Short Paper

Predicting Side Effect of Drug Molecules Using Recurrent Neural Networks

Collin Beaudoin, Koustubh Phalak, and Swaroop Ghosh, *Senior Member, IEEE*

***Abstract*—Identification of molecular properties, like side effects, is one of the most important and time-consuming steps in the process of molecule synthesis. Failure to identify side effects before submission to regulatory groups can cost millions of dollars and months of additional research to the companies. Failure to identify side effects during the regulatory review can also cost lives. The complexity and expense of this task have made it a candidate for a machine learning-based solution. Prior approaches rely on complex model designs and excessive parameter counts for side effect predictions. Reliance on complex models only shifts the difficulty away from chemists rather than alleviating the issue. Implementing large models is also expensive without prior access to high-performance computers. We propose a heuristic approach that allows for the utilization of simple neural networks, specifically the GRU recurrent neural network, with a 98+% reduction of required parameters compared to available large language models while obtaining near identical results as top-performing models.**

***Index Terms*—Molecular property prediction, drug evaluation, machine learning.**

I. Introduction

Molecular property prediction is one of the most fundamental tasks within the field of drug discovery [1], [2]. Applying in silico methods to molecular property prediction offers the potential of releasing safer drugs to the market while reducing test time and cost. Detecting molecular properties before development enables researchers to develop more effective new materials faster and with higher certainty. Detecting known causes of side effects in drugs before release can prevent unnecessary injury and save thousands of lives. Historically, these in silico approaches relied on complex feature engineering methods to generate their molecule representations for processing [3], [4]. The bias of the descriptors limits these approaches, which means the generated features may not be reusable for different tasks as some valuable identifiers may not be present. The feature vectors also depend on current molecular comprehension; upon discovery, the feature vectors could become redundant. Graph Neural Networks (GNN) remove the dependence on complex and temporal descriptors. GNNs became favorable due to the common practice of drawing molecules using graph representations, which offer a generic form for the input. The generic input format allows machine learning models to build their interpretation of information rather than rely on human capabilities. Through these advances GNNs perform well on multiple chem-informatic tasks, especially molecular property prediction [5], [6]. Despite these improvements GNNs still have limitations. Specifically, GNNs have difficulty understanding shared dependence and have scalability issues. The size of the graphical input increases exponentially with each additional molecule that is represented. With this growth, the cost of communication between graphical nodes also exponentially increases. Compared to other neural network types, GNNs can perform worse at molecular property prediction, despite their built-in generic representation [7]. With the recent success of large language models, newer attempts aim to build transformer-based approaches with promising signs of success [8]. While new large language models offer comparable performance to GNNs, they require up to 120 billion parameters.

Due to the rapid explosion of parameters caused by GNNs, feed-forward neural networks, and transformers, we propose a heuristic approach using a recurrent neural network, specifically the gated recurrent unit (GRU). Our approach can obtain close to state-of-the-art results with 99+% fewer parameters than large graph-based models or large language-based models, such as Galactica [8]. In the following sections, we review the MoleculeNet benchmark [9] and compare the SMILES and SELFIES formats and the basic concepts of a recurrent neural network, and also discuss a few of the related works that are evaluated using the MoleculeNet benchmark (Section II). We then discuss the data pre-processing and model implementation details (Section III), followed by model performances and a comparison to other state-of-the-art options (Section IV). Finally, we conclude the paper by giving a summary (Section V).

II. Background & Related Works

A. MoleculeNet Benchmark

MoleculeNet is a benchmark set used to evaluate machine learning techniques [9]. It curate's quantum mechanics, physical chemistry, biophysics, and physiology datasets. For each dataset, it establishes the preferred metric for evaluation to enable consistent comparison across models. We describe each dataset selected to evaluate our model.

1) Side Effect Resource (SIDER): The principal molecular property of human consumption is the side effects associated with the molecule. The Side Effect Resource (SIDER) dataset attempts to create a single source of combined public records for known side effects [10]. The dataset consists of 28 columns; the first column is the SMILES representation of a given molecule, and the 27 subsequent columns are affected system organ classes where side effects are classified by MedDRA .[1] The side effects of each molecule are marked with a one if it is known to have a side effect or a zero otherwise.

2) Bace: BACE is a collection of experimentally reported values from various journals for the binding results for inhibitors of human β-secretase 1 [11].

