Supporting Information

Generalized DeepONets for Viscosity Prediction Using Learned Entropy Scaling References

Maximilian Fleck[1], Marcelle B M Spera[1], Samir Darouich[2,3], Timo Klenk[1] and Niels Hansen[1]

[1]*Institute of Thermodynamics and Thermal Process Engineering*, University of Stuttgart, Pfaffenwaldring 9, Stuttgart, 70569, Germany

[2]*Institute for Artificial Intelligence*, University of Stuttgart, Universitätsstraße 32, Stuttgart, 70569, Germany

[3]*Institute for Theoretical Chemistry*, University of Stuttgart, Pfaffenwaldring 55, Stuttgart, 70569, Germany

maxi_fleck@posteo.com, hansen@itt.uni-stuttgart.de

1 Model Details

Table 1: Model details of the GenDeepONet with dimensions/nodes D and number of layers N. The dimensions of the passes between the sub networks are indicated separately after the +. We used ReLu activation, Adam Optimizer with learning rate of 0.001, and a batch size of 32 for training. Total number of weights is 18286, i.e., you should be able to train this model on your personal laptop CPU.

Unit	Network	D_{in}	D_{hidden}	N_{hidden}	D_{out}
entropy	branch	7+12	82	3	16+1
	trunk	1	16	3	16
reference/denoise (*)	branch	7	24	5	12+1
	trunk	1	16	5	12

2 Train/Test/Val splits

The train/test/val split strategy had to be determined considering that on top of the extrapolation behavior towards unknown species, the denoising capabilities and the prediction performance for unknown state points of known species was also desired. The latter is important because our approach can cover an extremely wide range of state points. Although the objectives overlap, they also differ. For example, a model may be good at extrapolating to unknown species without exhibiting very good denoising performance. To account for the test set contamination, we can compare the performance indication results. The real one containing only test set data points without leakage, and the perceived one containing data points with and without leakage, as was adopted by [1].

We show in SI Figure 1 our strategy for the split. This was used in all models shown in the main part (for Split 1 and Split 2) and in SI Figures 6 to 19. The dataset in use contains 77547 samples from 733 species. We used a train/test/val split of 50/25(15)/25(15), with the value inside parentheses showing the percentage of data not contaminated by leakage. The resulting train set has 38773 samples, the validation set 19387 samples, and the test set 19387 samples. Of the 733 species, 115 species with 8604 samples are uniquely found in the validation set, and 116 species with 10771 samples are uniquely found in the test set. Therefore, roughly 50% of the samples in the test (and validation) set are of species not part of the training (and validation) set. The validation and test sets do not contain exactly 15% of all species because we have incorporated a lower limit at the family level, i.e., only a sufficiently populated chemical family is split, while families that contain only a few species are initially placed in the training set in their entirety. More precisely, the species are split within a chemical family if it contains enough species. A family with 12 or more species is split into train/test/val. A family with 9 or more is split into train/val, and a family with fewer than 9 species remains in the test set. After splitting at the species level, the validation and test set are filled with random samples from the test set to achieve the desired 50/25/25 total ratio. In this random split, no substance- or family-specific characteristics are taken into account. In theory, therefore, large quantities of the substance-specific samples kept in the test set during the species split could be included in the validation and test sets. An average family in the dataset contains 18 species. We introduced a minimal molar weight (68 $^{g}/_{mol}$) and a minimum smiles length (4 characters). Only molecules with larger values are added to the validation or test set in the family split. We also want to point out that

the datasets largest family is the family with the label unknown, consisting of species that were not assigned to a family. The species in validation and test sets that are not part of the training set are also heterogeneous.

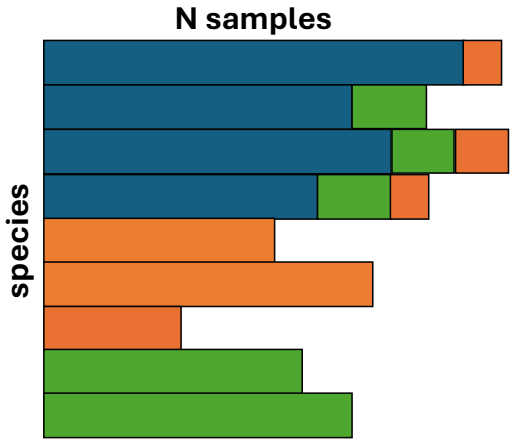

Figure 1: Visualization of train/test/val splits. Training data is blue and 50% of the total data. Validation data is orange and 25% of the total data. Test data is green and 25% of the total data. Both validation and test data contain complete data sets of species that do not appear in the training dataset (15%). A certain proportion of the species in the training set is allocated to the validation and test sets in order to test the denoising behavior and extrapolation to state points that are not part of the training set. We will denote the described split for train/test/val as 50/25(15)/25(15).

3 Outliers

For the 50/25(15)/25(15) training, validation, and test data split discussed in the main part of the work and used for benchmarking (see also SI Figure 6), species in the test set exhibiting deviations exceeding 100% relative to experimental values are listed in Supplementary Information (SI) Table 2. Many of these outliers contain molecular substructures uncommon within the dataset, such as aromatic or non-aromatic rings, and/or less frequently represented heteroatoms like iodine or fluorine. However, a notable subset of these discrepancies appears to result from inaccuracies in the reported experimental data. This includes compounds such as propane, propan-1-ol, propan-2-ol, and (e)-3-phenylprop-2-enal, 1-methyl-4-propan-2-

ylbenzene, dodecan-1-ol, and 1-methoxy-2-[2-[2-[2-(2-methoxyethoxy)ethoxy]ethoxy]ethoxy] ethane. Examples are provided in SI Figures 2–5. These outliers substantially influence the overall error metrics. When evidently erroneous experimental entries are excluded, the model's mean absolute error deviation (MARD) on the test set decreases from approximately 10% to around 8%.

Table 2: Species with samples with more than 100% deviation from experimental results for the model shown in the main the paper. N is the number of samples with more than 100% deviation from experimental results.

