Electronic Supplementary Material (ESI) for ChemComm.

Supporting Information for: Hybridizing physical and data-driven prediction methods for physicochemical properties

Fabian Jirasek,*[,†,‡] Robert Bamler,[†] and Stephan Mandt[†]

†*Department of Computer Science, University of California, Donald Bren Hall, Irvine, CA 92697, USA*

‡*Laboratory of Engineering Thermodynamics (LTD), TU Kaiserslautern, Erwin-Schrödinger-Straße 44, 67663 Kaiserslautern, Germany*

E-mail: fabian.jirasek@mv.uni-kl.de

Data

This work is based on the same data set as our previous work;[1] the following overview is therefore rather brief.

Data on activity coefficients at infinite dilution in binary mixtures γ_{ij}^{∞} at 298.15±1 K were adopted from the Dortmund Data Bank (DDB) 2019.[2] Only molecular components (and one ionic liquid that slipped through our filter) of which the molecular formula is known were considered. Furthermore, metals and data points that are indicated to be of poor quality in the DDB were rejected. If multiple data on γ_{ij}^{∞} for a specific binary mixture $i-j$ in the considered temperature range were available, the arithmetic mean of the available data was used. Furthermore, only components for which at least data in two different binary mixtures are available in the DDB were considered, which is a prerequisite for the application of leave-one-out cross-validation that was applied to evaluate predictive performances of the studied methods. The resulting data set comprises 240 solutes i and 250 solvents j. Experimental data on γ_{ij}^{∞} in the considered temperature range are available for 4,094 of the 60,000 possible binary mixtures. Some components appear as both solute and solvent in the data set. The entries of γ_{ij}^{∞} where i and j denote the same substance, i.e., for pure components, were set to unity to satisfy thermodynamic consistency. These data points were only included in the training data but not considered in the evaluation of the predictive performance. For information on the considered solutes and solvents, we refer to the Supporting Information of our previous work.[1]

For the comparison of the predictive performances of the different studied methods, the data set was further narrowed down since the physical base method, modified UNIFAC (Dortmund),[3,4] referred to as UNIFAC in the following, can (with its present publicly accessible parameterization[4]) only be applied to predict γ_{ij}^{∞} for 80% of the relevant mixtures. Figure S.1 shows the matrix representing all possible binary mixtures of the considered solutes and solvents. The color of each entry indicates availability of experimental data in the DDB[2] and applicability of UNIFAC to predict the respective data points (see figure caption).

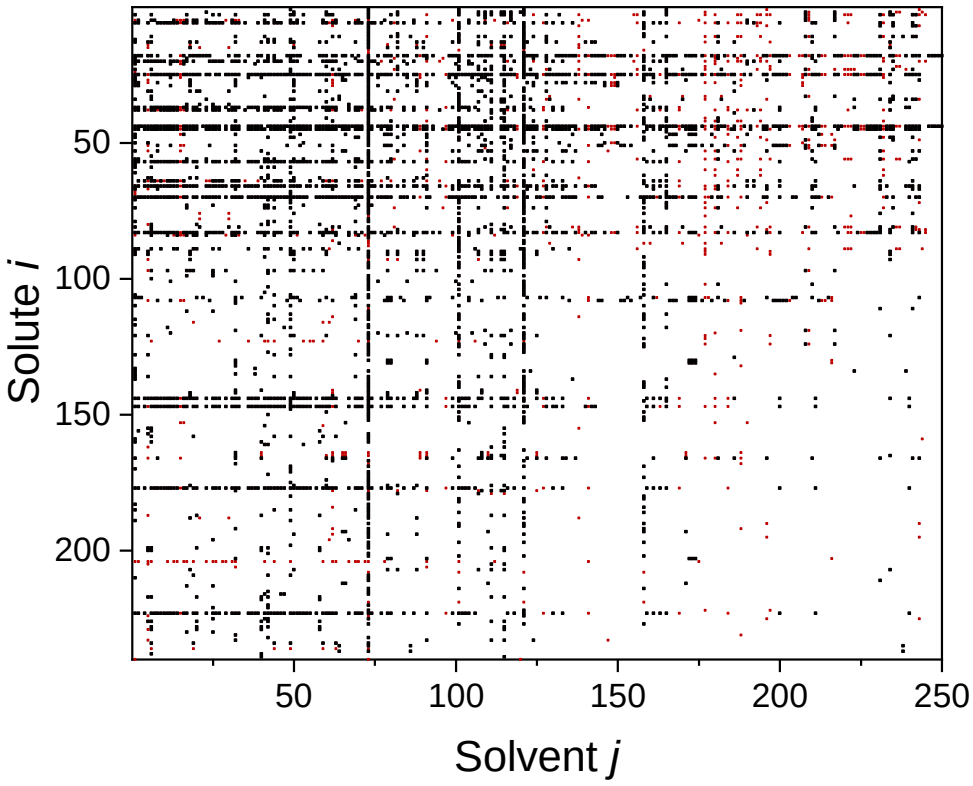

Figure S.1: Matrix representation of the available experimental data for γ_{ij}^{∞} at 298.15±1 K in the DDB 2019.[2] Whitespace: no experimental data available. Black squares: experimental data available, UNIFAC[3,4] can be applied. Red squares: experimental data available, UNIFAC cannot be applied.