[1]https://www.meddra.org/

Manuscript received 16 May 2023; revised 15 October 2023 and 10 February 2024; accepted 16 March 2024. This work was supported in part by NSF (CNS-2129675, CCF-2210963, DGE-1821766 and DGE-2113839) and gifts from Intel. *(Corresponding author: Collin Beaudoin.)*

The authors are with the Penn State University, Department of Computer Science, Engineering, University Park, PA 16802-1503 USA (e-mail: cpb5867@psu.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TETCI.2024.3389778, provided by the authors.

Recommended for acceptance by A. Lam.

Digital Object Identifier 10.1109/TETCI.2024.3389778

3) Blood-Brain Barrier Penetration (BBBP): Here molecules are classified by their ability to permeate through the blood-brain barrier. A drug's ability to permeate through the blood barrier is an important feature for drugs specifically targeting the central nervous system [12].

4) Clintox: MoleculeNet introduces ClinTox to evaluate drugs previously approved by the FDA and drugs that have failed clinical trials due to toxicity. [9]

5) Hiv: The HIV dataset is originally from the Drug Therapeutics Program (DTP)[2] consisting of molecules tested to inhibit HIV replication. There are roughly 40 k samples within the dataset, where MoleculeNet uses two labels, confirmed inactive and confirmed active.

6) Muv: The Maximum Unbiased Validation (MUV) dataset contains 17 labeling tasks and 90 k molecules. The dataset originates from PubChem [13].

B. Roc-Auc

The receiver operating characteristic curve (ROC curve) measures the true positive rate against the false positive rate at multiple threshold settings for a binary classifier. This measures the ability of a model to distinguish correctly between two classes. ROC-AUC is commonly preferred when evaluating models trained on imbalanced datasets, making it an ideal statistic to evaluate the MoleculeNet datasets.

C. Simplified Molecular-Input Line Entry System (SMILES)

Simplified molecular-input line-entry system (SMILES) uses characters to build a molecular representation [14]. Letters represent various elements within a molecule, where the first letter of an element can be uppercase, denoting that the element is non-aromatic, or lowercase, denoting that the element is aromatic. Assuming an element requires a second letter, it will be lowercase. Another possible representation of aromaticity is the colon, which is the aromatic bond symbol. Other potential bond symbols are a period (.), a hyphen (-), a forward slash (/), a backslash (\), an equal sign (=), an octothorpe (#), and a dollar sign ($). Periods represent a no bond, hyphens represent a single bond, and the forward slash and backslash represent single bonds adjacent to a double bond. However, the forward slash and backslash are only necessary when rendering stereochemical molecules. The equal sign represents a double bond, the octothorpe represents the triple bond, and the dollar sign represents a quadruple bond. In cases where stereochemical molecules are used, the asperand (@) can be used in a double instance to represent clockwise or in a single occurrence to represent counterclockwise. Numbers are used within a molecule to characterize the opening and closing of a ring structure, or if an element is within brackets, the number can represent the number of atoms associated with an element. Numbers appearing within brackets before an element represent an isotope. A parenthesis (()) denotes branches from the base chain.

D. Self-Referencing Embedded Strings (SELFIES)

Self-referencing embedded Strings (SELFIES) improve the initial idea of SMILES for usage in machine learning processes by creating a robust molecular string representation [15]. SMILES offered a simple and interpretable characterization of molecules that was able to encode the elements of molecules and their spatial features. The spatial features rely on an overly complex grammar where rings and branches are not locally represented features. This complexity causes issues, especially in generative models, where machines frequently produce syntactically invalid or physically invalid strings. To remove this non-locality, SELFIES uses a single ring or branch symbol, and the length of this spatial feature is directly supplied; ensuring that any SELFIES string has a valid physical representation.

[2]https://wiki.nci.nih.gov/display/NCIDTPdata/AIDS+Antiviral+Screen+Data

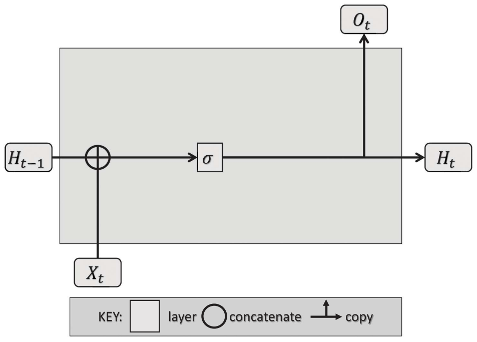

Fig. 1. Vanilla RNN architecture used for training; (H_{t-1}, H_t) represent the hidden state, (O_t) represents the output state, and (X_t) represents the input information. The σ represents the activation function that operates on the combined input and hidden state.

