Balancing Molecular Information and Empirical Data in the Prediction of Physico-Chemical Properties

Johannes Zenn [1 2 3] Dominik Gond [4] Fabian Jirasek [4] Robert Bamler [1 2]

Abstract

Predicting the physico-chemical properties of pure substances and mixtures is a central task in thermodynamics. Established prediction methods range from fully physics-based ab-initio calculations, which are only feasible for very simple systems, over descriptor-based methods that use some information on the molecules to be modeled together with fitted model parameters (e.g., quantitative-structure-property relationship methods or classical group contribution methods), to representation-learning methods, which may, in extreme cases, completely ignore molecular descriptors and extrapolate only from existing data on the property to be modeled (e.g., matrix completion methods). In this work, we propose a general method for combining molecular descriptors with representation learning using the so-called expectation maximization algorithm from the probabilistic machine-learning literature, which uses uncertainty estimates to trade off between the two approaches. The proposed hybrid model exploits chemical structure information using graph neural networks, but it automatically detects cases where structure-based predictions are unreliable, in which case it corrects them by representation-learning based predictions that can better specialize to unusual cases. The effectiveness of the proposed method is demonstrated using the prediction of activity coefficients in binary mixtures as an example. The results are compelling, as the method significantly improves predictive accuracy over the current state of the art, showcasing its potential to advance the prediction of physico-chemical properties in general.

1. Introduction

Information on physico-chemical properties is crucial for the conceptual design and optimization of processes in many industries, including chemistry, pharmacy, and biotechnology. Among the most important thermodynamic properties are the activity coefficients of the components in a mixture, which describe the deviation of a mixture from the ideal mixture and enable the accurate prediction of reaction and phase equilibria of mixtures. Activity coefficients at infinite dilution are more sensitive thermodynamic properties than activity coefficients at finite concentration (and the subsequently calculated reaction and phase equilibria). Knowing the activity coefficients at infinite dilution allows to predict the activity coefficients in binary mixtures of any finite concentration as well as the activity coefficients in multi-component mixtures. Unfortunately, measuring thermodynamic properties of mixtures, such as activity coefficients, is costly and time-consuming, and the number of relevant mixtures exceeds the ones that can be studied experimentally by orders of magnitude (Jirasek & Hasse, 2023). Consequently, prediction methods for thermodynamic properties of mixtures are paramount. Recently, research on such prediction methods has split into two branches.

On the one hand, descriptor-based methods correlate information on the molecules to be modeled with properties of interest. Among these, group-contribution methods, which use the composition of the components in terms of structural groups as molecular descriptors and whose underlying equations are usually derived from physical theories, are still the gold standard for property prediction in many (industrial) fields (Gmehling et al., 2015; Jirasek et al., 2023). The most successful group-contribution method for predicting activity coefficients is UNIFAC (Fredenslund et al., 1975; Weidlich & Gmehling, 1987; Constantinescu & Gmehling, 2016), which is available in different versions and established in most process simulation software. Besides the physics-based group-contribution methods, also other descriptor-based methods that rely on various descriptors, such as molecular weight or surface area, or boiling point, have been proposed for predicting activity coefficients (Katritzky et al.,

[1]AI Center Tübingen [2]University of Tübingen [3]IMPRS-IS [4]Laboratory of Engineering Thermodynamics (LTD), RPTU Kaiserslautern. Correspondence to: Johannes Zenn <johannes.zenn@uni-tuebingen.de>.

Preprint.

2010; Estrada et al., 2006; Giralt et al., 2004; Mitchell & Jurs, 1998; Paduszynski, 2016; Ajmani et al., 2008; Behrooz & Boozarjomehry, 2017; Medina et al., 2022).

In statistics parlance, such descriptor-based models are called *parametric* models as they fit model parameters that affect several related components (e.g., those sharing a given structural group). Thus, parametric models can leverage statistical strength across chemically similar components.

On the other hand, recent idea transfer from the machine learning community has led to an alternative approach to predicting activity coefficients and other mixture properties, which is based solely on learned representations without relying on descriptors. So-called matrix completion methods (MCMs) (Jirasek et al., 2020a;b; Hayer et al., 2022; Großmann et al., 2022; Damay et al., 2021) ignore the chemical structure of components and fit individual representation vectors for each mixture component that appears in a set of available experimental data. While this approach makes MCMs more flexible than descriptor-based methods, it prevents them from exploiting structural similarities across components, and, therefore, from extrapolating to new components. In statistics parlance, one says that MCMs are "*nonparametric* in the components" (note that nonparametric models tend to have many parameters, similar to how a "stepless" controller has infinitely many steps).

It was shown empirically (Jirasek et al., 2020a;b; 2022) that purely nonparametric MCMs make more accurate predictions for activity coefficients than the descriptor-based (parametric) state-of-the-art UNIFAC. However, since each fitted parameter (i.e., each representation vector) in an MCM only describes a single component, MCMs can only make predictions for mixtures where each component appears in some (other) mixtures in the available experimental data ("in-domain predictions"). By contrast, descriptor-based methods can exploit the structural similarity of components to extrapolate to components that appear in no mixture in the available experimental data ("out-of-domain predictions").

In this work, we propose a new method for predicting activity coefficients in binary mixtures that combines the strengths of both the parametric (descriptor-based) and the nonparametric (representation-based) approach while avoiding their respective weaknesses. To do this, we phrase both a parametric and a nonparametric model in a probabilistic framework, and we fit them jointly using the so-called variational expectation maximization (variational EM) algorithm. The algorithm finds an optimal compromise between the parametric and the nonparametric part of a model, taking into account how confident each part is in its fitted or predicted parameters (discussed in Section 2). Our evaluation shows that weighing off the respective confidences of the parametric and nonparametric models indeed improves the accuracy of both in-domain and out-of-domain predictions.

While this paper focuses on the concrete task of predicting activity coefficients of pure solutes at infinite dilution in pure solvents at room temperature, the proposed method can, e.g., be generalized to arbitrary temperatures and concentrations following the procedure described in Jirasek et al. (2022), and to other thermodynamic properties of binary mixtures by fitting it to a corresponding dataset. More generally, we argue that the variational EM algorithm is a valuable tool in thermodynamic modeling since it allows for combining the strengths of descriptor-based (parametric) and representation-based (nonparametric) models, which is a powerful approach beyond the modeling single thermodynamic properties of binary mixtures.

In the remaining sections of this paper, we first formalize the problem setup, present the proposed method, and discuss several variants of its concrete execution. We then empirically evaluate the accuracy of predicted activity coefficients and compare them to existing methods and simplified variants of our proposed method (ablation studies).

2. Method

2.1. Problem Setting

As in Jirasek et al. (2020a), we start from a data set of 4094 measured activity coefficients $\gamma_{i,j}^{\infty}$ of solutes i at infinite dilution in solvents j at $298.15\ (\pm 1)$ K. We use the same data set as in previous work (Jirasek et al., 2020a), which was taken from the Dortmund Data Bank (DDB) (Onken et al., 1989), the largest database for phsico-chemical properties covering the most relevant molecular components for technical processes. We refrain from using synthetic datasets because this only demonstrates how well a method can reproduce an available model and does not results in a practically useful new model. The DDB dataset is illustrated in the yellow/black matrix on the right of Figure 1 (which is discussed in more detail in Section 2.2 below). The matrix has $M = 240$ rows and $N = 250$ columns corresponding to the M distinct solutes and N distinct solvents that appear in the data set, and each black pixel indicates an available experimental data point $\gamma_{i,j}^{\infty}$. Our goal is to predict activity coefficients for the yellow parts of the matrix ("in-domain predictions"), and to also extend the rows and columns of the matrix, i.e., predict activity coefficients that involve yet unstudied solutes or solvents ("out-of-domain predictions").

A previous deep-learning-based approach (Medina et al., 2022) addressed this prediction problem with a combination of three neural networks. The first two networks are so-called graph neural networks (GNNs) that take as input the molecular graph structures of the solute and solvent, respectively, i.e., each atom kind, their hybridizations and formal charges, and the type of bond between each pair of atoms. The GNNs map the molecular graphs to so-called abstract