Name	family	N
fluorobenzene	C, H, F compounds	6
benzenethiol	mercaptans	5
3-methylbutan-1-ol	other aliphatic alcohols	1
1-methyl-4-propan-2-ylbenzene	other alkylbenzenes	3
1-methoxy-2-[2-[2-[2-(2-methoxyethoxy) ethoxy]ethoxy]ethoxy]ethane	other ethers/diethers	15
(2e)-3,7-dimethylocta-2,6-dien-1-ol	unknown	1
1-iodooctane	unknown	11
cyclooctanol	unknown	6
(e)-3-phenylprop-2-enal	unknown	90
1-bromobutane	C, H, Br compounds	1
propane	n-alkanes	1
propan-1-ol	n-alcohols	1
propan-2-ol	other aliphatic alcohols	1

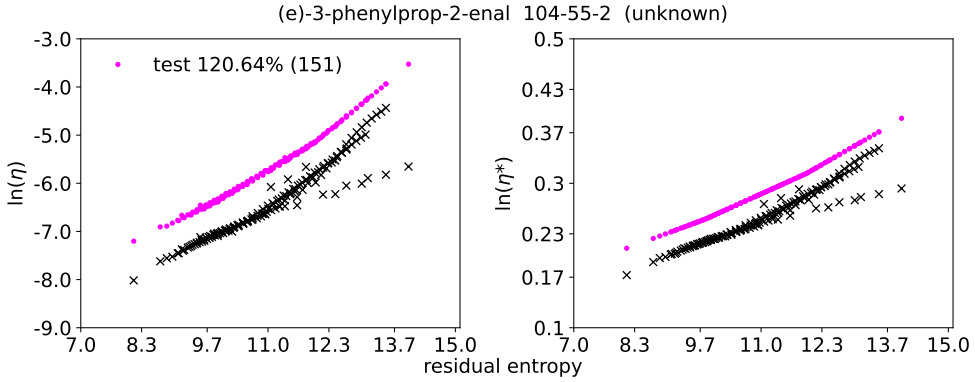

Figure 2: Predictions for (e)-3-phenylprop-2-enal. The viscosity is predicted very well qualitatively, but with systematic deviations (MARD% shown in the plot). However, the data set for this species also appears to contain experimental results that clearly deviate from the overall trends, and have a particularly strong impact on the error metrics.

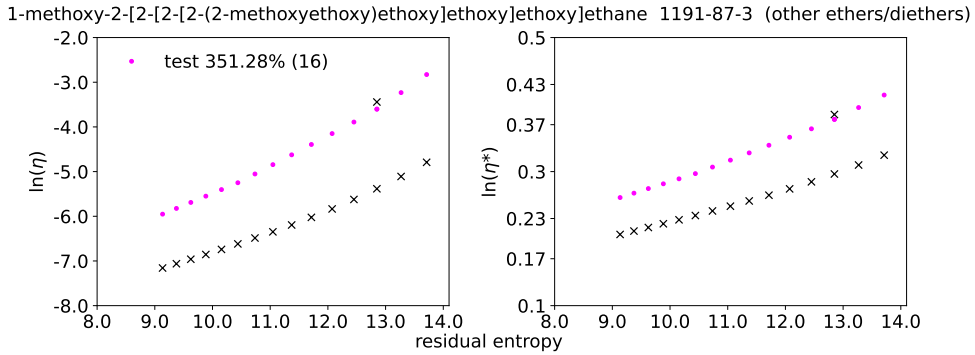

Figure 3: Predictions for 1-methoxy-2-[2-[2-[2-(2-methoxyethoxy)ethoxy]ethoxy]ethoxy] ethane. The viscosity is predicted very well qualitatively, but with systematic deviations. It is interesting to note that all other ethers are very well predicted. Here, one data point is also predicted very accurately and not just from the model shown here but also from all the models tested.

4 List of Species

A list of all species in the dataset plus the information to which split they belong to (train/test/val, referring to the 50/25(15)/25(15) split that was used throughout most parts of the study and for benchmarking) can be found on Github under train71/listofspecies.csv [2].

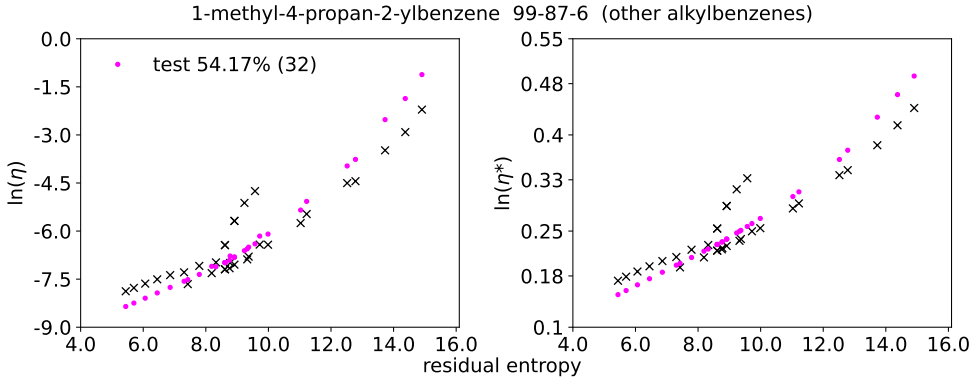

Figure 4: Predictions for 1-methyl-4-propan-2-ylbenzene. Significant deviations are apparent. They might be related to faulty experimental data.

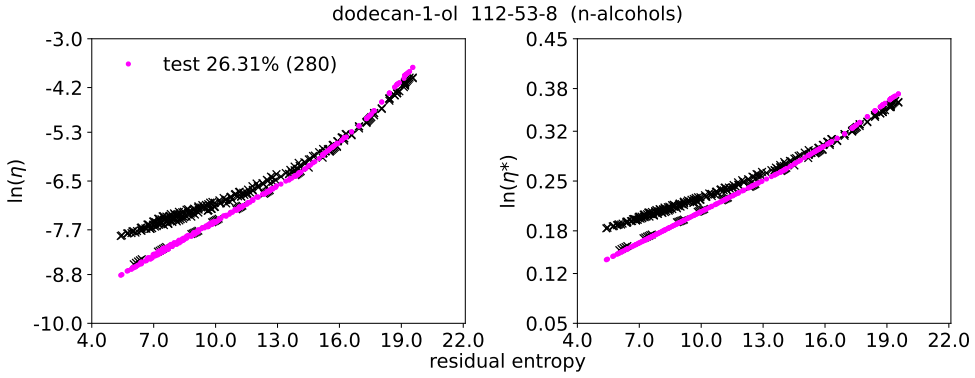

Figure 5: Predictions for dodecan-1-ol. Significant deviations are apparent. They might be related to the experimental data or to the equation of state used and the associated parameters.

5 Benchmarking

We tested our architecture against several other approaches. Models in SI Figures 6-13 were trained on the same 50/25(15)/25(15) split. Models in SI Figure 14 and 15 were trained on other 50/25(15)/25(15) splits. Models in SI Figures 16-18 were trained on the same two 20/40(25)/40(25) splits. Models in SI Figure 19 were trained on the same 40/30(20)/30(20) split. For all architectures and splits, several models were trained (including hyperparameter optimization based on the validation set) in order to assess and discuss the consistency of the results. The best results for each architecture are shown here. Significant deviations between the individual training runs are also shown or mentioned. The results are summarized on SI Table 3, and can be found in more details on DaRUS - the Data Repository of the University

of Stuttgart (link available on Github[2].)

Table 3: Benchmark against other architectures: performance comparison.

