Digital Discovery

ROYAL SOCIETY OF CHEMISTRY

PAPER

View Article Online
View Journal

Check for updates

Cite this: DOI: 10.1039/d5dd00369e

MolEncoder: towards optimal masked language modeling for molecules

Fabian P. Krüger,*[abc] Nicklas Österbacka,[a] Mikhail Kabeshov,[a] Ola Engkvist[ad] and Igor Tetko[c]

Received 18th August 2025
Accepted 25th October 2025

DOI: 10.1039/d5dd00369e

rsc.li/digitaldiscovery

Predicting molecular properties is a key challenge in drug discovery. Machine learning models, especially those based on transformer architectures, are increasingly used to make these predictions from chemical structures. Inspired by recent progress in natural language processing, many studies have adopted encoder-only transformer architectures similar to BERT (Bidirectional Encoder Representations from Transformers) for this task. These models are pretrained using masked language modeling, where parts of the input are hidden and the model learns to recover them before fine-tuning on downstream tasks. In this work, we systematically investigate whether core assumptions from natural language processing, which are commonly adopted in molecular BERT-based models, actually hold when applied to molecules represented using the Simplified Molecular Input Line Entry System (SMILES). Specifically, we examine how masking ratio, pretraining dataset size, and model size affect performance in molecular property prediction. We find that higher masking ratios than commonly used significantly improve performance. In contrast, increasing model or pretraining dataset size quickly leads to diminishing returns, offering no consistent benefit while incurring significantly higher computational cost. Based on these insights, we develop MolEncoder, a BERT-based model that outperforms existing approaches on drug discovery tasks while being more computationally efficient. Our results highlight key differences between molecular pretraining and natural language processing, showing that they require different design choices. This enables more efficient model development and lowers barriers for researchers with limited computational resources. We release MolEncoder publicly to support future work and hope our findings help make molecular representation learning more accessible and cost-effective in drug discovery.

Introduction

Developing new drugs is an expensive and time-consuming process. A major reason for this is that many drug candidates fail during development due to poor properties such as low solubility, high toxicity, or metabolic instability.[1] By enabling the profiling of compounds even before they are synthesized, computational approaches allow earlier identification of these issues, which reduces failed experiments and helps limit the need for animal testing. Recent regulatory developments encourage the use of computational models to reduce reliance on animal testing.[2] While early screening for undesirable properties alone cannot eliminate all failures, it can improve overall efficiency and reduce unnecessary resource expenditure.

[a]AstraZeneca R&D, Discovery Sciences, Molecular AI, 431 83 Mölndal, Sweden
[b]TUM School of Computation, Information and Technology, Department of Mathematics, Technical University of Munich, 80333 Munich, Germany
[c]Helmholtz Munich – German Research Center for Environmental Health (GmbH), Institute of Structural Biology, Molecular Targets and Therapeutics Center, 85764 Neuherberg, Germany
[d]Department of Computer Science and Engineering, Chalmers University of Technology and University of Gothenburg, Sweden

Therefore, predicting molecular properties using computational methods has the potential to make drug development more efficient, ultimately saving significant costs and resources.[3]

Machine learning models offer a promising approach to achieve this goal.[4] They have gained significant traction in drug discovery, with numerous tools now available to predict molecular properties from chemical structure.[4] Although traditional machine learning models for this have existed for decades, recent breakthroughs in deep learning have sparked interest in applying these techniques to chemistry.[5,6] This r-enewed attention is driven by progress in fields such as natural language processing and computer vision, where advances in model architectures and computing power have produced substantial improvements in performance.[7]

One of the most important developments in this context has been the transformer architecture introduced by Vaswani *et al.* in 2017.[8] This architecture became the foundation for many high-performing models in natural language processing by allowing them to capture contextual relationships in text through self-attention. A key variant of this architecture are bidirectional transformers for language understanding, first introduced by Devlin *et al.* in 2018 in the form of BERT (Bidirectional Encoder Representations from Transformers).[9] BERT is an encoder-only transformer trained using masked language modeling and next sentence prediction. In masked language modeling parts of the input sequence are hidden and the model learns to predict the missing parts from their context. In next sentence prediction, the model has to determine whether the second sentence follows the first in the original text or if it is a random sentence from the training corpus. These unsupervised training tasks allow the model to learn general features of the input domain, which can then be adapted to specific tasks such as classification or regression through fine-tuning.[9]

Inspired by BERT's success, researchers started applying this model architecture to molecular data.[10–20] To do so, molecules are represented as text sequences using the Simplified Molecular Input Line Entry System (SMILES).[21] In this setting, masked language modeling can be applied directly by masking parts of the SMILES string, whereas next sentence prediction is typically omitted because molecules lack a natural analogue to ordered sentences in text. Using masked language modeling, researchers have built molecular BERT-like encoder models that learn contextual representations of molecular structure.[10,11,13–19,22] These representations can be fine-tuned to predict a wide range of molecular properties. A related line of work has adapted the same approach to biological sequences, such as proteins represented as amino acid sequences,[23] and used combinations of BERT-like encoder models to predict interactions between molecules and proteins.[24] Recent studies have also shown that combining masked language modeling with auxiliary tasks during pretraining, such as predicting physical or chemical properties, can improve model performance. This can be achieved through joint training or by adapting the model to the auxiliary tasks after pretraining.[13,16,18]

Although these methods have gained significant attention, many modeling choices are still directly borrowed from natural language processing. One important example is the masking ratio. Molecular BERT models use a masking ratio of 15 percent, as in the original BERT paper.[10,11,13–19,22] However, there is little evidence that this value is optimal for SMILES strings, which differ from natural language in their structure and redundancy. In fact, even follow-up work in natural language processing found that higher masking ratios, such as 40 percent, lead to better model performance.[25] These findings raise the question on whether the historical masking ratio from natural language processing is appropriate for molecular data.

Another assumption that has been carried over is that scaling up model size and pretraining dataset size leads to better performance. Many existing BERT models for molecules are pretrained on tens or hundreds of millions of molecules, even up to over a billion.[10,13,15,17,24] This increases computational requirements significantly. However, previous studies have reported mixed findings on the benefits of larger pretraining datasets. Some observe performance improvements, while others find diminishing returns or no gains beyond certain thresholds.[15,18] These inconsistencies suggest that the relationship between pretraining dataset size and model performance in this domain is still unclear.

Similarly, the effect of scaling model size has not been systematically studied. Model sizes used in previous work vary widely, and it is unclear how many parameters are actually needed to achieve good performance. While too few parameters can prevent the model from learning the necessary patterns in the data, using more than needed results in wasted computational resources during both training and deployment.

In this work, we address these knowledge gaps by systematically evaluating how masking ratio, model size, and pretraining dataset size affect the performance of encoder-only transformers trained on SMILES strings using masked language modeling. To this end, we build upon the ModernBERT architecture,[26] the current state-of-the-art and most efficient variant of BERT in natural language processing. ModernBERT incorporates recent advances such as rotary positional embeddings, GeGLU activation functions, pre-normalization, the removal of bias terms, FlashAttention, and unpadding. These components are described in detail in the methods section.

Our experiments employ statistically robust procedures, including 5 × 5 cross-validation, to capture the impact of variability from both data splitting and fine-tuning, following the guidelines suggested by Ash *et al.*[27] Our evaluation spans several datasets that reflect key tasks in drug discovery, such as solubility, metabolic stability, permeability, and enzyme inhibition. Based on our findings, we propose MolEncoder, a model that achieves improved performance over existing BERT-based models while requiring significantly fewer computational resources.

Results

Here we present our findings on how different pretraining settings for masked SMILES strings influence performance on common tasks in drug discovery. Our goal is to provide practical guidance for future model development. The evaluation covers a diverse set of tasks, including prediction of human liver microsomal stability (*HCLint*), aqueous solubility (*Solubility*), membrane permeability (*Permeability*), lipophilicity (*Lipophilicity*), and CYP3A4 enzyme inhibition (*CYP*) for small molecules.[28–30] These five datasets are used consistently throughout all experiments.

We adopt a 5 × 5 cross-validation strategy, as recommended by Ash *et al.*,[27] to ensure robust and reliable evaluation. In preliminary experiments, we assessed the stability of this setup by repeating the analysis across different random seeds, which affect both model fine-tuning and dataset splitting. The results were almost identical, confirming the reliability of the evaluation (see Fig. S1). Based on this consistency, we used this setup for all subsequent experiments.

In order to improve masked language modeling for molecules and develop a better model, we first investigate several key design choices that influence pretraining effectiveness. We begin our analysis by examining how the masking ratio used during pretraining affects downstream performance. This ratio determines how much information is hidden from the model and is a design choice that has received little attention in existing work. We then explore two additional factors: the pretraining dataset size and the model size. To better understand the generalizability of our results, we also include ablation studies on our training hyperparameters.

Finally, we put our findings to the test by developing MolEncoder, a model that incorporates these insights to improve masked language modeling for molecules, and evaluating whether it achieves better performance at a lower training cost compared to existing approaches. This comparison allows us to assess the practical value of our design choices and understand the extent to which masked pretraining can be optimized for drug discovery tasks.