Model Details

Bayesian Matrix Completion

As in our previous work,[1] we used a Bayesian approach to matrix completion to predict $\ln \gamma_{ij}^{\infty}$ (the logarithm of the activity coefficient is used for scaling purposes throughout this work). This approach consists of three steps. In the first step, a generative probabilistic model for the variable of interest, i.e., $\ln \gamma_{ij}^{\infty}$, as a function of initially unknown (latent) features of the solutes i and solvents j is specified. $\ln \gamma_{ij}^{\infty}$ is thereby modeled as the dot product of the feature vector u_i of the solute i and the feature vector v_j of the solvent j:

$$\ln \gamma_{ij}^{\infty} = u_i \cdot v_j + \varepsilon_{ij} \tag{S.1}$$

where the random variable ε_{ij} captures both measurement noise and inaccuracies of the model. Both u_i and v_j are vectors of length K containing features of solute i and solvent j, respectively. The hyperparameter K is the number of considered features per component and also called latent dimension, and was set to $K = 4$ as in our previous work[1] for all approaches discussed here.

In the second step, the latent features are inferred by training the generative model to the available data for $\ln \gamma_{ij}^{\infty}$, which requires inverting the generative model. We use Gaussian meanfield variational inference[5–7] for this purpose, which was demonstrated to be robust and efficient in our previous work.[1] Since our generative model is probabilistic, the inferred latent features are random variables and a probability distribution, called posterior, for each latent feature is obtained. In the third step, we use the means μ_{u_i} and μ_{v_j} of the inferred approximate posterior distributions over u_i and v_j, respectively, to obtain predictions from the dot product:

$$\ln(\gamma_{ij}^{\infty})^{\text{pred}} = \mu_{u_i} \cdot \mu_{v_j} \tag{S.2}$$

We note that the feature vectors u and v represent a characterization of each solute and

solvent, respectively, that is exclusively inferred from the available data for $\ln \gamma^\infty$ in binary mixtures of these components. Hence, no explicit physical knowledge on the pure solutes or solvents, e.g., molecular or pure component descriptors like molar mass, dipole moment, or structural formula, was used to find suitable feature vectors. However, since the data for $\ln \gamma^\infty$ *implicitly* comprise physical information on the respective components (which is extracted during training the model and aggregated in the feature vectors), relationships between the learned features and physical descriptors of the components can be expected. Preliminary studies have shown that there are no direct correlations between features and physical descriptors. However, to unveil these (complex) relationships will be an exciting task for future work, possibly generating previously unknown physical insights.

All predictions were obtained by leave-one-out cross-validation, i.e., by training the model to all available data except for the one data point that is to be predicted, and repeating this procedure for each available data point. This procedure ensures that the model cannot cheat by training to the test data. In all cases, we used the Stan framework,[8] which allows the specification of user-defined generative models and automates the task of Bayesian inference. The following sections provide implementation details for each of the compared methods. For more information on the theoretical background of Bayesian matrix completion, the reader is referred to our previous work[1] and the literature.[9]

Data-driven Matrix Completion Method (MCM)

Figure S.2 shows the Stan code of the probabilistic generative model of the data-driven MCM from our previous work,[1] which is considered as a data-driven base method here.

For all component features u_i and v_j, the same Gaussian prior distribution with mean $\mu_0 = 0$ and standard deviation $\sigma_0 = 0.8$ was used. Furthermore, a Cauchy likelihood with scale parameter $\lambda = 0.15$ was used for all data points.

```
1  data {
2      int<lower=0> I;             // number of solutes
3      int<lower=0> J;             // number of solvents
4      int<lower=0> K;             // number of latent dimensions
5      real ln_gamma_exp[I, J];    // matrix of experimental activity coefficients
6      real<lower=0> sigma_0;      // prior standard deviation
7      real<lower=0> lambda;       // likelihood scale
8  }
9
10 parameters {
11     vector[K] u[I];             // solute feature vectors
12     vector[K] v[J];             // solvent feature vectors
13 }
14
15 model {
16     // Prior: draw feature vectors for all solutes and solvents:
17     for (i in 1:I) {
18         u[i] ~ normal(0, sigma_0);
19     }
20     for (j in 1:J) {
21         v[j] ~ normal(0, sigma_0);
22     }
23
24     // Likelihood: model the probability of ln_gamma_exp as a Cauchy
25     // distribution around the dot product of the feature vectors:
26      for (i in 1:I) {
27         for (j in 1:J) {
28             // Preprocessing uses a sentinel value of -99 for missing entries.
29             if (ln_gamma_exp[i, j] != -99) {
30                 ln_gamma_exp[i, j] ~ cauchy(u[i]' * v[j], lambda);
31             }
32         }
33     }
34 }
```

Figure S.2: Stan code of the probabilistic generative model of the data-driven MCM. Line 29 ensures that the model is only trained on available experimental data, since missing entries of the matrix were set to -99.

Whisky Method

The *whisky* method is proposed in this work as a novel generic approach for hybridizing physical and data-driven prediction methods, and it is applied to predict $\ln \gamma_{ij}^{\infty}$ here. Figure 1 in the manuscript illustrates the idea of the proposed hybrid approach.

The method consists of two steps: in the first step (*distillation* step, purple part of Figure 1 in the manuscript), UNIFAC is employed to predict $\ln \gamma_{ij}^{\infty}$ at 298.15 K in all possible combinations of the considered 240 solutes i and 250 solvents j. With the current publicly accessible parameterization of UNIFAC,[4] approx. 66% of all relevant binary mixtures can be modeled. Hence, a rather dense matrix with approx. 66% observed entries, i.e., UNIFAC predictions for $\ln \gamma_{ij}^{\infty}$, is obtained, cf. Figure S.3.