		MARD %				
split	**dataset split**	**training**	**validation**	**test**	**architecture**	**SI Figure**
		5.07 - 6.18	7.54 - 9.93	9.61 - 12.78	DeepESNet	6 - 7
		5.21 -5.52	7.13 - 7.33	10.1 - 4404.36	FF + PC-SAFT (s)	8 - 9
		5.79	7.47	10.84	DeepESNet mod	10
1	50/25(15)/25(15)	7.19	8.59 - 10.24	11.16 - 14.46	DeepESNet poly	11
		32.13	32.04	59.89	FF + PC-SAFT (p)	12 (left)
		20.73	24.02	30.93	FF + PC-SAFT (log p)	12 (right)
		6.58	8.52 - 10.93	13.12 - 18.6	FF + PC-SAFT (dens)	13
2	50/25(15)/25(15)	5.78	7.71 - 8.75	7.81 - 8.74	DeepESNet	14 (top)
		5.62	8.9 - 10.57	9.15 - 10.92	FF + PC-SAFT (s)	14 (bottom)
3	50/25(15)/25(15)	6.19	9.68 - 12.88	8.22 - 9.35	DeepESNet	15 (top)
		5.21	8.78 - 12.04	7.48 - 8.71	FF + PC-SAFT (s)	15 (bottom)
		8.7 - 13.4	19.36 - 17.96	14.15 - 19.6	DeepESNet	16
4	20/40(25)/40(25)	6.75 - 14.26	16.3 - 26.94	25.84 - 52.96	FF + PC-SAFT (s)	17
		9.61 - 10.01	17.78 - 24.05	17.12 - 38.65	DeepESNet mod	18
5	40/30(20)/30(20)	11.14	15.36 - 17.5	17.33 - 20.53	DeepESNet	19 (top)
		11.24	21.45 - 26.91	18.7 - 23.06	FF + PC-SAFT (s)	19 (bottom)

The Figures show that the DeepESNet architecture is easier to train and most consistent in its extrapolation performance toward unknown species. A key component allowing those results is that all networks in that multi-network architecture are able to communicate with each other in a sequential, directed, and meaningful order. This separates the DeepESNet model from the DeepESNet modified architecture missing the branch passing (SI Figures 10 and 18), as the latter has shown to be less reliable in training and extrapolation than the fully connected DeepESNet model. Additionally, trained a DeepESNet model with polynomials replacing the trunk neural networks. Those models showed slightly worse metrics than the original DeepESNet. To achieve those metrics polynomials of high order, for example 12, were used.

Of the feed forward models, the entropy model performs best, followed by molar density, log pressure, and pressure. This demonstrates above all the strength of the entropy scaling approach, which delivers good results even in the feed forward approach.

We trained the models on a much smaller dataset with a 20/40(25)/40(25) split. i.e, using only 20% of the available samples for training. Here, too, the DeepESNet model performs best for the performance metrics, extrapolation, and reliability. It also shows the lowest performance loss when the training data set size is reduced. Finally, a 40/30(20)/30(20) split was used to train the DeepESNet and the feed forward entropy. Results are shown in

SI Figure 19.

DeepESNet architectures use much less weights than feed forward architectures. The best performing DeepESNet on the 50/25(15)/25(15) split has 18286 weights, whereas the best performing entropy feed forward network has 50945 weights. This is a significant difference, even if one assumes that both models are equivalent in performance metrics.

5.1 Previous Work

Integrating Entropy Scaling into a Deep Neural Network Architecture to Predict Viscosities

We have published another architecture related to entropy scaling on ChemRxiv (Title: Integrating Entropy Scaling into a Deep Neural Network Architecture to Predict Viscosities)[3]. This architecture is closer to analytical entropy scaling approaches and incorporates the Chapman–Enskog reference and polynomial features of residual entropy. This work explicitly adopts the Loetgering, Lin et al. approach [4] and therefore does not represent a generalized entropy scaling. Instead, it represents an intermediate step between this work and analytical methods [4, 5, 6]. This is where its contribution lies. We did not find the manuscript sufficient for peer-reviewed publication and have instead decided to further develop a generalized approach (this work). Nevertheless, we consider appropriate to leave the work available on ChemRxiv, especially since it has the intermediate step between analytical entropy scaling and our current proposal.

Performance comparison: the architecture in previous work was trained on a smaller, more curated dataset containing fewer chemical families and only around 600 species (in other words: the around 100 missing species are the challenging ones). In addition, the test set is much smaller in relation to training and validation sets compared to this work. The main disadvantage of the model is that it contains a feed forward correction of the Chapman-Enskog reference with full state point information (temperature and density) as inputs. Without this correction, the performance of the model would be significantly worse, while with the correction there is a risk of overfitting. The model therefore suffers from the disadvantages of the feed forward approaches discussed in this paper (but is at least interpretable in comparison to them). In summary, we consider this model to be immature because it applies the entropy scaling approach but does not enforce it.

When comparing the results of this work with those of previous works, it becomes clear that the new approach - trained and tested on a much more demanding dataset, with a more demanding train/test/val split - easily achieves better metrics and is therefore superior. Even a simple feed forward approach using residual entropy as input should be superior in this regard. Results for DeepESNet architectures that use polynomial features in one or both of the trunk networks represent a middle ground between the initial work and this work with results shown in SI Figure 12.

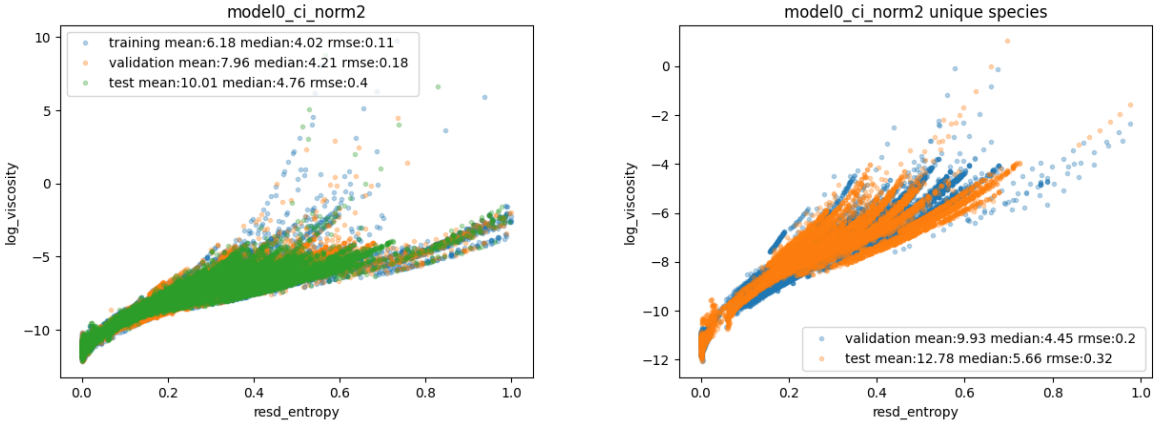

Figure 6: Detailed metrics for the DeepESNet model shown in the main part of the work.

