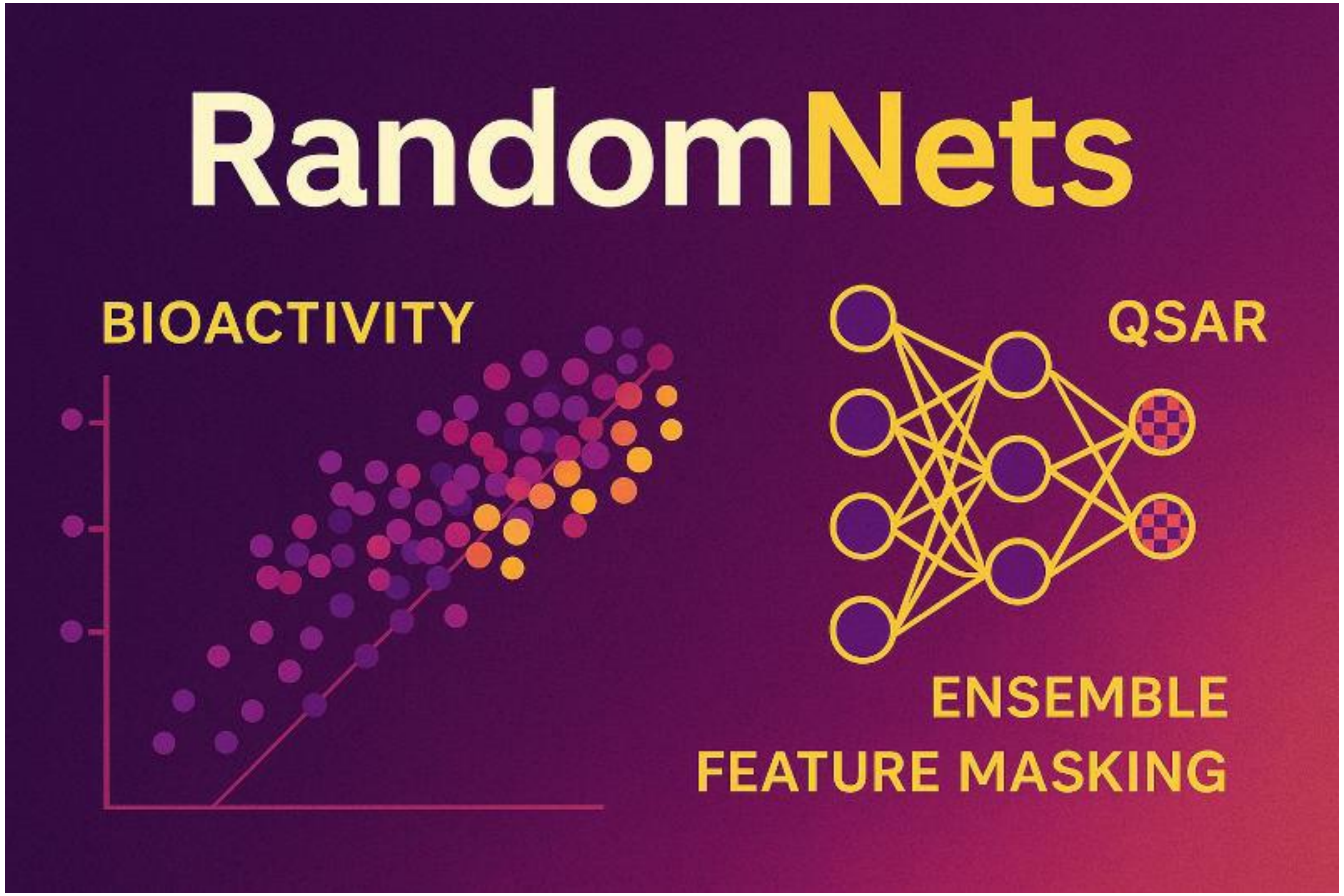
RandomNets – Implicit and Vectorized Ensemble Neural Networks