E. Recurrent Neural Networks (RNN)

Elman networks, more commonly known as vanilla recurrent neural networks (RNN), attempt to introduce the concept of a time-dependent dynamic memory [16]. The idea is to make predictions about inputs based on contextual information. Context-based predictions can be done for four input-output schemes: one-to-one, one-to-many, many-to-one, and many-to-many. One-to-one models are a variation of a classic neural network, one-to-many models are best for image caption generation, many-to-one models are best for sentiment analysis, and many-to-many models are best for translation or video frame captioning. Fig. 1 is an example of the basic structure of a vanilla RNN.

In Fig. 1, the X_t represents some input, H_{t-1}, H_t represents some hidden state (which is representative of memory), O_t represents some output, and σ represents some activation function. The current input information combines with the previous hidden state, and the resulting combined state is then fed to an activation function to insert some non-linearity. This non-linearity produces the next hidden state, which can be manipulated to create a desired output. The fundamental element is the hidden state. The hidden state theoretically allows for consideration of any historical input and its effects on the current input. For a mathematical description of an RNN, we refer to (1) and (2).

$$H_t = \sigma(W_{HH}H_{t-1} + W_{XH}X_t) \tag{1}$$

$$O_t = W_{HO}H_t \tag{2}$$

Unfortunately, Vanilla RNNs suffer from memory saturation issues, so they are not always reliable. There have been many methods proposed to overcome this issue, but one of the most popular is the Gated Recurrent Unit (GRU) [17]. The basic structure of a GRU is in Fig. 2. We can mathematically describe each of the components using (3), (4), (5), and (6). Equation (5) represents the candidate hidden state function, representing the potential updated state. Equation (6) performs the actual update to the hidden state based on the previous hidden state and the candidate hidden state. Both (3) and (4) allow the network to tune the importance of the contribution of the previous hidden state to the new hidden state. Because of the r_t and z_t parameters, the GRU can better control its memory state offering a practical improved performance

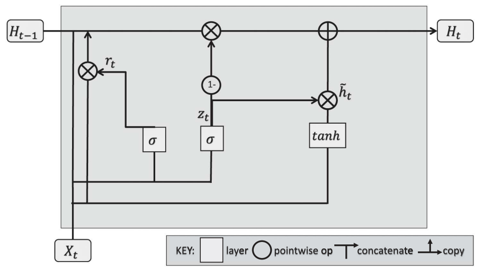

Fig. 2. GRU architecture used for training; (h_{t-1}, h_t) represent the hidden state, $(\tilde{h}_t)$ represents the candidate hidden state state, and (r_t) and (Z_t) represents the parameters to tune the importance of the previous hidden state versus the updated information. The σ represents the activation function that operates on the combined input and hidden state.

over RNNs.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{3}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{4}$$

$$\tilde{h}_t = tanh(W \cdot [r_t * h_{t-1}, x_t]) \tag{5}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \tag{6}$$

F. Related Works

a) GROVER: The graph representation from the self-supervised message passing transformer (GROVER) model takes two forms, GROVER$_{base}$ and GROVER$_{large}$ [18]. We only consider GROVER$_{large}$ as it achieves the highest performance of the two. GROVER bases its design on popular large language models such as, BERT and GPT, where a large corpus of information pre-trains a model and fine-tuning is applied for the completion of downstream tasks [19], [20]. However, they stray from prior works that attempt training using the SMILES string format [21] and instead use graphs, which they state are more expressive. Previous graph pre-training approaches use the available supervised labels to train their model [5], but GROVER prefers a self-supervised approach to achieve higher performance, so they suggest using contextual property prediction and graph-level motif prediction. Contextual property prediction takes a given element (node) within a molecular graph and predicts the connected elements and the type of bond used for the connection. Graph-level motif prediction takes a given molecule and attempts to predict the recurrent sub-graphs, known as motifs, that may appear within the molecule. To build the model, they designed a new architecture known as the GTransformer, which creates an attention-based understanding of molecular graphs. The pre-training process uses 10 million unlabeled molecules for training and 1 million molecules for validation. The molecules are taken from ZINC15 [22] and Chembl [23]. GROVER is fine-tuned on 11 benchmark datasets from MoleculeNet [9] for final evaluation. While this new architecture and self-supervised training approach offer appealing results the model uses 100 M parameters, uses 250 Nvidia V100 GPUs, and takes four days for pre-training.