Masking ratio

We first examined how the masking ratio used during pretraining affects downstream performance. To this end, we tested a range of masking ratios from 10% to 90% and evaluated the resulting models on all five of our benchmark datasets. Fig. 1 summarizes the mean absolute error (MAE) across the five tasks as a function of masking ratio. The confidence intervals reflect the variability due to data splitting and fine-tuning across cross-validation folds. They indicate how precisely we can distinguish between models' average performances given these sources of variation and are constructed using Tukey's Q critical value as proposed by Hochberg and Tamhane.[31] If two intervals overlap, the observed difference may be explained by this variation alone, and we cannot confidently conclude that the models differ in performance.

Our results show a clear pattern: models trained with masking ratios below 20% and above 70% perform significantly worse than those trained with higher ratios, as determined by repeated-measures ANOVA with *post hoc* Tukey HSD tests. This includes the widely used masking ratio of 15%. The performance drop is consistent across all five datasets and is reflected by significant differences in MAE. In contrast, for masking ratios between 20% and 60%, the MAEs remain relatively consistent. Although the 30% masking ratio achieves the best overall performance, the differences between 20% and 60% are small and mostly not statistically significant.

The observed trend is consistent across all five evaluation tasks, suggesting that the effect of masking ratio is robust to the specific molecular property being predicted. We confirmed this observation using alternative evaluation metrics, including mean squared error (MSE), coefficient of determination (R^2), and Spearman correlation (ρ), all of which show the same pattern (see Fig. S2–S4).

We also explored how this effect generalizes to variations in SMILES representation. Specifically, we repeated the experiments using SMILES strings that include explicit hydrogen atoms. While models trained on these SMILES strings performed notably worse than those using standard SMILES strings (Fig. S5), we again observed a strong dependence on the masking ratio. In particular, masking ratios above 20% again consistently outperformed lower ones (Fig. S6). In this setting, the 40% masking ratio gave the best results. These findings suggest that the observed benefits of higher masking ratios than

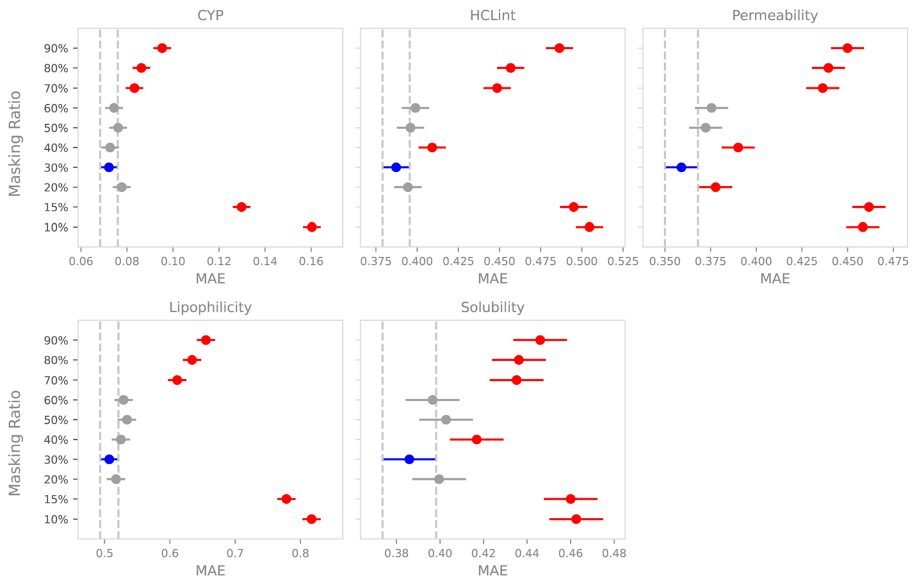

Fig. 1 Mean absolute error (MAE) for five molecular property prediction tasks, shown for models pretrained with different masking ratios. Points show MAE averaged over 5 × 5 cross-validation. Horizontal bars and color coding are based on 95% simultaneous confidence intervals for each mean, constructed using Tukey's Q critical value as proposed by Hochberg and Tamhane,[31] applied post-ANOVA. Within this construction, non-overlapping intervals indicate statistically significant differences. They reflect how well we can distinguish models given the variability from data splitting and fine-tuning across cross-validation folds. The best-performing model (lowest MAE over 5 × 5 CV) is shown in blue. Models not significantly different from the best are shown in gray; significantly worse models are shown in red. The models were trained on $\log_{10}$-transformed values of the original measurements (as provided and recommended by Polaris[27]). Therefore, an MAE of 0.25, for example, corresponds to a prediction error of approximately a factor of 1.8 in the original scale. Further details on the original measurement units are provided in the Methods section.

the commonly used 15% are robust, holding consistently across different downstream tasks, evaluation metrics, and even variations in SMILES representation.

Effect of pretraining dataset and model size

Next, we investigated how pretraining dataset size and model size affect downstream performance. We evaluated all combinations of three model sizes: 5 million, 15 million, and 111 million parameters, and three pretraining dataset sizes: half of the ChEMBL dataset, the full ChEMBL dataset, and the PubChem dataset (which includes ChEMBL). Fig. 2 summarizes the results for all five evaluation tasks.

Our first observation is that pretraining consistently improves performance. Models trained directly on the downstream task without any pretraining always performed significantly worse than the best pretrained model.

Second, we find that the smallest model with only 5 million parameters consistently performs worse. Independent of the pretraining dataset size, its performance is significantly worse than that of the best model.

However, increasing model size and dataset size beyond a certain point does not lead to better performance. The combinations that yield the best results, or are not significantly statistically different from the best, are those using the 15 million parameter model pretrained on either the full ChEMBL dataset or half of the ChEMBL dataset. This trend holds across all five evaluation tasks and is consistent for all other metrics, including mean squared error (MSE), coefficient of determination (R^2), and Spearman correlation (ρ) (see Fig. S7–S9).

The largest model, with 111 million parameters, performs well in some cases but not consistently. For some tasks and pretraining dataset sizes, its performance is significantly worse than the best model. However, this varies across evaluation datasets and does not hold consistently across metrics. This suggests that increasing both model size and pretraining dataset size does not reliably improve performance and may even lead to a decrease in model performance for the masked language learning pretraining task.

To investigate whether the observed differences in performance could be attributed to distributional differences rather than dataset size, we compared the chemical and drug-like property distributions of the pretraining datasets. Specifically, we examined quantitative estimates of drug-likeness (QED), adherence to Lipinski's Rule of Five, and overall coverage of chemical space (Fig. S11A–C). We found that ChEMBL and PubChem exhibit very similar distributions across these metrics, suggesting that their overall molecular characteristics are comparable. Notably, although PubChem contains nearly all SMILES strings present in the downstream tasks (>98%)—substantially more than ChEMBL—it does not lead to improved performance (Fig. S11D). This further supports that differences

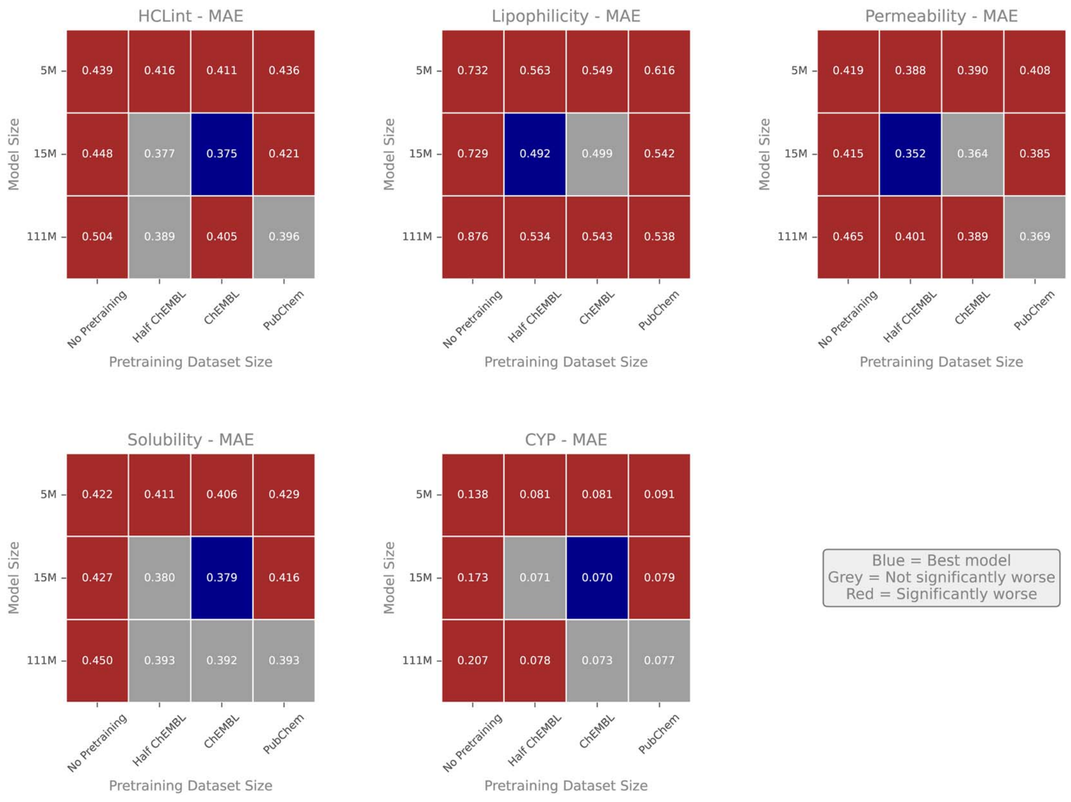

Fig. 2 Mean absolute error (MAE) for five molecular property prediction tasks, reported for models of different sizes (in number of parameters) pretrained on different datasets. The pretraining dataset sizes are 114.4M molecules for PubChem, 2.3M for ChEMBL, and 1.2M for Half ChEMBL. All models were trained using masked language modeling with a masking ratio of 30%. Each cell shows the average MAE over 5 × 5 cross-validation. Cell color indicates statistical difference relative to the best-performing model (see legend in figure).