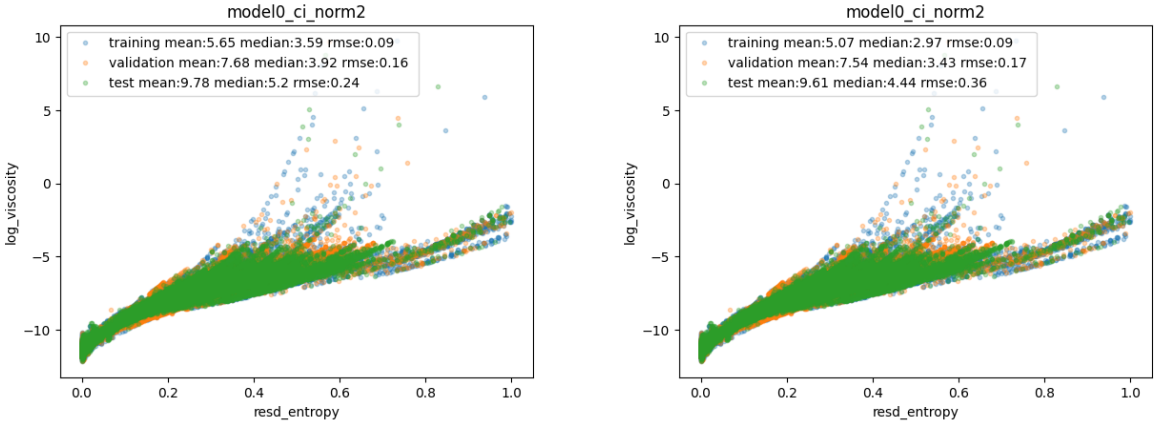

Figure 7: DeepESNet results from different training runs. Training high-performing models has proven to be very easy with this architecture.

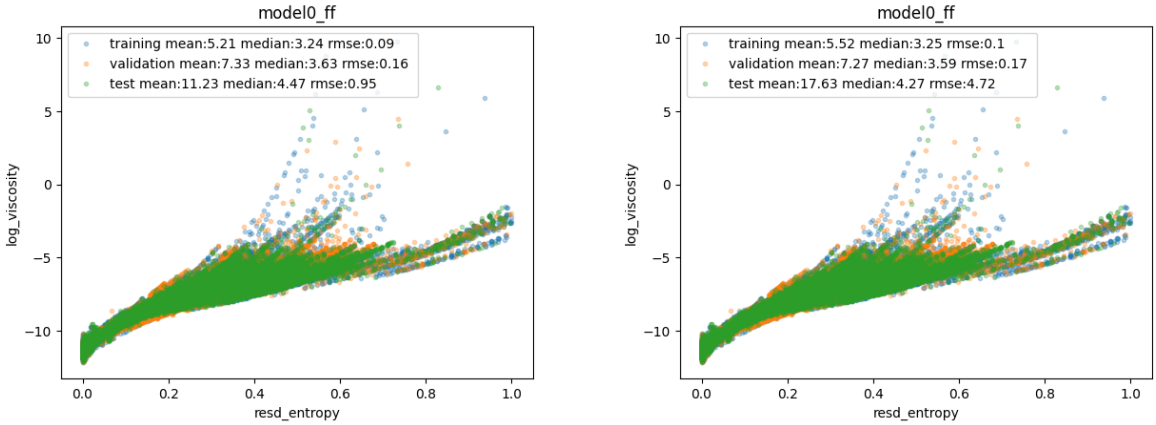

Figure 8: Two different feed forward results trained with PC-SAFT parameters, molar mass, temperature, and residual entropy as inputs and on the same dataset than our model. Another model trained the same way is shown in the next Figure (SI Figure 9). Each plot belongs to a model trained on the same data but comes from different training runs. The metrics of a feed forward model can be very good, which underscores the usefulness of the entropy scaling principle. However, it is also apparent that the extrapolation capability to unknown molecules in feed forward approaches can vary greatly with each training.