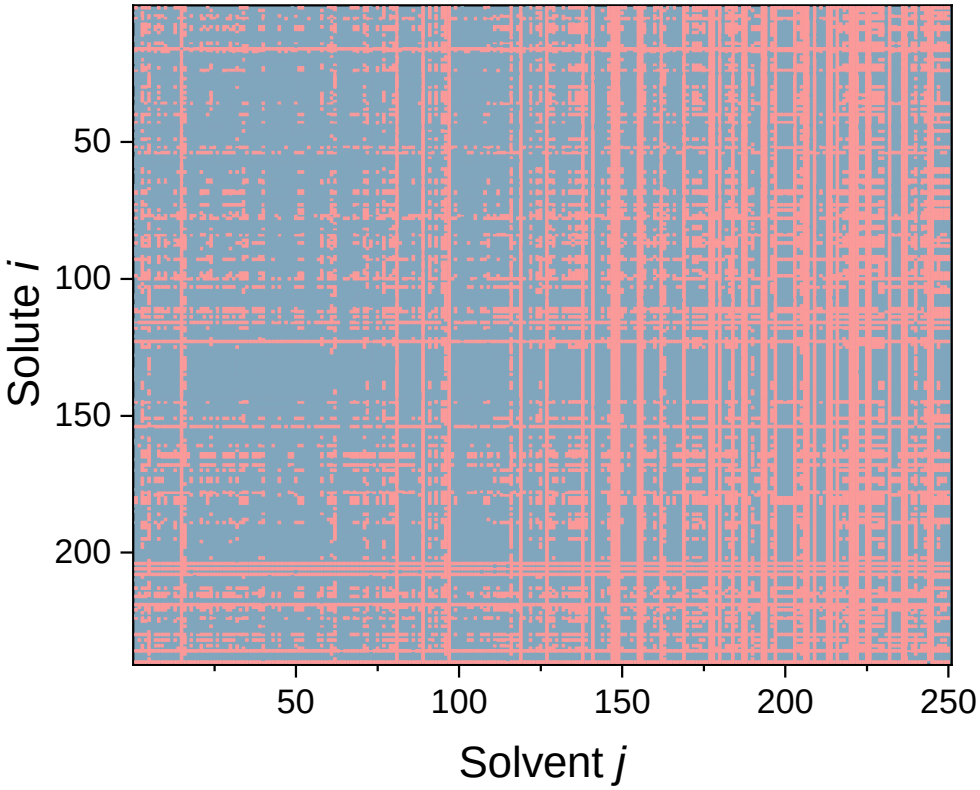

Figure S.3: Matrix representation of all possible binary mixtures of the considered solutes i and solvents j. Blue: UNIFAC can be applied to predict γ_{ij}^{∞}. Red: UNIFAC cannot be applied to predict γ_{ij}^{∞}.

At first, this rather dense matrix is used for training a Bayesian MCM in the distillation

step. In this step, the parameters (component features) and hyperparameters of the model are trained simultaneously. Therefore, a (strongly uninformative) Gaussian hyperprior with mean $\mu_{\mathrm{hp}} = 0$ and standard deviation $\sigma_{\mathrm{hp}} = 100$ was used for all hyperparameters: the mean μ_0 and standard deviation σ_0 of the Gaussian prior and the scale parameter λ of the Cauchy likelihood. Figure S.4 shows the Stan code of the generative model of the distillation step of the whisky method.

The posterior of the distillation step constitutes probability distributions for all component features, thus containing information from the UNIFAC predictions for $\ln \gamma_{ij}^{\infty}$. However, meaningful features were only obtained for the components that can, in principle, be modeled with UNIFAC, i.e., for which UNIFAC predictions were available during the distillation step. These posterior distributions were used to generate informative priors for the subsequent maturation step of the whisky method, the actual training of the approach to the available experimental data (green part of Figure 1 in the manuscript). In detail, the mean of each posterior distribution of the distillation step was adopted and used in combination with a standard deviation of $\sigma_0 = 0.5$ in a Gaussian prior in the maturation step. For those components that cannot be modeled with UNIFAC, i.e., for which no UNIFAC predictions were available during the distillation step, a Gaussian prior with mean $\mu_0 = 0$ and standard deviation $\sigma_0 = 3$ was used in the maturation step. Hence, we used a rather small prior standard deviation, i.e., a rather strong or informative prior, for those components for which we could extract and hand over information from the distillation to the maturation step. In contrast, we used a rather large prior standard deviation, i.e., a rather weak or uninformative prior, for those components for which no a-priori information could be generated with UNIFAC. The actual numbers for σ_0 for the informative and uninformative priors in the maturation step are to a certain degree arbitrary but its ratio matches the ratio of the mean and maximum posterior standard deviation of the distillation step. Furthermore, the proposed whisky method is quite robust with respect to σ_0 in the maturation step. Figure S.5 shows the Stan code of the maturation step, the actual training step, of the whisky method.

```
1  data {
2      int<lower=0> I;                // number of solutes
3      int<lower=0> J;                // number of solvents
4      int<lower=0> K;                // number of latent dimensions
5      real ln_gamma_UNIFAC[I, J];    // matrix of UNIFAC predictions
6      real<lower=0> sigma_hp;        // hyperprior standard deviation
7  }
8
9  parameters {
10     vector[K] u[I];                // solute feature vectors
11     vector[K] v[J];                // solvent feature vectors
12     real mu_0;                     // prior mean
13     real<lower=0> sigma_0;         // prior standard deviation
14     real<lower=0> lambda;          // likelihood scale
15 }
16
17 model {
18     // Fit hyperparameters:
19     mu_0 ~ normal(0, sigma_hp);
20     sigma_0 ~ normal(0, sigma_hp);
21     lambda ~ normal(0, sigma_hp);
22
23     // Prior: draw feature vectors for all solutes and solvents:
24     for (i in 1:I) {
25         u[i] ~ normal(mu_0, sigma_0);
26     }
27     for (j in 1:J) {
28         v[j] ~ normal(mu_0, sigma_0);
29     }
30
31     // Likelihood: model the probability of ln_gamma_UNIFAC as a Cauchy
32     // distribution around the dot product of the feature vectors:
33     for (i in 1:I) {
34         for (j in 1:J) {
35             // Preprocessing uses a sentinel value of -99 for missing entries.
36             if (ln_gamma_UNIFAC[i, j] != -99) {
37                 ln_gamma_UNIFAC[i, j] ~ cauchy(u[i]' * v[j], lambda);
38             }
39         }
40     }
41 }
```