in dataset coverage or distribution are not the primary drivers of model performance. Both datasets broadly span the chemical space relevant to the downstream evaluation tasks (Fig. S12 and S13). Most importantly, the fact that pretraining on half of ChEMBL yields performance statistically indistinguishable from pretraining on the full dataset indicates that the lack of improvement with larger datasets is unlikely to stem from a distributional mismatch. Since the half-ChEMBL subset was obtained by uniformly drawing molecules from the full dataset, it preserves the original distribution of molecular properties, reinforcing that the observed plateau in performance reflects a true scaling effect rather than differences in dataset composition.

To assess whether pretraining dataset size could be reduced further for the 15 million parameter model, we extended the study to include smaller subsets of ChEMBL: one fourth and one eighth of the dataset. These experiments resulted in worse performance (see Fig. S10), indicating that further reducing dataset size below half of ChEMBL results in worse models.

An additional question we investigated was whether downstream evaluation is necessary, or if pretraining performance alone can serve as a reliable proxy for downstream performance. To test this, we examined the correlation between pretraining performance and final performance on each of our five evaluation datasets. Pretraining performance was defined as the cross entropy loss for predicting the masked tokens and final performance as the mean absolute error for predicting the label. As shown in Fig. S14, we found no meaningful correlation between these metrics. Both Pearson and Spearman correlation coefficients were low and not statistically significant across all evaluation tasks. This result shows that pretraining loss does not predict downstream performance, highlighting the importance of directly evaluating models on the target tasks.

Comparison to existing models

To test whether our findings lead to practical improvements, we compared a model trained using our insights with widely used baselines. We selected ChemBERTa-2 (ref. 17) and MolFormer[15] because both are based on the BERT architecture and employ masked language modeling as their pretraining objective, making them directly comparable to our approach. In addition, both models provide publicly available pretrained weights on Hugging Face, which enabled us to evaluate the original released models without retraining and thereby ensured fair comparability. Other potential baselines did not fulfill all of these criteria, either differing in pretraining objectives or lacking publicly available model weights. Collectively, these considerations made ChemBERTa-2 and MolFormer the most appropriate baselines for assessing the effectiveness of our proposed approach.

We also included a classical baseline using XGBoost trained on extended connectivity fingerprints with a radius of 2 (ECFP4), a method that remains widely used in cheminformatics. We also extended our analysis by adding four more datasets for a more comprehensive comparison between the models. These datasets include human plasma protein binding rate (*hPP Binding*), drug half-life in the body (*Half-life*), inhibitory constant for the delta opioid receptor (*DOR Binding*), and inhibitory constant for the dopamine D3 receptor (*D3R Binding*).[29,32,33]

Our goal was to assess whether using a relatively small model, a modest pretraining dataset, and a higher masking ratio would result in competitive or superior downstream performance, while requiring much less computational effort. Therefore we used the model with 15M parameters pretrained on half of ChEMBL, which we call *MolEncoder* from now on. The results of this comparison are shown in Fig. 3.

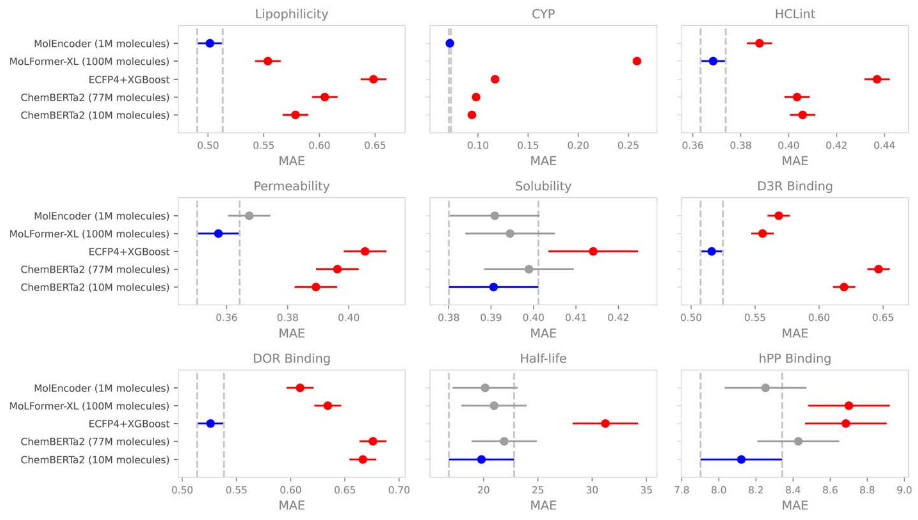

Fig. 3 Comparison of our model, *MolEncoder*, to ChemBERTa-2,[17] MoLFormer[15] and an XGBoost model. The number of molecules each model was pretrained on is shown in brackets after the model name. Model sizes are 15.2M parameters for *MolEncoder*, 3.4M parameters for ChemBERTa-2, and 45.5M parameters for MoLFormer. The best-performing model is shown in blue. Models not significantly different from the best are shown in gray; significantly worse models are shown in red. All MAEs are reported in $\log_{10}$-transformed scale, following the data format provided by Polaris. The only exceptions are the half-life dataset, reported in hours, and the hPP binding dataset, expressed as the percentage of a drug bound to plasma proteins in blood, as both were supplied in their original (non-log-transformed) scales.

Across 9 evaluation datasets, our model achieved the best performance or showed no statistically significant difference from the best in the majority of cases. It was best or not significantly different from the best on 6 datasets. Among the BERT-based models (MoLFormer, ChemBERTa-2, and ours), this number increased to 8 out of 9 datasets. In direct comparisons, our model performed significantly better than the ECFP4 with XGBoost baseline on 6 datasets. Against both versions of ChemBERTa-2, it was also significantly better on 6 datasets and showed no significant difference on the remaining 3. Compared to MoLFormer, it was significantly better on 4 datasets, showed no significant difference on another 4, and was significantly worse on 1 dataset.

Between the two ChemBERTa-2 variants, performance is either statistically indistinguishable or slightly better for the version pretrained on 10 million molecules, depending on the evaluation dataset. MoLFormer performs significantly better than ChemBERTa-2 on five datasets, significantly worse on two, and shows no significant difference on the remaining 2 datasets.

Overall, our model *MolEncoder* achieved the strongest and most consistent performance among all tested models. Comparing to MolFormer, our model is being pretrained on a dataset 100 times smaller and the parameter count of our model is only one third of MoLFormer. These results suggest that for achieving high performance in masked language modeling for molecular property prediction in drug discovery, large-scale pretraining datasets or model sizes are not required. Models trained with a small computational budget can outperform more resource-intensive models when pretraining settings are chosen carefully.

While this work mainly focuses on how to make masked language modeling better, we also wanted to see how masked language model pretraining compares to other modeling approaches. We therefore evaluated our final model, *MolEncoder*, on the official benchmark test sets for the five datasets used throughout this study. These benchmarks are hosted on the Polaris platform[34] and provide held-out test sets that we did not use at any point during model development. While we used the benchmark training data for our 5×5 cross-validation experiments, the benchmark test sets remained entirely untouched until this final evaluation.

MolEncoder achieved strong results across the benchmark tasks. It ranked third out of thirteen submissions on the permeability prediction task and fourth out of twelve on the liver microsomal stability prediction task. For the solubility prediction task, it placed thirteenth out of forty-two models. On the lipophilicity prediction task, it ranked fourth out of eight participants. Finally, it placed second out of two on the CYP3A4 inactivation rate task. Full benchmark results, including ranks, metric values, and number of competing models, are provided in Table S1, along with direct links to the corresponding public leaderboard pages at this time point.

These rankings confirm that our MLM pretraining performs reliably across a range of molecular property prediction tasks compared to other machine learning models. It consistently outperforms traditional machine learning baselines in the leaderboard. In several tasks, it also outperforms other modern deep learning models. While it is not the top-performing method on every task, it consistently delivers strong performance across tasks.

