Supporting Information
Ligand-Based Compound Activity Prediction via Few-Shot Learning

Peter Eckmann,* Jake Anderson,* Rose Yu,* and Michael K. Gilson*

E-mail: peckmann@ucsd.edu; jta002@ucsd.edu; roseyu@ucsd.edu; mgilson@health.ucsd.edu

Implementation details

Unless specifically stated, `FS-CAP` and all other baselines were trained with the same molecular representation (2048-bit Morgan fingerprints with a radius of 3). While we tried altering both the length and radius parameters of the Morgan fingerprint, we found that it did not change the performance of `FS-CAP` much. For `FS-CAP` and all baseline methods, we tuned hyperparameters once for BindingDB test set performance discussed in the first section of the Results using 8 context compounds, except without limiting the context activity range. We used the same hyperparameters for all other datasets and subsequent tasks. For all methods, the reported model performance in each experiment is measured after 2^{27} query molecules had been seen in training, or until the performance on the relevant task stopped improving (i.e. early stopping). For each method, hyperparameter tuning was done via a random search with 20 trials. The hyperparameters that we searched are shown below for each method. All models were trained on a server with 8 NVIDIA GTX 3080 GPUs.

FS-CAP

`FS-CAP` was implemented in PyTorch. We used an Adam optimizer with a base learning rate of 10^{-5} and 128 steps for learning rate warmup, and then cosine annealed the learning rate to 0 over all training steps. We used dropout ($p = 0.1$) and batch normalization following each layer in the predictor network (except in the last 2 layers), while the encoder networks used neither.

Hyperparameters: `learning rate, batch size, encoding dim, n layers, mlp width`

Tanimoto similarity

We used 2048-bit Morgan fingerprints with a radius of 2 for the calculation of Tanimoto similarity, which was chosen by conducting a sweep of different length and radius parameters and picking the one with the highest performance on the BindingDB test set with 8 context compounds. When using multiple context compounds, we calculate the Tanimoto similarity between each context compound and all query compounds, but only use the highest similarity context compound for each query compound. This is because if a query compound is similar to one of, but not all, the known actives (the context set), it is still presumed to be active.

MolBERT + ANP

We used the pretrained MolBERT model available from `https://github.com/BenevolentAI/MolBERT` to encode SMILES strings into a 768-dimensional vector. We then used this featurizer (which was not made trainable) in an attentive neural process architecture to represent the context and query features, x_i and x_*, respectively.[1] We re-implemented the attentive neural process architecture in PyTorch, following the original paper[1] and their published code (`https://github.com/deepmind/neural-processes/blob/master/attentive_neural_process.ipynb`) as closely as possible. We trained the model using an Adam optimizer with a base learning rate of 10^{-5} and 128 steps for learning rate warmup,

and then cosine annealed the learning rate to 0 over all training steps.

Hyperparameters: `learning rate, batch size, num attention heads, encoding dim, decoder layers, mlp width`

NGGP

We used the official PyTorch implementation of NGGP available at `https://github.com/gmum/non-gaussian-gaussian-processes`. Using the existing code available for the `QMUL` dataset, we modified the datalaoders for our task by outputting 2048-bit Morgan fingerprints. We trained only one model on BindingDB because the size of the context set is only relevant at test time. We also expanded the `MLP2` model used in the code to more layers, so the number of parameters was about equivalent to other baselines.

Hyperparameters: `all lr, meta batch size, update batch size, noise, cnf dims, mlp layers, nonlinearity, batch norm, mlp width`

MetaNet

We adapted the Chainer code provided in the official MetaNet implementation (`https://bitbucket.org/tsendeemts/metanet/src/master/`) to PyTorch. Most of the architectural choices were kept the same as the original code, although we changed each Block network to include two 2048-wide linear layers with ReLU nonlinearities so that the entire model used about the same number of parameters as other baselines. We trained the model using an Adam optimizer with a base learning rate of 10^{-5} and 128 steps for learning rate warmup, and then cosine annealed the learning rate to 0 over all training steps.