Esben Jannik Bjerrum

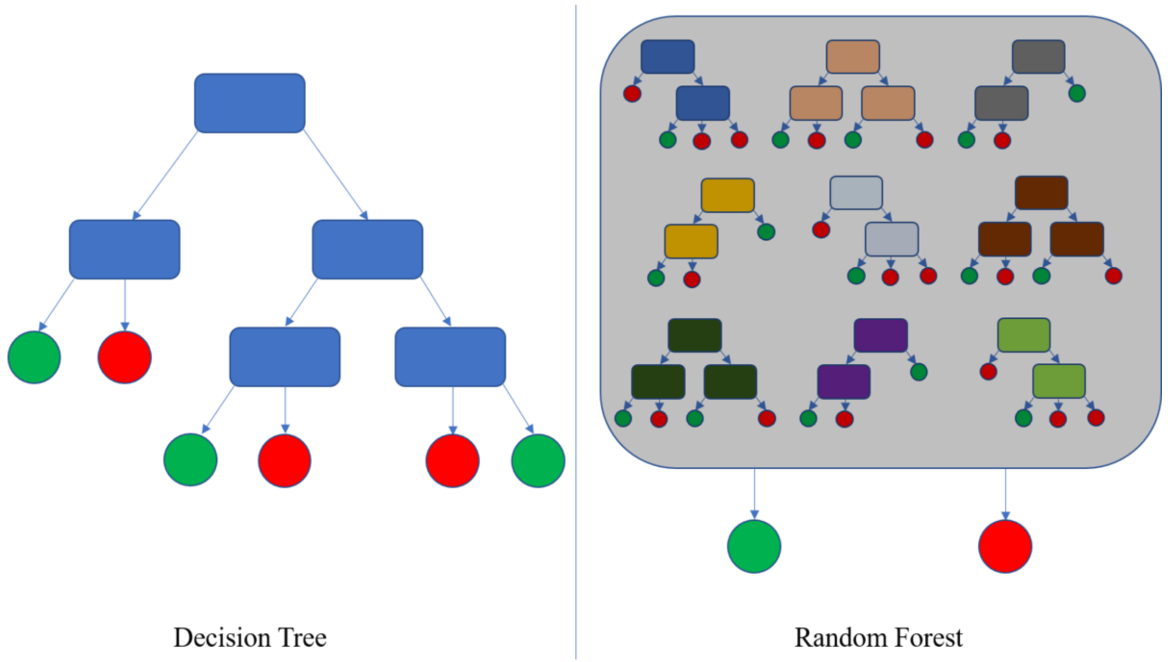
RDKit UGM

September 2025

Prague

Model Ensemble – Enhancing Model Performance

- **Averaging prediction** from multiple, slightly different, models is a known trick to enhance predictivity
- Well-known models such as **Random Forest** uses ensembles of decision trees
- **Bagging**: Bootstrap Aggregation
  - Sampling the dataset with replacement gives different datasets and thus models
- **Random subspace method** (feature bagging):
  - Models are presented with different subsets of features to induce differences.
- **Efficient approach**: Tox24 blind challenge showed top contenders to be ensembles or consensus models[1]



Weak model

Strong ensemble

Cirino, Thalita, Luis Pinto, Mateusz Iwan, et al. 'Consensus Modeling Strategies for Predicting Transthyretin Binding Affinity from Tox24 Challenge Data'. Chemical Research in Toxicology 38, no. 6 (2025): 1061–71. https://doi.org/10.1021/acs.chemrestox.5c00018.

Manual or Loop Approaches

- Often ensembles approached with looping or manually training different models and later averaging
- Efforts scale linearly with number of models
- Can be a hassle to manage: Training, saving, loading, prediction, averaging

  …. often need to develop some sort of framework if it doesn’t exist already
- But is maximum flexible….

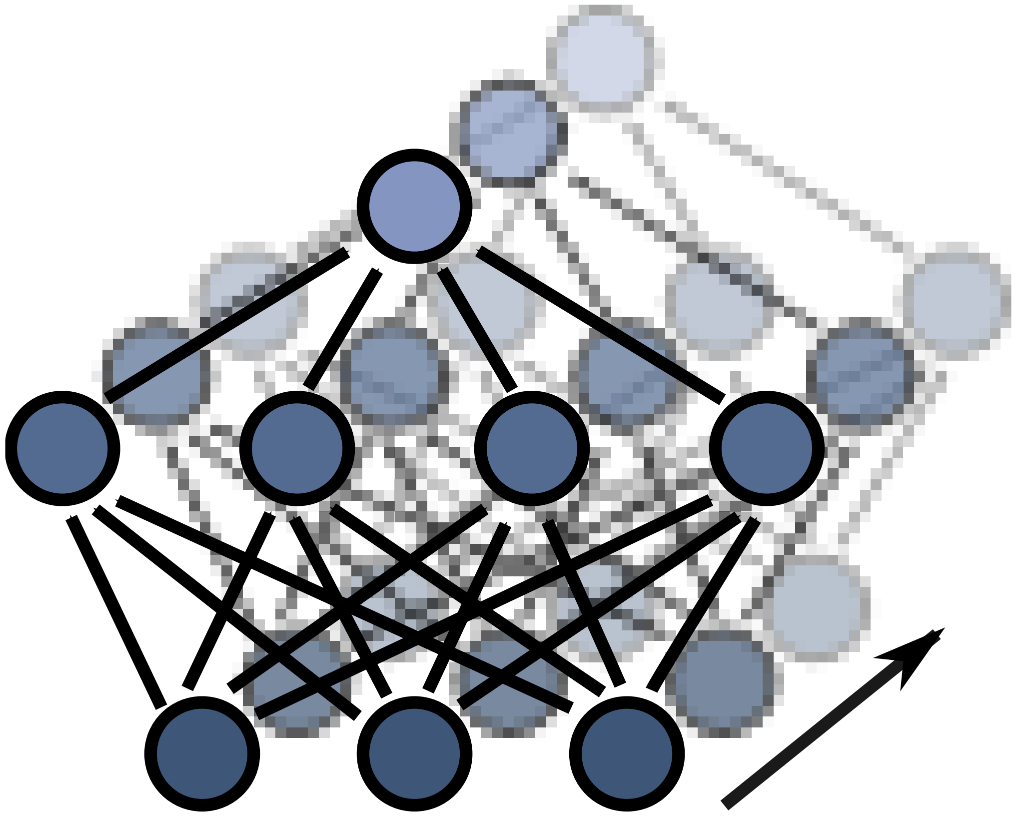
Vectorization for *Speed* 🏃‍♀️ 🏃‍♀️ 🏃‍♀️

- Vectorization of problems is a known approach to avoid slow for-loops (especially in Python)

  => “Simply” add a new dimension to the tensors going through the neural network.

- ```
  fp_reshaped =
  fp.unsqueeze(2).expand(
              -1, -1, n_ensemble_sel
          )   # -> samples, fp_size,
  n_nns_sel
  ```