It is important to note that these leaderboards come with limitations. Polaris does not enforce blind evaluation, and performance relative to the leaderboard results is evident before being uploaded to the platform. This may lead to a reluctance in sharing results that achieve a low rank. Moreover, the leaderboard does not include statistical significance testing, meaning that small differences in rank may not reflect meaningful differences in performance. The Polaris platform itself acknowledges these limitations and cautions against overinterpreting the public rankings.[34]

Despite these limitations, these rankings give some evidence that MLM pretraining is a strong approach that delivers consistently good results. While not delivering the best model for each task, it performs robustly across tasks and provides a solid foundation for further improvement as for example combination with other auxiliary prediction tasks, which we will discuss in the following section.

Discussion

In this study, we explored how to improve masked language modeling for molecular data by systematically varying pretraining conditions and evaluating their impact on downstream performance in drug discovery tasks. Our goal was to identify which design choices lead to models that achieve stronger downstream performance while reducing the computational cost required for pretraining, and to understand how these trade-offs differ from those observed in natural language processing.

We found that higher masking ratios during pretraining consistently led to better performance across downstream tasks. In contrast, increasing the size of the model or the pretraining dataset only provided benefits up to a point. Beyond that, further scaling did not improve performance and, in some cases, even reduced it. Finally, we demonstrated that by following our proposed pretraining guidelines, it is possible to train models that outperform the current state of the art while requiring significantly less computational resources. These results challenge some common assumptions in the field and offer a more efficient path forward. In the following paragraphs, we discuss each of these findings in more detail and place them in the context of existing work.

As with any empirical study, our findings are subject to certain limitations. Most notably, it is not feasible to explore all possible combinations of hyperparameters. This means we cannot guarantee that our conclusions hold universally across every conceivable setting. However, we aimed to control for the most influential factors within our computational budget. Our ablation studies indicate that common hyperparameter variations such as learning rate and batch size do not significantly alter the observed trends, which is consistent with previous work on hyperparameter optimization.[35] This suggests that our conclusions are likely to hold across a range of realistic conditions.

In addition, we focused on a variety of tasks within the drug discovery domain, which primarily involve drug-like molecules. While this provides a representative assessment across bioactive chemical space, future work will be needed to validate whether the same conclusions hold when applying our findings outside of the drug discovery setting, for example in material science or other chemically distinct domains.

A key finding of our study is the strong impact of the masking ratio on model performance. Most existing work that applies masked language modeling to SMILES strings uses a masking ratio of 15%.[10,11,13–19,22] This value was originally used in the BERT paper[9] and has likely been adopted into the chemistry domain without systematic investigation. Our results show that higher masking ratios lead to significantly better performance (Fig. 1). This effect is consistent across different downstream tasks, evaluation metrics, and even across variations in the SMILES representation. These findings have strong practical relevance, as simply increasing the masking ratio can substantially improve model quality without changing any other aspect of the training setup. A few years after the initial BERT paper, Wettig *et al.*[25] showed that the default 15% masking ratio is suboptimal for natural language. They found that a higher rate of 40% leads to better performance, which has influenced the development of subsequent models. Here we find a similar trend for the SMILES language. We hope that our findings will have a similar impact in the field of molecular representation learning and help guide future pretraining strategies.

Another key aspect we investigated is the effect of pretraining data size on model performance. Our results provide strong evidence that increasing the size of the pretraining dataset does not lead to better downstream performance beyond the size of 1 million molecules. In multiple settings, models pretrained on larger datasets performed similarly or slightly worse than those trained on smaller ones. Although the difference in performance could, in principle, be influenced by variations in molecular distributions between PubChem and ChEMBL, our additional analyses suggest that this is not the primary cause. ChEMBL and PubChem have similar trends in distributions of key drug-like properties and chemical space coverage, which is consistent with previous observations.[36] Yet, models pretrained on PubChem do not outperform those trained on ChEMBL. Even more compellingly, pretraining on only half of ChEMBL yields performance not significantly different than using the full dataset. Because this subset was uniformly drawn, it preserves the same molecular distribution, providing strong evidence that the observed performance plateau arises from scaling limitations rather than dataset composition. While this does not entirely exclude the possibility that subtle distributional biases play a role, the comparison closely mirrors a practical choice faced by model developers: whether to pretrain on a smaller, carefully curated dataset such as ChEMBL, or on a larger dataset such as PubChem. Our results challenge the common assumption that more pretraining data is always better.

Additional evidence for our claim comes from comparing two variants of ChemBERTa-2 in Fig. 3. One is pretrained on 10 million molecules and the other on 77 million. In our experiments, the one pretrained on the larger dataset did not have better performance. This aligns with the findings of the original paper,[17] which also showed that better performance varied across evaluation datasets, with no consistent advantage for the larger pretraining set, but did not investigate that further or more systematically.

Other previous work has reported mixed results regarding this topic. For example, Ross *et al.*[15] found that larger pretraining datasets led to better performance, but did not assess whether the improvement was statistically significant. Contrarily, a recent preprint by Sultan *et al.*[18] found no significant gains beyond 400 000 molecules. While the exact number is different, our findings agree mostly with the latter finding. We hope our statistically robust analysis helps clarify this ongoing debate and provides solid evidence that it is not necessary to increase pretraining dataset size beyond a relatively low amount, which we found to be one million molecules.

We also investigated the effect of model size, an aspect that has received little attention in previous work but is important because it directly impacts the computational cost of training and deploying models. We found that increasing the number of parameters from 5 million to 15 million led to a clear improvement in performance. However, scaling the model further to 111 million parameters, which matches the size of the original ModernBERT model that exhibits state-of-the-art results in natural language processing,[26] did not lead to further gains. In fact, we observed a slight drop in performance in some tasks. We thereby conclude that masked language modeling on SMILES strings does not require as large model sizes as in natural language processing. Smaller, more efficient, model sizes suffice.

When looking at the model comparison we can find some more evidence for our findings. Our model (15 million parameters) and MoLFormer (45.5 million parameters) performed better than ChemBERTa-2, which has only 3.4 million parameters. ChemBERTa-2's weaker performance may thus partially be explained by its small model size. While other factors may also contribute, our results offer a simple and plausible explanation based on model size.

The comparison between our model and MoLFormer highlights the practical value of our findings even more. Our model achieved better performance, despite having only one-third the number of parameters and being pretrained on a dataset 100 times smaller. This result demonstrates that larger models and massive pretraining datasets are not always necessary for strong downstream performance. We thereby show that our findings can both reduce computational costs and improve model effectiveness. We hope this encourages more inclusive research by showing that strong models can be developed even with limited computational resources, thereby making progress in this domain more accessible to more researchers.

After demonstrating how to improve masked language modeling (MLM) for molecular representations, we now place these findings within the broader context of property and activity prediction in cheminformatics. Our results show that a model pretrained with MLM performs better than the traditional approach based on ECFP4 fingerprints and XGBoost for most tasks. This traditional method remains widely used in the field, but our findings suggest that MLM provides a more effective and flexible foundation for molecular modeling tasks.

This trend persisted for other traditional models using handcrafted fingerprints, with our model consistently outperforming them on the Polaris benchmarks. While leaderboard comparisons have limitations due to the absence of rigorous statistical testing, the consistent ranking of our MLM-based model above traditional ones provides further evidence of its value. Compared to other modern architectures, our MLM model performed consistently well across tasks, although it was never the top-performing model.

This pattern is expected. MLM is a general and task-agnostic pretraining method, which provides a strong foundation for a model. However, pretraining tasks that are more closely aligned with the target downstream task or that incorporate more prior knowledge can lead to further improvements in performance.[16,18,37] Prior studies have shown that combining MLM with other objectives, such as physicochemical property prediction, improves downstream results.[16,18] These combinations have been implemented either through joint pretraining or through later task-specific adaptation.[16,18] We believe that our findings establish a solid foundation for developing these models further. By improving MLM for molecules, we offer a more reliable and efficient starting point that can be further extended with tailored pretraining strategies.

Beyond property prediction, MLM-pretrained models are also increasingly applied in more complex tasks. For instance, they are used to encode molecules in tasks such as drug–target interaction prediction[24] and molecular knowledge graph construction.[38] In these applications, molecular encodings produced by a model pretrained on MLM are integrated with representations of other biological entities to predict interactions. Our model's strong performance and, in particular, its lower computational requirements make it well suited for such computationally demanding applications, where scalability is often a critical concern. Moreover, our findings contribute to this area more broadly by offering practical insights into how to design effective and efficient molecular encoders, which form a key component of these multi-entity modeling approaches, as well as making our small and well performing model *MolEncoder* publicly available.

Conclusion

We systematically investigated key design choices for masked language modeling on molecular SMILES strings and found that (1) higher masking ratios than the widely used 15% consistently improve downstream performance; (2) increasing model or pretraining dataset size beyond moderate levels yields no consistent benefit and can even degrade performance; and (3) pretraining loss is not a reliable predictor of downstream performance. Putting these insights into practice, we trained *MolEncoder*, a 15M-parameter model pretrained on 1 million molecules, which outperforms or matches state-of-the-art models on most benchmark tasks while requiring vastly fewer computational resources. These results challenge common assumptions and provide practical guidelines for building more efficient, accessible molecular language models.

