i \Leftarrow i + len(\mathcal{C}[0])$
    $\mathcal{C} \Leftarrow \varnothing$
  else*#no candidates found*
    $\mathcal{C} \Leftarrow \varnothing$
    while $\Xi \neq \varnothing$ **do** *#backtrack to last available*
      $(\mathcal{C}, i) \Leftarrow$ **POP** from Ξ
      if $|\mathcal{C}| > 1$ **then**
        $\mathcal{C} \Leftarrow \mathcal{C} \setminus \mathcal{C}[0]$ *#remove failed branch*
        BREAK
      end if
    end while
    if $(\mathcal{C} = \varnothing) \wedge (\Xi = \varnothing)$ **then***#no available position*
      Return Error
    end if
  end if
end while

while $\Xi \neq \varnothing$ **do**
  $(\mathcal{C}, i) \Leftarrow$ **POP** from Ξ
  Insert $\mathcal{C}[0]$ to the head of $\mathcal{L}$
end while

Tokenization and Representation Generation

We generate the representation $\mathbf{x}$ of a drug/compound using its SMILES sequence. The first step is to split a sequence into a set of tokens. Although the idea is intuitive, there are only a few works in literature which have studied this problem. In [31], Goh *et al.* propose SMILES2vec in which every character in the SMILES sequence is considered as a token for embedding, while in [32], Zhang *et al.* propose SPVec which breaks a sequence into tokens using a sliding window of size 3. Tokens generated with those methods may not be physicochemically meaningful, and less representative when used for embedding. We address this issue with an interactive process between a statistical tokenizer and human experts.

The motivation is to find tokens that are both **statistically meaningful** so as to make the modeling effective and efficient, and **physicochemically meaningful** to human experts so as to increase the explainability of the method. As shown Figure 2, the proposed tokenization process consists of three parts: (1) a *Token Extractor* to propose candidate tokens and corresponding confidence scores; (2) a *Domain Knowledge Injector* in which human experts identify physicochemically meaningful tokens from the highly confident candidates so as to construct a token dictionary as well as generalize rules for further extraction; and (3) a *Sequence Validator* which finds the best partition among all combinations of tokens for a given sequence.

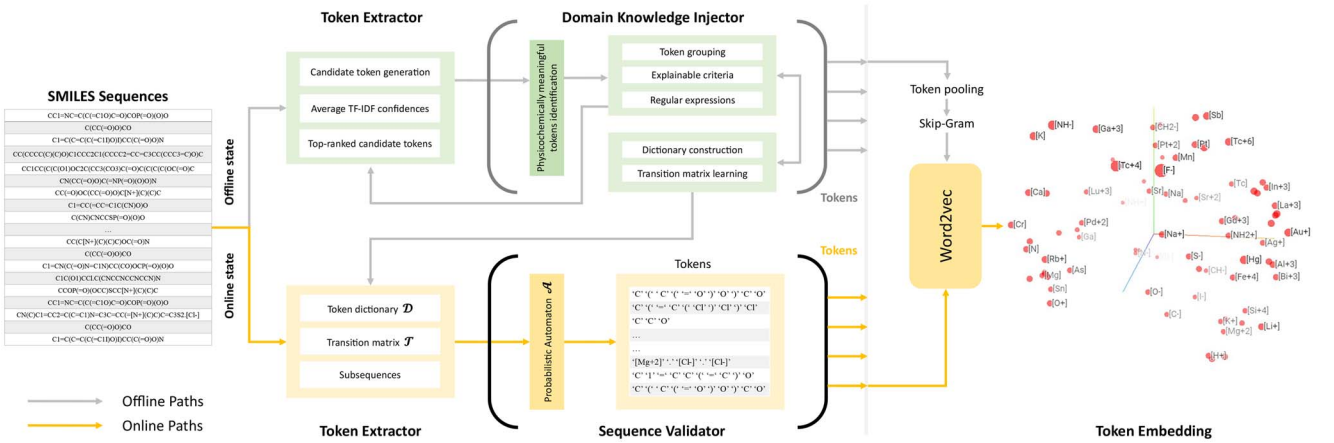

Figure 2. Token Generation and Embedding. A dictionary consisting of the best set of tokens is learned jointly by the statistical extractor and human experts (offline), and used to conduct the tokenization (online). The token embeddings are learned by retraining the Word2vec in an ATC setting.

Token Extractor

The *Token Extractor* has offline and online states. The offline state is for learning the token dictionary $\mathcal{D}$ by working with the domain knowledge injector, while the online state is for extracting tokens for ATC classifications by working the *Sequence Validator*. We will introduce the offline state and delay the introduction of online state until that of the validator.

In offline state, the extractor scans the sequences with a windows size k ranging from 1 to 6. This pools all the candidate tokens of length k together for dictionary learning. For each candidate, we calculate the term frequency-inverse document frequency (i.e. *tf-idf* [33–35]) and use its average *tf-idf* over all sequences as the statistical confidence which also indicates how much information the candidate carries when compared with other tokens. Denoting the sequences of ATC-SMILES as a set $\{s\}$ in which s is the SMILES sequence for a given drug/compound, the $t \in s$ is candidate token in s, the confidence $\phi(t)$ is calculated as

$$\phi(t) = \sum_{s \in \{s\}} \frac{tf(t,s) \cdot idf(t,\{s\})}{\|\{s\}\|}, \tag{4}$$

$$tf(t,s) = \frac{\rho(t,s)}{\sum_{t' \in s} \rho(t',s)}, \tag{5}$$

$$idf(t,\{s\}) = \log\left(\frac{\|\{s\}\|}{\|\{s \mid t \in s\}\|}\right), \tag{6}$$

where $\rho(t,s)$ counts the number of times that a candidate t appears in the sequence s. Once the confidence scores are calculated, we rank the candidates in a descending order based on the scores, and select the top-20 candidates to pass to the *Human Knowledge Injector*. This excludes the candidates like '(((" and "]))' which have a high term frequency but low inverse document frequency making them less informative.

