Leap: molecular synthesisability scoring with intermediates

Antonia Calvi
Exscientia
acalvi@exscientia.co.uk

Théophile Gaudin
Exscientia
tgaudin@exscientia.co.uk

Dominik Miketa
Exscientia
dmiketa@exscientia.co.uk

Dominique Sydow
Exscientia
dsydow@exscientia.co.uk

Liam Wilbraham
Exscientia
lwilbraham@exscientia.co.uk

Abstract

Assessing whether a molecule can be synthesised is a primary task in drug discovery. It enables computational chemists to filter for viable compounds or bias molecular generative models. The notion of *synthesisability* is dynamic as it evolves depending on the availability of key compounds. A common approach in drug discovery involves exploring the chemical space surrounding synthetically-accessible intermediates. This strategy improves the synthesisability of the derived molecules due to the availability of key intermediates. Existing synthesisability scoring methods such as SAScore, SCScore and RAScore, cannot condition on intermediates dynamically. Our approach, Leap, is a GPT-2 model trained on the depth, or longest linear path, of predicted synthesis routes that allows information on the availability of key intermediates to be included at inference time. We show that Leap surpasses all other scoring methods by at least 5% on AUC score when identifying synthesisable molecules, and can successfully adapt predicted scores when presented with a relevant intermediate compound.

1 Introduction

The ability to predict whether a compound can be made is key to accelerating drug discovery. Generative methods that neglect synthesisability often suggest compounds that are challenging or impossible to make [8]. Attempting to synthesise these problematic compounds is costly and time-consuming. Therefore, it is crucial to assess synthesisability as part of the scoring functions applied within generative methods to prioritise synthetically accessible compounds.

Furthermore, the concept of synthesisability should be regarded as dynamic and dependent on the compounds available at a specific time. Synthesis planning algorithms [7, 11, 1, 19] are used to identify possible synthetic pathways for specific molecules by using readily accessible compounds. However, these methods are computationally expensive and slow to apply exhaustively within generative workflows, which can involve the assessment of hundreds of thousands to millions of compounds [8]. Therefore, it makes them unsuitable to assess synthesisability when speed is required to provide feedback to generative models. In practice, this limits the applicability of generative workflows to real world scenarios where an fast, automated and accurate metric of chemical tractability

The NeurIPS 2023 Workshop on New Frontiers of AI for Drug Discovery and Development (AI4D3 2023), New Orleans, LA, USA, 2023.