Methods

In this work, we studied different ways to use masked language modeling (MLM) to pretrain chemical language models with the goal of improving performance on common downstream tasks in drug discovery. We focused on small molecules, which we represented using the Simplified Molecular Input Line Entry System (SMILES). Our models were based on an encoder-only transformer architecture, following the design of the recently improved version of the Bidirectional Encoder Representations from Transformers (ModernBERT).[26] This section first introduces the datasets and model configurations used in our experiments, and then outlines the training procedures, evaluation pipeline, and experimental setups used to assess the effects of masking ratio, model size, and dataset scale. An overview of our workflow is shown in Fig. 4.

Evaluation datasets

To evaluate the effect of pretraining strategies on downstream performance, we selected five benchmark prediction tasks that represent a broad range of challenges in drug discovery. All datasets were taken from the Polaris benchmark suite (Python package version 0.13.0). Three of the tasks, introduced by Fang *et al.*,[28] focus on pharmacokinetic properties. The first, *HCLint*, involves predicting human liver microsomal stability. It is reported as intrinsic clearance in mL min^{-1} kg^{-1}. The second, *Solubility*, targets aqueous solubility at pH 6.8, reported in μg mL^{-1}. The third, *Permeability*, assesses membrane permeability based on the MDR1-MDCK efflux ratio, a unitless value. All three targets are provided in logarithmic form in the benchmark suite.

The remaining two tasks cover physicochemical and safety-relevant endpoints. The *Lipophilicity* dataset, originally published by AstraZeneca on ChEMBL,[29] contains the logarithmic values of the distribution coefficient of a compound at pH 7.4. The *CYP* dataset, published by Fluetsch *et al.*[30] involves predicting the inactivation rate constant k_{obs} for time-dependent inhibition of the CYP3A4 enzyme, a major contributor to drug metabolism and potential drug–drug interactions. It includes log-transformed k_{obs} values (originally measured in min^{-1}) based on high-throughput *in vitro* assays with human liver microsomes. Together, these five tasks were selected to provide a diverse and representative set of benchmarks for evaluating the quality of molecular representations on common tasks in drug discovery.

For the comparison between our developed model and other models, we added four additional evaluation datasets. This was computationally feasible since it involved fewer models than the previous experiments. The four datasets are based on data from ChEMBL and include the human plasma protein binding rate as the percentage of a drug bound to plasma proteins in the blood (*hPP Binding*),[29] drug half-life in the body in hours (*Half-life*),[33] inhibitory constant for the delta opioid receptor (*DOR Binding*),[32] and inhibitory constant for the dopamine D3

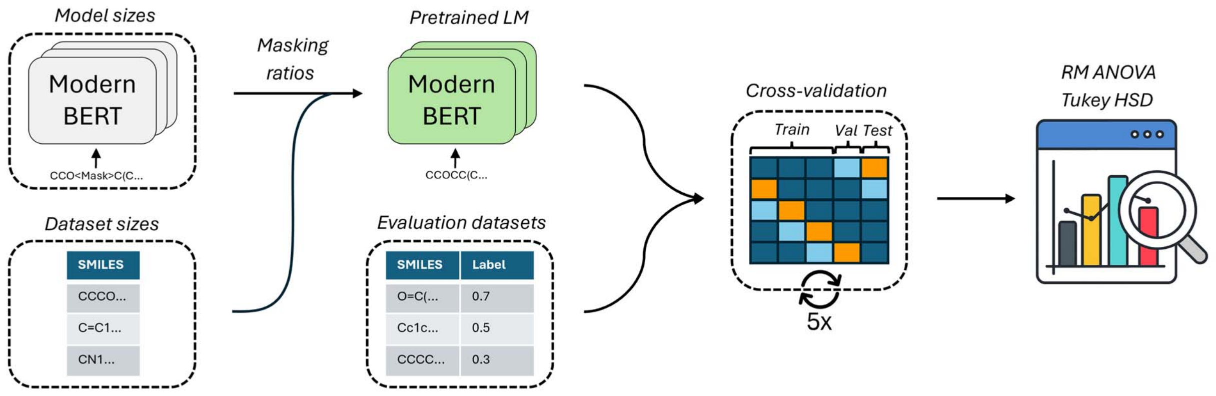

Fig. 4 Overview of the workflow used in this study. We pretrained encoder-only transformer models on SMILES representations of small molecules using a masked language modeling (MLM) objective. Pretraining experiments varied model size, masking ratio, and pretraining dataset size. We included datasets derived from ChEMBL[29] and PubChem.[39] All models were evaluated using a standardized 5 × 5 cross-validation pipeline on five benchmark tasks from Polaris.[34] Evaluation focused on downstream performance across pharmacokinetic, physicochemical, and safety-related endpoints. Statistical comparisons were made using repeated measures ANOVA and a *post hoc* Tukey HSD test if the former was significant.

Table 1 Number of molecules in each training and test split provided by Polaris. We only used the training part for our 5 × 5 cross validation procedure. The test set was only used for the final Polaris leaderboard evaluation

Dataset	Train size	Test size
CYP[30]	16 372	368
Lipophilicity[29]	3360	840
HCLint[28]	2229	575
Permeability[28]	1919	483
Solubility[28]	1578	400

receptor (*D3R Binding*).[32] The latter two datasets have previously been characterized as containing activity cliffs.[32]

All evaluation datasets were provided with predefined training and test splits by the Polaris benchmark suite. For all evaluation experiments involving model selection and comparison, we used only the training sets. This includes all experiments for which we used a 5 × 5 cross-validation procedure, which is described in detail later. The test sets were reserved exclusively for the final leaderboard evaluation in order to ensure an unbiased comparison with previously published models. Since the datasets were already preprocessed and cleaned as part of the Polaris benchmark, we used them as provided without applying any additional data cleaning. The number of molecules in each training and test split is summarized in Table 1.

Pretraining datasets

For pretraining, we used molecular data from ChEMBL (release 35),[29] and PubChem (downloaded in March 2025).[39] We cleaned molecules from both sources to ensure consistency and quality. This included removing isotope labels and explicit hydrogen atoms, sanitizing molecules to correct valences, aromaticity, and bond types, disconnecting metals, normalizing functional groups, reionizing, and assigning stereochemistry. We also removed common salts and solvents, but left mixtures of compounds in the data to promote generalization to different downstream tasks that would require it. All preprocessing steps were carried out using RDKit (release 02.03.2025) and the datasets Python package (version 3.3). To avoid biasing the model toward patterns introduced by canonicalization algorithms, we generated non-canonical (random) SMILES representations. We removed SMILES strings longer than 500 characters and eliminated duplicates, ensuring each SMILES string corresponds to a unique molecule.

For all pretraining datasets, we used a held-out test set consisting of 50 000 randomly selected SMILES strings. These molecules were excluded from all training datasets and used solely for early stopping during pretraining to detect convergence.

After cleaning and removing the molecules used for the test set, the ChEMBL training dataset contained approximately 2.32 million molecules. From this full dataset, we constructed three smaller subsets: one-half (1.2M), one-quarter (0.6M), and one-eighth (0.3M) of ChEMBL. Each subset was created by randomly sampling the corresponding fraction of molecules from the original cleaned ChEMBL training data. In addition, we prepared a large-scale dataset using PubChem. Although PubChem already contains ChEMBL data, we ensured it included the most recent entries by adding the molecules from our ChEMBL dataset to the PubChem dataset and then removing duplicates. The resulting PubChem dataset contained 114 million molecules. We also created a version of the

Table 2 Sizes of the pretraining datasets after cleaning and deduplication

Dataset	Size (in molecules)
PubChem	114.4M
ChEMBL	2.3M
Half of ChEMBL	1.2M
Quarter of ChEMBL	0.6M
Eighth of ChEMBL	0.3M

ChEMBL dataset where SMILES strings were written with all hydrogen atoms explicitly, to investigate the influence of explicit hydrogens. The sizes of all pretraining datasets are summarized in Table 2. We make all cleaned and processed versions of these datasets publicly available on Hugging Face at: https://huggingface.co/collections/fabikru/molencoder-mlm-for-molecules-689def77dadfb1379c34e210.

Model architecture

All models in this work are based on the ModernBERT architecture,[26] an encoder-only transformer designed to improve performance and efficiency over the original BERT model.[9] ModernBERT incorporates a number of architectural updates including rotary positional embeddings, GeGLU activation functions, pre-normalization, and the removal of bias terms from most linear layers. It also integrates Flash Attention[40] for efficient attention computation and supports unpadded sequence processing, which reduces memory and compute overhead by avoiding unnecessary padding.