Hyperparameters: `learning rate, num blocks, mlp width, hidden dim, batch size`

Meta-MGNN

We used the official Meta-MGNN code available at `https://github.com/zhichunguo/Meta-MGNN`, and modified it to produce continuous, as opposed to binary, activity predictions. To do this, we swapped the both the total and self-supervised loss to MSE, and removed the sigmoid and thresholding operation at the final layer. All other design choices in the model, such as the graph encoding of the molecules, were kept the same.

Hyperparameters: `learning rate, batch size, graph pooling, num layers`

MAML

We used the MAML implementation in the learn2learn library.[2] The base model was a simple multilayer perceptron that takes a 2048-bit Morgan fingerprint as input and produces a single scalar output, which is the activity value prediction. As in the original MAML paper,[3] we used an SGD optimizer with a constant learning rate, as well as applied dropout with $p = 0.1$ after all layers of the network during training.

Hyperparameters: `learning rate, maml learning rate, batch size, n layers, mlp width`

MetaDTA

Since there was no available implementation of MetaDTA, we re-implemented it in PyTorch. For information on the specifics of the MetaDTA architecture, see Section 3.2 of Lee et al.[4]. The context and query inputs, $\mathbf{x}_i$ and $\mathbf{x}_q$ as described in the paper, were represented with 2048-bit Morgan fingerprints, and the context target y_i used the same scalar activity representation as `FS-CAP`. As it does not specify in the original paper, similarly to `FS-CAP`, we used an Adam optimizer with a base learning rate of 10^{-5} and 128 steps for learning rate warmup, and then cosine annealed the learning rate to 0 over all training steps.

Hyerparameters: `learning rate, batch size, encoding dim, n layers, mlp width,`

`attention heads`. We used the same number of layers, `n_layers`, for the query and context set embedding networks, and the decoder network. We also used the same number of attention heads, `attention_heads`, for the multi-head cross and self-attention components of the model.

Dataset details

BindingDB

BindingDB data was downloaded from the TSV file available at (`https://www.bindingdb.org/rwd/bind/chemsearch/marvin/SDFdownload.jsp?download_file=/bind/downloads/BindingDB_All_202403_tsv.zip`). We extracted the K_d, K_i, IC50, or EC50 values, whichever was present, for each compound in the dataset. We used such a broad range of different activity types because all values are similarly determined by an underlying binding mechanism, it increased the amount of data we can train on, and allowed the trained models to generalize to both target-based and phenotypic data types. When activity was expressed as an upper or lower bound, we took the bound itself as the known activity. To reduce outlier activity values, we also clipped activity values with pActivity > 11.5 or < 2.5, as values surpassing those limits were rare. Then, we excluded all targets that include less than 10 measured compounds. We also excluded very small or very large molecules, defined as fewer than 10 atoms or more than 70.

To cluster the target sequences, we used the mmseqs2[5] library's `easy-cluster` mode with the following parameters: `-min-seq-id 0.2 -c 0.8 -cov-mode 0`. To form our dataset, we simply took all targets in BindingDB that were present in the outputted "representative sequences" from mmseqs2 and excluded all others.

PubChemHTS

To construct this dataset, we started with a list of Assay IDs (AIDs) obtained from the search function at `https://pubchem.ncbi.nlm.nih.gov/`, and then downloaded the top 100 AIDs with the highest number of tested substances with "BioAssay Type" equal to "Screening" and a linked "Protein Target" section in the "BioAssay Record." The quantitative data we used for the context compounds was obtained via the protein's "Chemicals and Bioactivities" section in PubChem, which combines results from assays (including quantitative assays) that have the same protein target. We excluded all proteins, and therefore assays, with less than 10 tested compounds with continuous activity values. After obtaining the context compounds, we downloaded the datatable for each assay, which contained Compound IDs (which were linked to SMILES strings, to be used as query compounds in our tests, via the PubChem API) and binary compound activity classifications ("Active" or "Inactive" in the datatable file, to be used for computing ROC-AUC and enrichment scores). We also excluded very small or very large molecules, defined as fewer than 10 atoms or more than 70.