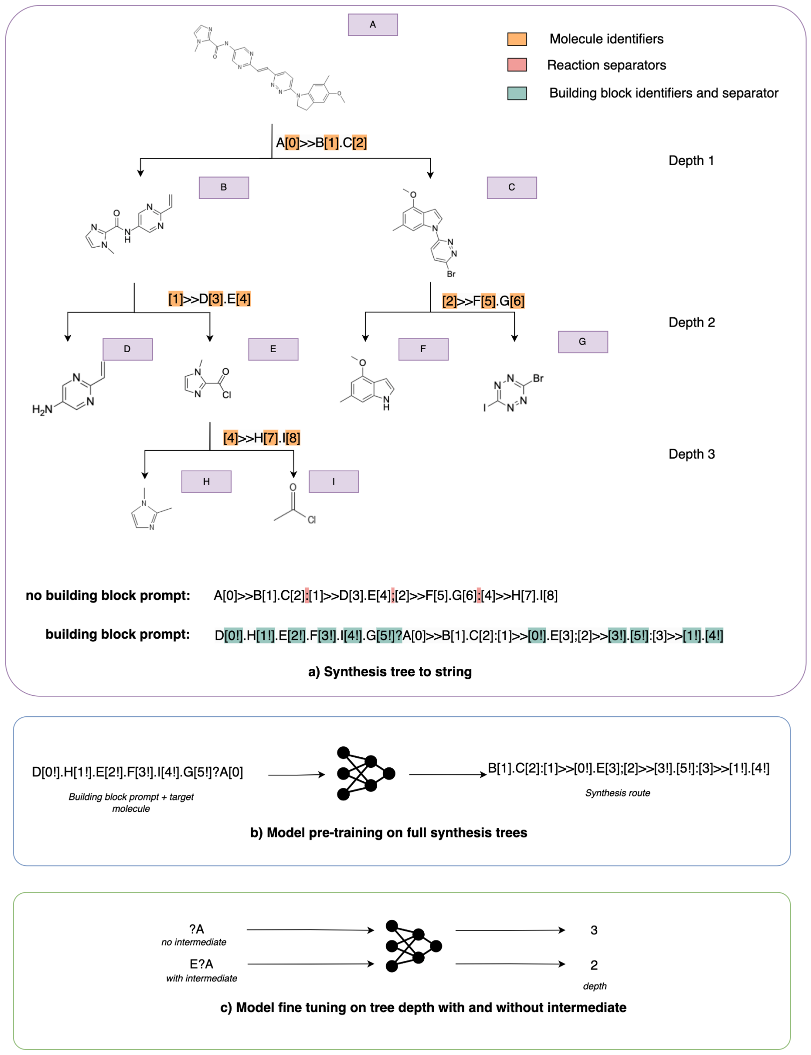

A
Molecule identifiers
Reaction separators
Building block identifiers and separator
A[0]>>B[1].C[2]
Depth 1
B
C
[1]>>D[3].E[4]
[2]>>F[5].G[6]
Depth 2
D
E
F
G
[4]>>H[7].I[8]
Depth 3
H
I
no building block prompt:
A[0]>>B[1].C[2]:[1]>>D[3].E[4];[2]>>F[5].G[6]:[4]>>H[7].I[8]
building block prompt:
D[0!].H[1!].E[2!].F[3!].I[4!].G[5!]?A[0]>>B[1].C[2]:[1]>>[0!].E[3];[2]>>[3!].[5!]:[3]>>[1!].[4!]
a) Synthesis tree to string
D[0!].H[1!].E[2!].F[3!].I[4!].G[5!]?A[0]
Building block prompt + target molecule
B[1].C[2]:[1]>>[0!].E[3];[2]>>[3!].[5!]:[3]>>[1!].[4!]
Synthesis route
b) Model pre-training on full synthesis trees
?A
no intermediate
E?A
with intermediate
3
2
depth
c) Model fine tuning on tree depth with and without intermediate

Figure 1: Schematic diagram of our workflow. a) shows an example of a synthesis tree of depth 3, and its representation as a string; b) shows input and expected output of our model during pre-training, while c) shows the input and expected output of our model fine-tuned to predict tree depth for a molecule with and without intermediate.

is critical to enable chemists to prioritise high quality compounds that minimise development time and cost.

To address this challenge, scoring methods such as SAScore [6], SCScore [4], and RAScore [20] were developed to assess the synthetic accessibility of molecules. SAScore assumes that the complexity of synthesising a molecule is correlated with the scarcity of its fragments in databases of bioactive compounds (e.g. PubChem [13] or ChEMBL [10]) and checks for complex chemistry like stereocenters and ring systems. SCScore is designed to ensure that, on average, the products of chemical reactions exhibit greater synthetic complexity compared to their reactants. RAScore [20] classifies whether a synthetic route can be identified for a particular compound or not by the retrosynthesis tool AiZynthFinder [11]. During inference, none of these methods are aware of possible available intermediates for the query molecules. In drug discovery projects, a common strategy is to identify, through the synthesis of molecules around a specific chemical space, large quantities of useful synthesisable intermediates from which further compounds can be readily made [12]. When failing to adapt to this information, existing scorers can overestimate the synthetic complexity of molecules produced in this way [16], hindering their ability to identify compounds as synthesisable, particularly when these intermediates are complex.

Our main contribution is a novel synthesisability scoring method, Leap, that can adapt to available intermediates to better estimate the practical synthetic complexity of a target molecule. We introduce a specific pre-training objective, where we train a GPT-2 model to predict the entire multi-step retrosynthesis route for a target compound. We use the retrosynthetic planning tool AiZynthFinder to compute the routes of compounds sampled from ChEMBL, and fine-tune the pre-trained model to predict a molecule's synthetic complexity with and without an intermediate. Figure 1 shows a schematic representation of our method.