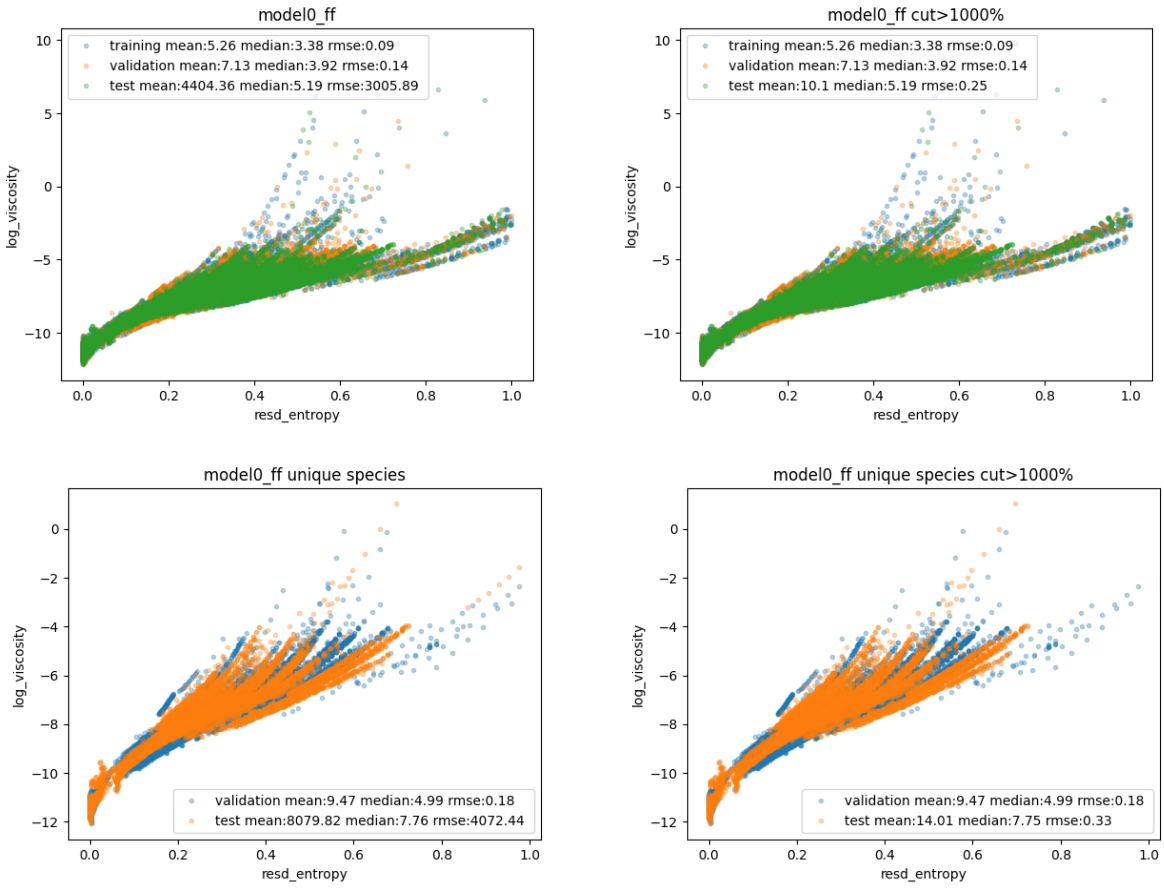

Figure 9: Feed forward model trained with PC-SAFT parameters, molar mass, temperature, and residual entropy as inputs and on the same dataset than our model. The model is trained on the same data as the models shown in the previous Figure (SI Figure 8) but comes from a different training run. This example shows how the extrapolation capability to unknown species in feed forward approaches can vary greatly with each training. It is also clear that individual incorrect predictions may have a strong influence on the model metrics: the metrics improve dramatically when the most extreme outliers (here MARD $> 1000\%$) are removed. Since other models, especially our architecture, can be trained more reliably, this is a clear indicator of the risk of overfitting in pure feed forward architectures. We would like to emphasize that we have not encountered any such outliers with the DeepESNet architecture, whereas they occur regularly with other architectures. It should be noted that the error metrics of the DeepESNet models also improve dramatically when, for example, the three worst predicted species are removed. This demonstrates the strength of the DeepESNet architecture, especially when extrapolating to unknown molecules. If only an existing dataset needs to be represented and interpolated, feed-forward approaches are a suitable alternative.

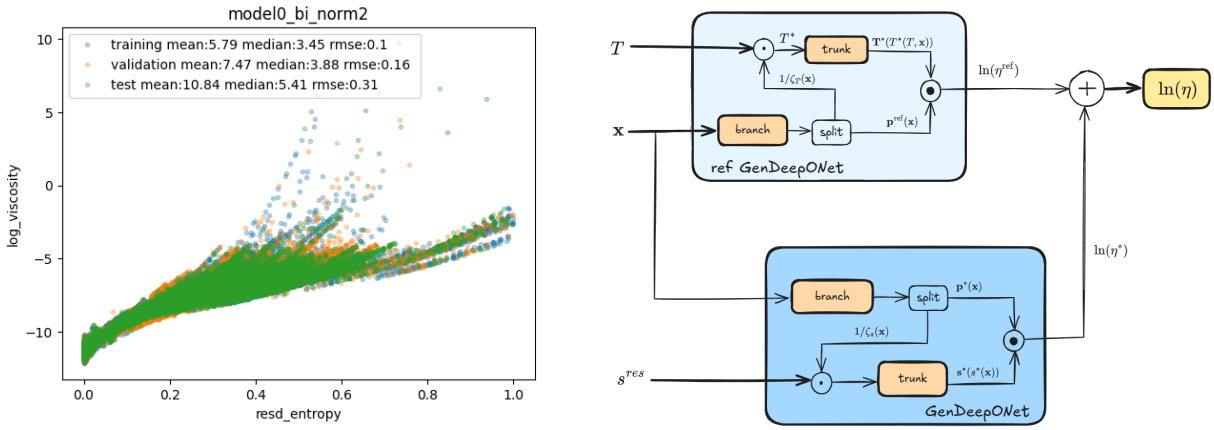

Figure 10: DeepESNet architecture results (left plot) for the case without passing information between the branch nets (illustrated on the right). This type also performed well but it was more difficult to reliably train well-extrapolating networks. Training two branch networks with identical inputs completely independently seems to destabilize the model and impair performance. These effects are further amplified when the ζ passes are omitted in the GenDeepONets, i.e., standard DeepONets are used. This demonstrates the usefulness of all networks in a multi-network architecture being designed to communicate with each other.

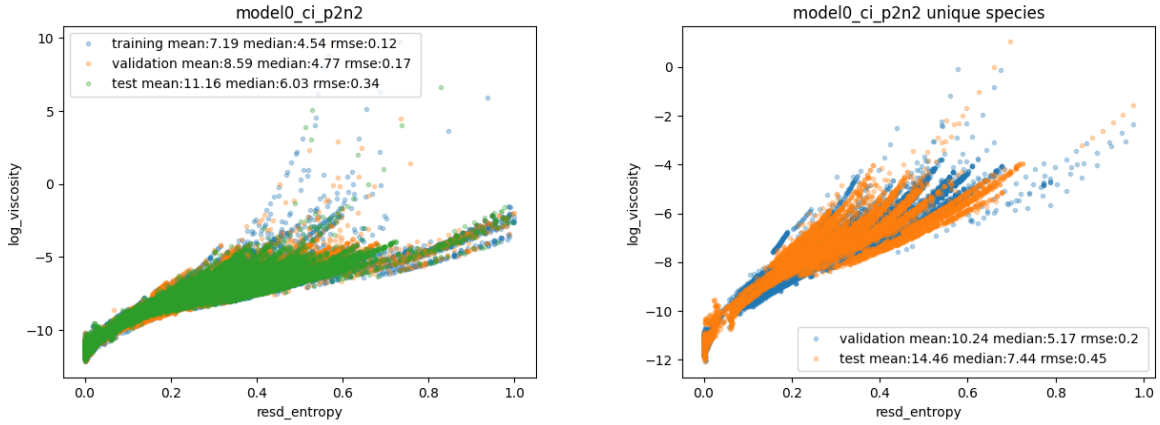

Figure 11: DeepESNet with polynomials replacing trunk networks. Performance is a little bit worse than for trunk neural networks. Extrapolation capabilities varied much more with each training than for the DeepESNet models. This is most likely due to the use of high order polynomials instead of linear ReLu activations in a neural network. In return, the models are slimmer - for example, 13078 weights - thanks to the elimination of two networks and smaller branch networks. This gives these models their raison d'être when computational efficiency is required.

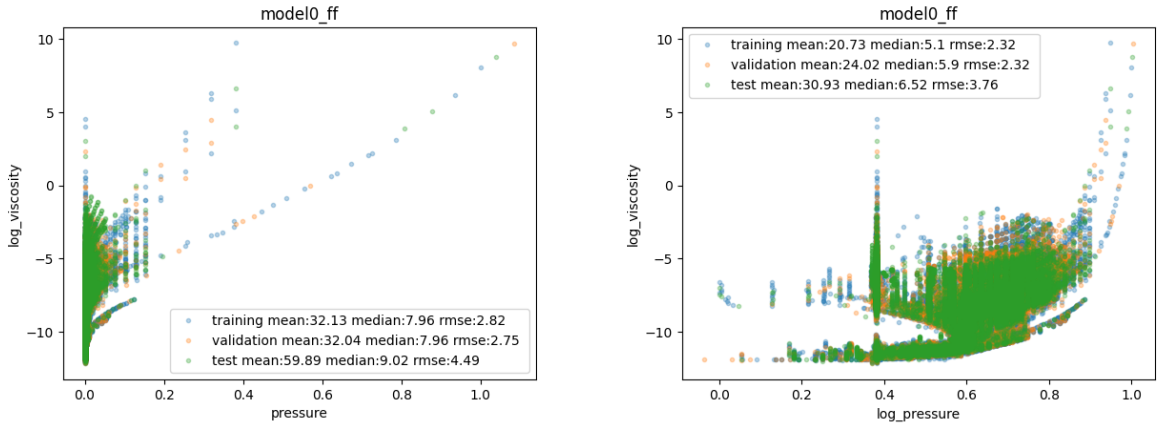

Figure 12: Feed forward results trained with PC-SAFT parameters, molar mass, temperature, and pressure (left) or log pressure (right) as inputs and on the same dataset as our model. These results show above all how useful entropy is as an input feature for describing transport properties.