Human Knowledge Injector

The Injector takes the top-20 candidates as the input and involves human experts in the learning. The human experts identify the physicochemically meaningful tokens from the candidates and add them into the token dictionary $\mathcal{D}$. In addition, the experts conduct two generalization steps to inject knowledge into the extractor and validator, respectively.

One is to categorize the tokens in the dictionary into a set of groups $\{g_i\}$ according to their physicochemically characteristics. For each group, the experts generate selection criteria to make the results explainable (as the results shown in Figure 2). A set of regular expressions is also generated for each group by observing the token patterns. The token extractor will use these expressions for more efficient extraction. This makes the results generalizable. For example, a regular expression '[??+?]' generated from tokens '[Fe+4]' can help identify '[Lu+3]', '[Cu+2]' and '[Al+3]' in the future. The dictionary learning process jointly conducted by the extractor and injector is then repeated until no tokens can be identified and the dictionary $\mathcal{D}$ is ready. This results in eight groups of 109 tokens as shown in Figure 3.

The injector also learns the inter-group relations by calculating the transition probabilities among groups. We encapsulate the probabilities into a transition matrix $\mathcal{T}$ in which the $(i,j)^{th}$ entity indicates the probability of observing a token from the group g_j when a token from the group g_i is observed previously. Denoting t_k and t_{k+1} as two consecutive tokens, we have

$$\mathcal{T}_{ij} = \Pr(t_{k+1} \in g_j \mid t_k \in g_i). \tag{7}$$

Note that the transition matrix $\mathcal{T}$ is asymmetric and we will inject it into the validator for the online tokenization process. The whole offline state algorithm is defined in Algorithm 1.

Sequence Validator

The validator works together with the *Token Extractor* to split a sequence into actual tokens based the token dictionary $\mathcal{D}$ and transition matrix $\mathcal{T}$. In the online tokenization process for a sequence s at the step k, the validator selects for next step $k+1$ the most possible token t_{k+1} that maximizes the probability $\Pr(t_0, t_1, \ldots, t_k, t_{k+1})$ of observing the subsequence $t_0, t_1, \ldots, t_k, t_{k+1}$. This is a typical Markov Chain process and can be represented as a probabilistic automaton

$$\mathbf{A} = (\{g_i\}, \mathcal{D}, \mathcal{T}), \tag{8}$$

where we use the $\{g_i\}, \mathcal{D}, \mathcal{T}$ as the states, symbol set and transition matrix, respectively. A visual illustration of the automaton can be found in Figure 3.

We use the automaton $\mathbf{A}$ to find the best partition of a sequence into tokens. At a given step k and a position i in the current

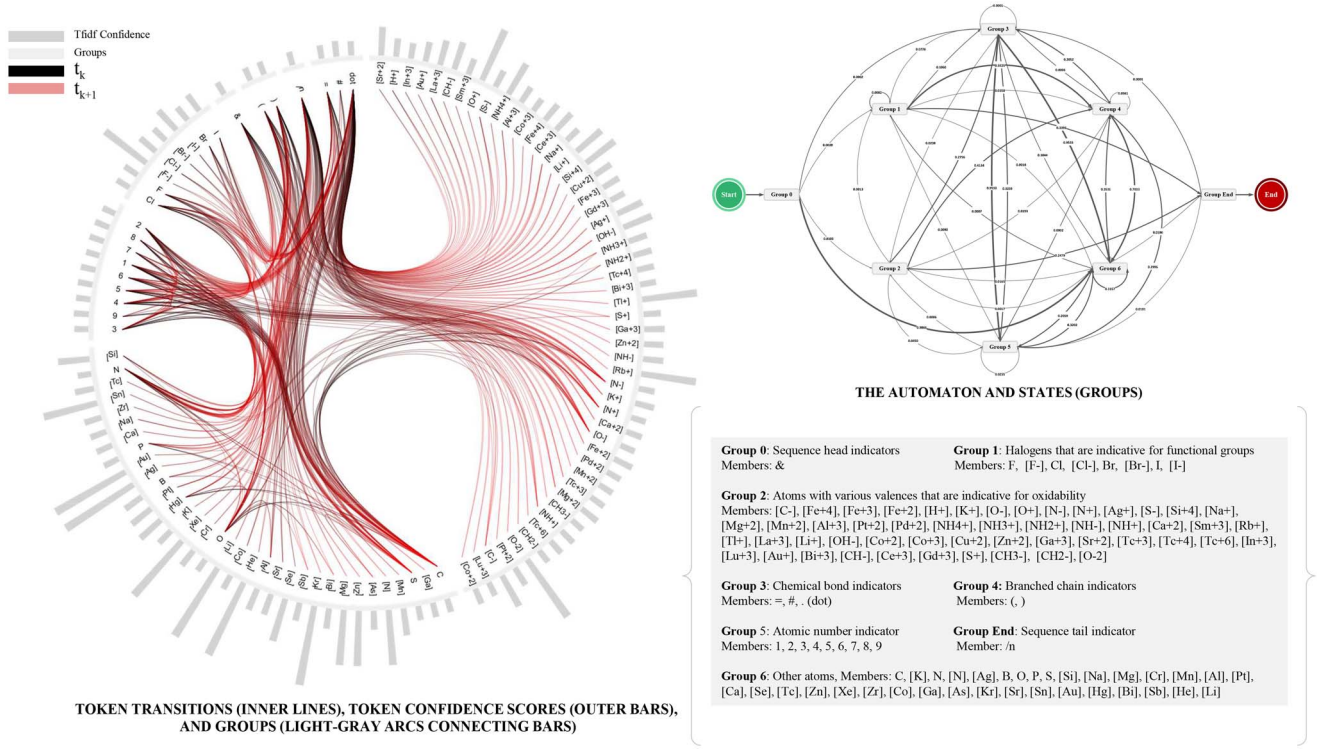

Figure 3. Tokens, Transitions, Groups and the Probabilistic Automaton.

sequence, the *Token Extractor* reads in n characters from current sequence (n equals to the maximum length of tokens in $\mathcal{D}$). The validator searches from the n-character sub-sequence for a set $\{\hat{t}_i\}$ of all possible candidate tokens that appear in the dictionary $\mathcal{D}$, and selects the one which maximizes the transition probability for the step $k+1$ as

$$t_{k+1} = \arg\max_{\hat{t}_i \in \{\hat{t}_{k+1}\}} \Pr(\hat{t}_i \in g_j \mid t_k \in g_i). \quad (9)$$

The rest of the candidate tokens and current position i are pushed into a stack Ξ. In case the candidate set is empty (i.e. no tokens can be detected at current position), we pop from the stack Ξ to find the last position that we have valid candidates. The validator returns to the position and restarts by evaluating the candidates at the position. The algorithm is a backtracking algorithm that we formally defined in Algorithm 2.