b) ChemRL-GEM: Geometry Enhanced Molecular representation learning method (GEM) for Chemical Representation Learning (ChemRL) (ChemRL-GEM) draws inspiration from previous works using a graph-based approach, especially GROVER [5], [18]. ChemRL-GEM uses a large corpus of information to pre-train a model and, like GROVER, finds the ambiguity of SMILES and lack of structural information hard to build a successful model using a string-based approach [24]. ChemRL-GEM blames the low performance of prior graph approaches on neglecting the available molecular 3D information and improper pre-training tasks. ChemRL-GEM pre-training splits tasks into geometry-level and graph-level tasks. The geometry level tasks are again split into two types where bond length prediction, and bond angle prediction are local spatial structure predictions, and atomic distance matrices prediction is a global spatial prediction. The graph-level predictions are the Molecular ACCess System (MACCS) key prediction and the extended-connectivity fingerprint (ECFP) prediction. To build the model they designed an architecture called GeoGNN which trains on the atom-bond graph and the bond-angle graph of molecules to build a 3D structure-based understanding of the molecular graphs. ChemRL-GEM achieves SOTA performance and is an early attempt at a large 3D graph model pre-trained network. The pre-training approach uses 18 million training samples from ZINC15 and 2 million for evaluation [22]. They state that pre-training a small subset of the data would take several hours using 1 Nvidia V100 GPU, and fine-tuning would require 1-2 days on the same GPU. As a rough estimate of the actual training process there was a follow-up work called LiteGEM which removed the 3D input of the model but still uses 74 million parameters and takes roughly ten days of training using 1 Nvidia V100 GPU [25].

c) Galactica: Galactica is inspired directly by previous large language models and their utilization of large datasets to pre-train models for downstream tasks [19], [20]. Differentiating from BERT, they use a decoder-only setup from Vaswani et al. [26]. Unlike GROVER or ChemRL, Galactica focuses on general scientific knowledge and wishes to apply it to the entirety of the scientific domain [8]. The Galactica model takes several forms, but the 120 billion parameter model offers the best performance. Galactica trains over 60 million individual scientific documents and 2 million SMILES strings. Galactica is trained with samples from MoleculeNet, where the molecular properties are converted to text prompts and responses. Galactica acknowledges using SMILES they receive reduced performance gains as their model size increases, but they state this could be overcome with more samples. Galactica offers a competitive performance to graph-based approaches while offering a simplified architecture design. Unfortunately, the model requires 120 billion parameters and trains using 128 Nvidia A100 80 GB nodes. Despite the massive model size, it is not SOTA for a single SMILES metric. Galactica states they need additional samples/fine-tuning to obtain SOTA results.

III. Methods

A. Data Pre-Processing

The available MoleculeNet benchmark [9] uses SMILES for its molecular representation. After reviewing some of the molecule strings, not all are canonical. Including non-canonical SMILES is problematic as SMILES grammar is already complex; the molecules are converted to RDKit's canonical form to reduce complexity. The next issue is caused by RNNs, one of the many advantages of RNN is the allowance of variable length inputs to account for a variable length of history. This is only true theoretically; in practice, RNN memory has limits, which is the focus of many newer works [27]. Despite this limitation, it has been recently shown that RNNs can handle input lengths of around 45–50 before the performance begins to degrade [28], [29]. Using this knowledge, we set a maximum SMILES length of 46 for the molecules. The limitation keeps a minor majority of the molecules while allowing us to ensure the RNN is performing well. After limiting the SMILES molecular length, the SMILES are converted to SELFIES. The intention of converting SMILES to SELFIES is to reduce the

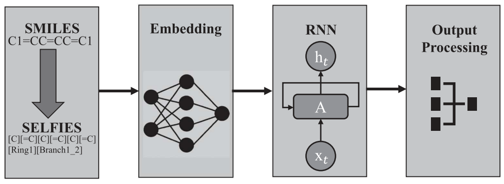

Fig. 3. Overview of the RNN process.

grammar complexity and simplify the learning process of the RNN. SELFIES converts each element and structural component, such as rings or branches, into their label. These labels are then encoded into a numerical value based on their dictionary index.