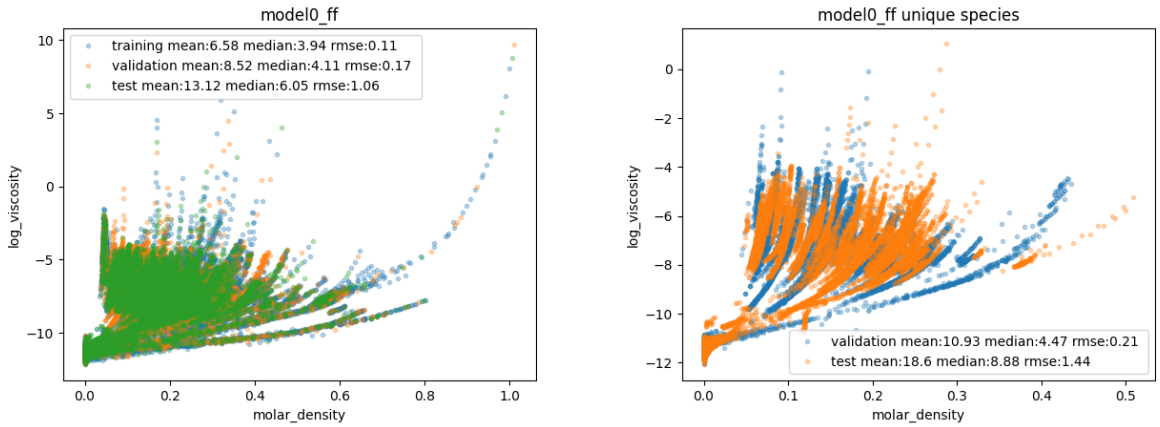

Figure 13: Feed forward results trained with PC-SAFT parameters, molar mass, temperature, and molar density. As most of the state points are given in temperature and pressure, we used the PC-SAFT EoS to compute the molar density. Using density as an input feature leads to better feed forward model performances than using pressure, still not as good as residual entropy.

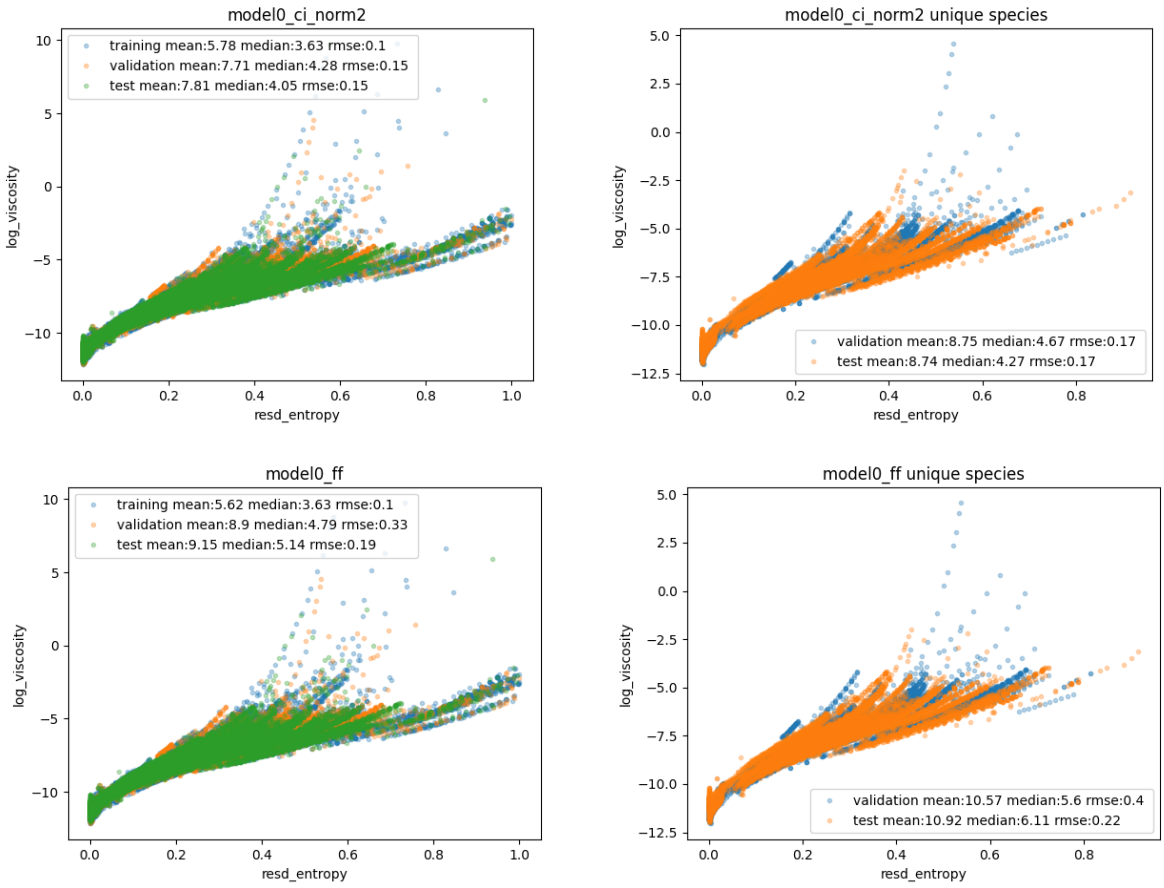

Figure 14: DeepESNet (top row) and feed forward entropy model (bottom row) trained on another train/test/val split of 50/25(15)/25(15) - split 2.