Two practically useful characteristics distinguish our method from prior art. First, compared to existing scorers, Leap adapts its synthetic complexity score in the presence of useful intermediates for a target molecule. There is currently no existing tool that offers this capability. Second, we demonstrate that even in the absence of intermediates, Leap outperforms existing scorers, providing a strong baseline for future research in this area.

The rest of the paper is organised as follows: in Section 2 we expand on our method and provide details on both the pre-training and fine-tuning steps; in Section 3 we outline our experiments and discuss the results on both publicly available data and molecules coming from real drug discovery projects; finally we conclude and discuss further work in Section 4.

2 Methods

2.1 Problem Setting

Within this work, we also refer to a synthesis route as a synthetic tree. Figure 1a. shows an example of this. The root of the tree is the target molecule to be synthesised. The leaves are the building blocks and consist of available compounds. All the other molecules are intermediates. The depth of the tree (tree depth) is given by the maximum number of steps needed to go from the target molecule to any of the building blocks.

Our method is based on the success of transformers [21] in predicting single-step forward reactions (reactants to product), as well as single-step retrosynthesis (product to reactants) [18, 22] by extending the approach to multi-step routes. We achieve this by encoding synthesis routes as strings, and training a transformer model (GPT-2 [17]) to predict them. To guide the model towards the expected route, we prompt it with the expected building blocks. Finally, we fine-tune this model using a regression head to predict tree depth for target molecules.

2.2 Pre-Training

2.2.1 Data Representation

Our dataset consists of synthesis routes that we represent with a custom string format. This representation encodes both the tree depth and intermediate molecules that act as reactants and products of two reactions as shown in Figure 1b.

Each molecule is encoded as SMILES. Inspired by the reaction smiles [2], we represent these in a `product>>reactants` format, where each reactant is separated by a dot. Reactions occurring at the same depth are separated by a `;` , while we use `:` to separate reactions at different depths of the tree.

Each molecule is given a unique identifier, in the format of `[0]`, `[1]`, `[2]` etc. These are assigned in ascending order as we traverse the tree from the root to the leaves. The target molecule (i.e. the root of the tree) is always assigned the identifier `[0]`. When a molecule is encountered for the first time, it is represented by its SMILES followed by its identifier. If that same molecule is used in a subsequent reaction, only the identifier is used. This shortens the length of the string, freeing GPT-2's context window from repeated SMILES. Figure 1a. shows an example.

To prompt the model with the expected building blocks for a route, we introduce these at the beginning of the string, separating them from the target molecule with a `?`. As before, each building block's SMILES is followed by a unique identifier. We represent these with `[0!]`, `[1!]`, `[2!]` etc. Then, when the building blocks are used along the synthesis route, we refer to them using their identifier as shown in Figure 1.

2.2.2 Model Architecture and Training

Our model is a GPT-2 model with 8 attention heads and 5 hidden layers. We use HuggingFace's implementation [23]. The hidden states have a dimension of 512, while the feed-forward layers have a dimension of 1024. We train the model for 16 epochs, use AdamW optimiser with a learning rate of 0.0003, and a cosine learning rate scheduler with 2000 warmup steps. We train the model on a language modelling task, where the goal is to predict the next token.

2.2.3 Dataset

To pre-train our model, we use a combination of two datasets. As no dataset of multi-step synthetic routes exists, methods have been developed to generate pathways from existing single-step reaction datasets. The first dataset from Liu et al. [14], uses dynamic programming to extract synthesis routes from patent reactions found within the USPTO-all [15] dataset. We use train, test and validation splits from Liu et al. [14] which gives 39717, 5479 and 5493 routes for each.

The dataset from Liu et al. [14] is skewed both towards short routes of an average depth of three, and routes that only extend down a single branch of the synthesis tree (i.e. linear rather than convergent synthesis routes). To increase diversity, and allow for longer and more branched routes, we extend the training set with a second dataset based on Gao et al. [9]. This models the problem of synthetic tree generation as a Markov decision process with a random policy over reactants and reaction templates. Reaction templates are rule-based patterns extracted from existing reactions to define transformations on molecules that represent valid chemical reactions. From this we obtain 116029 synthesis routes, which we add to the training set.