Generating the Representation $\mathbf{x}$ *using Word Embedding*

Once the tokens are extracted from sequences, we pool them for word embedding. It generates for each token a vector representation $\mathbf{t} \in \mathbb{R}^m$, so that the representation can be used to generate the drug/compound representation $\mathbf{x} \in \mathbb{R}^d$. To this end, word embedding learns a linear space $\mathbb{R}^m$, in which tokens, once being represented into the space, distribute closely with their contextually related tokens (i.e. those co-occur frequently within the same sliding windows with them) while far from the non-related ones. For example, the distance of representations of 'C' and 'O' is smaller than that of the '[Cl-]' and '='.

We adopt the Word2Vec [36] and Skip-gram model [37] to conduct the learning by setting the dimensionality of the target space $m = 300$, the context window size of 12 and the negative sample rate at 15. In this case, Skip-gram generates positive training examples by taking each token in a 15-token window as the predictor, the rest of 14 tokens as the context target and a label 1 indicating the contextual relatedness. By contrast, Skip-gram generates negative training example by replacing the context targets with those from different windows (randomly selected) and setting the label to 0 indicating the non-contextual relatedness.

After training, we have the token represented in a space $\mathbb{R}^{300}$. The drug/compound representation of a SMILES sequence s is constructed by simply stacking its member token representations as

$$\mathbf{x}_s = \{\mathbf{t}\} \in \mathbb{R}^{\|s\| \times 300}, \forall t \in s. \quad (10)$$

However, this makes the length of the representation vary when the length of the sequence $\|s\|$ varies. We use zero-padding to address this issue which extends the length of the representation to 787 (the maximum length of sequences in ATC-SMILES) using zero vectors $\mathbf{0}$ as

$$\mathbf{x} = \{\mathbf{x}_s \in \mathbb{R}^{\|s\| \times 300}, \mathbf{0} \in \mathbb{R}^{(787-\|s\|) \times 300}\}, \mathbf{x} \in \mathbb{R}^{787 \times 300}. \quad (11)$$

Then, the representation $\mathbf{x}$ is ready for the model learning.

Model f and Parameters θ

We design the model f with an ad hoc CNN by following the practice of TextCNN [38] family which is recognized with promising performance on text liked sequences. To ease the description, we call it ATC-CNN hereafter. As shown in Figure 4, ATC-CNN is a seven-stream and light-weight CNN. The seven streams take the $\mathbf{x} \in \mathbb{R}^{787 \times 300}$ as the input and process it in parallel, which results in seven feature maps. The feature maps are then flattened and concatenated as a single feature vector for the inference in the fully connected (FC) layers.