- Then of course “just” adapt the network layers as well …..


  ```

How to Create Different Models Within the Neural Network?

- **“The Blind Men and the Elephant** 🐘” TitthaSutta, Udāna 6.4, Khuddaka Nikaya Ca. 500 BC.

  => Each man describe only a part of the elephant, by talking they discover they only see part of the truth

  (in other versions they start fighting over who’s lying 🤼)

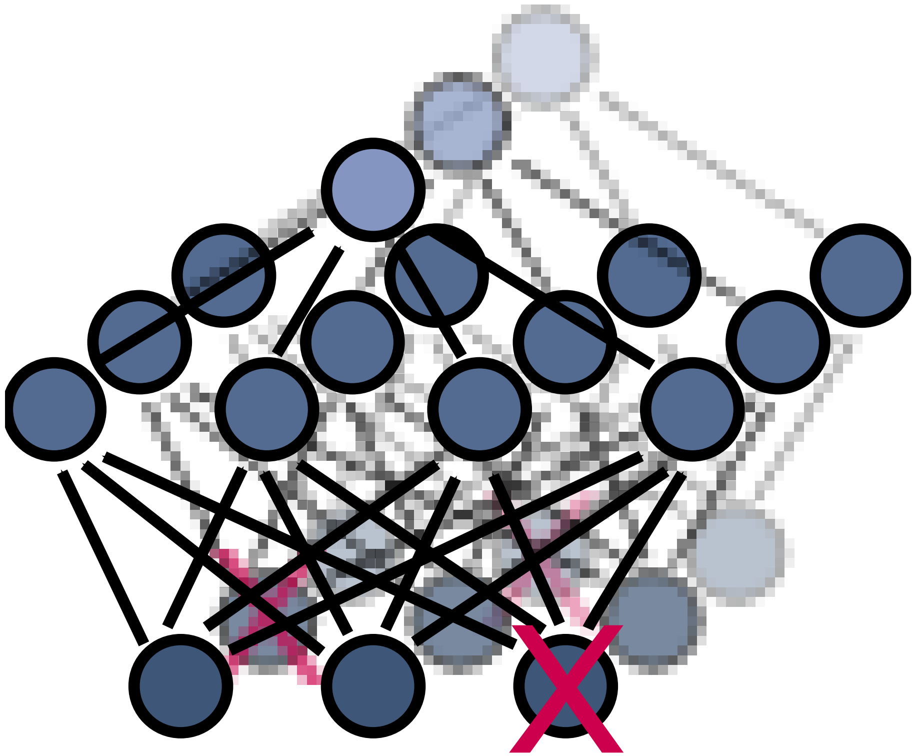
- Similar to Random Forests we can do:
- **input masking** (Random Subspace Method)
  - Was easy to implement!
- Use only some samples to some members (**subsampling**, so not 1:1 to bagging)
  - Was way more challenging to implement!

Implicit Ensembling Saves Memory (and Worked Better)

- Ran **out of memory** on my limited GPU with even low numbers of internal models (GTX 1060 with 6GB VRAM ~2017)
- Implemented **weight sharing** via Conv1D layers, turning the ensemble implicit