We convert all routes to the string format described in Section 2.2.1. We augment the data by randomising the SMILES as suggested in Arús-Pous et al. [3], and randomly permuting both the reactions located at the same depth of the route, and the building blocks in the prompt. We apply augmentation three times per route.

2.3 Fine-Tuning

The purpose of fine-tuning the model is to have it predict the expected tree depth of a target molecule both when an intermediate is provided and when it is not.

2.3.1 Data Representation

The input to the model consists of the target molecule and, when available, an intermediate. Both molecules are represented with their respective SMILES. The target molecule is prepended by the `?` token that is used to separate it from the potential intermediate. Therefore, the input format to the model will either be `intermediate?target` or `?target` depending on whether an intermediate is included.

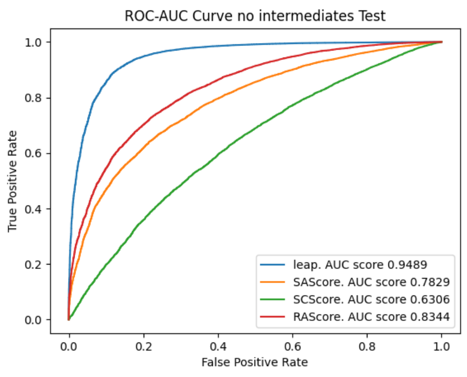
The output of the model is an integer that corresponds to the expected tree depth. A simplified example of this is shown in Figure 1c.

2.3.2 Model Architecture and Training

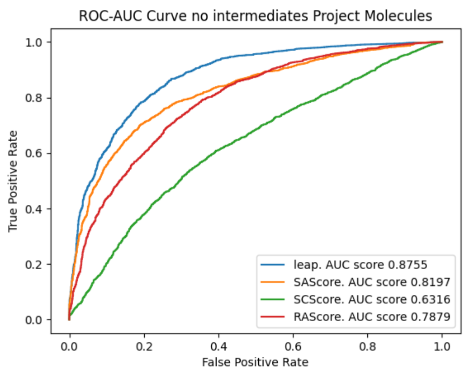
We add a regression head on top of the pre-trained GPT-2 model to predict tree depth. We tune the hyperparameters by hand. Fine-tuning only the regression head, and the the regression head with last two hidden layers, showed worse performance on the validation set than fine-tuning all layers of the model. We use an AdamW optimizer with a learning rate of 0.0001, and a linear scheduler with 4000 warmup steps. We minimise the mean squared error, and train for 16 epochs with early stopping on the validation loss.

2.3.3 Dataset

The fine-tuning dataset is a sample of 290418 ChEMBL [10] molecules that we use to train our model. In addition, we curate a held-out dataset of 4989 project molecules coming directly from internal drug discovery projects, to challenge the model on out-of-domain prediction.



(a) Test molecule



(b) Project molecules

Figure 2: ROC-AUC curves for all scorers on both test and project molecules.

We use AiZynthFinder [11] with default parameters to compute the routes of both datasets. A route is considered to be solved if all leaves are purchasable molecules. If more than one route is found for a molecule, the route scored the highest by AiZynthFinder is retained. For solved routes, we compute the depth of the tree. Since the maximum tree depth allowed by AiZynthFinder's default parameters is 7, we assign a depth of 10 for molecules where a route is not found.

To create target molecule-intermediate pairs, we randomly sample a maximum of three intermediate molecules for each route and recompute the depth accordingly. We do the latter by treating the sampled intermediate as an available building block, meaning that we can ignore the synthesis steps needed to create it. At a synthesis tree level, this effectively results in the removal of any nodes beyond the intermediate molecule. This reduces the depth of the tree when the intermediate is found along the longest branch of the tree.