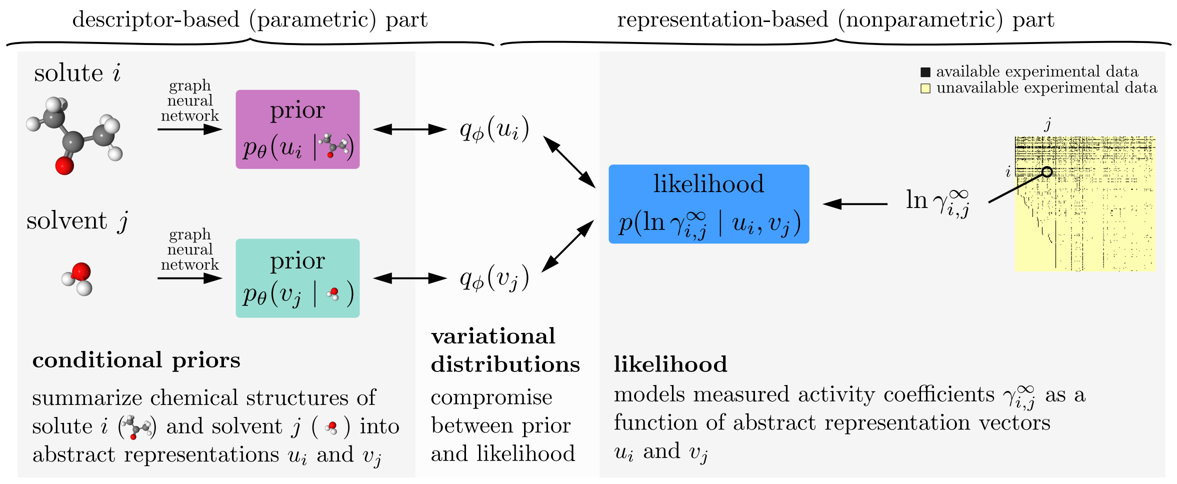
descriptor-based (parametric) part
representation-based (nonparametric) part
solute i
graph neural network
prior $p_\theta(u_i \mid \cdot)$
$q_\phi(u_i)$
solvent j
graph neural network
prior $p_\theta(v_j \mid \cdot)$
$q_\phi(v_j)$
likelihood $p(\ln \gamma_{i,j}^\infty \mid u_i, v_j)$
$\ln \gamma_{i,j}^\infty$
■ available experimental data
□ unavailable experimental data
j
i
conditional priors
summarize chemical structures of solute i ( ) and solvent j ( ) into abstract representations u_i and v_j
variational distributions
compromise between prior and likelihood
likelihood
models measured activity coefficients $\gamma_{i,j}^\infty$ as a function of abstract representation vectors u_i and v_j

Figure 1. Model and data flow for training the proposed model. Left: graph neural networks take chemical structure information and output the parameters of conditional prior probability distributions (Equation (1)) over abstract representation vectors. Right: the likelihood (Equation (2)) models how well given representation vectors explain experimentally measured activity coefficients $\gamma_{i,j}^\infty$. We use variational EM (Sections 2.3 and 2.4) to fit the neural network weights θ (parametric, descriptor-based part), and to find variational distributions for each solute and solvent (nonparametric, representation-based part).

representation vectors $\boldsymbol{u} \in \mathbb{R}^K$ and $\boldsymbol{v} \in \mathbb{R}^K$, respectively, where the dimension K is a modeling choice. The third neural network combines $\boldsymbol{u}$ and $\boldsymbol{v}$ and outputs a prediction for the activity coefficient for the respective solute at infinite dilution in the respective solvent. This existing approach can perform out-of-domain predictions because the neural networks can extrapolate to new molecular structures as long as they share some common substructures with the ones in the training data. But this approach has the downside that it uses an entirely parametric model, i.e., it is limited by the expressiveness of the neural networks and cannot make any exceptions in case some anomalous components behave very differently than structurally similar components. Our proposed method, described below, accounts for exceptions with anomalous behavior in a nonparametric way.

2.2. Probabilistic Model

Like in Medina et al. (2022), discussed in Section 2.1 above, our proposed model has a descriptor-based part (left half of Figure 1) that processes the chemical structures of the solute and solvent independently using two neural networks (one for solutes and one for solvents), and our main results were also obtained by using GNNs here (similar to the model of Medina et al. (2022)). Unlike in the previous work, these neural networks parameterize probabilistic models, i.e., their outputs are not representation vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ but instead parameters that define so-called conditional prior probability distributions $p_\theta(\boldsymbol{u} \mid r)$ and $p_\theta(\boldsymbol{v} \mid s)$, respectively. Here, θ are the neural network weights, and the bar "$|$" denotes conditioning on the chemical structure r and s of the solute and solvent, respectively. Specifically, the conditional priors in our empirical analysis are normal distributions,

$$\begin{aligned} p_\theta(\boldsymbol{u} \mid r) &= \mathcal{N}\big(\boldsymbol{u}; {}^{u}\mu_\theta(r), \operatorname{diag}({}^{u}\sigma_\theta^2(r))\big); \\ p_\theta(\boldsymbol{v} \mid s) &= \mathcal{N}\big(\boldsymbol{v}; {}^{v}\mu_\theta(s), \operatorname{diag}({}^{v}\sigma_\theta^2(s))\big) \end{aligned} \tag{1}$$

where the means ${}^{u}\mu_\theta(r), {}^{v}\mu_\theta(s) \in \mathbb{R}^K$ and variances ${}^{u}\sigma_\theta^2(r), {}^{v}\sigma_\theta^2(s) \in \mathbb{R}_{>0}^K$ are extracted from the outputs of the two neural networks. Here, $\operatorname{diag}({}^{u/v}\sigma_\theta^2(\,\cdot\,))$ is a covariance matrix with the components of ${}^{u/v}\sigma_\theta^2(\,\cdot\,)$ on its diagonal, and zeros on all off-diagonal entries. The inference algorithm, described in Section 2.3 and Section 2.4 below, ensures that ${}^{u/v}\sigma_\theta^2(\,\cdot\,)$ estimates an uncertainty region around the corresponding mean prediction ${}^{u/v}\mu_\theta(\,\cdot\,)$ of the parametric part of the model. These uncertainty estimates affect how strongly the parametric part of the model constrains ("regularizes") the nonparametric part of the model during training which we describe next.

The representation-based (nonparametric) part of our model is a probabilistic MCM (Jirasek et al., 2020a). It represents each solute i and each solvent j that appears in the experimental data with an individual representation vector $\boldsymbol{u}_i, \boldsymbol{v}_j \in \mathbb{R}^K$, respectively, which it uses to predict the activity coefficients $\gamma_{i,j}^\infty$. Since activity coefficients range over several orders of magnitude, we model their logarithm, $\ln \gamma_{i,j}^\infty$. We use a simple Gaussian likelihood,

$$p(\ln \gamma_{i,j}^\infty \mid \boldsymbol{u}_i, \boldsymbol{v}_j) = \mathcal{N}(\ln \gamma_{i,j}^\infty; \boldsymbol{u}_i \cdot \boldsymbol{v}_j, \lambda^2) \tag{2}$$

where "$\cdot$" denotes the dot product and $\lambda = 0.15$ as proposed in previous work (Jirasek et al., 2020a). While more expressive likelihoods are compatible with our setup, we found the simple choice of Equation (2) to be sufficient.

2.3. Fitting the Model: Intuition

While the inference algorithm that we use is easy to implement (see Algorithm 1 discussed in Section 2.4 below), understanding why it works requires more explanation. We therefore first motivate the algorithm in this section before formalizing it mathematically in Section 2.4.

We propose to train the nonparametric and parametric parts of the model jointly using the so-called variational expectation maximization (variational EM) (Dempster et al., 1977; Beal & Ghahramani, 2003) algorithm. Variational EM allows us to fit a model that can generalize across components with similar chemical structures while still being able to learn exceptions for individual components where the experimental data shows evidence for anomalous behavior.

The arrows in Figure 1 show the direction of data flow in the algorithm. It concurrently fits both the neural network weights θ of the conditional priors and a so-called variational distribution $q_\phi(\boldsymbol{u}_i)$ and $q_\phi(\boldsymbol{v}_j)$ for each solute i and each solvent j that appears in the experimental data. The weights θ of the conditional priors are fitted to model the data as well as one can with a parametric model. By contrast, the variational distributions are fitted in a nonparametric way. They are fitted to find a compromise between the conditional priors (which can share statistical strength across chemically similar components but cannot make exceptions for anomalous cases) and the experimental data (which may contain evidence for anomalous behaviors, but which is often scarce and generally affected by measurement errors).

Conceptual Remark on Empirical Bayes Methods. Readers who are experienced with Bayesian inference may find it strange that we fit the prior distribution to the data. In normal Bayesian inference, one seeks the posterior distribution of some experimental data under a given probabilistic model, and one assumes that the prior of the probabilistic model is given (e.g., informed by expert knowledge). Variational EM falls into the class of so-called empirical Bayes methods, which differ from normal Bayesian inference in that they estimate the prior distribution from the data as well. This would be an underspecified problem if the prior was unconstrained, in which case the prior would overfit to the data, and the resulting posterior would be equal to the prior and thus also overfit, i.e., perfectly explain the available data but fail to generalize beyond it. To avoid this collapse of empirical Bayes, one has to constrain the prior to a smaller class of distributions than the posterior.

In our setup, the necessary constraint on the prior comes from the finite expressiveness of the neural networks: unless the neural network for, e.g., solute representation vectors $\boldsymbol{u}_i$ is exorbitantly large, it cannot output completely independent prior parameters $\left({}^u\mu_\theta(r_i), {}^u\sigma^2_\theta(r_i)\right)$ for all solutes i in the dataset. Thus, fitting the neural network weights θ cannot perfectly overfit the prior to the data. By contrast, the variational distributions $q_\phi(\boldsymbol{u}_i)$ and $q_\phi(\boldsymbol{v}_j)$ are fitted nonparametrically, i.e., with individual parameters for each solute and solvent. The reason why these do not perfectly overfit the data is because they are not fitted solely to the data but instead obtained by (approximate) Bayesian inference with the (non-overfitting) prior (explained in Section 2.4).