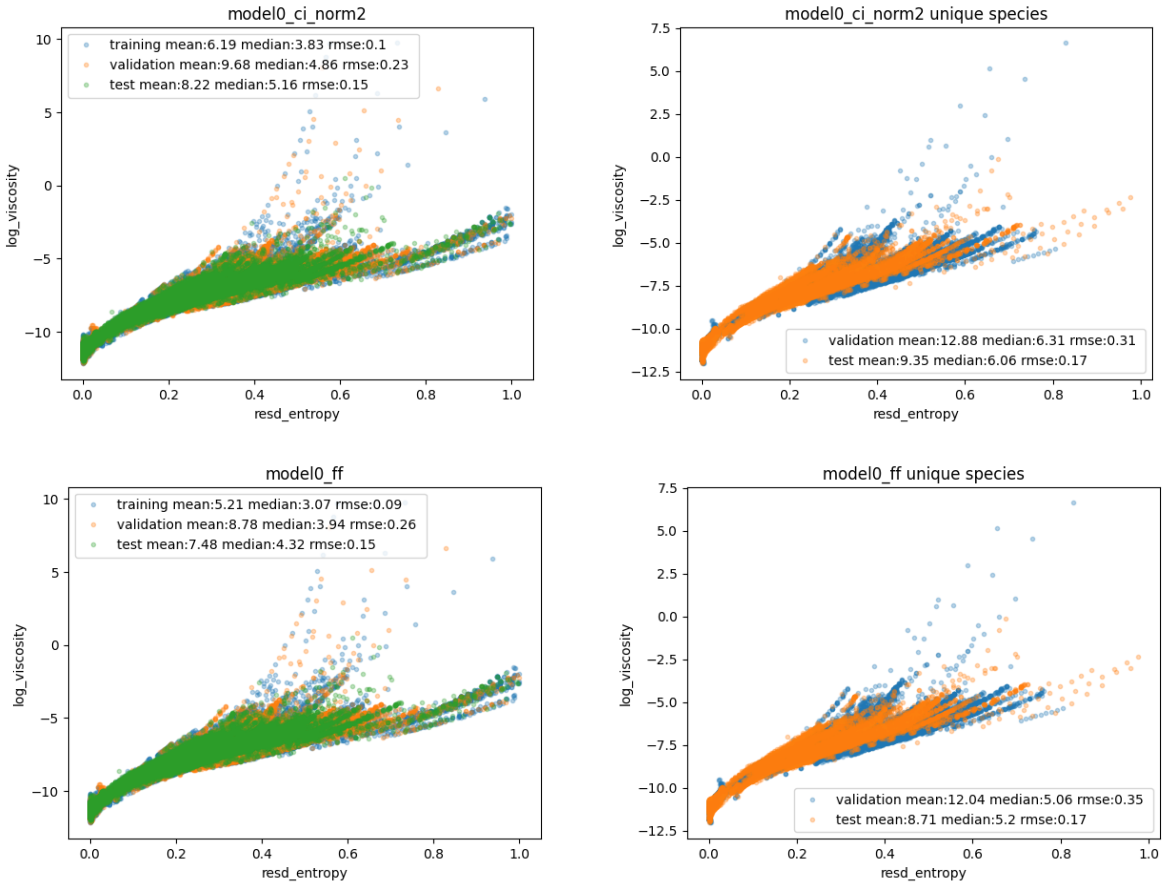

Figure 15: DeepESNet (top row) and feed forward entropy model (bottom row) trained on another train/test/val split of 50/25(15)/25(15) - split 3. In this case, the feedforward approach performs slightly better than the DeepESNet architecture, although the differences are very small and the models extrapolate equally well in terms of RMSE.

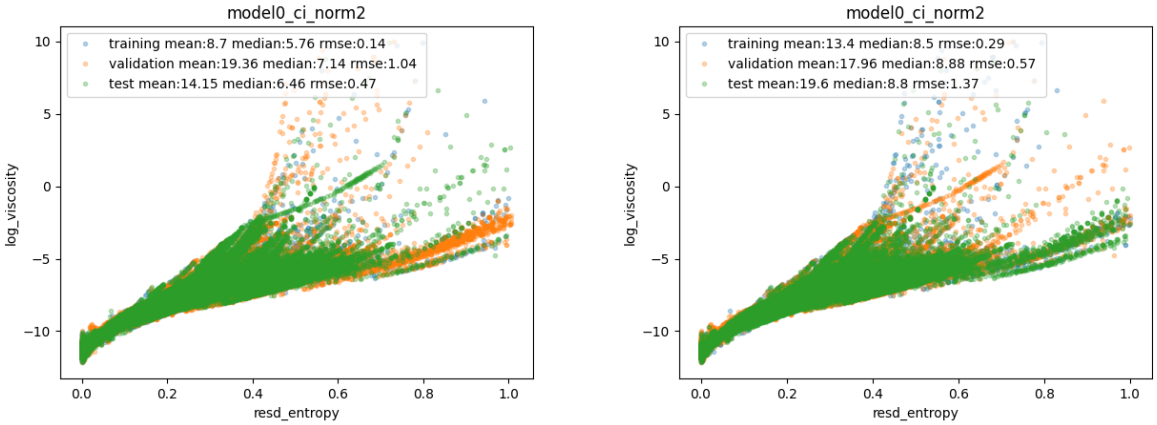

Figure 16: DeepESNet model trained with only 20% of all data, i.e., a train/test/val split of 20/40(25)/40(25). We show the results of two different splits. Both plots show that the validation (orange) and test (green) set contains special species (top center and high entropy area) covering areas not covered by the training set (blue).

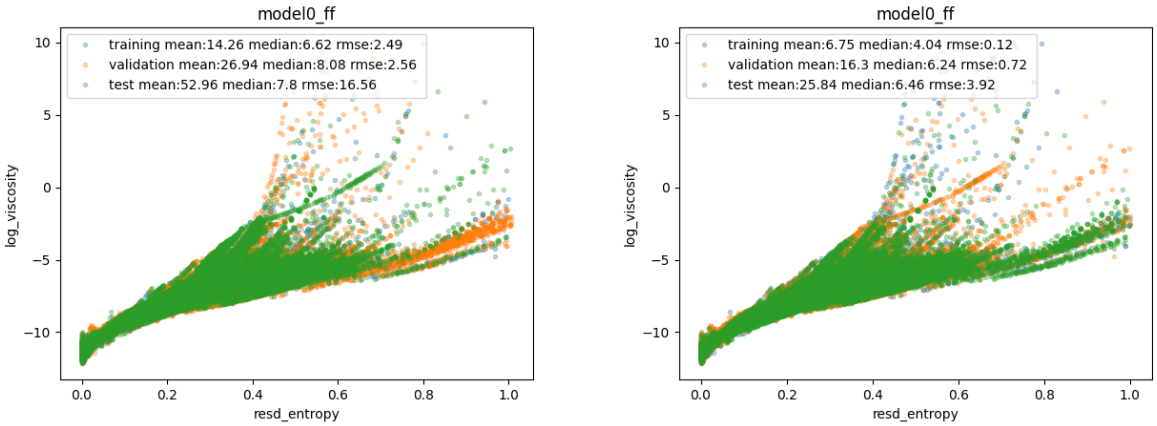

Figure 17: Results for a feed forward model with PC-SAFT parameters, molar mass, temperature, and residual entropy as inputs trained with only 20% of all data, i.e., a train/test/val split of 20/40(25)/40(25). We show the results of the model trained on two different splits. Both plots show that the validation (orange) and test (green) set contains special species (top center and high entropy area) covering areas not covered by the training set (blue).