To train the model to recognise that not every molecule is a viable intermediate and pathologically reduce predicted route depth in response to supplied intermediates, we also generate a set of "false" intermediates. We do this by pairing target molecules with a random molecule sampled across all intermediates in the dataset. For these negative pairs, the expected depth is the same as that of the target molecule when no intermediate is supplied.

ChEMBL molecules are randomly split 70/15/15 into train, test, validation sets, leaving 202421, 43563, 44434 molecules, respectively. After augmentation we have a total of 646421, 139594, 142600 data points per set. The project molecules are augmented in the same way, totalling 20277 project-intermediate data points.

3 Experiments

We run experiments to assess the ability of Leap to adapt to the knowledge of intermediates, its ability to classify the synthetic feasibility of molecules, and whether pre-training helps. We compare to SAScore, SCScore and RAScore. SAScore is normalised between 0 and 1 through min-max scaling. We explore results on the ChEMBL test set as well as internal molecules set to assess generalisation to potentially out-of-domain novel compounds.

3.1 Identifying synthesisable molecules

To determine molecular synthesisability we compute routes using AiZynthFinder [11] for both test and project molecules. If a route can be found, the molecule is classified as synthesisable.

To permit comparison between Leap and RAScore, SAScore and SCScore, and to establish a baseline for further experiments, we do not consider intermediates in this experiment. We score all molecules with all scorers, and use these to compute the ROC-AUC curves shown in Figure 2. Leap outperforms all other methods in terms of AUC score for both test and project molecules. This demonstrates

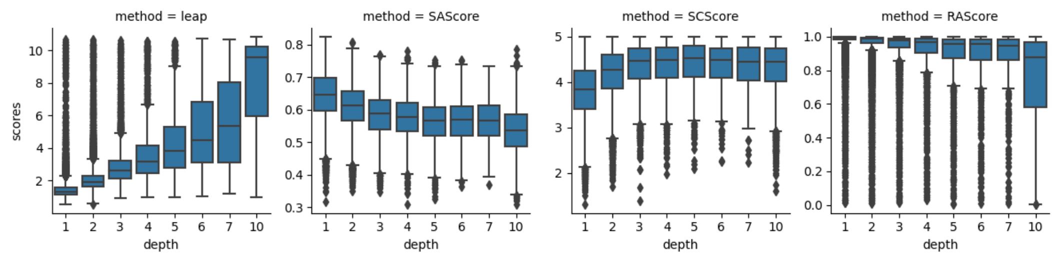


Figure 3: Distribution of scores assigned by Leap, SAScore, SCScore and RAScore for molecules with synthetic routes of different depths.

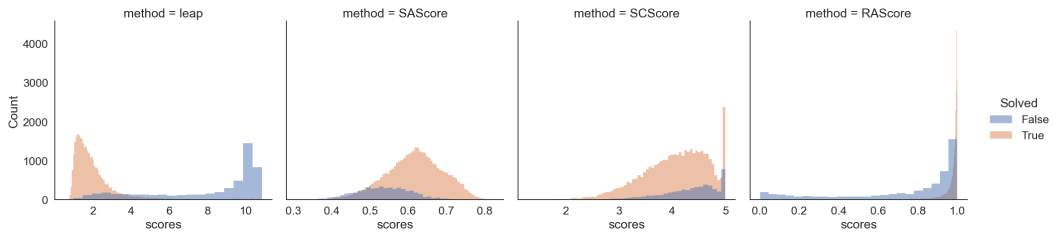


Figure 4: Distribution of scores assigned by Leap, SAScore, SCScore and RAScore for solved and unsolved compounds.

that our model remains useful even when intermediates are not present. Leap also shows superior performance on the hold-out set of project molecules, showing its ability to generalise beyond its training domain. Conversely, even though RAScore was also trained on ChEMBL molecules, its predictive performance deteriorates on out-of-domain molecules as previously observed in Thakkar et al. [20].