We implemented three model variants with approximately 5 million, 15 million, and 111 million parameters. We refer to these models by their parameter count. The largest model directly follows the ModernBERT-base configuration. The smaller variants were constructed by scaling down the ModernBERT-base configuration while preserving architectural ratios and following best practices for GPU efficiency. Specifically, hidden sizes and feedforward dimensions were chosen to be divisible by 64. This improves the performance of matrix multiplications on GPUs by enabling better tiling and utilization of tensor cores, which helps speed up computation and reduce overhead during training and inference.[41]

All models use a maximum sequence length of 502 tokens and apply global attention in every layer. Although ModernBERT uses alternating global and local attention to improve long-context efficiency, we opted for full global attention to better capture dependencies in the comparatively relatively short SMILES sequences. We also adopted ModernBERT's unpadding strategy, where padding tokens are removed prior to the embedding layer, allowing batches to be concatenated into a single stream for more efficient processing. This setup, combined with flash attention, significantly improved training and inference speed. Our models use a character level tokenizer that we construct to cover all possible symbols that can occur in SMILES strings, guaranteeing full coverage. It also includes some unused symbols that can be repurposed if needed for some downstream tasks. A summary of all model configurations is provided in Table S2.

Model pretraining

All models were pretrained using the masked language modeling (MLM) objective. We investigated a range of masking ratios across different model and dataset sizes, as detailed in the experiments part of the methods section. Based on the masking ratio, each input token was randomly selected for replacement with the corresponding probability. Of the selected tokens, 80% were replaced with the `[MASK]` token, 10% with a random token, and 10% were left unchanged. Learning rate and batch size were selected using the scaling rules proposed by Li *et al.*,[42] which provide heuristic formulas based on dataset and model size. According to their findings, the optimal batch size depends on the size of the training data, while the optimal learning rate depends on both the dataset and model size.

Although these recommendations were developed for natural language models and differ slightly in optimizer setup, we found them to transfer well to our setting. In our ablation studies we observed no statistically significant differences in downstream performance when comparing models trained with the derived values to those using manually tuned settings (Fig. S15).

To accelerate training further, we used mixed-precision computation and compiled all models using `torch.compile` with the inductor backend in a nightly PyTorch build (2025-04-01), which included the latest compiler and Triton optimizations.[43,44] Training was implemented using the transformers (version 4.5) and accelerate (version 1.7) libraries from Hugging Face. We used the schedule-free variant of the AdamW optimizer, which avoids the need for learning rate scheduling, thereby eliminating the requirement for a predefined training length.[45] This was chosen because we trained each model until convergence based on validation loss rather than a fixed number of steps or tokens.

Since our focus was on downstream task performance, we did not use a separate validation set during pretraining. Instead, each model was pretrained until its performance on the validation set converged. We defined convergence as no decrease in test loss greater than 0.001 for two consecutive evaluations. For all downstream experiments, we used the checkpoint with the lowest test loss. Full details of the pretraining configuration, including training parameters for each model size and dataset, are provided in Table S3.

All models were trained on a single NVIDIA GH200 Grace Hopper Superchip, featuring a 96 GB H100 GPU and an ARM64 CPU architecture with 64 cores. The system was equipped with 432 GB of shared memory and 4 TiB of SSD storage. This configuration allowed for efficient pretraining without the need for distributed computation.

Model finetuning

All models were finetuned on the downstream prediction tasks described earlier. Each evaluation dataset was split into training, validation, and test sets. The validation set was used exclusively for early stopping.

To improve training stability, we applied robust label scaling to the training and validation splits. Specifically, labels were rescaled using the robust scaler transformation (y − median)/IQR, where both the median and interquartile range (IQR) were computed on the training set. This normalization was applied to prevent large gradients at the start of training. For evaluation on the test set, we applied the inverse transformation to the model's predictions to match the original label scale. For models pretrained on SMILES strings that include explicit hydrogen atoms, the input strings were adjusted accordingly before finetuning. Each model was fine-tuned until convergence, defined as no decrease in validation loss (mean squared error) greater than 0.001 over five consecutive epochs.

We used the same optimizer and training infrastructure as for pretraining. However, we fixed the batch size to 64, the learning rate to 8×10^{-4}, and the number of warmup steps to 100. These values were not tuned for performance and were chosen solely for their training stability. The influence of hyperparameter optimization during fine-tuning was examined in a dedicated ablation study (described in detail later), which showed that fine-tuning hyperparameter optimization does not significantly change the model performance (Fig. S16).

Evaluation pipeline

To evaluate the impact of different pretraining configurations, we applied a consistent evaluation procedure across all model size and dataset size combinations. Each model was first pretrained as described above, then tested on all five benchmark datasets. For each dataset, we conducted a 5 × 5 cross-validation procedure, following the recommendations of Ash *et al.*[27] Specifically, this involved five repetitions of 5-fold cross-validation using different data splits between repetitions. In each fold, 60% of the dataset was used for training, 20% for validation, and 20% for testing. Models were finetuned as described in the previous section, and predictions were evaluated on the test split. Our evaluation pipeline is depicted in Fig. 4.

To ensure a fair comparison, all models within each experimental run used identical data splits and folds. This alignment is essential for the statistical assumptions of the repeated measures ANOVA,[46] which we used to compare model performance. When the ANOVA indicated a significant difference, we applied a *post hoc* Tukey HSD test[47] to identify which model pairs differed significantly. All statistical analyses were conducted for four different performance metrics: mean absolute error (MAE), mean squared error (MSE), coefficient of determination (R^2), and Spearman's rank correlation coefficient (ρ).

From the Polaris benchmark suite, we used only the provided training split for each dataset. This portion was further divided into our training, validation, and test subsets as described above. Unless stated otherwise, we did not use the benchmark test sets provided by Polaris in our evaluation pipeline. Preliminary analyses of our evaluation pipeline showed that it was highly robust. As an example, we repeated the entire evaluation with different random seeds for the same 111M parameter model. Our evaluation pipeline showed negligible variation in performance, and our statistical tests correctly confirmed no difference in model performance (see Fig. S1).

Experiments

In the following paragraphs, we describe the setup for each experiment in detail.

Effect of masking ratio

In our first experiment, we investigated how different masking ratios affect the quality of chemical language model pretraining. We used our medium-sized model with 15 million parameters and pretrained it on half of the cleaned ChEMBL dataset, using the procedure described above. We included masking ratios ranging from 10% to 90% in increments of 10%, and additionally included the commonly used 15% masking ratio due to its prevalence in prior work. Each model was evaluated on all five benchmark datasets using the evaluation pipeline described earlier.

To assess whether these findings generalize to different SMILES based representations, we repeated the experiment using the same 15M model architecture but with SMILES strings that include explicit hydrogen atoms. In this setup, models were pretrained on the full ChEMBL dataset with explicit hydrogens, described earlier. The same masking ratios were used, up to a masking ratio of 70%. During evaluation, the model pretrained at 50% masking exhibited unstable behavior under our default finetuning setup. Rather than adjusting hyperparameters for this one case, we excluded it from downstream evaluation to preserve comparability across models. In addition to varying the masking ratio, we also compared models pretrained on SMILES with and without explicit hydrogen atoms using the same procedure.

We visualized the results by plotting the group means for each metric alongside 95% confidence intervals, computed using Tukey's Q critical value as proposed by Hochberg and Tamhane.[31] We chose this approach to summarize variability and highlight statistically meaningful differences across masking ratios in a single plot per dataset and metric, rather than visualizing all pairwise comparisons explicitly.

Influence of model and dataset size

In our second experiment, we investigated how model size and pretraining dataset size influence downstream performance. We considered three model sizes: 5 million, 15 million, and 111 million parameters. Each model was pretrained on one of three dataset sizes: half of ChEMBL, the full ChEMBL dataset, or the combined ChEMBL + PubChem dataset. This resulted in a total of nine model–dataset combinations, all of which were trained and evaluated. For all configurations, we used a fixed masking ratio of 30%, which was identified as the most effective setting in the previous experiment.

As before, each pretrained model was evaluated using the 5 × 5 cross-validation procedure on all five benchmark datasets. To visualize the results, we plotted the mean performance across folds for each configuration and highlighted three categories: the best-performing model, models not statistically significantly worse than the best (based on *post hoc* Tukey HSD tests), and models that performed significantly worse. This provided a structured way to compare the effects of model and dataset scaling on downstream performance.

In a follow-up experiment, we extended the dataset size comparison. We used the 15M model, which showed the strongest overall performance. In addition to the previously tested datasets, we included smaller subsets corresponding to one quarter and one eighth of the ChEMBL to better understand how performance scales in the low-data regime.

To evaluate potential distributional differences between pretraining datasets, we compared key molecular properties and chemical space representations of ChEMBL and PubChem. Both datasets were uniformly subsampled to 100 000 molecules to reduce computational demand while maintaining representative coverage. Molecular fingerprints were computed using RDKit, and dimensionality reduction was performed by principal component analysis (scikit-learn 1.7.2) followed by uniform manifold approximation and projection (UMAP) with 20 nearest neighbors, a minimum distance of 0.4, and two output dimensions (umap-learn 0.5.9.). The first 200 PCA components, explaining approximately 55% of the total variance, were used as input to the UMAP embedding. The same UMAP projection was employed for both pretraining–pretraining (ChEMBL *vs.* PubChem) and pretraining–downstream (ChEMBL or PubChem *vs.* evaluation tasks) comparisons to ensure a consistent embedding space. Quantitative estimates of drug-likeness (QED) and Lipinski's Rule of Five adherence were also computed with RDKit to assess drug-like characteristics.