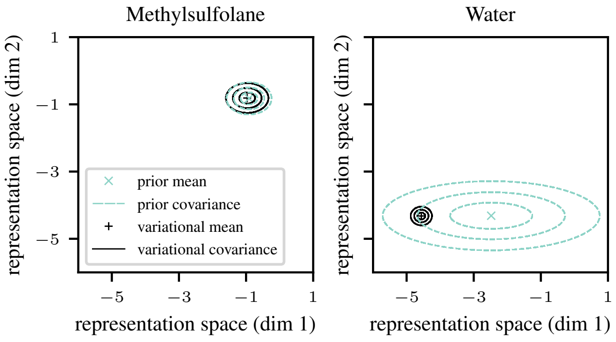

Figure 2. Influence of prior uncertainty estimates (turquoise) on the final fitted parameters (black) for methylsulfolane (least frequent solvent, left) and water (most frequent solvent, right). Concentric ellipses show 25%, 75%, and 95% quantiles, respectively. For low prior uncertainty (small turquoise ellipses, left), the final fit is forced to closely match the prior, while a large prior uncertainty (right) admits more freedom to the final fit. Discussion in Section 2.3 and model architectures in Section 3.

The role of prior uncertainties. Our ablation studies in the results section show that it is indeed crucial that the compromise between the parametric and the nonparametric fit takes the uncertainty estimates ${}^u\sigma^2_\theta(r)$ and ${}^v\sigma^2_\theta(s)$ of the conditional priors into account. Figure 2 shows two examples of how prior uncertainties affect the training in variational EM. The two panels show 2-dimensional cuts of the representation spaces for the solvents methylsulfolane (left) and water (right). We picked the two dimensions in representation space in which prior and variational distribution differ most (measured by Kullback-Leibler divergence (Kullback & Leibler, 1951; Murphy, 2022)). The dashed turquoise and solid black ellipses show 25%, 75%, and 95% quantiles of the conditional priors (Equation (1)) and variational distributions, respectively.

The positions of the ellipses in representation space are not directly interpretable, but their sizes indicate uncertainty estimates. For example, the prior predictions for methylsulfolane (left panel in Figure 2) have low uncertainty (the turquoise ellipses are small). This is expected to happen for solvents (and equally for solutes) where the dataset contains structurally similar solvents that empirically behave similarly in mixtures, thus allowing the neural network to effectively and confidently interpolate between them. The low prior uncertainty causes variational EM to trust the prior predictions, and to fit a variational distribution $q_\phi(\boldsymbol{v}_j)$ (solid black ellipses) that closely follows the prior.

By contrast, the prior predictions for water (right panel in Figure 2) have high uncertainty (the turquoise ellipses are large). This is expected to happen if the dataset contains solvents that are structurally similar to water but behave very differently in mixtures. Such anomalous cases prevent the neural network from interpolating effectively. However, as we discuss in Section 2.4 below, the neural network is at least fitted to detect such cases, and to reflect them by outputting a large uncertainty estimate ${}^v\sigma_\theta^2(s)$. As can be seen in the right panel of Figure 2, the high prior uncertainty allows variational EM to fit the variational distribution (black ellipses) more freely, thus, in a sense, overriding the descriptor-based prior (mean) prediction ${}^v\mu_\theta(s)$ (the turquoise cross) in this case. Note that the uncertainty of the (approximate) *posterior* (the size of the black ellipses) for water is small despite the large prior uncertainty. This is expected since the dataset contains a lot of experimental data where the solvent is water.

2.4. Inference Algorithm

We now formally discuss the variational EM algorithm (Dempster et al., 1977; Beal & Ghahramani, 2003) in the concrete context of the model for activity coefficients in binary mixtures which has been introduced in Section 2.2. Combining the conditional priors (Equation (1)) and the likelihood (Equation (2)), our probabilistic model defines a joint probability density over all representation vectors $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$, and all logarithmic activity coefficients $\ln\gamma_{i,j}^\infty$ in all binary mixtures i–j in the dataset,

$$p_\theta(\boldsymbol{u},\boldsymbol{v},\ln\boldsymbol{\gamma}^\infty\,|\,\boldsymbol{r},\boldsymbol{s}) = \Big(\prod_{i=1}^{M} p_\theta(u_i\,|\,r_i)\Big)\times \\ \times\Big(\prod_{j=1}^{N} p_\theta(v_j\,|\,s_j)\Big)\times\Big(\prod_{(i,j)\in\mathcal{D}} p(\ln\gamma_{i,j}^\infty\,|\,u_i,v_j)\Big). \tag{3}$$

Here, our notation of boldface symbols $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{r}$, $\boldsymbol{s}$, and $\ln\boldsymbol{\gamma}^\infty$ on the left-hand side denotes the collection of all representation vectors u_i and v_j and all chemical structures r_i and s_j for all solutes i and all solvents j, respectively, that appear at least once in the experimental data $\mathcal{D}$, and all logarithmic activity coefficients $\ln\gamma_{i,j}^\infty$ of all binary mixtures i–j for which experimental data is available. Similarly, the first two products on the right-hand side of Equation (3) run over all M solutes i and all N solvents j, respectively, and the third product runs over all pairs (i,j) where we have experimental data for the binary mixture i–j (i.e., the black pixels in the yellow/black matrix on the right of Figure 1).

A naive approach to training the neural networks would attempt to find the network weights θ that maximize the so-called marginal likelihood $p_\theta(\ln\boldsymbol{\gamma}^\infty\,|\,\boldsymbol{r},\boldsymbol{s})$ which is the probability density of predicting the experimentally measured logarithmic activity coefficients $\ln\boldsymbol{\gamma}^\infty$ for all binary systems that are contained in the available experimental dataset. Unfortunately, the marginal likelihood is not accessible in our model because obtaining it would require marginalizing Equation (3) over $\boldsymbol{u}$ and $\boldsymbol{v}$,

$$p_\theta(\ln\boldsymbol{\gamma}^\infty\,|\,\boldsymbol{r},\boldsymbol{s}) = \iint p_\theta(\boldsymbol{u},\boldsymbol{v},\ln\boldsymbol{\gamma}^\infty\,|\,\boldsymbol{r},\boldsymbol{s})\,\mathrm{d}\boldsymbol{u}\,\mathrm{d}\boldsymbol{v} \tag{4}$$

which is a high-dimensional integral that is prohibitively computationally expensive to calculate. Variational EM instead resorts to an approximate method called variational inference (Blei et al., 2017; Zhang et al., 2018), which provides a lower bound on the log marginal likelihood, called the evidence lower bound (ELBO),

$$\mathrm{ELBO}(\theta,\phi) \le \ln p_\theta(\ln\boldsymbol{\gamma}^\infty\,|\,\boldsymbol{r},\boldsymbol{s}) \quad \forall\theta,\phi. \tag{5}$$

Here, ϕ are the so-called variational parameters. We discuss ϕ and define the ELBO below. The ELBO is useful because—unlike the marginal likelihood—it can be estimated efficiently, and maximizing it over both θ and ϕ serves as a proxy for maximizing the log marginal likelihood on the right-hand side of Equation (5): since the bound in Equation (5) holds for all values of ϕ, and ϕ only appears on the left-hand side, maximizing the ELBO over ϕ makes the bound as tight as possible. Maximizing the ELBO also over θ thus finds neural network weights for which we can at least give the best guarantee for the marginal likelihood.

To derive a valid expression for the ELBO, variational inference replaces the integral on the right-hand side of Equation (4) with a form of biased importance sampling (Bamler et al., 2017). One first chooses a family of typically simple probability distributions $q_\phi(\boldsymbol{u},\boldsymbol{v})$ that are parameterized by ϕ and called variational distributions. For simplicity, we use the so-called Gaussian mean-field approximation, i.e., we choose a family of fully factorized normal distributions $q_\phi(\boldsymbol{u},\boldsymbol{v}) = \big(\prod_{i=1}^{M} q_\phi(u_i)\big)\big(\prod_{j=1}^{N} q_\phi(v_j)\big)$ with

$$q_\phi(\boldsymbol{u}_i) = \mathcal{N}\big(\boldsymbol{u}_i;{}^u\tilde{\mu}_i,\mathrm{diag}({}^u\tilde{\sigma}_i^2)\big); \\ q_\phi(\boldsymbol{v}_j) = \mathcal{N}\big(\boldsymbol{v}_j;{}^v\tilde{\mu}_j,\mathrm{diag}({}^v\tilde{\sigma}_j^2)\big) \tag{6}$$

where the variational means ${}^u\tilde{\mu}_i, {}^v\tilde{\mu}_j \in \mathbb{R}^K$ and variances ${}^u\tilde{\sigma}_i^2, {}^v\tilde{\sigma}_j^2 \in \mathbb{R}_{>0}^K$ together make up the variational parameters ϕ. The ELBO is then (Blei et al., 2017)

$$\mathrm{ELBO}(\theta,\phi) = \sum_{(i,j)\in\mathcal{D}} \mathbb{E}_{q_\phi(\boldsymbol{u}_i)\,q_\phi(\boldsymbol{v}_j)}\big[\ln p(\ln\gamma_{i,j}^\infty\,|\,\boldsymbol{u}_i,\boldsymbol{v}_j)\big] \\ - \sum_i D_{\mathrm{KL}}\big(q_\phi(\boldsymbol{u}_i)\,\big\|\,p_\theta(\boldsymbol{u}_i\,|\,r_i)\big) \\ - \sum_j D_{\mathrm{KL}}\big(q_\phi(\boldsymbol{v}_j)\,\big\|\,p_\theta(\boldsymbol{v}_j\,|\,s_j)\big). \tag{7}$$

Here, the first term in the sum is the expectation value $\mathbb{E}[\,\cdot\,]$ of the log likelihood under the variational distribution, which can be estimated by averaging the logarithm of Equation (2) over samples $u_i \sim q_\phi(u_i), v_j \sim q_\phi(v_j)$. The following two terms in the sum are Kullback-Leibler (KL) divergences (Kullback & Leibler, 1951; Murphy, 2022), which quantify how much the variational distributions differ from the conditional priors. For normal distributions, the KL divergence can be calculated analytically (Murphy, 2022)

$$D_{\mathrm{KL}}\big(q_\phi(\boldsymbol{u}_i)\,\big\|\,p_\theta(\boldsymbol{u}_i\,|\,r_i)\big) = \frac{1}{2}\sum_{\alpha=1}^{K}\Bigg[\frac{\big({}^u\tilde{\mu}_{i,\alpha} - {}^u\mu_\theta(r_i)_\alpha\big)^2}{{}^u\sigma_\theta^2(r_i)_\alpha} + \frac{{}^u\tilde{\sigma}_{i,\alpha}^2}{{}^u\sigma_\theta^2(r_i)_\alpha} + \ln\big({}^u\sigma_\theta^2(r_i)_\alpha\big) - \ln\big({}^u\tilde{\sigma}_{i,\alpha}^2\big) - 1\Bigg] \tag{8}$$

(analogously for $D_{\mathrm{KL}}\big(q_\phi(\boldsymbol{v}_j)\,\big\|\,p_\theta(\boldsymbol{v}_j\,|\,s_j)\big)$), where α indexes the coordinate in K-dimensional representation space.