We hypothesise that the ability of our model to better distinguish between easy- and difficult-to-synthesise molecules is due to Leap using tree depth as a proxy for synthetic complexity. We show, from Figure 3 that scores predicted by Leap correlate with the tree depth predicted by AiZynthFinder. All other scorers show very similar scores across molecules that require different depths of synthesis. Therefore, this can make it harder to establish the optimal threshold to differentiate between synthesisable and non-synthesisable molecules: as shown in Figure 4 Leap is the only scorer that shows a clear distinction in scores assigned to synthesisable and non-synthesisable molecules.

3.2 Effects of intermediates

In the previous experiments, we showed that even when not supplying intermediates Leap performs better than other scorers in identifying whether a molecule is synthesisable. Here, we explore how Leap performs when intermediates are supplied. We use the test and project molecule sets obtained from Section 2.3.3, both with and without intermediates. We then score all the molecules with the scorers, however only Leap can make use of intermediates. From Section 2.3.3, we mark molecules as synthesisable if their assigned depth is not 10 and non-synthesisable otherwise. Note that the same molecule can go from non-synthesisable to synthesisable if a useful intermediate is supplied. It should also be noted that not all intermediates have this effect: aside from including false intermediates, selecting intermediates at random can result in some of them not contributing to the reduction of tree depth.

We compute the AUC for each scorer in identifying whether a molecule is synthesisable, both when no intermediate is known, as well as when an intermediate is provided. From Figure 5 we can see that for molecules for which a key intermediate is given, Leap can differentiate between synthesisable molecules with an AUC score of 0.89. Conversely, all other scorers incur a decrease in their AUC scores. This is to be expected, but highlights the limitations of such scorers: complex synthesis routes become simpler if chemists already possess key intermediates. We further check that our model does not pathologically lower the predicted score if any intermediate is supplied by comparing the distributions between supplying false and no intermediates. Figure 6 shows that distributions of

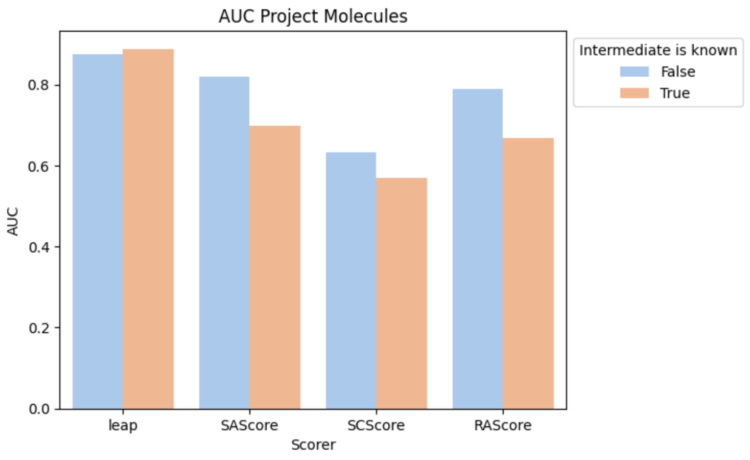
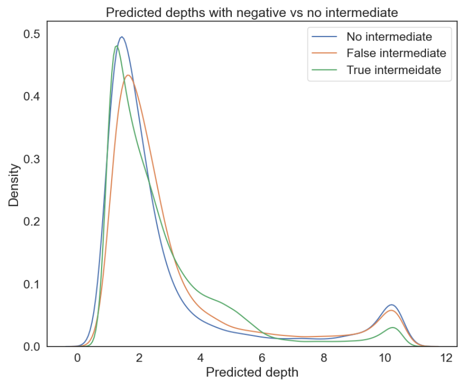
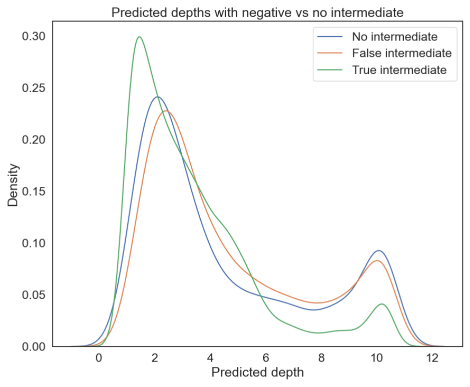


Figure 5: Barplot showing the AUC score for the various scorers when we do and do not have a key intermediate for molecules.