Figure S.4: Stan code of the probabilistic generative model of the distillation step of the proposed whisky method. Line 36 ensures that the model in only trained on available UNIFAC predictions, since missing entries of the matrix were set to -99.

```
1  data {
2      int<lower=0> I;             // number of solutes
3      int<lower=0> J;             // number of solvents
4      int<lower=0> K;             // number of latent dimensions
5      real ln_gamma_exp[I, J];    // matrix of experimental activity coefficients
6      vector[K] mu_u[I];          // prior mean of solute features
7      vector[K] mu_v[J];          // prior mean of solvent features
8      vector[K] sigma_u[I];       // prior standard deviation of solute features
9      vector[K] sigma_v[J];       // prior standard deviation of solvent features
10     real<lower=0> lambda;       // likelihood scale
11 }
12
13 parameters {
14     vector[K] u[I];             // solute feature vectors
15     vector[K] v[J];             // solvent feature vectors
16 }
17
18 model {
19     // Prior: draw feature vectors for all solutes and solvents:
20     for (i in 1:I) {
21         u[i] ~ normal(mu_u[i], sigma_u[i]);
22     }
23     for (j in 1:J) {
24         v[j] ~ normal(mu_v[j], sigma_v[j]);
25     }
26
27     // Likelihood: model the probability of ln_gamma_UNIFAC as a Cauchy
28     // distribution around the dot product of the feature vectors:
29     for (i in 1:I) {
30         for (j in 1:J) {
31             // Preprocessing uses a sentinel value of -99 for missing entries.
32             if (ln_gamma_UNIFAC[i, j] != -99) {
33                 ln_gamma_UNIFAC[i, j] ~ cauchy(u[i]' * v[j], lambda);
34             }
35         }
36     }
37 }
```

Figure S.5: Stan code of the probabilistic generative model of the maturation step of the proposed whisky method. Line 32 ensures that the model is only trained on available experimental data, since missing entries of the matrix were set to -99.

The proposed whisky method can be applied to predict $\ln \gamma_{ij}^{\infty}$ for any mixture of the considered solutes and solvents. Hence, predictions for the complete data set on $\ln \gamma_{ij}^{\infty}$ from our previous work[1] that contains 4,094 experimental data points and covers 240 solutes i and 250 solvents j can be obtained with the whisky approach. In the manuscript, we only show results for the data points that can also be predicted with UNIFAC for the reason of comparability. However, the performance of the whisky method to predict all available 4,094 data points is demonstrated below and compared to the data-driven base method MCM.[1]

Bagging

For obtaining predictions with the bootstrap aggregation (bagging) approach, the arithmetic mean of the predictions of the two base methods UNIFAC[3,4] and data-driven MCM[1] was calculated for each available data point:

$$\ln(\gamma_{ij}^{\infty})^{\text{Bagging}} = \frac{1}{2}\left(\ln(\gamma_{ij}^{\infty})^{\text{UNIFAC}} + \ln(\gamma_{ij}^{\infty})^{\text{MCM-data}}\right) \tag{S.3}$$

For the UNIFAC predictions, an inhouse MATLAB implementation with the latest publicly accessible parameterization[4] was used. The MCM predictions were adopted from our previous work.[1]

With the data-driven MCM, predictions for all 4,094 available experimental data points are obtained; with UNIFAC, predictions for only about 80% of the these can be obtained. Hence, the applicability of the bagging approach is directly limited by the applicability of UNIFAC. The relevant data set covers 231 solutes and 205 solvents.

Boosting

Additionally, another established machine learning ensemble method, namely boosting, is adopted here as hybrid baseline. The basis for the boosting method is the matrix containing UNIFAC predictions for $\ln \gamma_{ij}^{\infty}$ for the mixtures for which also experimental data on $\ln \gamma_{ij}^{\infty}$

are available. As described above, UNIFAC yields only predictions for about 80% of the experimental data, which can also be arranged in a partially observed matrix, whose rows and columns correspond to the solutes i and solvents j, respectively. Since the results of the boosting method were also evaluated using leave-one-out cross-validation, at least two observed entries, i.e., entries for which experimental data *and* UNIFAC prediction are available, per row and column were required. The data set therefore slightly reduced further, covering 224 solutes and 205 solvents. For all applicable mixtures, the residuals r_{ij} of UNIFAC, i.e., the differences between the experimental data points and the UNIFAC predictions, were calculated:

$$r_{ij} = \ln(\gamma_{ij}^{\infty})^{\text{exp}} - \ln(\gamma_{ij}^{\infty})^{\text{UNIFAC}} \tag{S.4}$$

These UNIFAC residuals were arranged in a partially observed matrix to which the concept of Bayesian matrix completion was applied in the second step. Hence, previously unknown features of the solutes and solvents that describe the deviation of the UNIFAC predictions from the experimental data were learned from the training data. For all parameters a Gaussian prior with mean $\mu_0 = 0$ and standard deviation $\sigma_0 = 1$ was used here. Furthermore, a Cauchy likelihood with scale parameter $\lambda = 0.15$ was used. Figure S.6 shows the Stan code for the boosting method.