Maximizing the ELBO in Equation (7) over both θ and ϕ trades off between three objectives:

(i) maximizing the first term on the r.h.s. of Equation (7) over ϕ tries to fit variational distributions $q_\phi(u_i)$ and $q_\phi(v_j)$ in a way that samples from these distributions explain the experimental data in $\mathcal{D}$;

(ii) maximizing the last two terms in Equation (7) over ϕ (which amounts to minimizing the KL-divergences over ϕ) regularizes the fits, i.e., it keeps the variational distributions $q_\phi(u_i)$ and $q_\phi(v_j)$ close to the conditional priors. Here, the first term on the r.h.s. of Equation (8) penalizes deviations between prior mean and variational mean stronger for smaller prior variance ${}^u\sigma_\theta^2(r_i)_\alpha$. Thus, the (parametric) prior model has a stronger effect on the (nonparametrically fitted) variational distributions when it is confident in its prediction, as claimed in the discussion of Figure 2;

(iii) minimizing the KL-divergences in Equation (7) also over θ fits the neural networks that define the conditional priors to the variational distributions, and thus indirectly to the data. This includes fitting the prior variances ${}^{u/v}\sigma_\theta^2(\cdot)$ to model the aleatoric uncertainty observed in the data plus any changes between the variational distributions of structurally similar components that cannot be resolved by the prior due to the finite expressiveness of the neural networks.

We maximize the ELBO over θ and ϕ with stochastic gradient descent, using reparameterization gradients (Kingma & Welling, 2013) for the first term on the right-hand side of Equation (7), and automatic differentiation provided by common software frameworks for machine learning (Paszke et al., 2019). Algorithm 1 summarizes the algorithm. Our implementation is available online (see section "Data and Software Availability"). Training our largest model variant (GNN MCM, see below) took about four hours on a single GPU (Nvidia GeForce RTX 2080 Ti).

Algorithm 1 Variational Expectation Maximization for GNN MCM

```
Input: dataset $\mathcal{D}$ of activity coefficients $\gamma_{i,j}^\infty$ in binary mixtures, involving $M$ distinct solutes $i$ and $N$ distinct solvents $j$; model $p_\theta$ as defined in Equations (1) and (2); variational family $q_\phi$ as defined in Equation (6); dimension $K$ of the abstract representation space; learning rate $\alpha$; size $m$ of minibatches.
Output: optimized parameters $\theta$ and $\phi$.

Initialize $\theta$ and $\phi \equiv \big(({}^u\tilde{\mu}_i, {}^u\tilde{\sigma}_i)_{i=1}^M, ({}^v\tilde{\mu}_j, {}^v\tilde{\sigma}_j)_{j=1}^N\big)$ randomly.
repeat
    Draw a minibatch $\mathcal{B}$ of $m$ index pairs $(i,j)$ for which experimental data $\gamma_{i,j}^\infty$ exists in $\mathcal{D}$.
    Set $\mathcal{I} \leftarrow \{i : (i,j) \in \mathcal{B}\}$ and $\mathcal{J} \leftarrow \{j : (i,j) \in \mathcal{B}\}$.
    Draw standard normal noise ${}^u\epsilon_i \sim \mathcal{N}(0, I_{K\times K})\ \forall i \in \mathcal{I}$
        and set $u_i \leftarrow {}^u\tilde{\mu}_i + {}^u\tilde{\sigma}_i \odot {}^u\epsilon_i\ \ \forall i \in \mathcal{I}$.
        ("$\odot$" denotes elementwise multiplication.)
    Draw standard normal noise ${}^v\epsilon_j \sim \mathcal{N}(0, I_{K\times K})\ \forall j \in \mathcal{J}$
        and set $v_j \leftarrow {}^v\tilde{\mu}_j + {}^v\tilde{\sigma}_j \odot {}^v\epsilon_j\ \ \forall j \in \mathcal{J}$.
    Set ${}^\gamma\mathcal{L} \leftarrow \frac{|\mathcal{D}|}{m}\sum_{(i,j)\in\mathcal{B}} \ln p(\ln\gamma_{i,j}^\infty\,|\,u_i, v_j)$.
        ▷ see Equation (2)
    Set ${}^u\mathcal{L} \leftarrow \frac{M}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} D_{\mathrm{KL}}\big(q_\phi(\boldsymbol{u}_i)\,\|\,p_\theta(\boldsymbol{u}_i\,|\,r_i)\big)$.
        ▷ see Equation (8)
    Set ${}^v\mathcal{L} \leftarrow \frac{N}{|\mathcal{J}|}\sum_{j\in\mathcal{J}} D_{\mathrm{KL}}\big(q_\phi(\boldsymbol{v}_j)\,\|\,p_\theta(\boldsymbol{v}_j\,|\,s_j)\big)$.
    Set $\mathrm{ELBO}_\mathcal{B}(\theta,\phi) \leftarrow {}^\gamma\mathcal{L} + {}^u\mathcal{L} + {}^v\mathcal{L}$.
    Compute gradients $\nabla_\theta\,\mathrm{ELBO}_\mathcal{B}(\theta,\phi), \nabla_\phi\,\mathrm{ELBO}_\mathcal{B}(\theta,\phi)$
        using automatic differentiation.
    Update $\theta \leftarrow \theta + \alpha\nabla_\theta\,\mathrm{ELBO}_\mathcal{B}(\theta,\phi)$.
    Update $\phi \leftarrow \phi + \alpha\nabla_\phi\,\mathrm{ELBO}_\mathcal{B}(\theta,\phi)$.
until convergence.
```

2.5. Predictions

Once our model is trained with variational EM, we use it for predicting activity coefficients for binary mixtures whose components can each be either in-domain (i.e., appearing in other mixtures in the available experimental data) or out-of-domain (i.e., previously unstudied components). Figure 3 shows an example where the solute i is out-of-domain whereas the solvent j is in-domain. For the out-of-domain solute i, we apply the trained neural network to its chemical structure r_i, which outputs the means and variances of the conditional prior $p_\theta(u_i\,|\,r_i)$ (Equation (1)). For the

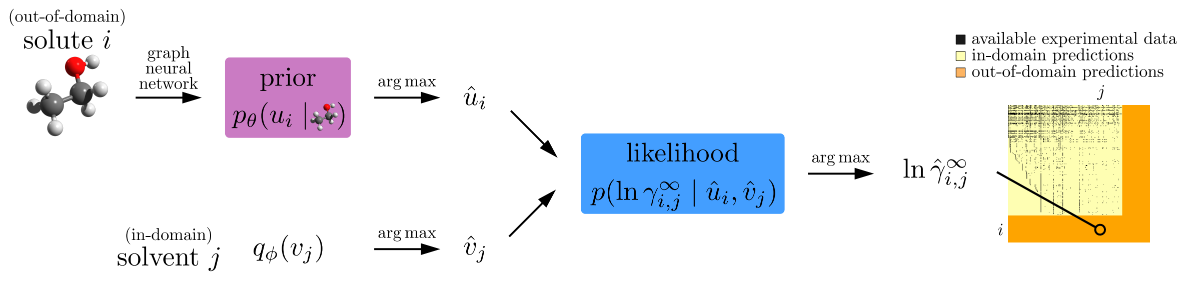

Figure 3. Data flow for a prediction where the solvent appears in the training set (in-domain) but the solute does not (out-of-domain). We thus predict the solute representation vector $\hat{u}_i$ from the prior, and the solvent representation vector $\hat{v}_j$ from the variational distribution q_ϕ, see Equation (9).

in-domain solvent j, we directly use the variational distribution $q_\phi(v_j)$ (Equation (6)), which was fitted to the data under consideration of its conditional prior. We then obtain a prediction $\hat{\gamma}^\infty_{i,j} = \exp(\ln\hat{\gamma}^\infty_{i,j})$ by calculating the modes,

$$\begin{aligned}\hat{u}_i &:= \arg\max_{u_i} p_\theta(u_i \mid r_i) = {}^{u}\mu_\theta(r_i);\\ \hat{v}_j &:= \arg\max_{v_j} q_\phi(v_j) = {}^{v}\tilde{\mu}_j;\\ \ln\hat{\gamma}^\infty_{i,j} &:= \arg\max_{\ln\gamma^\infty_{i,j}} p(\ln\gamma^\infty_{i,j} \mid \hat{u}_i, \hat{v}_j) = \hat{u}_i \cdot \hat{v}_j;\end{aligned} \tag{9}$$

where $p(\ln\gamma^\infty_{i,j} \mid \hat{u}_i, \hat{v}_j)$ is the likelihood (Equation (2)). For different combinations of in-domain and out-of-domain mixture components, we adapt Equation (9) accordingly.