- Fixed **input mask**
- Output masking (**sample masking**) where each sample was only allowed backpropagation for predefined output neurons.

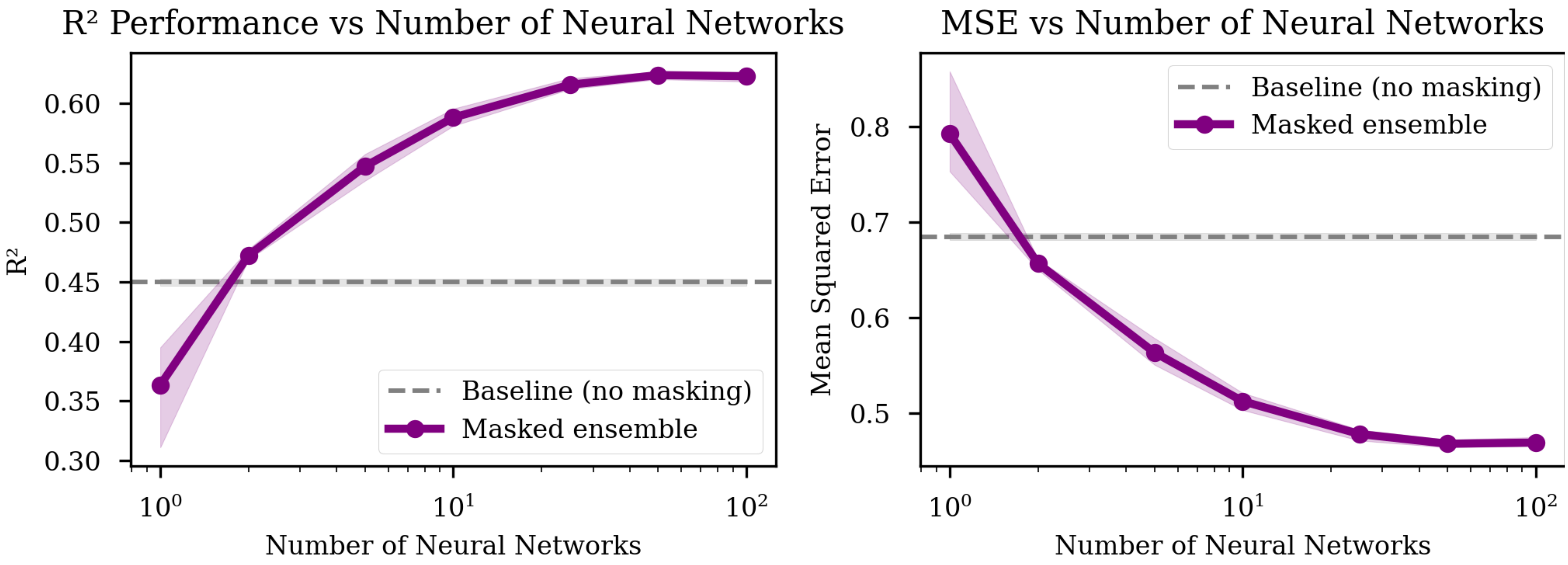
- Features were Morgan Fingerprints Radius 2
- Any Scikit-Mol transformer/Pipeline can be used.

Open-Source Cheminformatics and Machine Learning



I have no more excuses

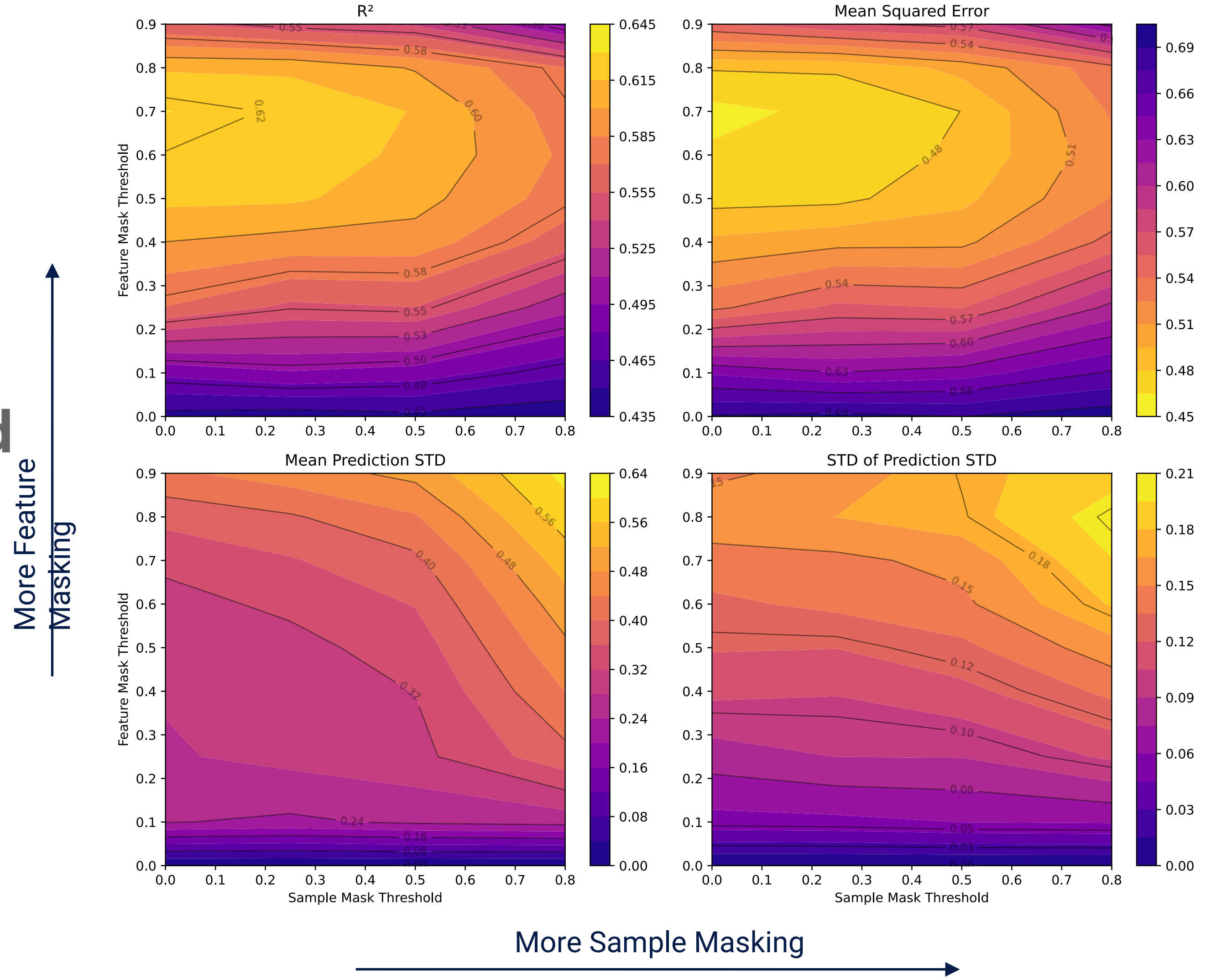
Ensembling leads to better model predictions



Already two members in ensemble is better than single network with no masking.
Effect plateaus after 50 implicit networks. Tested on ExcapeDB SLC6A3 dataset.

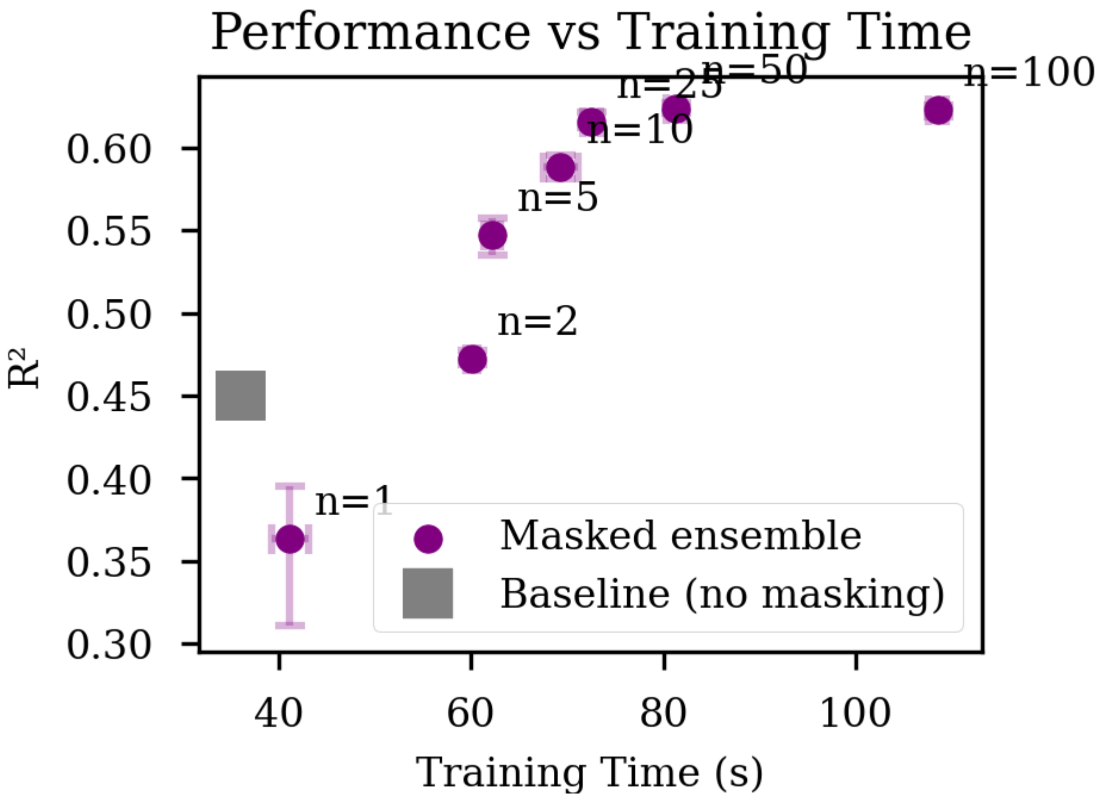
Feature masking >> Sample-masking

- **Feature-masking** around 0.7 **improved** ensemble performance the most.
- **Sample-masking** only **decreased** performance
- (**Inverse proportional** to difficulty of implementation, 🤷)