B. RNN Implementation

Fig. 3 offers a visualization of the methodology used to train the RNN. The molecules are first loaded in from a dataset from the MoleculeNet benchmark [9] and converted to SELFIES representation using the method described in Section III-A. The converted SELFIES are then processed through an embedding layer with a dimensional space matching the size of the label dictionary. The dictionary consists of all the unique SELFIES components within the dataset and the embedding dimension equals the dictionary size to maintain as much information as possible. The input, hidden, and output dimensions of the RNN are also equal to the size of the dictionary. Maintaining the dimensional space and not reducing it before output generation gives the RNN a chance of learning the molecular context. Figs. 1 and 2 are visualizations of the RNN architectures used to process the SELFIES. RNNs historically use the Tanh activation function, but we use the LeakyReLU as it reduces saturation possibilities and typically results in higher performance [30], [31]. In addition to this, we also include a dropout layer on the output of the RNN which helps prevent overfitting and reduce the error rate of RNNs [32]. After processing the SELFIES through the RNN, the final state should have all important prior information encoded into it. This vector then passes through an additional LeakyReLU and dropout layer before being fed to a fully connected layer. The fully connected layer reduces the vector from the dictionary-sized dimension down to the number of classes present in the molecular property. Subsequently, a soft-max operation finds the most likely class.

IV. Results and Comparative Analysis

A. Results

Before training on the selected MoleculeNet datasets referenced in Section II-A, we perform an additional reduction to the dataset by setting the lower bound of 31 molecules to the SMILES string allowing for the search space to remain sufficiently complex while reducing the overall run time. The lower bound reduces the datasets before stratified splitting the data using 80% for training and 20% for testing [33]. The stratified splitting intends to maintain the known sample rate of a given side effect to model real-world testing. However, during training, we want to remove the sampling bias to ensure our model accurately learns the causes of a side effect. The minority samples within the training set are duplicated to have an even sample count between the side effect present and the side effect not present to reduce the sampling bias. After replicating training samples, the SMILES conversion to SELFIES occurs. Typical natural language processing (NLP) methods use a word, sub-word, or character tokenization to convert strings into

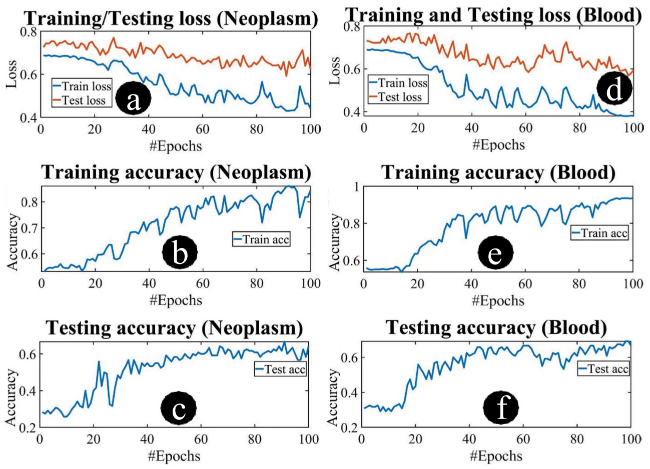

Fig. 4. Results of three tasks: (a) loss curves, (b) training accuracy, (c) testing accuracy for neoplasms benign, malignant and unspecified (incl cysts and polyps) disorders, (d), (e), (f) for blood and lymphatic system disorders.