(a) Test molecule



(b) Project Molecules

Figure 6: Distributions of predicted depths for both project and test molecules when false, true and no intermediates are supplied.

predicted scores are very close for both test and project molecules, indicating that Leap's predictions are robust against irrelevant intermediates.

3.3 Ranking molecules with intermediates

The purpose of this experiment is to investigate whether our model maintains the ranking of the expected tree depth as different intermediates are supplied. We group both test and project sets by their target molecules, and then compare the expected ranking of tree depth for each target molecule with the different intermediates, to that obtained by the prediction of our model. If everything is ranked correctly, then that is considered a correct ranking. For this experiment, we do not consider false intermediates. We obtained 77% correct ranking on the test set, and 69% on the project set.

3.4 Comparison to baseline models

To validate our architecture and our pre-training method, we conduct experiments using three baseline models: CatBoost, GPT-2 (i.e. Leap with no pre-training) and BERT [5]. We compare their performance on R2, mean absolute error (MAE) and mean squared error (MSE). For CatBoost, intermediate and target molecules are converted to ECFP4 fingerprints, and then concatenated. For

	Test			Project		
Model	R2	MAE	MSE	R2	MAE	MSE
CatBoost	0.42	1.32	3.63	0.19	1.98	7.18
BERT	0.54	1.05	2.89	0.29	1.78	6.77
GPT-2	0.41	1.23	3.71	0.17	1.92	7.48
Leap	**0.64**	**0.88**	**2.29**	**0.41**	**1.56**	**5.59**

Table 1: Comparison between performance of our model Leap to the baseline models on regression metrics: R2, mean absolute error (MAE), mean squared error (MSE)

BERT, they are passed in as two sentences, separated by `SEP` token. The results are reported in Table 1. We show that pre-training helped boost the performance of Leap, and that the fine-tuned Leap outperforms both CatBoost and BERT on all the metrics.

4 Conclusion

We introduced Leap, the first method for scoring the synthesisability of molecules that considers intermediates. We have pre-trained a model on the task of generating synthesis routes, and fine-tuned it on tree depth prediction with and without intermediates. We have shown the ability of our model to generalise to out-of-domain molecules and to take advantage of intermediates provided from various points along a synthesis route. Future work includes testing this scorer paired with a generative model, as well as testing the effect of more challenging false intermediates, for example by modifying intermediates along a synthesis route of a target molecule.

References

[1] IBM RXN. URL `https://rxn.res.ibm.com/`.

[2] Daylight theory: Reaction smiles and smirks. URL `https://www.daylight.com/meetings/summerschool01/course/basics/smirks.html`.

[3] Josep Arús-Pous, Simon Viet Johansson, Oleksii Prykhodko, Esben Jannik Bjerrum, Christian Tyrchan, Jean-Louis Reymond, Hongming Chen, and Ola Engkvist. Randomized SMILES strings improve the quality of molecular generative models. *Journal of Cheminformatics*, 11(1): 71, November 2019.

[4] Connor W. Coley, Luke Rogers, William H. Green, and Klavs F. Jensen. Scscore: Synthetic complexity learned from a reaction corpus. *Journal of Chemical Information and Modeling*, 58 (2):252–261, 2018. doi: 10.1021/acs.jcim.7b00622. URL `https://doi.org/10.1021/acs.jcim.7b00622`. PMID: 29309147.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

[6] Peter Ertl and Ansgar Schuffenhauer. *Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions*, volume 1. June 2009.

[7] Hanyu Gao, Thomas J. Struble, Connor W. Coley, Yuran Wang, William H. Green, and Klavs F. Jensen. Using machine learning to predict suitable conditions for organic reactions. *ACS Central Science*, 4(11):1465–1476, 2018. doi: 10.1021/acscentsci.8b00357. URL `https://doi.org/10.1021/acscentsci.8b00357`. PMID: 30555898.