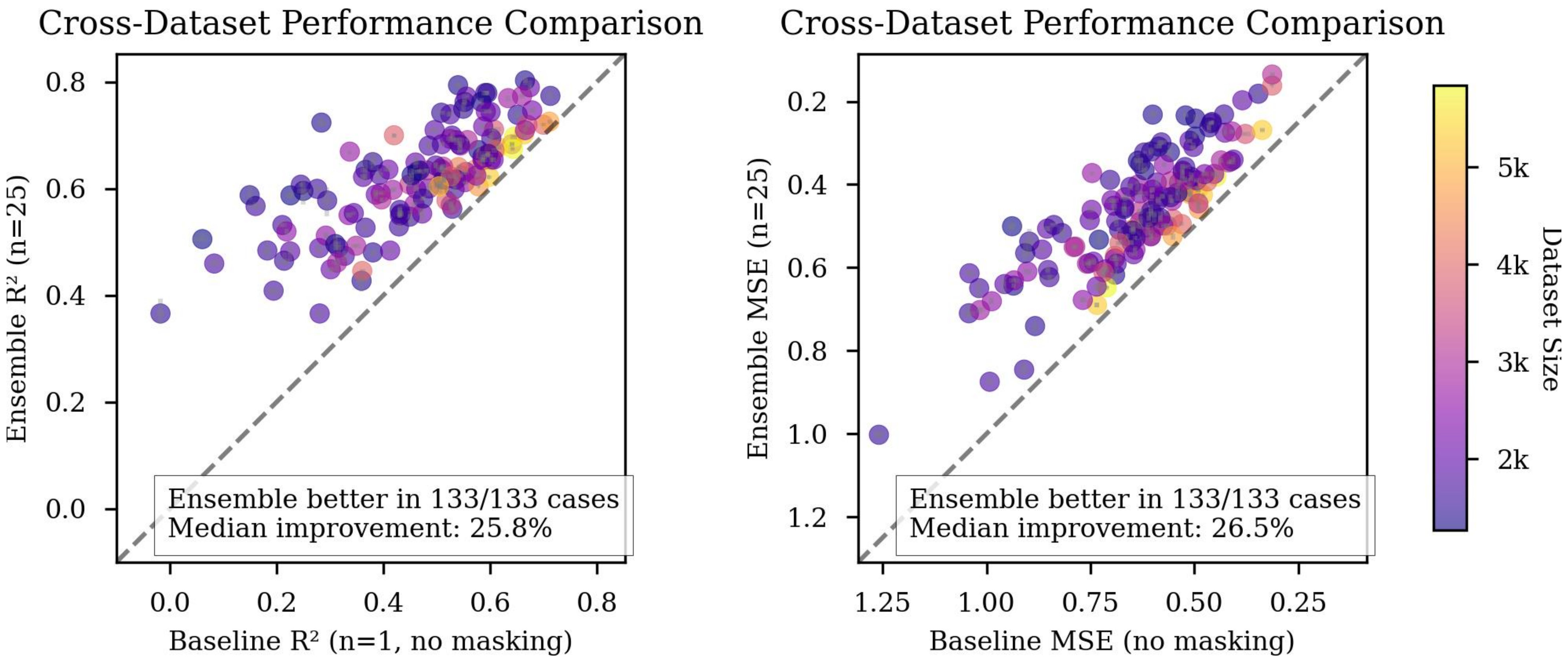


Very Efficient Training

- Performance levels off after 50 members in ensemble.
- But corresponds to only **~2x training time** compared to single network!
- Fingerprint and y_value tensors moved to GPU *before* expansion along n_ensemble dimension
- Effective **minibatch size increased** significantly
- => Maybe why we see more robust training in fewer epochs?



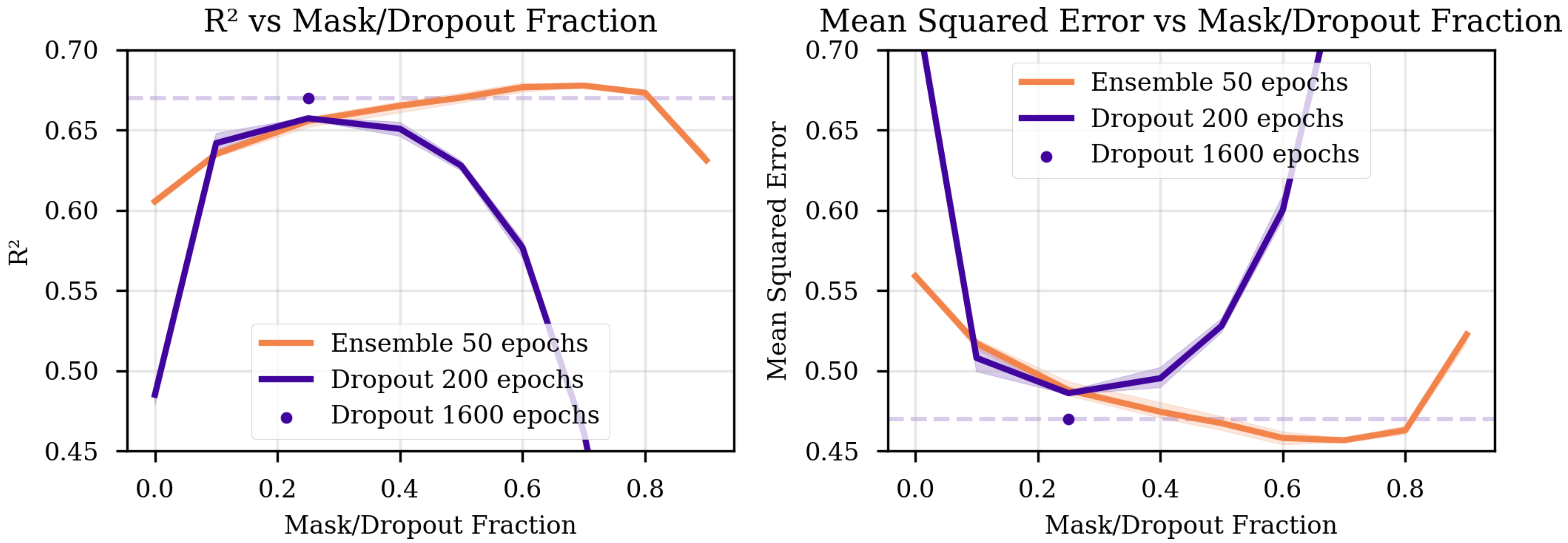
Benchmarking on a collection of datasets



- Tests on **133** regression datasets from **ExcapeDB** (binding and activity).
- Using RandomNets ensembles **unidirectionally improved performance**, being most pronounced for smaller datasets.

Sun, Jiangming, Nina Jeliazkova, Vladimir Chupakin, et al. ‘ExcapeDB: An Integrated Large Scale Dataset Facilitating Big Data Analysis in Chemogenomics’. Zenodo, 29 November 2016.

Comparison with Dropout