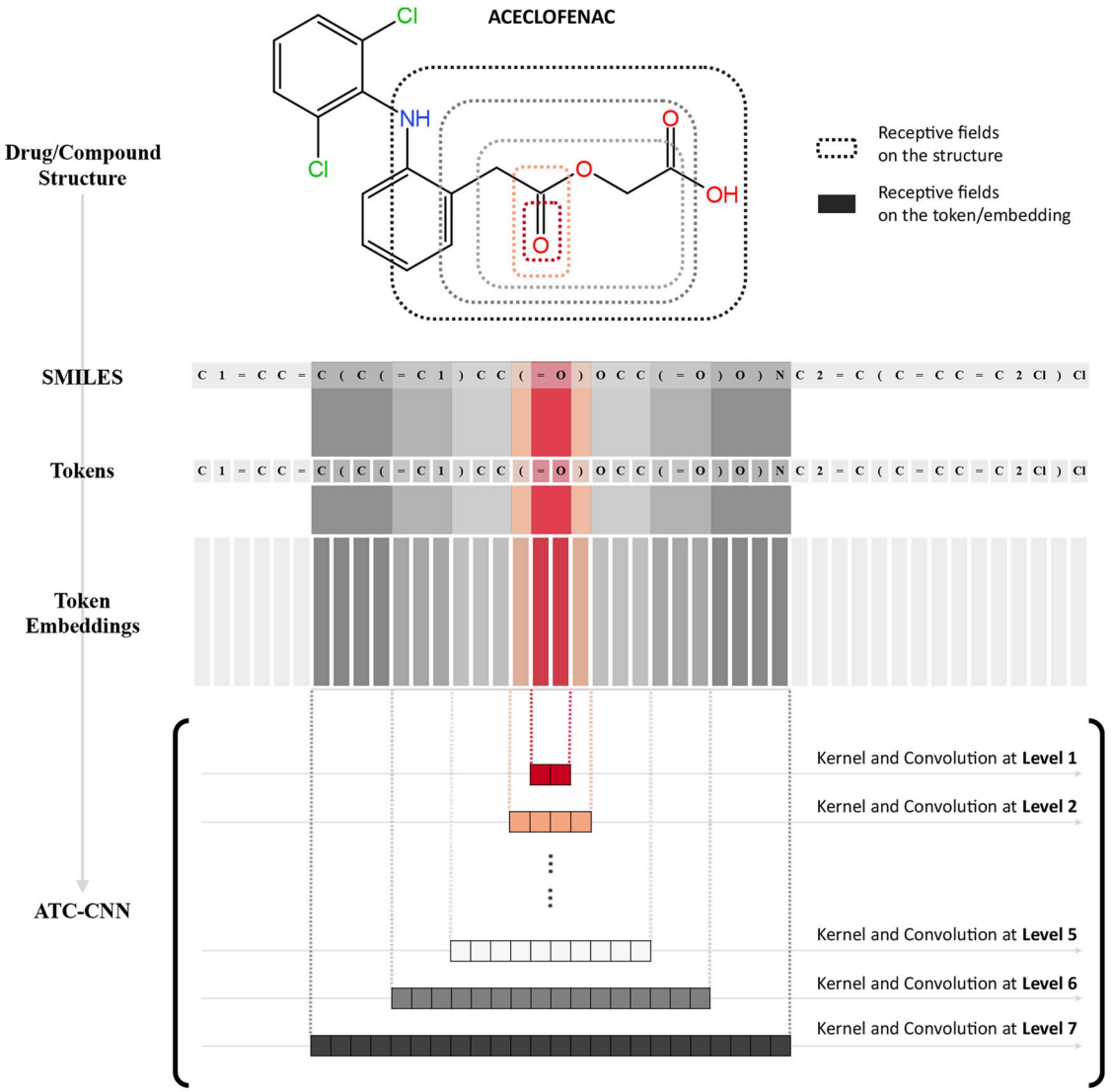

Figure 4. Representation generation using ATC-CNN and the receptive fields captured for Aceclofenac. The kernel pyramid enables a multi-resolution modeling and embedding of compound structures. It captures a double-bond and an Oxygen atom at the kernel size 2, expands to a Carbanyl group at size 4 and includes a functional group of Esters at size 6. More branches and groups are included and jointly embedded while the kernel is expanding to size 24.

Convolutional Layers

The convolution kernel $\boldsymbol{\kappa}_i^c$ of the c^{th} channel of the i^{th} stream is with a size of $787 \times 2m_i$, which constraints the convolution to be conducted only along the first dimension of $\mathbf{x}$ (i.e. the kernel moves across tokens). This makes the convolution work as a local structure extractor to summarize the relation of the $2m_i$ adjacent tokens.

As shown in Figure 4, while the m_i varies from 1 to 12, the seven kernels work together to form a pyramid-liked extraction scheme for sensing the structure at different scales. This is especially useful for modeling the functional groups and subbranches. In Figure 4, it captures a double-bond and an Oxygen atom at the kernel size 2, together with a Carbonyl group at size 4, and a functional group of Esters at size 6. More and more branches and connected structures are included and jointly embedded while the kernel is expanding. This is also different from the LSTM [39–42] or Transformers [43–46], which model the structure implicitly. ATC-CNN is thus with better explainability. The convolution is defined as

$$\mathbf{x} * \boldsymbol{\kappa}_i^c = \left\{ \left[\sum_{k=-m_i}^{m_i} \sum_{l=0}^{299} \mathbf{x}(r + m_i + k, l) \cdot \boldsymbol{\kappa}_i^c(k, l) \right]_r \right\}, \quad \forall r \in [0, 787 - 2m_i + 1), \forall c \in [0, 256) \quad m_i \in \{1, 2, 3, 4, 5, 8, 12\}, \tag{12}$$

where the tuple $(,)$ is used to represent the element index of a matrix. By repeating the convolution in Eq. (12) for 256 channels, we have a feature map of shape $(787 - 2m_i + 1) \times 256$ for the i^{th} stream.

After the convolution, we use max-pooling (1-Max) to obtain the feature map for each stream. It indeed takes the maximum from each channel to construct a feature map of shape 1×256. Finally, the seven feature maps are then concatenated into a single vector of the length $256 \times 7 = 1792$ as the feature representation

$\Phi \in \mathbb{R}^{1792\times 1}$. The whole process can be formulated as

$$\mathbf{\Phi} = \bigoplus_{i=1}^{7} Flatten\left(Maxpool\left(\bigoplus_{c=0}^{255} [\mathbf{x} * \boldsymbol{\kappa}_i^c] \right) \right), \tag{13}$$

where $\bigoplus$ denotes the concatenation operator.

FC Layers and Predictions $\hat{\mathbf{y}}$

In the FC layers, the feature map $\mathbf{\Phi}$ is fully connected to another layer of 14 neurons. The values on these 14 neurons are used for generating predictions $\hat{\mathbf{y}}$. This creates 1792×14 connections on which we encapsulate a weight matrix $\mathbf{W} \in \mathbb{R}^{1792\times 14}$. The predictions are then written

$$\hat{\mathbf{y}} = Dropout(\mathbf{W}^\top \mathbf{\Phi}) \in \mathbb{R}^{14}, \tag{14}$$

where the dropout probability is set at 0.2.

Loss Function $L(\hat{\mathbf{y}}, \mathbf{y})$ and Parameter Learning

We measure the predictions with Binary Cross Entropy using Logits Loss Function as

$$L(\hat{\mathbf{y}}, \mathbf{y}) = -\frac{1}{14}\left(\mathbf{y}^\top \ln\left(g(\hat{\mathbf{y}}) \right) + (1-\mathbf{y})^\top \ln\left(1 - g(\hat{\mathbf{y}}) \right) \right), \tag{15}$$

$$g(x) = \frac{1}{1+e^{-x}}. \tag{16}$$

Up to here, the parameter set $\boldsymbol{\theta}$ includes a set of 256×7 kernels from the convolutional layers and the weights from the FC layers as

$$\boldsymbol{\theta} = \left\{ \{\boldsymbol{\kappa}_i^c\}, \mathbf{W} \right\}. \tag{17}$$

We use Adam optimizer to learn the $\boldsymbol{\theta}$ with $batchsize = 16$ and learning rate $1e-3$.