Correlation between pretraining and finetuning loss

To assess whether pretraining performance is predictive of downstream performance, we analyzed the relationship between pretraining loss and finetuning loss across all models trained in the previous experiments. Models trained on SMILES with explicit hydrogens were excluded from this analysis, as they represent a different pretraining task. For each combination of pretrained model and evaluation dataset, we calculated the mean and standard deviation of the loss across the 5×5 cross-validation.

We then computed the Pearson and Spearman rank correlations between pretraining and finetuning loss for each evaluation dataset to assess whether lower pretraining loss is consistently associated with better downstream performance. This analysis evaluates the extent to which pretraining loss can serve as a proxy for downstream model performance.

Comparison to existing models

To contextualize our results, we compared our model to two state-of-the-art chemical language models: MolFormer-XL by Ross *et al.*[15] and ChemBERTa-2 by Ahmad *et al.*[17] For all models, including ours, we applied the same finetuning strategy described above to ensure a fair comparison. We want to emphasize that this finetuning setup was not optimized for our model's performance but only chosen to provide stable training.

In contrast to the original publications, we did not perform task-specific hyperparameter optimization. While the referenced works tuned hyperparameters on individual datasets, our evaluation relied on a 5×5 cross-validation protocol, which introduces a large number of finetuning runs. Applying hyperparameter optimization within this framework for each model, fold, and dataset exceeded our computational resources.

We used the Hugging Face implementations provided by the respective authors. For MolFormer-XL, this included a version pretrained on approximately 100 million molecules, drawn from a combination of 10% of the ZINC database and 10% of PubChem. A fully pretrained version on the entire dataset was not made publicly available. For ChemBERTa-2, we evaluated both the variants trained on 10M and 77M molecules provided by the authors. All models were evaluated using the same 5×5 cross-validation pipeline applied throughout our study.

We also included a baseline using XGBoost and extended connectivity fingerprints (ECFPs). For calculating the fingerprints, we used RDKit's Morgan fingerprints of radius 2 with 2048 bits, which are an open source analog to the original ECFP4s. We will refer to them as ECFP4 in the main text. For the XGBoost model we used the xgboost python package version 3.0.0. For the XGBoost baseline we included a hyperparameter optimization, since it is computationally more feasible. The hyperparameter space included: number of estimators: 50, 100, 200, 300; maximum tree depth: 4, 6, 8; and learning rate: 0.05, 0.1, 0.2. The best hyperparameters for each 5×5 cross validation fold were selected based on the MSE of the validation set in each the fold.

We also compared our model to the existing Polaris leaderboard.[34] To do so, we trained a version of our model on the full benchmark training split for each dataset. First, we performed 5-fold cross-validation on the training data to determine the average optimal number of epochs (based on the minimum evaluation loss). We then trained on the entire benchmark training set using this fixed number of epochs and generated predictions on the held-out Polaris test sets. These test sets were not used at any stage before in this whole study, ensuring that no information leakage occurred. Results were submitted to the Polaris platform and compared against all leaderboard entries up available at the time of writing (03.07.2025).

Ablation studies on hyperparameter sensitivity

To test whether our results depend on the specific hyperparameters we used, we conducted two ablation studies. In our setup, hyperparameters can affect performance during both pretraining and finetuning. We therefore designed one ablation study for each stage. In both cases, we changed hyperparameters to see whether they affected our results or conclusions.

Pretraining. For the pretraining ablation, we wanted to check whether the formulas by Li *et al.*[42] actually give good hyperparameters in our setting. These formulas were developed for a different context, so we tested them against other values. We selected one model for this comparison: the 15M parameter model pretrained on the full ChEMBL dataset. We trained five versions of this model using different combinations of learning rate and batch size, listed in Table S4. We then evaluated each version on three benchmark datasets using the 5×5 cross-validation procedure described earlier.

Finetuning. In the second ablation, we tested whether our fixed finetuning hyperparameters affected the model performance. We compared a model trained with our standard settings to one trained with optimized settings. For the optimization, we used the Tree-structured Parzen Estimator.[48] It searched for hyperparameters that minimized the validation loss within each fold. We tried 20 different combinations per fold, using the ranges shown in Table S5. Because this procedure is very compute-intensive, we ran it on one evaluation dataset only.

As in all other experiments, we used repeated measures ANOVA and *post hoc* Tukey HSD tests to compare the performance of the model when finetuned using fixed against using optimized hyperparameters.

Author contributions

Fabian Krüger: conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing – original draft, writing – review & editing, visualization. Nicklas Österbacka: conceptualization, methodology, writing – review & editing, supervision. Mikhail Kabeshov: conceptualization, supervision, project administration. Ola Engkvist: conceptualization, writing – review & editing, supervision, project administration, funding acquisition. Igor Tetko: conceptualization, writing – review & editing, supervision, project administration, funding acquisition.

Conflicts of interest

The authors declare that there are no conflicts of interest.

Data availability

Our model, *MolEncoder*, is publicly available on Hugging Face at **https://huggingface.co/fabikru/MolEncoder**. The version of the model at the time of the publication is available at **https://doi.org/10.57967/hf/6263**. A curated Hugging Face collection that includes the model together with all pretraining datasets can be found at **https://huggingface.co/collections/fabikru/molencoder-mlm-for-molecules-689def77dadfb1379c34e210**. The individual dataset links are: **https://doi.org/10.57967/hf/6265**, **https://doi.org/10.57967/hf/6266**, **https://doi.org/10.57967/hf/6267**, **https://doi.org/10.57967/hf/6268**, **https://doi.org/10.57967/hf/6269**, and **https://doi.org/10.57967/hf/6270**. The corresponding GitHub repository (**https://github.com/FabianKruger/MolEncoder**) provides the complete source code for this study, as well as ready-to-use examples for downloading the model, fine-tuning it on custom data, and performing predictions. The state of the repository at the publication of the paper is available at **https://doi.org/10.5281/zenodo.16894614**. All benchmark datasets employed in this study are accessible *via* Polaris (**https://doi.org/10.5281/zenodo.13652587**).

Supplementary information is available. See DOI: **https://doi.org/10.1039/d5dd00369e**.

Acknowledgements

This study was partially funded by the Horizon Europe funding programme under the Marie Skłodowska-Curie Actions Doctoral Networks grant agreement "Explainable AI for Molecules – AiChemist", no. 101120466. A preliminary version of parts of this work appeared as an extended abstract at the 34th International Conference on Artificial Neural Networks (ICANN), AI in Drug Discovery workshop, in accordance with its partnership with Digital Discovery.[49] NÖ is a fellow of the AstraZeneca Postdoctoral Research Programme.

References

1 I. Kola and J. Landis, Can the pharmaceutical industry reduce attrition rates?, *Nat. Rev. Drug Discovery*, 2004, **3**(8), 711–716.
2 U.S. Food and Drug Administration, Roadmap to reducing animal testing in preclinical safety studies, 2025.
3 S. Dara, S. Dhamercherla, S. Singh Jadav, C. H. Madhu Babu and J. A. Mohamed, Machine learning in drug discovery: a review, *Artif. Intell. Rev.*, 2022, **55**(3), 1947–1999.
4 E. N. Muratov, J. Bajorath, R. P. Sheridan, I. V. Tetko, D. Filimonov, V. Poroikov, T. I. Oprea, I. I. Baskin, A. Varnek, A. Roitberg, *et al.*, Qsar without borders, *Chem. Soc. Rev.*, 2020, **49**(11), 3525–3564.
5 K.-K. Mak, Y.-H. Wong, and M. Rao Pichika, Artificial intelligence in drug discovery and development, *Drug discovery and evaluation: safety and pharmacokinetic assays*, 2024, pp. 1461–1498.
6 P. Karpov, G. Godin and I. V. Tetko, Transformer-cnn: Swiss knife for qsar modeling and interpretation, *J. Cheminf.*, 2020, **12**, 17.
7 B. Min, H. Ross, E. Sulem, A. P. B. Veyseh, T. H. Nguyen, O. Sainz, E. Agirre, I. Heintz and D. Roth, Recent advances in natural language processing via large pre-trained language models: A survey, *ACM Comput. Surv.*, 2023, **56**(2), 1–40.
8 A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, Attention is all you need, *Adv. Neural Inf. Process. Syst.*, 2017, vol. 30.
9 J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.
10 S. Wang, Y. Guo, Y. Wang, H. Sun, and J. Huang, 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, BCB '19*, Association for Computing Machinery, New York, NY, USA, 2019, pp. 429–436.
11 J. Li and X. Jiang, Mol-bert: An effective molecular representation with bert for molecular property prediction, *Wireless Commun. Mobile Comput.*, 2021, **2021**(1), 7181815.

12 B. Li, M. Lin, T. Chen and L. Wang, Fg-bert: a generalized and self-supervised functional group-based molecular representation learning framework for properties prediction, *Briefings Bioinf.*, 2023, **24**(6), bbad398.

13 H. Kim, J. Lee, S. Ahn and J. R. Lee, A merged molecular representation learning for molecular properties prediction with a web-based service, *Sci. Rep.*, 2020, **11**, 11028.