Following the concept of leave-one-out cross-validation, the solute and solvent features were trained to all r_{ij} except for one, which was then considered as test data point and predicted. Each data point served once as test data point. The prediction of the logarithmic activity coefficient $\ln(\gamma_{ij}^{\infty})^{\text{Boosting}}$ was calculated in a straightforward manner:

$$\ln(\gamma_{ij}^{\infty})^{\text{Boosting}} = \ln(\gamma_{ij}^{\infty})^{\text{UNIFAC}} + \mu_{u_i} \cdot \mu_{v_j} \tag{S.5}$$

where $\ln(\gamma_{ij}^{\infty})^{\text{UNIFAC}}$ is the UNIFAC prediction for the activity coefficient, and μ_{u_i} and μ_{v_j} are the posterior means of the corresponding solute and solvent feature vectors, respectively, obtained from the MCM of the residuals r_{ij}.

```
1  data {
2      int<lower=0> I;                // number of solutes
3      int<lower=0> J;                // number of solvents
4      int<lower=0> K;                // number of latent dimensions
5      real ln_gamma_exp[I, J];       // matrix of experimental activity coefficients
6      real ln_gamma_UNIFAC[I, J];    // matrix of UNIFAC predictions
7      real<lower=0> sigma_0;         // prior standard deviation
8      real<lower=0> lambda;          // likelihood scale
9  }
10
11 parameters {
12     vector[K] u[I];                // solute feature vectors
13     vector[K] v[J];                // solvent feature vectors
14 }
15
16 model {
17     // Prior: draw feature vectors for all solutes and solvents:
18     for (i in 1:I) {
19         u[i] ~ normal(0, sigma_0);
20     }
21     for (j in 1:J) {
22         v[j] ~ normal(0, sigma_0);
23     }
24
25     // Likelihood: model the probability of ln_gamma_exp as a Cauchy
26     // distribution around the dot product of the feature vectors + ln_gamma_UNIFAC:
27     for (i in 1:I) {
28         for (j in 1:J) {
29             // Preprocessing uses a sentinel value of -99 for missing entries.
30             if (ln_gamma_exp[i, j] != -99 && ln_gamma_UNIFAC[i, j] != -99) {
31                 ln_gamma_exp[i, j] ~ cauchy(u[i]' * v[j] + ln_gamma_UNIFAC[i, j], lambda);
32             }
33         }
34     }
35 }
```

Figure S.6: Stan code of the probabilistic generative model of the boosting method. Line 30 ensures that the model is only trained on available data, since missing entries of the matrix were set to -99.

Additional Results

Results of Bagging and Boosting

In Figure S.7, the predictions with the hybrid baselines bagging and boosting are represented in parity plots and compared to the predictions of the base methods UNIFAC and data-driven MCM. Both hybrid approaches only partially compensate for outliers of the data-driven MCM but severely suffer from outliers of UNIFAC. The worst outliers of UNIFAC, bagging, and boosting lie outside the depicted ranges, cf. Figure S.8. Slight improvements with bagging and boosting compared to the data-driven MCM can only be achieved, if the worst UNIFAC outliers are ignored, cf. the coefficients of determination R^2 in Figure S.7 and Figure 2 in the manuscript.

The slight improvements that are possible with the bagging approach compared to MCM if (and only if) the worst UNIFAC outliers are ignored can mainly be attributed to error cancellations by averaging the predictions of the two base methods (UNIFAC and MCM). For approx. 67% of the applicable data points, better predictions are obtained with bagging than with UNIFAC, whereas for approx. 52% of the data points, the predictions with bagging are better than those of the data-driven MCM. Hence, the bagging approach improves the predictions of both base methods for more data points than impairs the predictions. This can be explained by the observation that for almost 19% of the data points, an improved prediction accuracy compared to *both* UNIFAC and MCM is observed. Hence, for these 19%, the bagging approach benefits from the effect of error cancellation, as the respective data points are overestimated by MCM and underestimated by UNIFAC, or vice versa. Incidentally, it is in the nature of the bagging approach that it cannot impair the predictions of UNIFAC *and* MCM for a specific data point at the same time.

However, in all cases, i.e., with or without ignoring the worst UNIFAC outliers, the performances of bagging and boosting are significantly worse than the performance of the proposed whisky method, cf. Figures 2 and 3 in the manuscript.