2.6. Model Details And Variants

In our experiments, we investigate two model variants that differ in how they represent chemical information of solutes and solvents, and how they parameterize the means and variances of the conditional prior distributions (Equation (1)) as functions of these chemical structures. A simple model variant, which we call "MoFo MCM", represents chemical structures by the molecular formula (MoFo) (e.g., water is represented as H_2O). A more expressive model variant, which we call "GNN MCM", represents chemical structures by their topological molecular graphs (e.g., water is represented as the graph H−O−H), and the model employs graph neural networks (GNNs) (Gori et al., 2005; Scarselli et al., 2008; Bronstein et al., 2021).

In detail, the MoFo MCM uses two neural networks (one for solutes and one for solvents) that receive a fixed-size integer-valued vector as input. Each entry of the input vector corresponds to a given atom or bond type, and the values at these entries count the number of occurrences of the given atom or bond type in the molecule. Specifically, we use 16-dimensional input vectors for the 12 atoms O, Si, I, F, Br, P, H, S, Sn, N, C, and Cl present in the dataset and the 4 bond types single, double, triple, and aromatic. The network outputs a $2K$-dimensional vector that is the concatenation of the prior means ${}^{u/v}\mu_\theta(\cdot)$ and variances ${}^{u/v}\sigma^2_\theta(\cdot)$.

The GNN MCM uses two graph neural networks, whose inputs are the molecular graphs of the solute and solvent, respectively. More specifically, we encode atoms and bonds from the same vocabulary as in the MoFo MCM with learnable real-valued vectors, which we use as initial node and edge features for the GNN. In general, message-passing GNNs operate on such graph-structured inputs by performing transformations of the node and edge features over multiple layers via a message-passing scheme (Gilmer et al., 2017). The output of a GNN is computed from all node features (and possibly edge features) at the last layer with a readout function. Generally, the message-passing scheme consists of a message step, an aggregation step, and an update step. In each layer, a message is computed for each directed edge utilizing a message function whose parameters are part of the learnable neural network parameters θ. Incoming messages are aggregated by a sum for each node. The update function produces new node features depending on the previous node features and the aggregated message, and its parameters are also part of θ.

Message, aggregation, and update steps are specific to the architecture of the GNN. In this work, we utilize the Feature-wise Linear Modulation GNN (Brockschmidt, 2020). This model uses the target features of a directed edge as input to a hyper-network that determines element-wise affine transformation parameters. Messages are computed by scaling and shifting the input features with the element-wise affine transformation parameters, where the input features result from multiplying a learnable matrix with previous features. The update function of a node sums over the aggregated messages for each edge type, where also transformation parameters are computed for each edge type.

3. Evaluation Setup

In Section 4 below, we compare the two variants of our proposed method ("MoFo MCM" and "GNN MCM") to other existing prediction methods (which are called "baselines") and to simplified variants of our models that have parts removed (which are called "ablations").

Models And Baselines. We evaluate the two variants "MoFo MCM" and "GNN MCM" of our proposed method. As baselines, we compare to the group-contribution method modified UNIFAC (Dortmund) (Weidlich & Gmehling, 1987; Constantinescu & Gmehling, 2016) (which we will refer to simply as "UNIFAC" in the following) and to two machine-learning based methods: the fully nonparametric MCM method by Jirasek et al. (2020a) and the fully parametric neural-network-based method by Medina et al. (2022). The latter uses a different GNN architecture (Gilmer et al., 2017) than our conditional priors, and it includes more chemical information in the prediction (such as orbital hybridizations and formal atom charges).

Ablation Studies. We perform two ablation studies where we remove parts of our method to investigate their contribution to the method's performance. For the first ablation study, we use the same trained MoFo MCM and GNN MCM models as for our main results, but we perform predictions for in-domain components as if they were out of domain, i.e., using the mode of the conditional prior as representation vector (see first line of Equation (9)), thus ignoring the variational distributions at prediction time.

For the second ablation study, we simplify the model by removing its nonparametric part, and we train it by maximum likelihood estimation (MLE) rather than variational EM. Thus, in this ablation, the neural networks only output means ${}^{u}\mu_\theta(r_i)$ and ${}^{v}\mu_\theta(s_j)$ and no variances, and we use these means directly as representation vectors u_i and v_j, respectively, in the likelihood (Equation (2)), which our training objective maximizes over the neural network parameters θ (similar to Medina et al. (2022)). Since there are no variational distributions, predictions are again done as if all mixture components were out of domain.

Training. We train our models with 10-fold cross validation (Goodfellow et al., 2016). For each split, we use 80% of the dataset for training, 10% for testing, and 10% for a validation set. The 10 resulting test set splits are pairwise disjoint, and their union equals the full dataset. We use the test set splits to evaluate the accuracy of model predictions, where we consider a mixture "$i{-}j$" in the test set to be out-of-domain if at least one of solute i or solvent j does not appear in the corresponding training split. Note that our 10-fold cross validation is different from the work of Jirasek et al. (2020a), which uses more computationally expensive leave-one-out cross validation.

We implement our models in PyTorch (Paszke et al., 2019) and use PyTorch Geometric (Fey & Lenssen, 2019) for the GNN. All models are trained for $15,000$ epochs using the Adam (Kingma & Ba, 2015) optimizer. In the MoFo (MLE) ablation study, we use early stopping (Morgan & Bourlard, 1989; Zhang et al., 2023), i.e., we compute validation errors every 10 epochs and use the model with the lowest validation mean squared error (MSE) to compute evaluation errors on the test set. This is done to be as lenient as possible to the ablation study, and because MLE training is more prone to overfitting than variational EM. When training with variational EM, we do use the validation set.

To find well-performing hyperparameters (e.g., the learning rate schedule and the dimension K of the abstract representation space), we utilize a sparse random grid search. We provide more information on this process in the supplementary information. We choose the best model of the grid search according to its MSE on a predefined dataset split that is the same for all models and different from any other split. In order to fairly compare all models, we exclude the test data of the predefined dataset (that is used to determine the hyperparameters) from the evaluation.

Medina et al. (2022) use a different dataset and train an ensemble of 30 models for prediction where each model has been trained on randomized train/validation splits. For a fair comparison against this baseline, we train a separate GNN MCM for each of these train/validation splits, using again a sparse random grid search for hyperparameter tuning.

4. Results and Discussion

Figure 4 and Table 1 summarize our results by showing the mean absolute error (MAE) and mean squared error (MSE) of the predicted logarithmic activity coefficients of all evaluated models. Solid bars in Figure 4 show evaluations based on our full dataset. As the UNIFAC baseline cannot be applied to all mixtures in our dataset, we also trained and evaluated all models on a reduced dataset, which contains only those data points that can be modeled by UNIFAC (light hatched bars). The neural-network baseline method by Medina et al. (2022) uses yet a different dataset. Therefore, we trained and evaluated an additional instance of our proposed GNN MCM method on their dataset, and we compare its predictive accuracy to the results reported by the authors of Medina et al. (2022) in Table 1. In the following, we discuss all results in detail.

Comparison to Baselines. The proposed GNN MCM (highlighted in gold in Figure 4) provides more accurate predictions than all considered baselines, both in terms of MAE and MSE, and for both in-domain and out-of-domain predictions. Compared to UNIFAC (Weidlich & Gmehling, 1987; Constantinescu & Gmehling, 2016) (first row in Figure 4), the GNN MCM makes significantly more accurate predictions even if we restrict the test set for GNN MCM to the more difficult out-of-domain predictions (eighth row in Figure 4). Predictive accuracy is further improved significantly for in-domain predictions (fifth row in Figure 4). Recall that, even for an in-domain prediction, the training

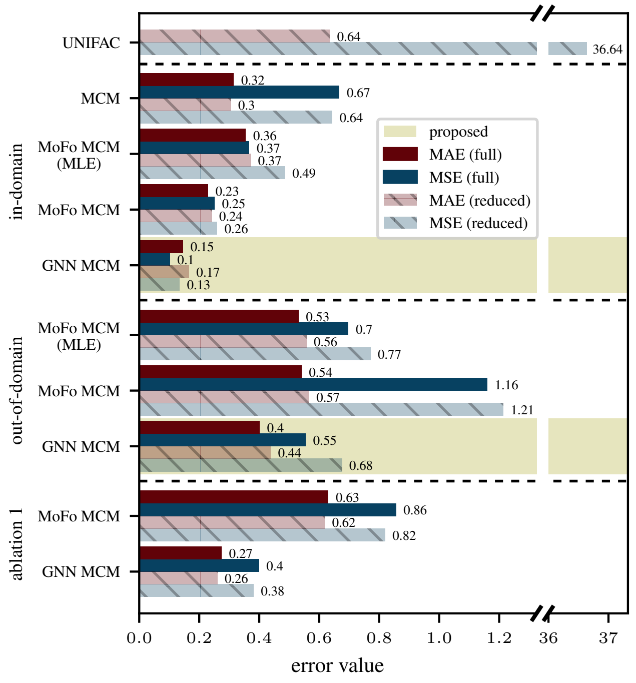

Figure 4. In-domain prediction errors (upper part), out-of-domain prediction errors (middle part), and ablations (lower section). The "reduced" dataset (light hatched bars) contains only mixtures to which UNIFAC is applicable. The proposed GNN MCM (gold highlighting) has the best predictive accuracy for both in-domain and out-of-domain predictions. Results labeled "ablation 1" (lower section) show errors for in-domain prediction tasks only, but performed as if these were out of domain.

data never contains the precise mixture i–j for which we make a prediction; it only contains other mixtures that may involve either solute i or solvent j or neither, but never both. This is in sharp contrast to UNIFAC, whose training set has not been disclosed. However, one must assume that a significant share of the dataset considered in this work was used to fit UNIFAC, so the UNIFAC results should rather be seen as correlations than predictions, making the performance of the proposed GNN MCM even more impressive.