14 Y. Liu, R. Zhang, T. Li, J. Jiang, J. Ma and P. Wang, Molrope-bert: An enhanced molecular representation with rotary position embedding for molecular property prediction, *J. Mol. Graphics Modell.*, 2022, **118**, 108344.

15 J. Ross, B. Belgodere, V. Chenthamarakshan, I. Padhi, Y. Mroueh and P. Das, Large-scale chemical language representations capture molecular structure and properties, *Nat. Mach. Intell.*, 2022, **4**(12), 1256–1264.

16 B. Fabian, T. Edlich, H. Gaspar, M. H. S. Segler, J. Meyers, M. Fiscato and M. Ahmed, Molecular representation learning with language models and domain-relevant auxiliary tasks, *arXiv*, 2020, preprint, arXiv:2011.13230, DOI: 10.48550/arXiv.2011.13230.

17 W. Ahmad, E. Simon, S. Chithrananda, G. Grand and B. Ramsundar, Chemberta-2: Towards chemical foundation models, *arXiv*, 2022, preprint, arXiv:2209.01712, DOI: 10.48550/arXiv.2209.01712.

18 A. Sultan, M. Rausch-Dupont, S. Khan, O. Kalinina, A. Volkamer and D. Klakow, Transformers for molecular property prediction: Domain adaptation efficiently improves performance, *arXiv*, 2025, preprint, arXiv:2503.03360, DOI: 10.48550/arXiv.2503.03360.

19 X.-C. Zhang, C. Wu, Z.-J. Yang, Z. Wu, J. Yi, C.-Y. Hsieh, T. Hou and D. Cao, Mg-bert: leveraging unsupervised atomic representation learning for molecular property prediction, *Briefings Bioinf.*, 2021, **22**(6), bbab152.

20 Z. Wu, D. Jiang, J. Wang, X. Zhang, H. Du, L. Pan, C.-Y. Hsieh, D. Cao and T. Hou, Knowledge-based bert: a method to extract molecular features like computational chemists, *Briefings Bioinf.*, 2022, **23**(3), bbac131.

21 D. Weininger, Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules, *J. Chem. Inf. Comput. Sci.*, 1988, **28**(1), 31–36.

22 T. Tran and C. Ekenna, Molecular descriptors property prediction using transformer-based approach, *Int. J. Mol. Sci.*, 2023, **24**, 11948.

23 N. Brandes, D. Ofer, Y. Peleg, N. Rappoport and M. Linial, Proteinbert: a universal deep-learning model of protein sequence and function, *Bioinformatics*, 2022, **38**(8), 2102–2110.

24 H. Yang, Z. Feng, X. Song, X.-J. Wu and J. Kittler, Mmdg-dti: Drug-target interaction prediction via multimodal feature fusion and domain generalization, *Pattern Recogn.*, 2025, **157**, 110887.

25 A. Wettig, T. Gao, Z. Zhong, and D. Chen, Should you mask 15% in masked language modeling?, *arXiv*, 2022, preprint, arXiv:2202.08005, DOI: 10.48550/arXiv.2202.08005.

26 B. Warner, A. Chaffin, B. Clavié, O. Weller, O. Hallström, S. Taghadouini, A. Gallagher, R. Biswas, F. Ladhak, T. Aarsen, *et al.*, Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference, *arXiv*, 2024, preprint, arXiv:2412.13663, DOI: 10.48550/arXiv.2412.13663.

27 J. R. Ash, C. Wognum, R. Rodríguez-Pérez, M. Aldeghi, A. C. Cheng, D.-A. Clevert, O. Engkvist, C. Fang, D. J. Price, J. M. Hughes-Oliver, *et al.*, Practically significant method comparison protocols for machine learning in small molecule drug discovery, *ChemRxiv*, 2024, preprint, DOI: 10.26434/chemrxiv-2024-6dbwv-v2.

28 C. Fang, Y. Wang, R. Grater, S. Kapadnis, C. Black, P. Trapa and S. Sciabola, Prospective validation of machine learning algorithms for absorption, distribution, metabolism, and excretion prediction: An industrial perspective, *J. Chem. Inf. Model.*, 2023, **63**(11), 3263–3274.

29 B. Zdrazil, E. Felix, F. Hunter, E. J. Manners, J. Blackshaw, S. Corbett, M. de Veij, H. Ioannidis, D. Mendez Lopez, J. F. Mosquera, *et al.*, The chembl database in 2023: a drug discovery platform spanning multiple bioactivity data types and time periods, *Nucleic Acids Res.*, 2024, **52**(D1), D1180–D1192.

30 A. Fluetsch, M. Trunzer, G. Gerebtzoff and R. Rodríguez-Pérez, Deep learning models compared to experimental variability for the prediction of cyp3a4 time-dependent inhibition, *Chem. Res. Toxicol.*, 2024, **37**(4), 549–560.

31 Y. Hochberg and A. C. Tamhane, *Multiple Comparison Procedures*, John Wiley & Sons, Inc., 1987.

32 D. Van Tilborg, A. Alenicheva and F. Grisoni, Exposing the limitations of molecular machine learning with activity cliffs, *J. Chem. Inf. Model.*, 2022, **62**(23), 5938–5951.

33 R. Scott Obach, F. Lombardo and N. J. Waters, Trend analysis of a database of intravenous pharmacokinetic parameters in humans for 670 drug compounds, *Drug Metab. Dispos.*, 2008, **36**(7), 1385–1405.

34 C. Wognum, J. R. Ash, M. Aldeghi, R. Rodríguez-Pérez, F. Cheng, A. C. Cheng, D. J. Price, D.-A. Clevert, O. Engkvist and W. P. Walters, A call for an industry-led initiative to critically assess machine learning for real-world drug discovery, *Nat. Mach. Intell.*, 2024, **6**(10), 1120–1121.

35 I. V. Tetko, R. van Deursen and G. Godin, Be aware of overfitting by hyperparameter optimization, *J. Cheminf.*, 2024, **16**(1), 139.

36 A. Lin, D. Horvath, V. Afonina, G. Marcou, J.-L. Reymond and A. Varnek, Mapping of the available chemical space versus the chemical universe of lead-like compounds, *ChemMedChem*, 2018, **13**(6), 540–554.

37 M. Sypetkowski, F. Wenkel, F. Poursafaei, N. Dickson, K. Suri, P. Fradkin and D. Beaini, On the scalability of gnns for molecular graphs, *Adv. Neural Inf. Process. Syst.*, 2024, **37**, 19870–19906.

38 T. L. Hoang, M. L. Sbodio, M. M. Galindo, M. Zayats, R. Fernández-Díaz, V. Valls, G. Picco, C. Berrospi, and V. López, Knowledge enhanced representation learning for drug discovery, in *AAAI Conference on Artificial Intelligence*, 2024.

39 S. Kim, J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B. A. Shoemaker, P. A. Thiessen, Bo Yu, *et al.*, Pubchem 2025 update, *Nucleic Acids Res.*, 2025, **53**(D1), D1516–D1525.

40 T. Dao, Flashattention-2: Faster attention with better parallelism and work partitioning, *arXiv*, 2023, preprint arXiv:2307.08691, DOI: **10.48550/arXiv.2307.08691**.

41 NVIDIA, Matrix Multiplication Background User's Guide, 2023, Accessed: 2025-06-10.

42 H. Li, W. Zheng, Q. Wang, H. Zhang, Z. Wang, S. Xuyang, Y. Fan, S. Zhou, X. Zhang, and D. Jiang, Predictable scale: Part i–optimal hyperparameter scaling law in large language model pretraining, *arXiv*, 2025, preprint, arXiv:2503.04715, DOI: **10.48550/arXiv.2503.04715**.

43 A. Paszke, Pytorch: An imperative style, high-performance deep learning library, *arXiv*, 2019, preprint, arXiv:1912.01703, DOI: **10.48550/arXiv.1912.01703**.

44 P. Tillet, H.-T. Kung, and D. Cox, Triton: an intermediate language and compiler for tiled neural network computations, in *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*, 2019, pp. 10–19.

45 A. Defazio, X. Yang, K. Ahmed, K. Mishchenko, H. Mehta and A. Cutkosky, The road less scheduled, *Adv. Neural Inf. Process. Syst.*, 2024, **37**, 9974–10007.

46 E. R. Girden, *ANOVA: Repeated measures*, Sage, 1992.

47 J. W. Tukey, Comparing individual means in the analysis of variance, *Biometrics*, 1949, 99–114.

48 J. Bergstra, R. Bardenet, Y. Bengio and B. Kégl, Algorithms for hyper-parameter optimization, *Adv. Neural Inf. Process. Syst.*, 2011, vol. 24.

49 F. P. Krüger, N. Österbacka, M. Kabeshov, O. Engkvist, and I. Tetko, Molencoder: Improved masked language modeling for molecules, in *Artificial Neural Networks and Machine Learning. ICANN 2025 International Workshops and Special Sessions*, ed. W. Senn, M. Sanguineti, A. Saudargiene, I. V. Tetko, A. E. P. Villa, V. Jirsa, and Y. Bengio, Springer Nature, Switzerland, Cham, 2026, pp. 42–44.