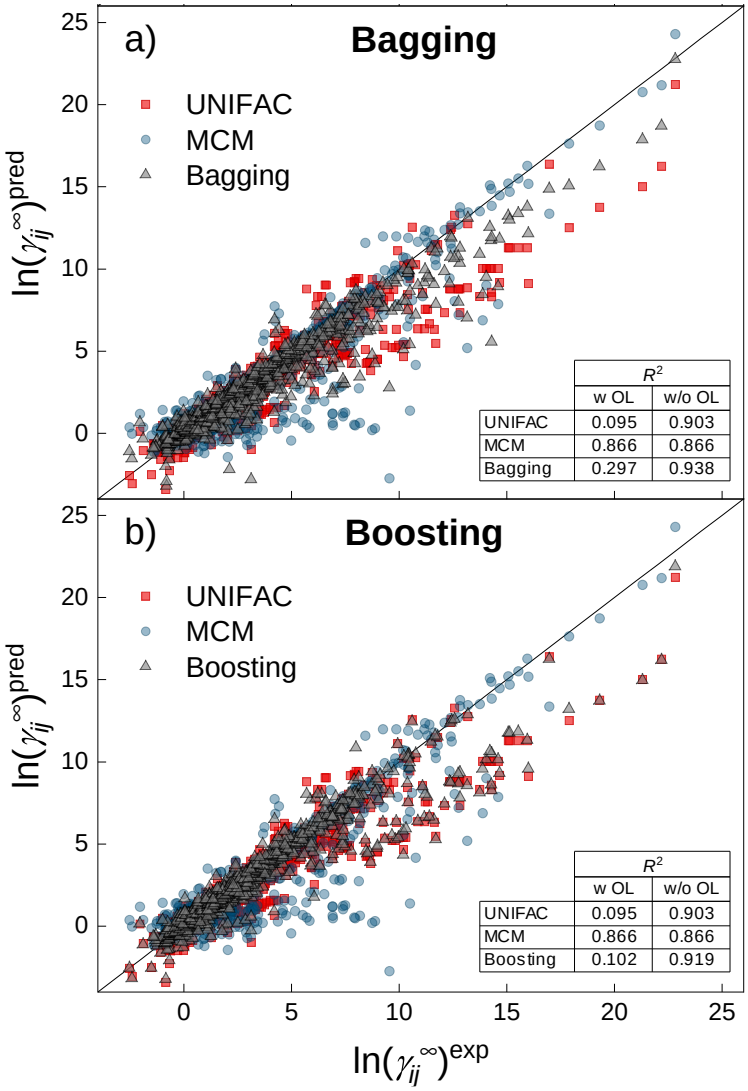

Figure S.7: Parity plots of the predictions (pred) for $\ln \gamma_{ij}^{\infty}$ with the bagging (a) and boosting (b) approaches over the corresponding experimental values (exp) and comparison to UNIFAC and data-driven MCM. Coefficients of determination R^2 (higher is better, 1 implies perfect correlation) are given, both including and excluding the worst eight UNIFAC outliers (OL).

UNIFAC Outliers

In Figure S.8, predictions for all applicable $\ln \gamma_{ij}^{\infty}$ *including* the worst eight UNIFAC outliers (OL) with all studied methods are shown in a parity plot representation. For these UNIFAC outliers, marked by red boxes in Figure S.8, the hybrid methods bagging and boosting, shown in panels b) and c), respectively, give poor predictions, while the whisky method, shown in panel a), as well as the purely data-driven MCM are demonstrated to be much more robust and do not exhibit such outliers.

The performance of UNIFAC strongly depends on the quality of the fitted binary group-interaction parameters. The worst eight outliers of UNIFAC are associated to binary mixtures of a cyclic alkane, cyclic alkene, or furane as solute and a heterocyclic compound in which the ring structure is formed by carbon and nitrogen as solvent. According to the group-division scheme of UNIFAC,[3,4] all aforementioned solutes contain at least one 'CY-CH2' main group (UNIFAC main group no. 42), all aforementioned solvents contain at least one 'PYRIDINE' main group (UNIFAC main group no. 18). We therefore attribute the poor predictions of UNIFAC for these outliers mainly to the UNIFAC group-interaction parameters between the 'CY-CH2' and the 'PYRIDINE' group, which have presumably been overfitted to parts of the data that were used for training UNIFAC (and that are not necessarily part of the data set considered here). We assume that this is not an isolated case but likely to occur for other group-interaction parameters for UNIFAC as well, depending on the data sets that are studied. As the data-driven MCM is an orthogonal approach to UNIFAC, it is conclusive that it does not suffer from the same difficulties. Furthermore, the prior distributions in the probabilistic approach of the data-driven MCM serve as regularization terms and can therefore be expected to prevent overfitting of the MCM to single data points or parts of the data set.