The comparison to the fully nonparametric MCM (Jirasek et al., 2020a) (second row in Figure 4) is only possible for in-domain predictions as this baseline cannot perform out-of-domain predictions. Here, the GNN MCM (fifth row) approximately halves MAE, and it reduces MSE (which is more sensitive to outliers) even more significantly.

The published evaluation results in Medina et al. (2022) do not distinguish between in-domain and out-of-domain predictions, effectively averaging over both. Using the same training data and evaluation setup, our proposed GNN MCM significantly reduces both MAE and MSE (Table 1).

The fourth and seventh rows in Figure 4 show prediction errors of the MoFo MCM variant of our model, whose conditional priors only utilize the molecular formula of mixture components but not on their chemical structures. We find that this model variant performs worse than the GNN MCM model on both in-domain and out-of-domain prediction tasks. For in-domain predictions, we can compare again to the fully nonparametric MCM (second row in Figure 4), which does not exploit any chemical information about the mixture components. We find, as expected, that performance improves with increasing granularity of exploited chemical information: MoFo MCM performs better than the fully nonparametric MCM but worse than GNN MCM.

Table 1. Comparison of the proposed GNN MCM with the model from We Medina et al. (2022) in terms of MAE and MSE. We use the same dataset and splits as Medina et al. (2022). The table shows mean and standard deviations over 30 splits.

model	MAE	MSE
GNN MCM	$\mathbf{0.1542}_{\pm \mathbf{0.0046}}$	$\mathbf{0.0905}_{\pm \mathbf{0.0071}}$
Medina et al. (2022)	$0.1973_{\pm 0.0067}$	$0.1196_{\pm 0.0074}$

Ablation 1: Predicting Without the Nonparametric Model Part. Our first ablation discards the nonparametric part of the model after training and performs predictions for in-domain mixtures as if they were out-of-domain (i.e., only using the conditional priors). The last two rows in Figure 4 show prediction errors of the MoFo MCM model and the GNN MCM model for this ablation. Here, we evaluate on the same dataset splits as in the in-domain predictions since the splits for out-of-domain predictions contain tasks where this ablation study would not change anything. Comparing the last two rows of Figure 4 to rows four and five, we observe that discarding the nonparametric part of the model at prediction time hurts predictive accuracy significantly, thus confirming that the nonparametric fits of our method are useful where they are available.

Ablation 2: Relevance of Variational EM. Our second ablation study goes one step further and removes the nonparametric part of the model already at training time. As a result, the model can no longer be trained with variational EM and has to be trained with standard maximum likelihood estimation (MLE) instead (see "Ablation Studies" above). We performed this ablation study only on the MoFo MCM model as performing the same ablation for the GNN MCM model would result in a simplified variant of the method by Medina et al. (2022), which we already compare to as part of our baselines (see Table 1 and "Comparison to Baselines").

The results for in-domain and out-of-domain predictions are labeled "MoFo MCM (MLE)" in Figure 4. For in-domain predictions, we observe that models trained with MLE per-

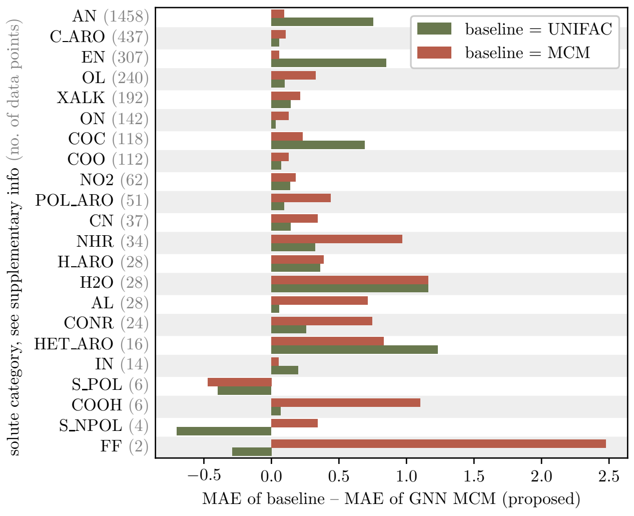

Figure 5. Improvement (in terms of mean average error, MAE) of the proposed GNN MCM method over UNIFAC and MCM, grouped by chemical category of the solute (see Table 7 in the supplementary information for a definition of the categories). Our method consistently improves over both UNIFAC and MCM across almost all solute categories; we see regressions (negative improvements) only on three categories with poor statistics in this evaluation due to small sample sizes (see gray numbers).

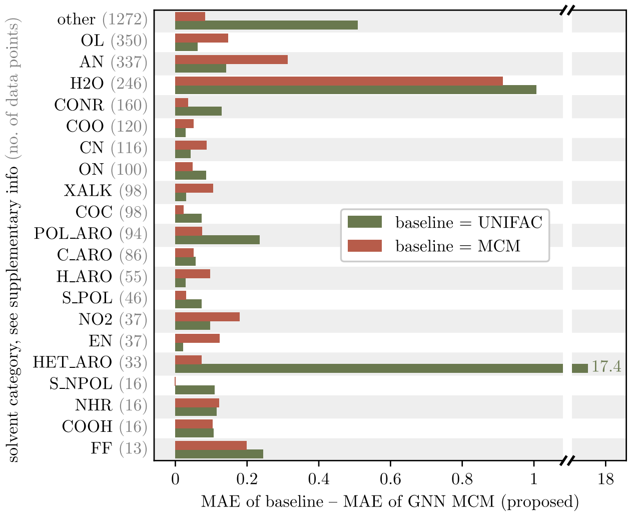

Figure 6. Improvement of the proposed GNN MCM over UNIFAC and over MCM, grouped by chemical category of the solvent (see Table 7 in the supplementary information for a definition of the categories). Our method consistently improves over both UNIFAC and MCM across all solvent categories.

form significantly worse than models trained with variational EM, but better than if we train with variational EM and then discard the nonparametric part (see Ablation 1 above). For out-of-domain predictions, the picture is less clear. Here, models trained with MLE perform slightly better than their variational EM counterparts, in particular in terms of MSE, which penalizes outliers more strongly. A possible explanation is that variational EM allows the parametric prior models to effectively ignore any mixture components that can be better modeled in a nonparametric way. This would make the priors in variational EM less regularized, so they are more susceptible to overfitting to the training data, which can result in worse generalization to unseen mixture components in out-of-domain predictions. However, this slight improvement of MoFo MCM (MLE) over MoFo MCM on out-of-domain predictions comes at the cost of significantly reduced performance on in-domain predictions, where the lack of a nonparametric model part prevents MoFo MCM (MLE) from specializing to components showing an anomalous behavior. Further, the proposed GNN MCM model further improves performance over both MoFo MCM and MoFo MCM (MLE) significantly on both in-domain and out-of-domain predictions.

Comparison by Chemical Structure. We analyze whether the improved predictive accuracy of our proposed GNN MCM method is systematic across all mixture types or limited to specific types of mixtures. For this purpose, we manually assign each solute and solvent to a category based on its chemical structure, e.g., category “XALK” for substituted alkanes and alkenes, or category “HET_ARO” for heteroaromatic compounds. Figure 5 and Figure 6 show the improvement of our proposed GNN MCM over both UNIFAC (olive) and MCM (terracotta), grouped by the category of the solute and solvent, respectively. We show in-domain predictions here so that we can compare to MCM. Positive values in the figures indicate that GNN MCM has a lower mean average error (MAE) within the corresponding category than the baseline, whereas negative values indicate that the baseline performs better within a given category.

We find that the improvements of our proposed GNN MCM are systematic across almost all categories of solutes and solvents. The only regressions occur within the solute categories “S_POL” (strongly polar sulfurous compounds), “S_NPOL” (weakly polar sulfurous compounds), and “FF” (perfluorinated compounds). The results in these categories should be taken with a grain of salt as the dataset contains only very few mixtures that involve a solute from one of these categories (gray numbers in Figure 5). Thus, within these categories, the MAE averages only over 6, 4, or 2 values, respectively, making it highly susceptible to outliers.