Cancer Cell Line Encyclopedia

As previously mentioned, we used the dataset reported in Table S11 of Barretina et al. [6], and extracted IC50 measurements for each drug measured against each cell line. We excluded compounds with less than 10 or more than 70 atoms, and cell lines with less than 10 drugs with measured activity. We also excluded all compound-activity pairs if there was no continuous activity value reported.

Table 1: **Model performance around activity cliffs**. The left column shows the average $RMSE_{cliff}$[7] across all BindingDB test set targets. The right column shows the same, except RMSE is computed across all test set compounds (not just cliff compounds). 8 context compounds were used.

Method	Mean per-target $RMSE_{cliff}$	Mean per-target RMSE
Meta-MGNN	1.82	1.23
MetaDTA	1.57	1.01
MAML	1.32	0.84
FS-CAP	**1.29**	**0.82**

Additional results

Activity cliff performance

Activity cliffs occur in cases where chemically similar compounds have very different activities, and are a well-known failure case of many machine learning-based chemical property predictors.[8] To evaluate the predictive power of FS-CAP and baseline methods around activity cliffs, we used the $RMSE_{cliff}$ proposed by van Tilborg et al.[7]. $RMSE_{cliff}$ measures the root mean square error (RMSE) between experimental and predicted activity on a subset of the test set deemed to be "cliff molecules." A pair of test set molecules are deemed cliff molecules if they have similarity above some threshold and a difference in activity above some threshold. While the thresholds used by van Tilborg et al.[7] were 0.9 for similarity and 10x for activity difference, none of our test set molecules were deemed cliff compounds under these thresholds. Instead, we used 0.4 for the similarity and 3x for the activity difference thresholds to ensure enough compounds were deemed cliff compounds. 275 of 9419 (3%) of all test compounds were found to belong to a pair of cliff compounds using these new thresholds.

Table 1 reports the results for FS-CAP and baseline methods on the activity cliff test. Using the already trained models, we measured the $RMSE_{cliff}$ (in pActivity) separately for each target in the BindingDB test set, and then averaged the RMSEs across the targets (left column). 8 context compounds were used, and they were chosen randomly. For reference, we also measured the RMSE across all test set compounds, not just the cliff compounds

(right column). As shown, our method achieves the lowest $\text{RMSE}_{\text{cliff}}$ among the chosen baselines, suggesting that it is most capable at distinguishing between two chemically similar compounds with a large activity difference.

References

(1) Kim, H.; Mnih, A.; Schwarz, J.; Garnelo, M.; Eslami, A.; Rosenbaum, D.; Vinyals, O.; Teh, Y. W. Attentive neural processes. *arXiv.org, e-Print Archive* **2019**,

(2) Arnold, S. M. R.; Mahajan, P.; Datta, D.; Bunner, I.; Zarkias, K. S. learn2learn: A Library for Meta-Learning Research. *arXiv.org, e-Print Archive* **2020**,

(3) Finn, C.; Abbeel, P.; Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. *International Conference on Machine Learning* **2017**, 1126–1135.

(4) Lee, E.; Yoo, J.; Lee, H.; Hong, S. MetaDTA: Meta-learning-based drug-target binding affinity prediction. *ICLR Machine Learning for Drug Discovery Workshop* **2022**,

(5) Steinegger, M.; Söding, J. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat. Biotechnol.* **2017**, *35*, 1026–1028.

(6) Barretina, J.; Caponigro, G.; Stransky, N.; Venkatesan, K.; Margolin, A. A.; Kim, S.; Wilson, C. J.; Lehár, J.; Kryukov, G. V.; Sonkin, D., et al. The Cancer Cell Line Encyclopedia enables predictive modelling of anticancer drug sensitivity. *Nature* **2012**, *483*, 603–607.

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

(8) Deng, J.; Yang, Z.; Wang, H.; Ojima, I.; Samaras, D.; Wang, F. A systematic study of key elements underlying molecular property prediction. *Nat. Commun.* **2023**, *14*, 6395.