numerical values, but we opt for a slightly different method, which we explain by referring to (7). It shows the SELFIES representation of benzene where each molecule and structural element are between brackets. Using this representation, we decide to tokenize based on each set of brackets that exist within the SELFIES converted dataset. This results in a total of 47 unique values. After tokenizing the SELFIES, the embedding dimension, input dimension of the RNN, and the hidden dimension of the RNN are set to a size of 47 to match the dimensional space of the tokens. To give the RNN model the best opportunity to make accurate classifications, we use a single model to perform a single side effect classification prediction. For SIDER, instead of predicting all 27 potential side effect classifications, we opt to predict 20 side effect classifications due to extreme imbalances present in the side effect data. The vanilla RNN architecture results in a model with 11.5 K parameters and the GRU architecture results in a model with 18.8 k parameters. Both train in under 2 minutes on an Nvidia GeForce RTX 3090. To compare our performance with other works that use MoleculeNet we evaluate using the suggested metric, the receiver operating characteristic curve (ROC) [1], [34]. While ROC is helpful for comparison, it is commonly misunderstood [35], [36] so we include a small sample of 2 training/testing accuracy and loss curves in Fig. 4 as a simple spot check of model performance. Examining Fig. 4, we note that training and testing loss is decreasing across all three side effect properties. There are spikes within each of the loss curves, but this is known to have occurred since the inception of RNNs [16]. The training loss for all three side effects saturate faster than the testing. There can be some gap in performance in loss based on the difficulty of new samples, but the gap here is likely accentuated as an unfortunate side effect of the minority sample duplication process. The duplicate samples within the training set help the model learn what molecular components help detect a side effect, but during training, the repeated samples become easier to predict for the model. In the case of accuracy, both training and testing show an upward trending curve where improvement starts to attenuate between the 20th–40th epoch. This attenuation roughly matches the attenuation that occurs with the loss curves. Comparing training and testing accuracy there appears to be a roughly 20+% gap in performance at nearly every epoch, which we again attribute to the duplicate samples within the training set.

$$[C][=C][C][=C][C][=C][Ring1][=Branch1] \tag{7}$$

TABLE I
TABLE OF ROC PERFORMANCE PER MOLECULAR PROPERTY PREDICTION ACROSS DATASETS

Model	**RNN**	**bi-RNN**	**GRU**	**CNN**	**MLP**	ChemRL-GEM [24]	GROVER$_{large}$ [18]	Galactica [8]	RF [9]	GCN [9]	DMPNN [1]	Pre-trained GIN [5]
SIDER	*.557 ± .05*	*.56 ± .071*	**.818 ± .084**	**.814 ± .059**	**.731 ± .063**	.672 ± .004	.658 ± .023	.632	.684 ± .009	.638 ± .012	.676 ± .014	.627 ± .08
BBBP	*.643 ± .133*	*.5 ± 0*	*.937 ± .025*	*.923 ± .068*	*.681 ± .039*	.724 ± .004	.940 ± .019	.661	.714 ± .000	.690 ± .009	.737 ± .001	.687 ± .013
Clintox	*.637 ± .101*	*.6 ± .17*	**.98 ± .012**	**.976 ± .021**	*.860 ± .029*	.901 ± .013	.944 ± .021	.826	.713 ± .056	.807 ± .047	.864 ± .017	.726 ± .015
BACE	*.623 ± .015*	*.733 ± .196*	**.94 ± .006**	**.923 ± .006**	**.943 ± .017**	.856 ± .011	.894 ± .028	.617	.867 ± .008	.783 ± .014	.852 ± .053	.845 ± .007
HIV	*.583 ± .032*	*.61 ± .095*	**.653 ± .031**	**.713 ± .081**	**.683 ± .041**	-	-	.632	-	.763 ± .016	.776 ± .007	.799 ± .007
MUV	*.621 ± .089*	*.627 ± .107*	*.923 ± .034*	*.931 ± .037*	*.867 ± .058*	-	-	-	-	.046 ± .031	.041 ± .007	.813 ± .021
Pre-train data	-	-	-	-	-	18M Mols	10M Mols	60M Docs	-	-	-	2.4M Mols
Rep/Est Train Time	$\sim$ 2m	$\sim$ 2m	$\sim$ 2m	$\sim$ 2m	$\sim$ 2m	$\sim$ 10D	4D	30D	-	-	-	-
GPU Req.	RTX3090	RTX3090	RTX3090	RTX3090	RTX3090	$\sim$ 1 V100	250 V100	512 A100	-	-	-	-

B. Comparisons

To understand the performance of the proposed approach, we compare it across multiple datasets to two top-performing GNN models, ChemRL-GEM [24] and GROVER$_{large}$ [18], and a top-performing NLP model, Galactica [8]. In addition to the top-performing large models, we include the random forest and GCN model from MoleculeNet [9], the DMPNN model [1], and the pre-trained GIN model [5]. For each heuristic model we evaluate we train using 20 different random seeds and evaluate the model by taking the top 3 performing ROC scores per metric. We include standard deviation as a way to account for uncertainty in model performance. Overall results are shown in Table I. Beginning with the SIDER test, the results in Table I show our approach using the GRU achieves SOTA performance with a 17.8% higher performance over the best model not using the proposed method (RF [9]). While there are no direct statistics available for ChemRL-GEM, we use roughly 99.7% fewer parameters than its follow-up work, LiteGEM [24], [25]. It is worth noting that applying our proposed approach to a CNN network offers a 17.3% higher performance over RF [9]. For the BBBP test, the GRU outperforms ChemRL-GEM and Galactica but performs 0.32% worse than GROVER$_{large}$. While it may be possible that GROVER achieves better performance due to their usage of graph representation, it more likely stems from having 100 M parameters, over 5,000x more parameters than our model [18]. For the Clintox test, our performance was again the best of all the models. This test is one of Galactica's best performances, yet we can achieve a 17.05% higher performance with 6 Mx less parameters [8]. Comparing the GRU approach to the best performing approach for Clintox and BACE, GROVER$_{large}$ [18], it achieves a 3.74% better performance for Clintox and a 5.01% better performance for BACE.