Results and Discussion

Metrics for Multi-label ATC Classification

We adopt the five metrics, which were established in [19] and are extensively employed in literature, to evaluate the performance of ATC classification in multi-label prediction setting as follows:

$$Aiming = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\|L_i \cap \hat{L}_i\|}{\|\hat{L}_i\|} \right) \tag{18}$$

$$Coverage = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\|L_i \cap \hat{L}_i\|}{\|L_i\|} \right) \tag{19}$$

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\|L_i \cap \hat{L}_i\|}{\|L_i \cup \hat{L}_i\|} \right) \tag{20}$$

$$Absolute\ True = \frac{1}{N}\sum_{i=1}^{N}\left(\Delta(L_i, \hat{L}_i) \right) \tag{21}$$

$$\Delta(L_i, \hat{L}_i) = \begin{cases} 1, & if\ L_i\ is\ identical\ with\ \hat{L}_i. \\ 0, & otherwise. \end{cases} \tag{22}$$

$$Absolute\ False = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\|L_i \cup \hat{L}_i\| - \|L_i \cap \hat{L}_i\|}{M} \right) \tag{23}$$

where N donates the total number of all samples, and M represents the number of labels. $\|\ \|$ is the operator acting on the set therein to count the number of its elements. L_i is the true label of the i^{th} drug, while $\hat{L}_i$ donates the predicated label. $\cup$ and $\cap$ represent the union and intersection operation, respectively. To ease the reading, we use $\uparrow$ as the indicator for positive indices (i.e. Aiming, Coverage, Accuracy and Absolute True) when presenting the results. Similarly, we use $\downarrow$ for negative indices (e.g. Absolute False).

Cross-validation

We use jackknife test[47] for cross-validation, which is considered the least arbitrary method that outputs unique outcome for the ATC benchmark dataset [48]. Therefore, jackknife test is commonly adopted for evaluating the ATC predictors in almost all the previous studies [4–7, 9–15].

Comparison with SOTA Methods

We compare the performance of the proposed ATC-CNN with 14 SOTA methods that are with performance reported in the five metrics. The 14 methods include those using various representations (chemical interactions, chemical structural features, molecular fingerprint features, pre-trained word embedding, ATC codes association information and drug ontology information) and models (SVM, ML-GKR, LIFT, NLSP, RAKEL, RR, CNN, GCN, hMuLab and LSTM). It is the most comprehensive comparison that we can find in literature. To be consistent with previous studies, we have also conducted experiments on the Chen-2012 benchmark. However, due to the aforementioned missing SIMILES issue, we have to set the representations of the 98 out of 3883 drugs with absent SIMILES structures (2.52% of the dataset) to zero vectors. The results are shown in Table 2.

ATC-CNN outperforms the SOTA methods by 1.62%, 6.40%, 7.15%, 7.68% and 0.22% on Chen-2012[4] in Aiming, Coverage, Accuracy, Absolute True and Absolute False, respectively. The superiority of the proposed method is more obvious on Absolute True, and Absolute False, which are two of the strictest metrics. This is an indication that ATC-CNN is better in providing the exactly matched labels to these of the ground-truth (measured by Absolute True), and is less possible to make all labels wrong (measured by Absolute False). It is a preferable characteristic in drug development because the risk-benefit ratio of developing a new drug can be better evaluated before starting the costly experimental process.

Comparison on Aligned Dataset

Although ATC-SMILES is with a larger scale than that of Chen-2012[4], there are some mis-aligned items. We remove these items to generate a subset consisting of 3785 drugs/compounds which are shared in common by the ATC-SMILES and Chen-2012. We call this set ATC-SMILES-Aligned hereafter. With ATC-SMILES-Aligned, we can study the characteristics of ATC-CNN in more detail by comparing its performance with that of CGATCPred[15]. CGATCPred is the only one in literature that has source code available and allows users to retrain the model by themselves (see statistics in Figure 1).

Overall Performance: As shown in Table 3, ATC-CNN outperforms the CGATCPred by 16.63%, 12.73%, 15.53%, 16.91% and 1.84% in Aiming, Coverage, Accuracy, Absolute True and Absolute

Table 2. Performance comparison with SOTA methods. The best results are in bold font.

Method	Year	Dataset	#Drugs	Rep.				Model	Aiming ↑	Coverage ↑	Accuracy ↑	Absolute True ↑	Absolute False ↓
Chen *et al.* [4]	2012	Chen-2012	3883	I	S			Similarity Search	50.76%	75.79%	49.38%	13.83%	8.83%
iATC-mISF[5]	2017	Chen-2012	3883	I	S	F		ML-GKR	67.83%	67.10%	66.41%	60.98%	5.85%
iATC-mHyb[6]	2017	Chen-2012	3883	I	S	F	O	ML-GKR	71.91%	71.46%	71.32%	66.75%	2.43%
EnsLIFT[7]	2017	Chen-2012	3883	I	S	F		LIFT	78.18%	75.77%	71.21%	63.30%	2.85%
EnsANet_LR[9]	2018	Chen-2012	3883	I	S	F		CNN,LIFT,RR	75.40%	82.49%	75.12%	66.68%	2.62%
EnsANet_LR⊗DO[9]	2018	Chen-2012	3883	I	S	F	O	CNN,LIFT,RR	79.57%	83.35%	77.78%	70.90%	2.40%
ATC-NLSP[10]	2019	Chen-2012	3883	I	S	F		NLSP	81.35%	79.50%	78.28%	74.97%	3.43%
iATC-NRAKEL[11]	2020	Chen-2012	3883	I	S			RAKEL,SVM	78.88%	79.36%	77.86%	75.93%	3.63%
iATC-FRAKEL[12]	2020	Chen-2012	3883	F				RAKEL,SVM	78.51%	78.40%	77.21%	75.11%	3.70%
FUS3[14]	2020	Chen-2012	3883	I	S	F		CNN,LSTM,LIFT,RR	87.55%	69.73%	73.46%	68.71%	2.38%
FUS3⊗DO[14]	2020	Chen-2012	3883	I	S	F	O	CNN,LSTM,LIFT,RR	79.79%	84.22%	79.64%	73.04%	2.09%
iATC_Deep-mISF[13]	2020	Chen-2012	3883	I	S	F	O	DNN	74.70%	73.91%	71.57%	67.01%	**0.00%**
CGATCPred[15]	2021	Chen-2012	3883	I	S	E	A	CNN,GCN	81.94%	82.88%	80.81%	76.58%	2.75%
EnsATC[18]	2022	Chen-2012	3883	I	S	F		hMuLab,LSTM	91.39%	84.32%	83.38%	80.09%	1.31%
ATC-CNN	2022	Chen-2012	3883	S				CNN	93.01%	90.72%	90.53%	87.77%	1.53%
ATC-CNN	2022	ATC-SMILES	4545	S				CNN	**95.83%**	**94.14%**	**93.99%**	**91.77%**	0.94%

Representation (Rep.) abbreviations: I - Chemical interactions, S - Chemical structural features, F - Molecular fingerprint features, O - Drug ontology information, E - Pre-trained word embedding, and A - ATC codes association information.