[8] Wenhao Gao and Connor W. Coley. The synthesizability of molecules proposed by generative models. *Journal of Chemical Information and Modeling*, 60(12):5714–5723, 2020. doi: 10.1021/acs.jcim.0c00174. URL `https://doi.org/10.1021/acs.jcim.0c00174`. PMID: 32250616.

[9] Wenhao Gao, Rocío Mercado, and Connor W. Coley. Amortized tree generation for bottom-up synthesis planning and synthesizable molecular design, 2022.

[10] Anna Gaulton, Louisa J. Bellis, A. Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, and John P. Overington. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Research*, 40(D1):D1100–D1107, 09 2011. ISSN 0305-1048. doi: 10.1093/nar/gkr777. URL `https://doi.org/10.1093/nar/gkr777`.

[11] Samuel Genheden, Amol Thakkar, Veronika Chadimová, Jean-Louis Reymond, Ola Engkvist, and Esben Bjerrum. AiZynthFinder: a fast, robust and flexible open-source software for retrosynthetic planning. *Journal of Cheminformatics*, 12(1):70, November 2020.

[12] J P Hughes, S Rees, S B Kalindjian, and K L Philpott. Principles of early drug discovery. *Br J Pharmacol*, 162(6):1239–1249, March 2011.

[13] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, Leonid Zaslavsky, Jian Zhang, and Evan E Bolton. PubChem 2023 update. *Nucleic Acids Research*, 51(D1):D1373–D1380, 10 2022. ISSN 0305-1048. doi: 10.1093/nar/gkac956. URL `https://doi.org/10.1093/nar/gkac956`.

[14] Songtao Liu, Zhitao Ying, Zuobai Zhang, Peilin Zhao, Jian Tang, Lu Lin, and Dinghao Wu. Metro: Memory-enhanced transformer for retrosynthetic planning via reaction tree, 2023. URL `https://openreview.net/forum?id=9JjGZsDvHb`.

[15] Daniel Mark Lowe. Extraction of chemical structures and reactions from the literature. 2012. doi: 10.17863/CAM.16293. URL `https://www.repository.cam.ac.uk/handle/1810/244727`.

[16] Gergely M Makara, László Kovács, István Szabó, and Gábor Pőcze. Derivatization design of synthetically accessible space for optimization: In silico synthesis vs deep generative design. *ACS Med Chem Lett*, 12(2):185–194, January 2021.

[17] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. URL `https://api.semanticscholar.org/CorpusID:160025533`.

[18] Philippe Schwaller, Teodoro Laino, Thé ophile Gaudin, Peter Bolgar, Christopher A. Hunter, Costas Bekas, and Alpha A. Lee. Molecular transformer: A model for uncertainty-calibrated chemical reaction prediction. *ACS Central Science*, 5(9):1572–1583, aug 2019. doi: 10.1021/acscentsci.9b00576. URL `https://doi.org/10.1021%2Facscentsci.9b00576`.

[19] Marwin H S Segler, Mike Preuss, and Mark P Waller. Planning chemical syntheses with deep neural networks and symbolic ai. *Nature*, 555(7698):604—610, March 2018. ISSN 0028-0836. doi: 10.1038/nature25978. URL `https://doi.org/10.1038/nature25978`.

[20] Amol Thakkar, Veronika Chadimová, Esben Jannik Bjerrum, Ola Engkvist, and Jean-Louis Reymond. Retrosynthetic accessibility score (rascore) – rapid machine learned synthesizability classification from ai driven retrosynthetic planning. *Chem. Sci.*, 12:3339–3349, 2021. doi: 10.1039/D0SC05401A. URL `http://dx.doi.org/10.1039/D0SC05401A`.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[22] Xiaorui Wang, Yuquan Li, Jiezhong Qiu, Guangyong Chen, Huanxiang Liu, Benben Liao, Chang-Yu Hsieh, and Xiaojun Yao. Retroprime: A diverse, plausible and transformer-based method for single-step retrosynthesis predictions. *Chemical Engineering Journal*, 420:129845, 2021. ISSN 1385-8947. doi: https://doi.org/10.1016/j.cej.2021.129845. URL `https://www.sciencedirect.com/science/article/pii/S1385894721014303`.

[23] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2020.