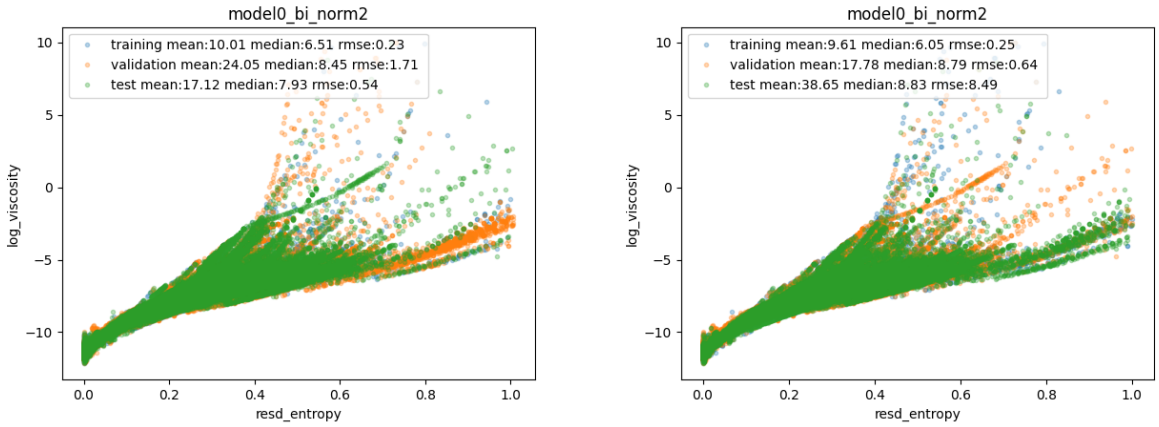

Figure 18: DeepESNet model without the passing of information between the branch networks with a train/test/val split of 20/40(25)/40(25). We show the result of two different splits. Both plots show that the validation (orange) and test (green) set contains special species (top center and high entropy area) covering areas not covered by the training set (blue).

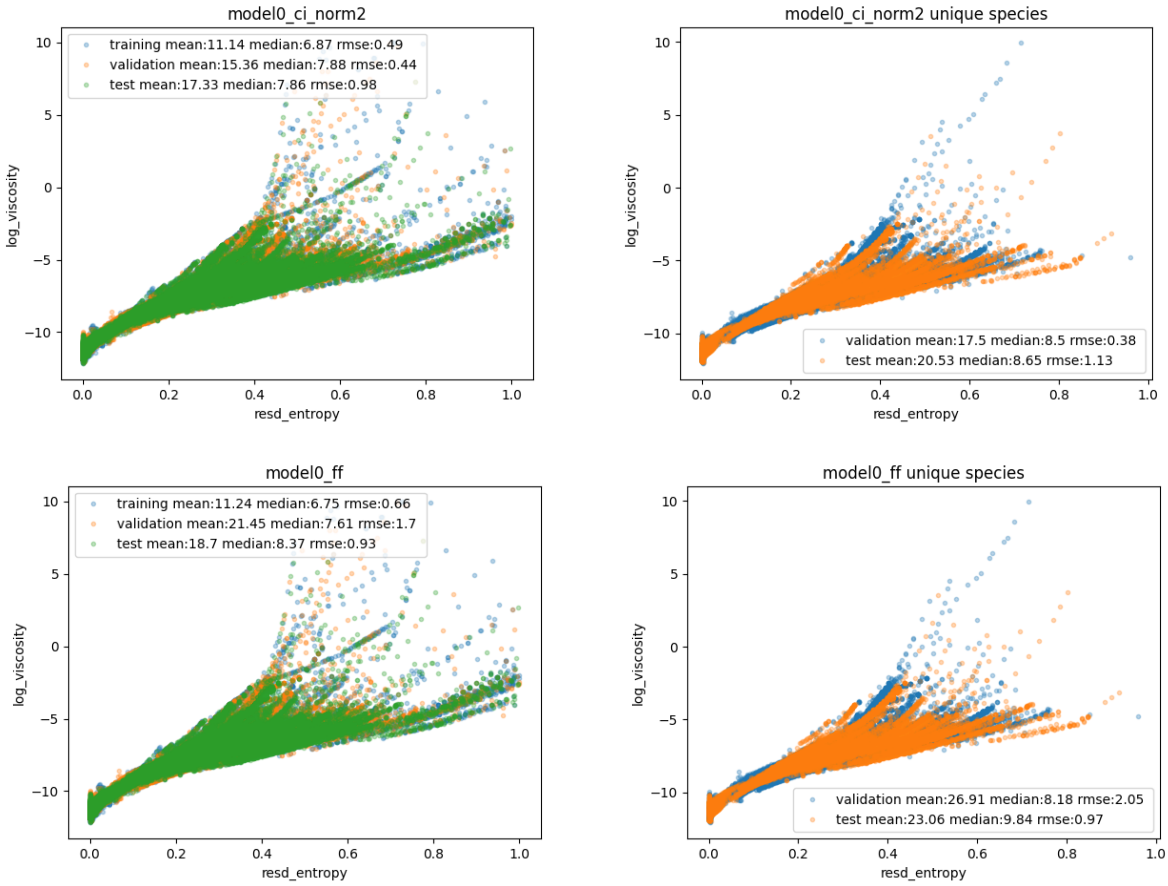

Figure 19: DeepESNet (top row) and feed forward entropy model (bottom row) trained on a train/test/val split of 40/30(20)/30(20) - split 5. Here too, various training sessions with the feed forward architecture showed fluctuations in the test and validation set performance.

References

[1] Esther Heid, Charles J McGill, Florence H Vermeire, and William H Green. Characterizing uncertainty in machine learning for chemistry. *Journal of Chemical Information and Modeling*, 63(13):4012–4029, 2023.

[2] Maximilian Fleck. https://github.com/maxfleck/deep-entropy-scaling.git (accessed on 08/07/2025).

[3] Maximilian Fleck, Joachim Gross, and Niels Hansen. Integrating entropy scaling into a deep neural network architecture to predict viscosities. *ChemRxiv*, pages doi: 10.26434/chemrxiv–2024–8982t, 2024. doi: 10.26434/chemrxiv-2024-8982t.

[4] Oliver Lötgering-Lin and Joachim Gross. Group contribution method for viscosities based on entropy scaling using the perturbed-chain polar statistical associating fluid theory. *Ind. Eng. Chem. Res.*, 54(32):7942–7952, 2015.

[5] Daniel J Carlson, Neil F Giles, W Vincent Wilding, and Thomas A Knotts IV. Liquid viscosity oriented parameterization of the mie potential for reliable predictions of normal alkanes and alkylbenzenes. *Fluid Phase Equilib.*, 561:113522, 2022.

[6] Daniel J Carlson, Neil F Giles, W Vincent Wilding, and Thomas A Knotts IV. Improved liquid viscosity prediction with the novel tlvmie force field for branched hydrocarbons. *Fluid Phase Equilib.*, 566:113681, 2023.