Table 3. Performance comparison on ATC-SMILES-Aligned.

Predictor		Effectiveness					Efficiency		
Model	#parameters (million)	Aiming ↑	Coverage ↑	Accuracy ↑	Absolute True ↑	Absolute False ↓	Training ↓ (ms./epoch)	Training ↓ (ms./sample)	Testing ↓ (ms./sample)
CGATCPred[15]	211.98	78.18%	79.91%	76.92%	72.84%	3.04%	67036.75	16.84	3.95
ATC-CNN	5.44	94.81%	92.64%	92.45%	89.75%	1.20%	19284.53	5.11	1.90

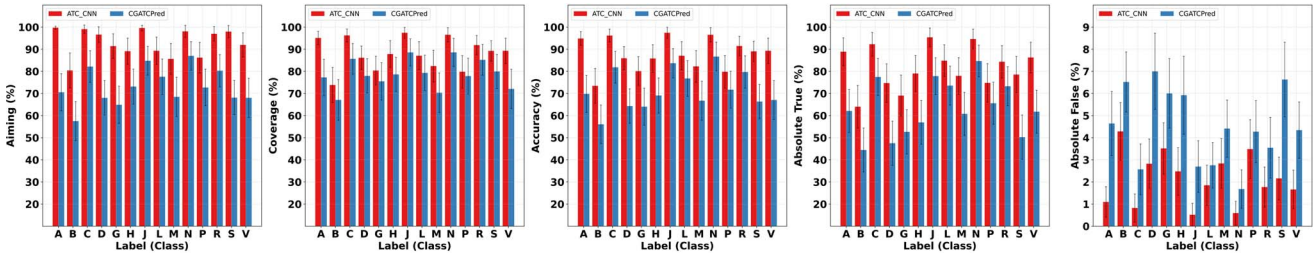

Figure 5. Performance comparison over labels/classes. ATC-CNN outperforms CGATCPred and is with smaller standard deviation.

False, respectively. Given the fact that the number of parameters of ATC-CNN (5.44 *million*) is 97.43% less than that of CGATCPred (211.98 *million*), this is a surprising result. Further benefiting from the light-weight model, ATC-CNN is 229.55% and 107.89% faster than CGATCPred in training and testing, respectively.

Class-dependent Performance: To investigate the generalizability of the proposed method, we compare the performance of the two methods over labels/classes. The results are shown in Figure 5.

The superiority of the proposed method is observed over all classes. For example, ATC-CNN obtains an accuracy above 80% on 13 out of the 14 labels/classes, while there are only four classes on which CGATCPred demonstrates a comparable performance. Furthermore, ATC-CNN appears more stable than CGATCPred, indicated by its smaller standard deviation than that of CGATCPred.

It is worth mentioning that both methods show inferior performance on class B with the accuracy below 80%. This is due to fact that class B is with the largest portion of inorganic salt (23.81%) when compared with that of other classes (less than 1%). Inorganic salt (e.g. $MgCl_2$, NaCl, KCl, $CaCl_2$) are with a SMILES length ranging from 3 to 7, which is much shorter than the standard representation length 787. This introduces an overwhelming number of zeros into their representations (because of the padding) which makes them less informative than others.

In addition, the most significant performance difference of the two methods is observed in class A, where ATC-CNN outperforms CGATCPred by 25.03%. The reason is that class A contains 28.85% of multi-label instances of ATC-SMILES-Aligned. As we will see in the next section that the majority of performance gain of ATC-CNN over CGATCPred is obtained on multi-label instances, it is not surprising that the performance difference on class A is also unique among the 14 classes.

Performance over #Labels: We also compare the performance over the number of labels. ATC-CNN outperforms CGATCPred significantly as expected. However, on the metric Coverage, ATC-CNN shows inferior performance over CGATCPred. The reason is

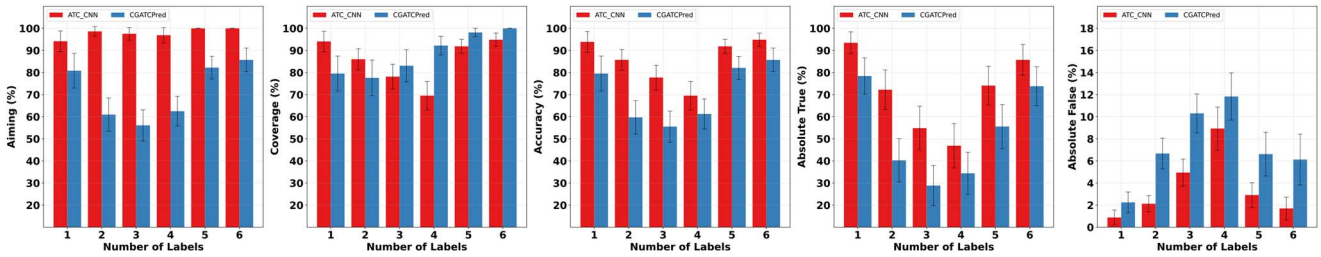

Figure 6. Performance comparison over number of labels/classes. ATC-CNN outperforms CGATCPred and is with smaller standard deviation.

that CGATCPred is a method with preference on the recall. It tends to output more labels and puts less focus on the precision. When investigating jointly with the metrics of Aiming and Accuracy, it is more evident that ATC-CNN obtains a better balance between the precision and recall.