- **Dropout-In** seems conceptually very similar to input-masking of implicit ensembles.
- Dropout network sampled multiple times with dropout-in active and prediction averaged.
- However, **ensemble performance could not be reached**, even with extended training to account for higher effective mini batch-size of ensemble approach.

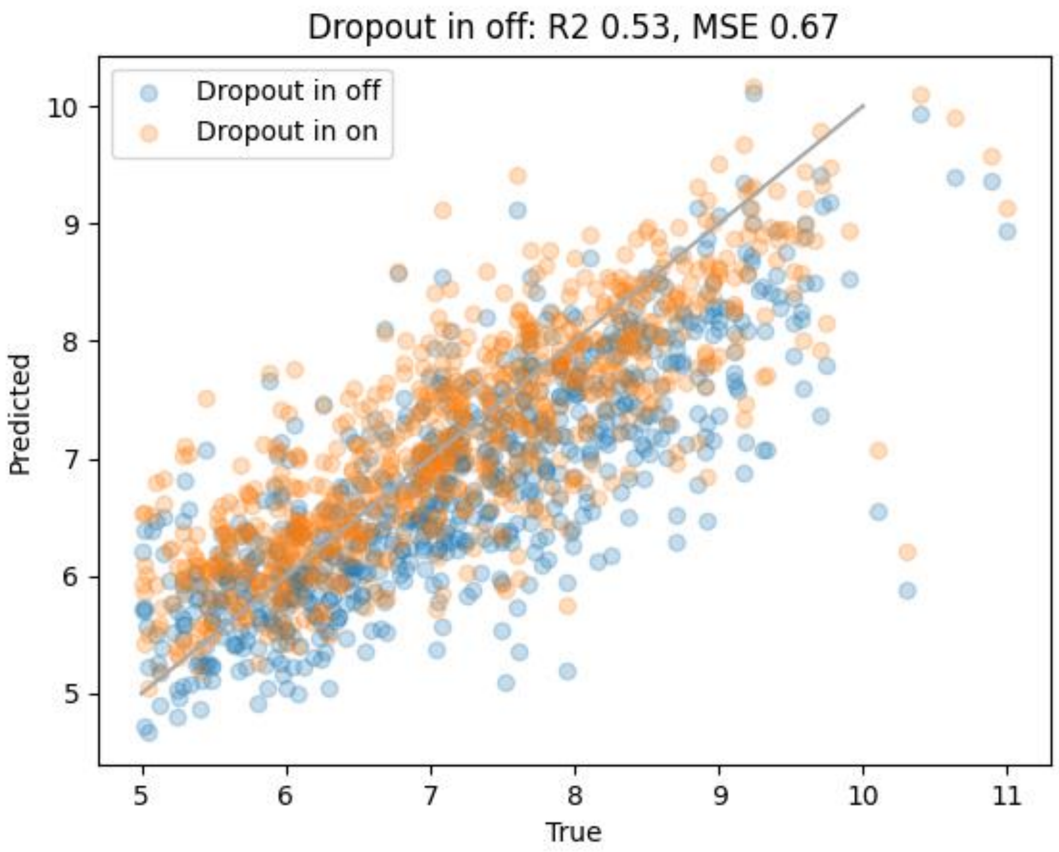
Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 'Dropout: A Simple Way to Prevent Neural Networks from Overfitting'. Journal of Machine Learning Research 15, no. 1 (**2014**): 1929–58.

Surprising effect of Dropout-In

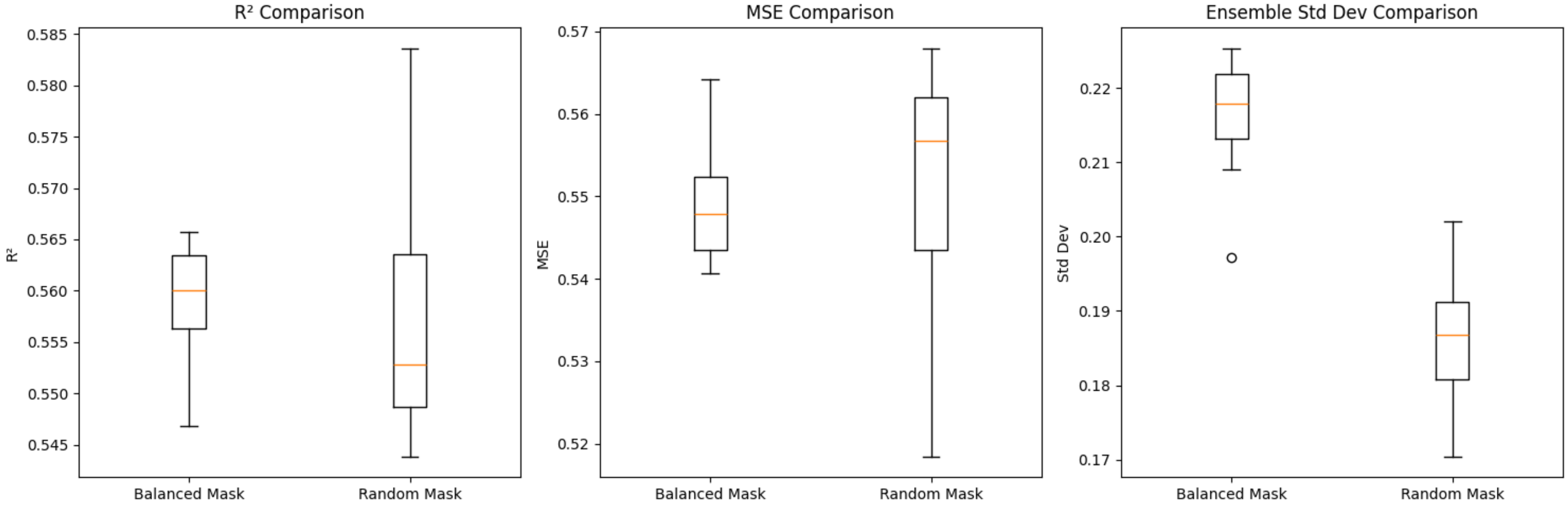
- When training with dropout-in, this could **NOT** be switched off during inference without seeing a **systematic lower prediction.**

  => Dropout_in networks was **sampled n_ensemble times** with dropout_in active and **prediction averaged**.