Further comparing the models we include the pre-train data requirement, the train time of the model and the GPU requirement to achieve the listed runtime in Table I.

V. Discussion & Conclusion

While large data models may offer coverage of larger molecular lengths, we have shown that small models (specifically GRUs) are still viable candidates for molecular property prediction. Smaller models are cheaper, more practical, and more accessible solutions as they don't require multiple GPUs and several days of training prior to obtaining a result. There are limitations with RNN based models, but when these limitations are carefully considered and more descriptive languages, such as SELFIES [15], are used RNNs offer SOTA/near SOTA results.

A. Clinical Insights

Property prediction models allow chemists to perform molecular evaluations prior to physical experimentation. Effective property evaluation can prevent months of wet lab research being spent on molecules that will not be feasible. Reducing failures realized during synthesis has the potential to greatly reduce the drug to market run time, enabling clinical researchers to rapidly treat patients for their medical conditions.

B. Constraints

Despite the RNN's learning capabilities and ability to process variable length input with no additional parameters required, using such an architecture does have drawbacks. RNN models can scale when dealing with large datasets via batching or even model parallelization, but they do not scale well when considering larger input sequences. Theoretically, RNN models can process large sequences of information with no problem, but in practice, RNNs can suffer from vanishing or exploding gradients causing them to "forget" important information. Even if we could implement the perfect memory model, the RNN still suffers from long run times where each addition to the sequence increases the run time due to the sequential nature of recurrence. One possible method to mitigate the long run time would be chunking, where the sequences are partitioned into smaller processable pieces. Unfortunately, this is unreliable, as sometimes vital state information may be separated from chunks causing inaccurate results.

C. Ethical Statement

While machine learning models can help identify potential molecular properties, they are not without flaws. Even if machine learning models can accurately identify all molecular properties of the datasets, they are trained with are fully dependent on previous human discoveries. The datasets are subject to flawed understandings of chemistry and even political choices. For example, the NIH only classifies drugs as toxic to the liver after successfully ruling out other potential causes.[3] Therefore, machine learning models should only be used for preliminary evaluation of molecules and not as the only form of molecular evaluation.

References

[1] K. Yang et al., "Analyzing learned molecular representations for property prediction," *J. Chem. Inf. Model.*, vol. 59, no. 8, pp. 3370–3388, 2019.

[2] O. Wieder et al., "A compact review of molecular property prediction with graph neural networks," *Drug Discov. Today: Technol.*, vol. 37, pp. 1–12, 2020.

[3] T. Lengauer, C. Lemmen, M. Rarey, and M. Zimmermann, "Novel technologies for virtual screening," *Drug Discov. Today*, vol. 9, no. 1, pp. 27–34, 2004.

[4] C. Merkwirth and T. Lengauer, "Automatic generation of complementary descriptors with molecular graph networks," *J. Chem. Inf. Model.*, vol. 45, no. 5, pp. 1159–1168, 2005.

[5] W. Hu et al., "Strategies for pre-training graph neural networks," in *Proc. Int. Conf. Learn. Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=HJlWWJSFDH

[6] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[7] A. Mayr et al., "Large-scale comparison of machine learning methods for drug target prediction on chembl," *Chem. Sci.*, vol. 9, no. 24, pp. 5441–5451, 2018.

[8] R. Taylor et al., "Galactica: A large language model for science," 2022, *arXiv:2211.09085*.

[9] Z. Wu et al., "Moleculenet: A benchmark for molecular machine learning," *Chem. Sci.*, vol. 9, no. 2, pp. 513–530, 2018.