Both methods obtain a U-shape performance when the number of labels increases. This is counterintuitive, because drugs with more labels (ATC codes) usually raise more challenges to the models than those with less. The proposed method makes correct predictions for drugs with six labels (and ATC codes up to 11) such as Chlorhexidine Gluconate (five labels or eight codes), Prednisolone sodium phosphate (6 labels or 10 codes) and Dexamethasone acetate (6 labels or 11 codes). The performance drops down to the valley at point 3 and climbs up afterwards. This can be explained by a Bernoulli process in which the network outputs six binary variables corresponding to the six labels/classes, and the variables are independent of each other. It becomes intuitive instantly that the maximum entropy obtains when three out of the six variables are with values of 1, while the rest of others are all 0.

In terms of single ($\#labels = 1$) versus multiple ($\#labels > 1$) labels, the performance gain of ATC-CNN over CGATCPred on multi-label ones is more significant than on single-label instances with 35.47%, 2.49%, 22.29%, 27.61% and 4.55% in Aiming, Coverage, Accuracy, Absolute True and Absolute False, respectively.

Web Server

In addition to making the source code of ATC-CNN open on Github.com, we develop a web server at http://www.aimars.net:8090/ATC_SMILES/ to increase the availability of the method and dataset. The web server takes a drug/compound ID, or SMILES sequence as the input, and predicts the labels and top-five related drugs/compounds. The ID and sequence are not necessarily from ATC-SMILES, the server is capable of predicting labels for any drugs or compounds with valid IDs or sequences.

Conclusion

We present a pilot study to explore the possibility of conducting ATC classification solely based on the structural information (i.e. SMILES sequences). A new dataset, which is with larger scale than the traditional one, is constructed for the study. We also propose a light-weight and *ad hoc* framework for ATC classification. The framework is with better explainability than previous methods because it extracts and embeds tokens that are both statistical and physicochemically meaningful, and generates compound representations by capturing the multi-resolution structural characteristics. Its efficacy has been validated in the experiments. This indicates that the ATC codes of drug/compound can be predicted prior to the costly biochemical trails/experiments to save the effort of drug development or basic research.

Key Points

- Construct a new benchmark ATC-SMILES for ATC classification which is with larger scale than transitional benchmarks and eliminates the reliance on STITCH database.
- Propose a new tokenization process which extracts and embeds statistically and physicochemically meaningful tokens.
- Propose a molecular structure-only deep learning method which is with better explainability.
- The proposed method outperforms the state-of-the-art methods.

Code and data availability

The dataset, source code, and web server are open to public at https://github.com/lookwei/ATC_CNN for easier production of this study.

Funding

This work was supported by the National Natural Science Foundation of China under Grant 61872256 and in part by the Internal Research Fund from the Hong Kong Polytechnic University (No. P0036200).

References

1. Dunkel M, Günther S, Ahmed J, *et al.* Superpred: drug classification and target prediction. *Nucleic Acids Res* 2008; **36**(suppl_2):W55–9.
2. Wang Y-C, Chen S-L, Deng N-Y, *et al.* Network predicting drug's anatomical therapeutic chemical code. *Bioinformatics* 2013;**29**(10):1317–24.
3. Nickel J, Gohlke B-O, Erehman J, *et al.* Superpred: update on drug classification and target prediction. *Nucleic Acids Res* 2014;**42**(W1):W26–31.
4. Chen L, Zeng W-M, Cai Y-D, *et al.* Predicting anatomical therapeutic chemical (atc) classification of drugs by integrating chemical-chemical interactions and similarities. *PloS one* 2012;**7**(4):e35254.
5. Cheng X, Zhao S-G, Xiao X, *et al.* iatc-misf: a multi-label classifier for predicting the classes of anatomical therapeutic chemicals. *Bioinformatics* 2017;**33**(3):341–6.
6. Cheng X, Zhao S-G, Xiao X, *et al.* iatc-mhyb: a hybrid multi-label classifier for predicting the classification of anatomical therapeutic chemicals. *Oncotarget* 2017;**8**(35):58494.