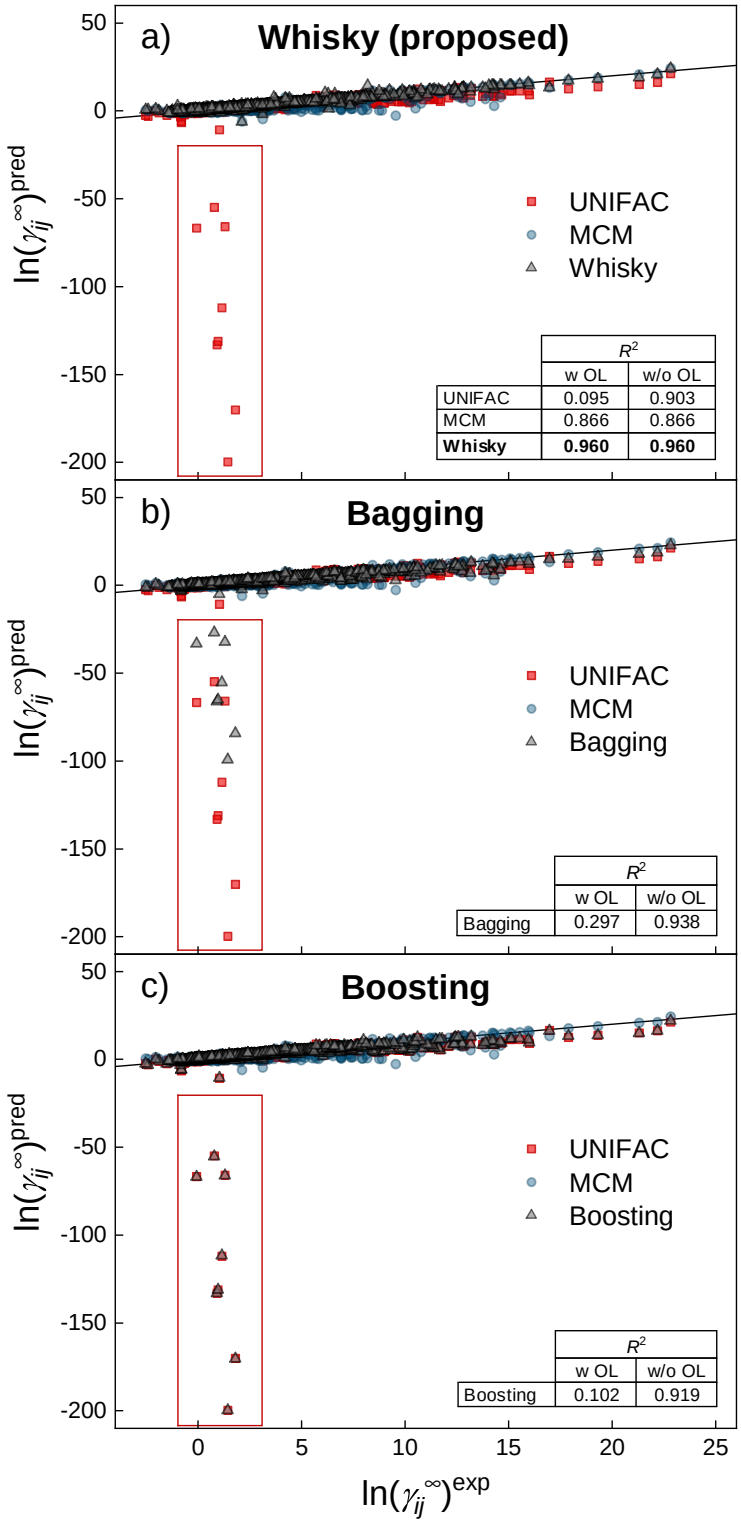

Figure S.8: Parity plots of the predictions (pred) for $\ln \gamma_{ij}^{\infty}$ with the hybrid approaches over the corresponding experimental values (exp) and comparison to UNIFAC and MCM. a) whisky (proposed), b) bagging, c) boosting. The worst eight UNIFAC outliers (OL) are marked by red boxes. Coefficients of determination R^2 (higher is better, 1 implies perfect correlation) are given, both including and excluding OL.

The proposed hybrid whisky approach, like the data-driven MCM, does not exhibit such outliers, although it, in contrast to the data-driven MCM, considers information from UNIFAC. In whisky, information from UNIFAC is taken into account in the prior distributions used in the *maturation* step, cf. Figure 1 in the manuscript. The nonzero variance of the Gaussian priors that we have used here allows the whisky method to 'correct' poor UNIFAC predictions by combining them with experimental data evidence in the maturation step. This procedure seems to work extremely efficiently. Ultimately, we emphasize again that even with omitting these systematic outliers of UNIFAC in the evaluation, the proposed whisky method significantly outperforms the individual and hybrid baselines studied here.

MCM and Whisky Predictions for All Available Data

As described above, the whisky method yields predictions for all available experimental $\ln \gamma_{ij}^{\infty}$ for the considered solutes i and solvents j, while the bagging and boosting methods are limited by the applicability of UNIFAC. In Figure S.9, the performance of the whisky approach to predict $\ln \gamma_{ij}^{\infty}$ for all 4,094 available data points is demonstrated and compared to the performance of the data-driven MCM. Significant improvements with respect to mean square error (MSE), mean absolute deviation (MAD), and coefficient of determination (R^2) are obtained with the whisky method.

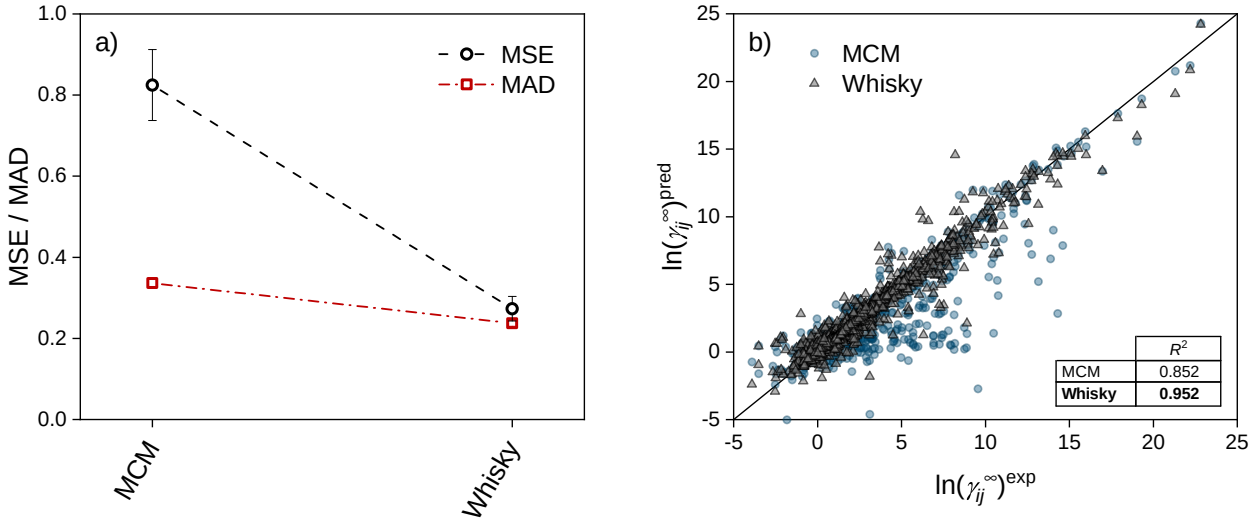

Figure S.9: Comparison of the data-driven MCM and the proposed whisky method considering all 4,094 available experimental data points for $\ln \gamma_{ij}^{\infty}$. a) Mean square error (MSE) and mean absolute deviation (MAD) of the predictions; lower is better for both metrics, error bars represent the standard errors of the means. b) Parity plot of the predictions (pred) over corresponding experimental values (exp) and coefficients of determination R^2 (higher is better, 1 implies perfect correlation).