[3]https://www.ncbi.nlm.nih.gov/books/NBK548049/

[10] M. Kuhn et al., "The sider database of drugs and side effects," *Nucleic Acids Res.*, vol. 44, no. D1, pp. D1075–D1079, 2016.

[11] G. Subramanian, B. Ramsundar, V. Pande, and R. Aldrin Denn, "Computational modeling of β-secretase 1 (BACE-1) inhibitors using ligand based approaches," *J. Chem. Inf. Model.*, vol. 56, no. 10, pp. 1936–1949, 2016.

[12] I. F. Martins, A. L. Teixeira, L. Pinheiro, and A. O. Falcao, "A Bayesian approach to in silico blood-brain barrier penetration modeling," *J. Chem. Inf. Model.*, vol. 52, no. 6, pp. 1686–1697, 2012.

[13] S. G. Rohrer and K. Baumann, "Maximum unbiased validation (MUV) data sets for virtual screening based on pubchem bioactivity data," *J. Chem. Inf. Model.*, vol. 49, no. 2, pp. 169–184, 2009.

[14] D. Weininger, "Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules," *J. Chem. Inf. Comput. Sci.*, vol. 28, no. 1, pp. 31–36, 1988.

[15] M. Krenn, F. Háse, A. K. Nigam, P. Friederich, and A. Aspuru-Guzik, "Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation," *Mach. Learn.: Sci. Technol.*, vol. 1, no. 4, 2020, Art. no. 045024.

[16] J. L. Elman, "Finding structure in time," *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, 1990.

[17] K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. 2014 Conf. Empirical Method Natural Lang. Process. (EMNLP)*, 2014, pp. 1724–1734.

[18] Yu Rong et al., "Self-supervised graph transformer on large-scale molecular data," in *Proc. 34th Conf. Neural Inf. Process. Syst.*, 2020, pp. 12559–12571.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. North Amer. Chapter Assoc. Comput. Linguistics (NAACL-HLT)*, vol. 1, 2019, p. 2.

[20] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[21] S. Wang et al., "Smiles-bert: Large scale unsupervised pre-training for molecular property prediction," in *Proc. 10th ACM Int. Conf. Bioinf., Computat. Biol. Health Inform.*, 2019, pp. 429–436.

[22] T. Sterling and J. J. Irwin, "ZINC 15–ligand discovery for everyone," *J. Chem. Inf. Model.*, vol. 55, no. 11, pp. 2324–2337, 2015.

[23] A. Gaulton et al., "CheMBL: A large-scale bioactivity database for drug discovery," *Nucleic Acids Res.*, vol. 40, no. D1, pp. D1100–D1107, 2012.

[24] X. Fang et al., "ChemRL-GEM: Geometry enhanced molecular representation learning for property prediction," *Nature Mach. Intell.*, vol. 4, pp. 127–134, 2022.

[25] S. Zhang et al., "LiteGEM: Lite geometry enhanced molecular representation learning for quantum property prediction," *arXiv:2106.14494*, 2021.

[26] A. Vaswani et al., "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[28] X. Yao, Q. Cheng, and G.-Q. Zhang, "A novel independent RNN approach to classification of seizures against non-seizures," 2019, *arXiv:1903.09326*.

[29] W. Yin, K. Kann, M. Yu, and H. Schütze, "Comparative study of CNN and RNN for natural language processing," 2017, *arXiv:1702.01923*.

[30] A. L. Maas, Y. H. Awni, and Y. N. Andrew, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. 30th Int. Conf. Mach. Learn.*, vol. 28, 2013. [Online]. Available: https://ai.stanford.edu/~amaas/papers/relu_hybrid_icml2013_final.pdf

[31] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," 2015, *arXiv:1505.00853*.

[32] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Proc. 30th Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 1027–1035.

[33] S. Thompson, *Sampling* (Wiley Series in Probabilty and Statistics). Hoboken, NJ, USA: Wiley, 2012, pp. 141–156. [Online]. Available: https://books.google.com/books?id=-sFtXLIdDiIC

[34] Z. Zhou et al., "Uni-mol: A universal 3D molecular representation learning framework," in *Proc. 11th Int. Conf. Learn. Representations*, 2022.

[35] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (ROC) curve," *Radiol.*, vol. 143, no. 1, pp. 29–36, 1982, doi: 10.1148/radiology.143.1.7063747.

[36] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PLoS One*, vol. 10, no. 3, 2015, Art. no. e0118432.