7. Nanni L, Brahnam S. Multi-label classifier based on histogram of gradients for predicting the anatomical therapeutic chemical class/classes of a given compound. *Bioinformatics* 2017;**33**(18): 2837–41.
8. Chen L, Liu T, Zhao X. Inferring anatomical therapeutic chemical (atc) class of drugs using shortest path and random walk with restart algorithms. *Biochimica et Biophysica Acta (BBA)-Molecular Basis of Disease* 2018;**1864**(6):2228–40.
9. Lumini A, Nanni L. Convolutional neural networks for atc classification. *Curr Pharm Des* 2018;**24**(34):4007–12.
10. Wang X, Wang Y, Xu Z, *et al.* ATC-NLSP: prediction of the classes of anatomical therapeutic chemicals using a network-based label space partition method. *Front Pharmacol* 2019;**10**:971.
11. Zhou J-P, Chen L, Guo Z-H. iatc-nrakel: an efficient multi-label classifier for recognizing anatomical therapeutic chemical classes of drugs. *Bioinformatics* 2020;**36**(5):1391–6.
12. Zhou J-P, Chen L, Wang T, *et al.* iatc-frakel: a simple multi-label web server for recognizing anatomical therapeutic chemical classes of drugs with their fingerprints only. *Bioinformatics* 2020;**36**(11):3568–9.
13. Zhe L, Chou K-C. iatc_deep-misf: a multi-label classifier for predicting the classes of anatomical therapeutic chemicals by deep learning. *Advances in Bioscience and Biotechnology* 2020;**11**(5): 153–9.
14. Nanni L, Brahnam S, Lumini A. Ensemble of deep learning approaches for atc classification. In: Satapathy SC, Bhateja V, Mohanty JR, Udgata SK, (eds). *Smart Intelligent Computing and Applications*. Springer, Singapore, 2020, 117–25.
15. Zhao H, Li Y, Wang J. A convolutional neural network and graph convolutional network-based method for predicting the classification of anatomical therapeutic chemicals. *Bioinformatics* 2021;**37**(18):2841–7.
16. Wang X, Liu M, Zhang Y, *et al.* Deep fusion learning facilitates anatomical therapeutic chemical recognition in drug repurposing and discovery. *Brief Bioinform* 2021;**22**(6):bbab289.
17. Nanni L, Lumini A, Manfe A, *et al.* Gated recurrent units and temporal convolutional network for multilabel classification. arXiv preprint arXiv:2110.04414. 2021.
18. Nanni L, Lumini A, Brahnam S. Neural networks for anatomical therapeutic chemical (atc) classification. *Applied Computing and Informatics* 2022. Vol. ahead-of-print, No. ahead-of-print. https://doi.org/10.1108/ACI-11-2021-0301.
19. Chou K-C. Some remarks on predicting multi-label attributes in molecular biosystems. *Mol Biosyst* 2013;**9**(6):1092–100.
20. Chen L, Jing L, Zhang N, *et al.* A hybrid method for prediction and repositioning of drug anatomical therapeutic chemical classes. *Mol Biosyst* 2014;**10**(4):868–77.
21. Zixin W, Chen L. Similarity-based method with multiple-feature sampling for predicting drug side effects. In: Karaman R, (ed). *Computational and mathematical methods in medicine*. London, Hindawi, 2022;**2022**.
22. Coley CW, Green WH, Jensen KF. Rdchiral: An rdkit wrapper for handling stereochemistry in retrosynthetic template extraction and application. *J Chem Inf Model* 2019;**59**(6):2529–37.
23. Szklarczyk D, Santos A, Von Mering C. Lars Juhl Jensen, Peer Bork, and Michael Kuhn. Stitch 5: augmenting protein–chemical interaction networks with tissue and affinity data. *Nucleic Acids Res* 2016;**44**(D1):D380–4.
24. Morgan HL. The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. *J Chem Doc* 1965;**5**(2):107–13.
25. Durant JL, Leland BA, Henry DR, *et al.* Reoptimization of mdl keys for use in drug discovery. *J Chem Inf Comput Sci* 2002;**42**(6): 1273–80.
26. Weininger D. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *J Chem Inf Comput Sci* 1988;**28**(1):31–6.
27. Chou K-C. Some remarks on protein attribute prediction and pseudo amino acid composition. *J Theor Biol* 2011;**273**(1):236–47.
28. Kanehisa M, Furumichi M, Sato Y, *et al.* Kegg: integrating viruses and cellular organisms. *Nucleic Acids Res* 2021;**49**(D1): D545–51.
29. Cock PJA, Antao T, Chang JT, *et al.* Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics* 2009;**25**(11):1422–3.
30. Kim S, Chen J, Cheng T, *et al.* (eds). Pubchem in 2021: new data content and improved web interfaces. *Nucleic Acids Res* 2021;**49**(D1):D1388–95.
31. Goh GB, Hodas NO, Siegel C, *et al.* Smiles2vec: An interpretable general-purpose deep neural network for predicting chemical properties. arXiv preprint arXiv:1712.02034. 2017.
32. Zhang Y-F, Wang X, Kaushik AC, *et al.* Spvec: a word2vec-inspired feature representation method for drug-target interaction prediction. *Front Chem* 2020;**7**:895.
33. Salton G, Fox EA, Harry W. Extended boolean information retrieval. *Communications of the ACM* 1983;**26**(11):1022–36.
34. Salton G, Buckley C. Term-weighting approaches in automatic text retrieval. *Inf Process Manag* 1988;**24**(5):513–23.
35. Ramos J. Using tf-idf to determine word relevance in document queries. *Department of Computer Science*, Rutgers University, 2000.
36. Mikolov T, Sutskever I, Chen K, *et al.* Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* 2013;**26**:3111–9.
37. Huang X, Alleva F, Hon H-W, *et al.* The sphinx-ii speech recognition system: an overview. *Computer Speech & Language* 1993;**7**(2):137–48.
38. Kim Y. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, 2014, 1746–51.
39. Zheng S, Yan X, Yang Y, *et al.* Identifying structure–property relationships through smiles syntax analysis with self-attention mechanism. *J Chem Inf Model* 2019;**59**(2):914–23.
40. Arús-Pous J, Johansson SV, Prykhodko O, *et al.* Randomized smiles strings improve the quality of molecular generative models. *J Chem* 2019;**11**(1):1–13.
41. Xue D, Gong Y, Yang Z, *et al.* Advances and challenges in deep generative models for de novo molecule generation. *Wiley Interdisciplinary Reviews: Computational Molecular Science* 2019;**9**(3):e1395.
42. Wu C-K, Zhang X-C, Yang Z-J, *et al.* Learning to smiles: Ban-based strategies to improve latent representation learning from molecules. *Brief Bioinform* 2021;**22**(6):bbab327.
43. Honda S, Shi S, Ueda HR. Smiles transformer: Pre-trained molecular fingerprint for low data drug discoveryarXiv preprint arXiv:1911.04738. 2019.
44. Schwaller P, Laino T, Gaudin T, *et al.* Molecular transformer: a model for uncertainty-calibrated chemical reaction prediction. *ACS central science* 2019;**5**(9):1572–83.
45. Wang S, Guo Y, Wang Y, *et al.* Smiles-bert: large scale unsupervised pre-training for molecular property prediction. In: *Proceedings of the 10th ACM international conference on bioinformatics,*

computational biology and health informatics. ACM, New York, 2019, 429–36.

46. Yang Q, Sresht V, Bolgar P, *et al.* Molecular transformer unifies reaction prediction and retrosynthesis across pharma chemical space. *Chem Commun* 2019;**55**(81):12152–5.
47. Chou K-C, Zhang C-T. Prediction of protein structural classes. *Crit Rev Biochem Mol Biol* 1995;**30**(4):275–349.
48. Chou K-C. Some remarks on protein attribute prediction and pseudo amino acid composition. *J Theor Biol* 2011; **273**(1): 236–47.