Comparison by Data Availability. We finally analyze if the improvement in predictive accuracy for a given mixture depends on the amount of training data that is available for the two mixture components. This analysis is motivated by

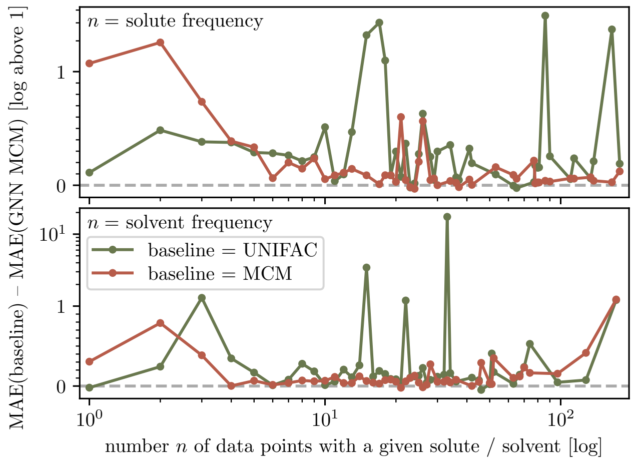

Figure 7. Improvement (in terms of mean average error, MAE) of the proposed GNN MCM method over UNIFAC and over MCM, grouped by the frequency n of solutes (top) or solvents (bottom) in the dataset. Our method consistently improves over both UNIFAC and MCM across both common and uncommon mixture components. Improvement over MCM (terracotta) is most pronounced for uncommon solutes and solvents (with small n), which is as expected since MCM is fully nonparametric.

the fact that our proposed GNN MCM trades off between a parametric and a nonparametric part, where parametric models tend to perform better in the low-data regime (because they can share statistical strength across structurally similar components) while nonparametric models tend to perform better in the high-data regime (because they can fit the data better due to fewer constrains). We therefore investigate whether our method manages to use the best of both approaches in both regimes.

For each solute i, we set $n(i)$ to the number of times that i appears as a solute in the data set $\mathcal{D}$. The inverse of this function, $i(n) := \{i \in \mathcal{D} : n(i) = n\}$ maps each possible solute count $n \in \mathbb{N}$ to all solutes in the data set that appear n times in the dataset $\mathcal{D}$. For each n where $i(n)$ is not empty, we now calculate the average prediction error for all binary mixtures in the data set whose solute is in $i(n)$. We proceed analogously for the solvents.

Figure 7 shows the improvement of our proposed GNN MCM over both UNIFAC (olive) and over MCM (terracotta) as a function of the frequency n of solutes (top) and solvents (bottom). We observe that GNN MCM improves over both UNIFAC and MCM consistently across all solute and solvent frequencies n (i.e., almost all points in the plots lie above the dashed zero line). The improvement over MCM (terracotta lines) is most pronounced in the regime of small n, i.e., where few data points with the same solute or solvent exist. This is to be expected since this is the regime where a fully nonparametric model like MCM tends to perform poorly. The strong improvement for the solvent with highest n (right end of lower plot) can be explained as this solvent is water, for which a lot of data points with infrequent solutes exist in the dataset.

In summary, we find that our proposed method significantly improves predictive accuracy over both fully parametric and fully nonparametric baselines (Figure 4 and Table 1), and that this improvement is consistent across mixture components from different chemical categories (Figure 5 and Figure 6) and across varying amounts of training data for involved components (Figure 7).

5. Summary and Outlook

In this work, we propose a method for predicting physico-chemical properties that combines a structure-based approach using graph neural networks (which are able to extrapolate across substances with similar chemical structure) with a representation-learning based approach (which allows the model to override structure-based predictions in anomalous cases). The method significantly improves predictive accuracy over the state of the art in the studied problem of predicting activity coefficients in binary mixtures.

Our ablation studies identify the variational EM algorithm to be crucial for the success of the prediction method. We think that variational EM can be a useful tool for many physico-chemical prediction problems since it balances structure-based and representation-learning based predictions by weighing off their respective uncertainties.

Future work should explore the application of our method to other properties such as diffusion coefficients or even fundamental quantities like interaction energies, which are at the core of established physical models of mixtures and based on which diverse mixture properties can be described. In a broader context, our work provides additional evidence for the efficacy of graph neural networks for processing chemical structure information. It would be interesting to study whether activations of hidden layers of the graph neural networks can be made interpretable to human domain experts, whether correlations between the hidden activations of different atoms can be used to identify relevant substructures of molecules, and whether such substructures correspond to the structural groups that are considered in established group contribution methods like UNIFAC.

Data and Software Availability

We evaluate our methods on two datasets: a dataset that is licensed from the Dortmund Data Bank (DDB) (Onken et al., 1989) and a dataset collected by Brouwer et al. (2021). The software packages for preprocessing and training our models are freely available. We provide the source code to replicate our results at https://github.com/jzenn/gnn-mcm. For the comparison of our method to the model proposed by Medina et al. (2022), we directly use the files available from their GitHub repository.

Acknowledgements

Johannes Zenn thanks Tim Xiao for helpful comments and discussions. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC number 2064/1 – Project number 390727645. This work was supported by the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039A. Robert Bamler acknowledges funding by the German Research Foundation (DFG) for project 448588364 of the Emmy Noether Program. The authors would like to acknowledge support of the 'Training Center Machine Learning, Tübingen' with grant number 01—S17054. Fabian Jirasek gratefully acknowledges funding by DFG for project 528649696 of the Emmy Noether Program and for project 497201843 in the Priority Program Molecular Machine Learning (SPP 2363). The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Johannes Zenn.

References

Ajmani, S., Rogers, S. C., Barley, M. H., Burgess, A. N., and Livingstone, D. J. Characterization of mixtures part 1: Prediction of infinite-dilution activity coefficients using neural network-based qspr models. *QSAR & Combinatorial Science*, 27(11-12):1346–1361, 2008.

Bamler, R., Zhang, C., Opper, M., and Mandt, S. Perturbative black box variational inference. In *Advances in Neural Information Processing Systems*, 2017.

Beal, M. J. and Ghahramani, Z. The variational bayesian em algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian statistics*, 7: 453–464, 2003.

Behrooz, H. A. and Boozarjomehry, R. B. Prediction of limiting activity coefficients for binary vapor-liquid equilibrium using neural networks. *Fluid Phase Equilibria*, 433:174–183, 2017.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112:859–877, 2017.

Brockschmidt, M. GNN-FiLM: Graph neural networks with feature-wise linear modulation. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Bronstein, M. M., Bruna, J., Cohen, T., and Veličković, P. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

Brouwer, T., Kersten, S. R., Bargeman, G., and Schuur, B. trends in solvent impact on infinite dilution activity coefficients of solutes reviewed and visualized using an algorithm to support selection of solvents for greener fluid separations. *Separation and purification technology*, 272: 118727, 2021.

Constantinescu, D. and Gmehling, J. Further development of modified unifac (dortmund): Revision and extension 6. *Journal of Chemical & Engineering Data*, 61(8):2738–2748, 2016. doi: 10.1021/acs.jced.6b00136.

Damay, J., Jirasek, F., Kloft, M., Bortz, M., and Hasse, H. Predicting activity coefficients at infinite dilution for varying temperatures by matrix completion. *Industrial & Engineering Chemistry Research*, 60(40):14564–14578, 2021.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (methodological)*, pp. 1–38, 1977.

Estrada, E., Díaz, G. A., and Delgado, E. J. Predicting infinite dilution activity coefficients of organic compounds in water by quantum-connectivity descriptors. *Journal of computer-aided molecular design*, 20(9):539–548, 2006.

Fey, M. and Lenssen, J. E. Fast graph representation learning with PyTorch Geometric. In *Workshop on Representation Learning on Graphs and Manifolds, ICLR*, 2019.

Fredenslund, A., Jones, R. L., and Prausnitz, J. M. Group-contribution estimation of activity coefficients in nonideal liquid mixtures. *AIChE Journal*, 21(6):1086–1099, 1975.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, 2017.

Giralt, F., Espinosa, G., Arenas, A., Ferre-Gine, J., Amat, L., Girones, X., Carbó-Dorca, R., and Cohen, Y. Estimation of infinite dilution activity coefficients of organic compounds in water with neural classifiers. *AIChE journal*, 50(6):1315–1343, 2004.

Gmehling, J., Constantinescu, D., and Schmid, B. Group contribution methods for phase equilibrium calculations. *Annual Review of Chemical and Biomolecular Engineering*, 6(Volume 6, 2015):267–292, 2015.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016.

Gori, M., Monfardini, G., and Scarselli, F. A new model for learning in graph domains. In *IEEE International Joint Conference on Neural Networks*, 2005.

Großmann, O., Bellaire, D., Hayer, N., Jirasek, F., and Hasse, H. Database for liquid phase diffusion coefficients at infinite dilution at 298 k and matrix completion methods for their prediction. *Digital Discovery*, 1:886–897, 2022.

Hayer, N., Jirasek, F., and Hasse, H. Prediction of henry's law constants by matrix completion. *AIChE Journal*, 68 (9):e17753, 2022.

Hoffman, M. D. and Johnson, M. J. Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in Advances in Approximate Bayesian Inference, NIPS*, 2016.

Jirasek, F. and Hasse, H. Combining machine learning with physical knowledge in thermodynamic modeling of fluid mixtures. *Annual Review of Chemical and Biomolecular Engineering*, 14:31–51, 2023.

Jirasek, F., Alves, R. A., Damay, J., Vandermeulen, R. A., Bamler, R., Bortz, M., Mandt, S., Kloft, M., and Hasse, H. Machine learning in thermodynamics: Prediction of activity coefficients by matrix completion. *The Journal of Physical Chemistry Letters*, 11(3):981–985, 2020a.

Jirasek, F., Bamler, R., and Mandt, S. Hybridizing physical and data-driven prediction methods for physicochemical properties. *Chemical Communications*, 56(82):12407–12410, 2020b.