Influence of Latent Dimension

Figure S.10 shows MSE and MAD scores that are obtained with the whisky method considering all available experimental data points for different latent dimensions K, specifically for varying numbers of learned solute and solvent features ranging from two to six. For $K = 4$, which was used to obtain all other results throughout this work (cf. Section 'Model Details' in the ESI), and $K = 5$, very similar scores are obtained. Also the scores for $K = 6$ are similar, which indicates that the whisky method is rather robust to small enlargements of K. However, the slightly worse MSE score for $K = 6$ indicates commencing overfitting at larger numbers for K. Hence, at very large numbers of K, the method is likely to overfit to the training data dropping predictive performance (on unseen test data) due to too high flexibility. On the other hand, also lowering the number of K can result in significant deterioration of the scores. For $K = 3$, only slightly worse scores, for $K = 2$, substantially worse scores are observed. This indicates that for very small numbers of K, the approach is insufficiently flexible for describing the data well. Hence, only two features per solute and

solvent are not adequate for characterizing the components well, which results in poor scores due to underfitting.

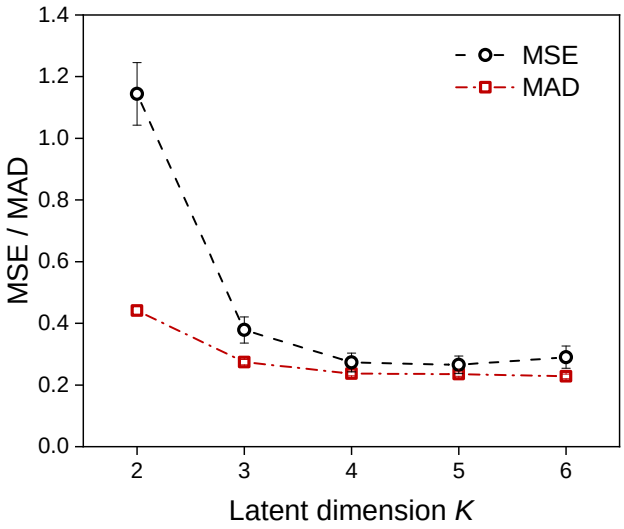

Figure S.10: Influence of the latent dimension K on mean square error (MSE) and mean absolute deviation (MAD) of the predictions with the proposed whisky method for all 4,094 available experimental data points for $\ln \gamma_{ij}^{\infty}$; lower is better for both metrics, error bars represent the standard errors of the means.

The optimal number of K strongly depends on the data that are considered and can, if necessary, be determined by cross-validation. In this work, we have simply adopted $K = 4$ from our previous work in which we introduced the data-driven MCM and which is also a good choice for the whisky method as demonstrated in Figure S.10. We consider the observed robustness towards small variations of K ($K = 3$ to $K = 6$ here) as a major strength of the whisky method, as it shows excellent predictive performance on $\ln \gamma^{\infty}$ data without requiring extensive hyperparameter optimization.

We note that for applying the whisky method to other physicochemical properties than activity coefficients, significantly different numbers for the latent dimension K might be required. I.e., the number of features that are required for adequately characterizing components with regard to a specific property is likely to depend on the considered property. The whisky method is not restricted to specific numbers of K; in principle any number can be chosen, which can be determined by cross-validation to prevent both under- and overfitting

as described above. Larger numbers of K might hamper the computation time required to train the method, which is, however, for $K = 4$ and the data set considered here, in the range of seconds or minutes (using a custom laptop).

References

(1) Jirasek, F.; Alves, R. A. S.; Damay, J.; Vandermeulen, R. A.; Bamler, R.; Bortz, M.; Mandt, S.; Kloft, M.; Hasse, H. Machine Learning in Thermodynamics: Prediction of Activity Coefficients by Matrix Completion. *J. Phys. Chem. Lett.* **2020**, *11*, 981–985.

(2) Onken, U.; Rarey-Nies, J.; Gmehling, J. The Dortmund Data Bank: A Computerized System for the Retrieval, Correlation, and Prediction of Thermodynamic Properties of Mixtures. *Int. J. Thermophys.* **1989**, *10*, 739–747.

(3) Weidlich, U.; Gmehling, J. A Modified UNIFAC Model. 1. Prediction of VLE, h^E, and γ^∞. *Ind. Eng. Chem. Res.* **1987**, *26*, 1372–1381.

(4) Constantinescu, D.; Gmehling, J. Further Development of Modified UNIFAC (Dortmund): Revision and Extension 6. *J. Chem. Eng. Data* **2016**, *61*, 2738–2748.

(5) Blei, D. M.; Kucukelbir, A.; McAuliffe, J. D. Variational Inference: A Review for Statisticians. *J. Am. Stat. Assoc.* **2017**, *112*, 859–877.

(6) Zhang, C.; Bütepage, J.; Kjellström, H.; Mandt, S. Advances in Variational Inference. *IEEE. T. Pattern. Anal.* **2019**, *41*, 2008–2026.

(7) Kucukelbir, A.; Tran, D.; Ranganath, R.; Gelman, A.; Blei, D. M. Automatic Differentiation Variational Inference. *J. Mach. Learn. Res.* **2017**, *18*, 1–45.

(8) Carpenter, B.; Gelman, A.; Hoffman, M. D.; Lee, D.; Goodrich, B.; Betancourt, M.; Brubaker, M.; Guo, J.; Li, P.; Riddell, A. Stan: A Probabilistic Programming Language. *J. Stat. Softw.* **2017**, *76*, 1–32.

(9) Murphy, K. P. *Machine Learning: A Probabilistic Perspective*; MIT press, 2012.