- Anyone got any idea as to why this effect comes when using dropout on input, in contrast to on layer outputs? Please reach out!



Balancing the Mask



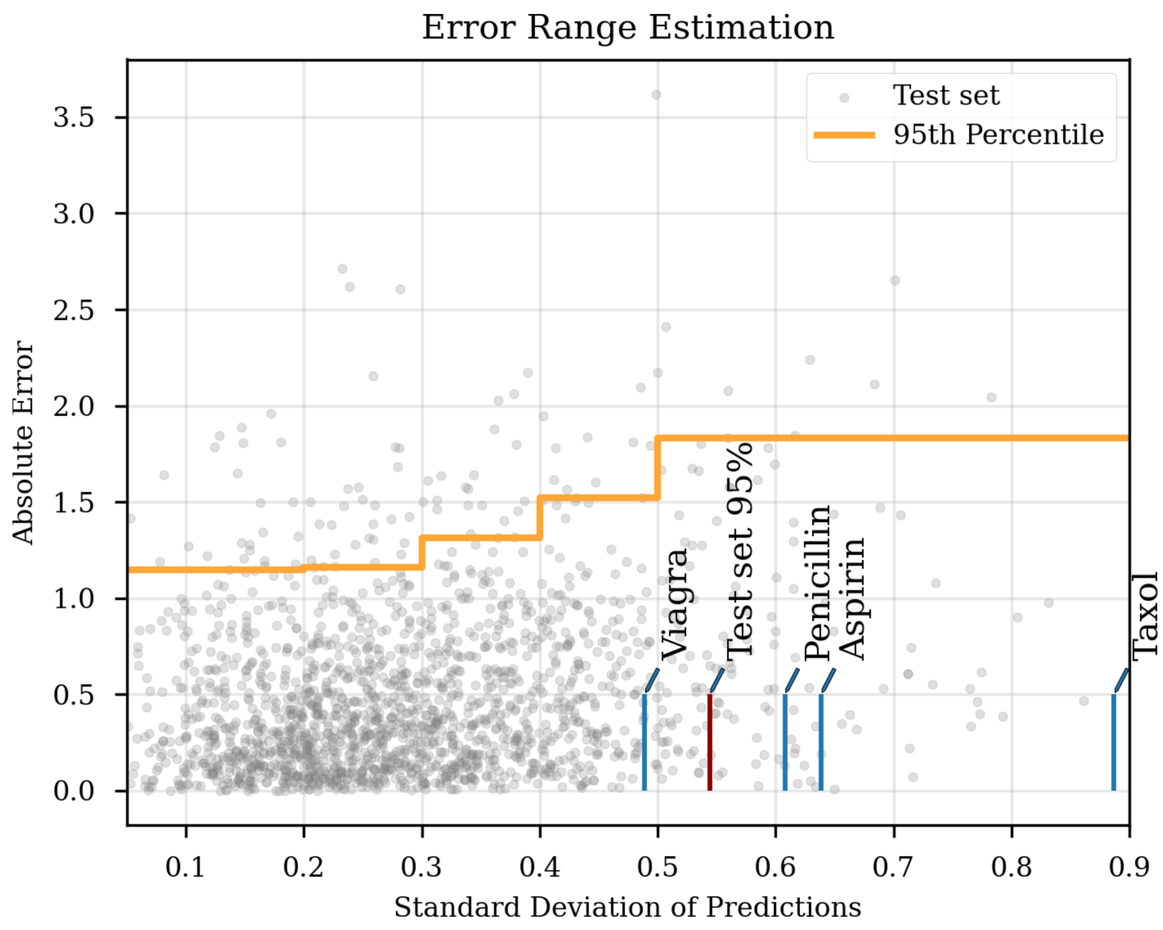
When creating the random feature mask, some features gets **more masked** than others (i.e. is used in fewer ensemble members).

Designing the input mask to have a **more even distribution** of masked features, reduces variability and improves predictive power on average.

(Results on previous slides done with fully random feature masks)

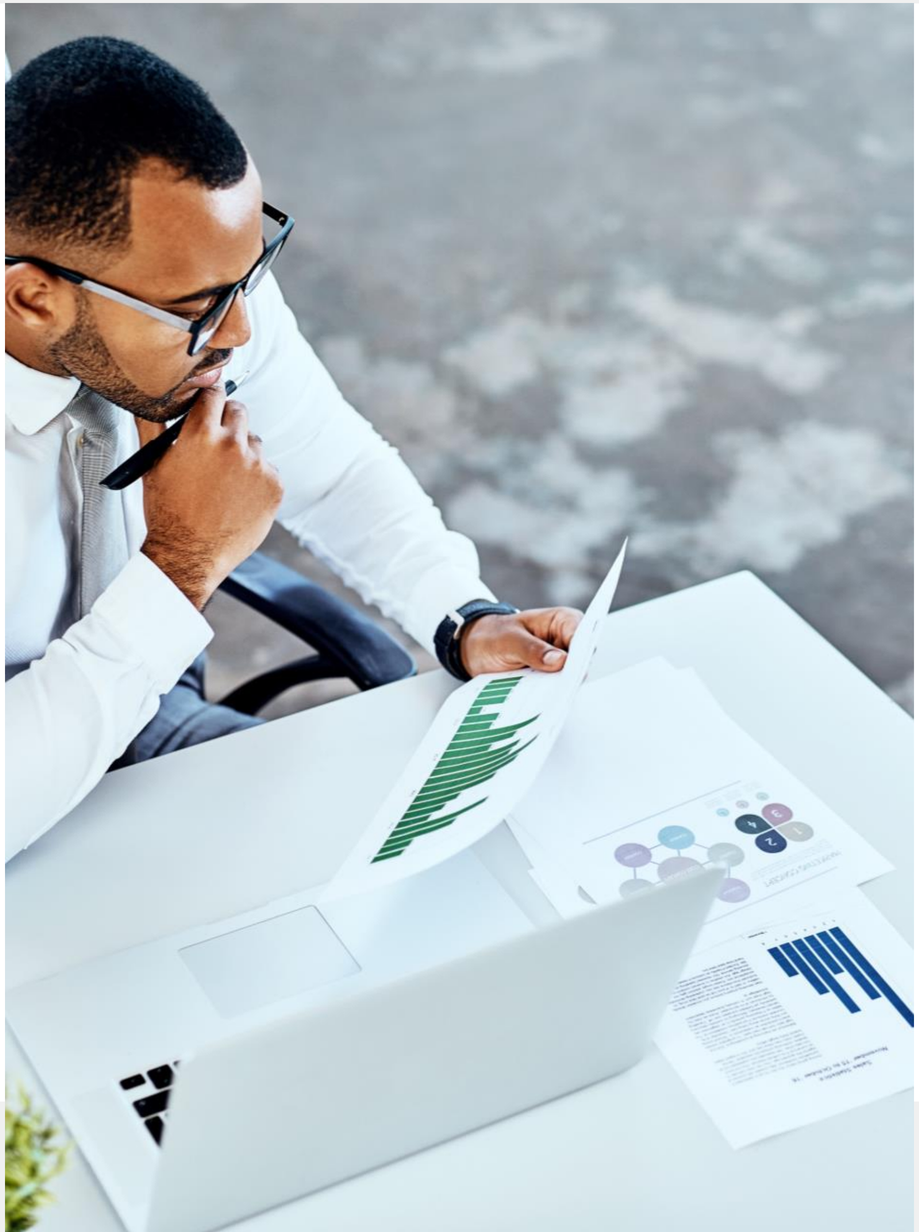
Standard Deviation of Predictions Hints of Uncertainty and Applicability Domain

- For this dataset, SLC6A4, a serotine transporter:
- Standard deviation < 0.3 have 95% of predictions below ~1.2 units
- Standard deviations > 0.5 have 95% of predictions below ~1.75 units in absolute error
- Challenges with believed adversarial compounds associated with higher standard deviation in prediction.



Conclusions

- **Feature masking**, but not sample masking, gave most improvement of ensemble predictions.
- Training show **sub-linear scaling**
- **Improved performance** on 133 out of 133 Bio regression datasets.
- Compared to feature dropout, shows less sensitivity to hyperparameter choice and better performance.
- RandomNets is an **attractive approach compared to single feed-forward neural networks**
- Is open-sourced at https://github.com/EBjerrum/RandomNets
- Currently only regression modelling but can be easily adapted (Reach out if interested).



Acknowledgements

- Open Source Contributors of the World
- Releasers of open molecular datasets

RDKit
Open-Source Cheminformatics and Machine Learning

python™

PyTorch

NumPy

Scikit Mol

scikit learn

pandas

PyTorch Lightning

Thank you for your attention!

Questions?