Jirasek, F., Bamler, R., Fellenz, S., Bortz, M., Kloft, M., Mandt, S., and Hasse, H. Making thermodynamic models of mixtures predictive by machine learning: matrix completion of pair interactions. *Chemical Science*, 13 (17):4854–4862, 2022.

Jirasek, F., Hayer, N., Abbas, R., Schmid, B., and Hasse, H. Prediction of parameters of group contribution models of mixtures by matrix completion. *Phys. Chem. Chem. Phys.*, 25:1054–1062, 2023.

Katritzky, A. R., Kuanar, M., Slavov, S., Hall, C. D., Karelson, M., Kahn, I., and Dobchev, D. A. Quantitative correlation of physical and chemical properties with chemical structure: utility for prediction. *Chemical reviews*, 110 (10):5714–5789, 2010.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kullback, S. and Leibler, R. A. On information and sufficiency. *The Annals of Mathematical Statistics*, 22:79 – 86, 1951.

Medina, E. I. S., Linke, S., Stoll, M., and Sundmacher, K. Graph neural networks for the prediction of infinite dilution activity coefficients. *Digital Discovery*, 2022.

Mitchell, B. E. and Jurs, P. C. Prediction of infinite dilution activity coefficients of organic compounds in aqueous solution from molecular structure. *Journal of Chemical Information and Computer Sciences*, 38(2):200–209, 1998.

Morgan, N. and Bourlard, H. Generalization and parameter estimation in feedforward nets: Some experiments. In *Advances in neural information processing systems*, 1989.

Murphy, K. P. *Probabilistic machine learning: an introduction*. MIT press, 2022.

Onken, U., Rarey-Nies, J., and Gmehling, J. The dortmund data bank: A computerized system for retrieval, correlation, and prediction of thermodynamic properties of mixtures. *International Journal of Thermophysics*, 10(5): 739–747, 1989. doi: 10.1007/BF00507993.

Paduszynski, K. In silico calculation of infinite dilution activity coefficients of molecular solutes in ionic liquids: critical review of current methods and new models based on three machine learning algorithms. *Journal of Chemical Information and Modeling*, 56(8):1420–1437, 2016.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

Robbins, H. and Monro, S. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. Computational capabilities of graph neural networks. *IEEE Transactions on Neural Networks*, 20 (1):81–102, 2008.

Smith, L. N. Cyclical learning rates for training neural networks. In *IEEE winter conference on applications of computer vision (WACV)*, 2017.

Weidlich, U. and Gmehling, J. A modified unifac model. 1. prediction of vle, he, and .gamma..infin. *Industrial & Engineering Chemistry Research*, 26(7):1372–1381, 1987. doi: 10.1021/ie00067a018.

Zhang, A., Lipton, Z. C., Li, M., and Smola, A. J. *Dive into deep learning*. Cambridge University Press, 2023.

Zhang, C., Bütepage, J., Kjellström, H., and Mandt, S. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

A. Supplementary Information

A.1. Model and Training Hyperparameters

To find well-performing hyperparameters we employ sparse random grid searches (RGS) for all models we consider in this work. We train 30 models with hyperparameters sampled uniformly at random from a defined search space. Each of the models is trained on the same training set and evaluated on the same test set. We pick the best model according to its mean squared error (MSE) on the test set. To report fair error estimates, we exclude all data points of this test set from the 10-fold cross validation. Since the sample size of 30 models is very small compared to the size of the search space, one might be able to further improve on our results by extending the RGS.

We schedule the learning rate either using a constant schedule or one out of nine schedules. All schedules $S_i \in \mathcal{S}$ (except for the constant one) start from an initial learning rate ϵ that is decreased by a factor of (maximally) 10^{-1} (multiple times) during training. After some initial warm-up steps T_w the Robbins-Monro Learning Rate Scheduler (Robbins & Monro, 1951) computes the learning rate at epoch t from an initial learning rate of ϵ_0 as follows.

$$\epsilon_t = \epsilon_0 \ / \ \left(\frac{t - T_w}{b} + a\right)^{\gamma} \tag{10}$$

The three parameter combinations we consider are shown in Table 2. The Cyclical Learning Rate Scheduler (Smith, 2017) linearly increases the learning rate from an initial $\epsilon_- = 10^{-1}\epsilon_0$ to $\epsilon_+ = \epsilon_0$ and back to ϵ_-. This is done T_c number of times during training. We consider $T_c \in \{1, 2, 4\}$ The Step Learning Rate Scheduler decays an initial learning rate ϵ_0 by γ every T_e epochs. Consequently, at epoch t the learning rate equals $\epsilon_0^{\gamma\lfloor t/T_e \rfloor}$. We search over three combinations of parameters that are depicted in Table 3.

T_w	γ	a	b
$1.5 \cdot 10^3$	0.5	1.0	150
$1.5 \cdot 10^3$	0.6	1.0	300
$1.5 \cdot 10^3$	0.8	1.0	900

Table 2. Combinations of parameters for the RM scheduler.

γ	T_e
0.8	1500
0.45	3750
0.1	7500

Table 3. Combinations of parameters for the Step scheduler.

The ELBO (that is maximized) can be formulated in various ways (Hoffman & Johnson, 2016), two of which we include in the grid search: ELBO-KL maximizes the data log-likelihood under the variational distribution and simultaneously minimizes a KL divergence between the variational posterior distribution and the prior distribution. ELBO-Entropy maximizes the data log-likelihood as well as the log-prior under the variational distribution while simultaneously maximizing the entropy of the variational distribution.

We apply skip connections (skip con. $\neq$ none) either every layer or every second layer. Additionally in the GNN MCM model, we experiment with mean aggregation besides the sum aggregation.

Table 4 lists hyperparameter values we search over for the GNN MCM model. Table 5 lists hyperparameter values we search over for the MoFo MCM model. Table 6 lists hyperparameter values we search over for the MoFo MCM (MLE) model.

GNN MCM (in-domain) ELBO-Entropy, 0.005, 16, 0, 0.1, 64, sum, 2, none, true

GNN MCM (out-of-domain) ELBO-Entropy, 0.0005, 8, 7, 0.0, 64, mean, 8, 1, true

Medina et al. (2022) ELBO-Entropy, 0.001, 16, 7, 0.1, 16, sum, 6, none, false

parameter	value
loss	ELBO-KL, ELBO-Entropy
learning rate	0.005, 0.001, 0.0005, 0.0001
K	4, 8, 16
lr-scheduler	0, 1, 2, 3, 4, 5, 6, 7, 8
dropout prob.	0.0, 0.1
representation dim.	16, 32, 64, 128
aggregation	sum, mean
L	1, 2, 4, 6, 8
skip con.	none, every $\{1, 2\}$-th layer
bias	true, false

Table 4. Hyperparameters of grid search for GNN MCM models.

parameter	value
loss	ELBO-KL, ELBO-Entropy
learning rate	0.005, 0.001, 0.0005, 0.0001
K	4, 8, 16
lr-scheduler	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
dropout prob.	0.0, 0.1
representation dim.	16, 32, 64, 128
L	1, 2, 4, 6, 8
skip con.	none, every $\{1, 2\}$-th layer

Table 5. Hyperparameters of grid search for MoFo MCM models.

parameter	value
learning rate	$10^{-\{2,3,4\}}$,$5 \cdot 10^{-\{4,5\}}$
K	$4, 8, 16$
lr-scheduler	$0, 1, 2, 3, 4, 5, 6, 7, 8$
dropout prob.	$0.0, 0.1$
representation dim.	16, 32, 64, 128
L	$1, 2, 4, 6, 8$
skip con.	none, every $\{1, 2\}$-th layer

Table 6. Hyperparameters of grid search for MoFo MCM (MLE) models.

MoFo MCM (in-domain) ELBO-Entropy, 0.0005, 8, 4, 0.1, 16, 1, none

MoFo MCM (out-of-domain) ELBO-Entropy, 0.001, 8, 0, 0.1, 16, 6, 1

MoFo MCM (MLE) (in-domain) 10^{-2}, 8, 1, 0.0, 64, 8, none

A.2. Solute and Solvent Categories

Table 7 defines the chemical categories used in the “Comparison by Chemical Structure” of the main text. We assigned each solute and solvent to one of these categories manually based on their chemical structure formula using human expert knowledge. These assignments were only used in the evaluation; the model is unaware of our assignments. Due to data licensing, we do not provide the solutes and solvents that fall into the chemical categories listed in Table 7 but only provide the total number of compounds in each category.

abbreviation	description
AN	alkanes (including compounds with long alkyl groups ≥ 10 C-atoms)
EN	alkenes and dienes
IN	alkynes
C_ARO	aromatic compounds without heteroatoms
POL_ARO	aromatic compounds with substituent heteroatoms, pi-systems with inductive and mesomeric effects
H_ARO	aromatic compounds with substituent heteroatoms that can build stable hydrogen bonds
HET_ARO	heteroarenes
XALK	chlorine, bromine, and iodine alkanes
OL	alcohols
COO	esters
NHR	amines
ON	ketones
AL	aldehydes
COC	ethers
CN	nitrils
NO2	nitro compounds
COOH	short carboxylic acids
FF	perfluorinated compounds
CONR	amides
S_NPOL	weakly polar sulfurous compounds
S_POL	strongly polar sulfurous compounds
H2O	water and heavy water

Table 7. Manually assigned chemical categories for solutes and solvents.