Proceedings of Machine Learning Research vol 145:1–23, 2021 2nd Annual Conference on Mathematical and Scientific Machine Learning

# Reconstruction of Pairwise Interactions using Energy-Based Models

**Christoph Feinauer** CHRISTOPH.FEINAUER@UNIBOCCONI.IT
**Carlo Lucibello** CARLO.LUCIBELLO@UNIBOCCONI.IT
*Department of Decision Sciences,*
*Bocconi Institute for Data Science and Analytics (BIDSA)*
*Bocconi University, Milan, Italy*

**Editors:** Joan Bruna, Jan S Hesthaven, Lenka Zdeborova

## Abstract

Pairwise models like the Ising model or the generalized Potts model have found many successful applications in fields like physics, biology, and economics. Closely connected is the problem of inverse statistical mechanics, where the goal is to infer the parameters of such models given observed data. An open problem in this field is the question of how to train these models in the case where the data contain additional higher-order interactions that are not present in the pairwise model. In this work, we propose an approach based on Energy-Based Models and pseudolikelihood maximization to address these complications: we show that hybrid models, which combine a pairwise model and a neural network, can lead to significant improvements in the reconstruction of pairwise interactions. We show these improvements to hold consistently when compared to a standard approach using only the pairwise model and to an approach using only a neural network. This is in line with the general idea that simple interpretable models and complex black-box models are not necessarily a dichotomy: interpolating these two classes of models can allow to keep some advantages of both.

**Keywords:** Pairwise Models, Neural Networks, Inverse Ising, Energy Based Models

## 1. Introduction

An important class of distributions used in the modeling of natural systems is the exponential family of pairwise models. Commonly investigated in the statistical physics community, pairwise models are a popular method for the analysis of categorical sequence data. Examples of data on which they have been successfully applied include protein sequence data (Morcos et al., 2011; Marks et al., 2012; Cocco et al., 2018), neuronal recordings (Roudi et al., 2009; Tkačik et al., 2014), magnetic spins (Fisher and Huse, 1986), economics and social networks (Stauffer, 2008; Sornette, 2014; Hall and Bialek, 2019).

One main advantage of these models is their relative simplicity: The probability assigned to a sequence $s$ of binary or categorical variables is of the form $p(s) \propto \exp(-E(s))$, where the energy $E$ is a *simple* function of $s$, meaning that it consists of terms that depend on only one or two variables. The parameters quantifying the pairwise interactions are typically called *couplings*.

Given this simple form, the parameters can often be given a direct interpretation in terms of the underlying system. Especially the couplings have been shown to contain highly non-trivial information in many cases: The couplings in the so-called Potts Models for protein sequence data can be seen as a measure for the strength of co-evolutionary pressure between parts of the sequence and can be used for the prediction of structural features (Morcos et al., 2011); the couplings in models for neuronal recordings can be seen as the functional couplings between neurons (Roudi et al., 2009); the couplings in magnetic systems of interacting spins can be seen as describing their physical interaction strength.

While pairwise models have been surprisingly successful in many fields, they have clear limitations: If the data generating process contains important interactions that cannot be described as pairwise interactions, the models might fail to capture important variability. Even worse, if such interactions are strong enough, the pairwise models might even stop to describe the pairwise interactions properly since they might contain effective pairwise interactions that try to include the variability of the higher-order interactions. In fact, it is known in literature that for example some variability in protein sequences is due to higher-order epistasis, including more than 2 residues (Waechter et al., 2012). Several methods have been proposed to address such problems, for example the 'manual' addition of higher-order interaction terms based on a close look at the data (Feinauer et al., 2014), or the addition of complete sets of higher-order interactions, for example all terms involving triplets of variables (Schmidt and Hamacher, 2018).

Another option is to abandon simple pairwise models and adopt more complicated, but also more expressive methods, for example neural networks. These models can in principle capture interactions of all orders and can be trained for a specific task, for example the extraction of structural information (Peng and Xu, 2011), the generation of new samples (Riesselman et al., 2019) or the creation of generic embeddings (Rives et al., 2019). While this strategy has lead to unprecedented successes in many fields, it also comes at the cost of a higher computational demand and the loss of the interpretability of the single parameters defining the distribution. Moreover, failing to encode the prior knowledge on the data generative process, these black-box methods are far away from being optimal in terms of sample efficiency. Moreover, while numerous knowledge integration approaches have been proposed in the past (Von Rueden et al., 2019), literature is scarce when it comes to generative modelling and density estimation.

*Energy-Based Models* (EBM) are a class of machine learning methods which specify the *unnormalized* negative log-probability of the distribution to be trained on the data (Song and Kingma, 2021; LeCun et al., 2006). This unnormalized negative log-probability is equivalent to an energy, but can take on more complex forms than in a pairwise model. The absence of an explicit normalization allows one to use any nonlinear regression function for specifying the energy. Models based on neural networks, for example, have recently been applied with success in the field of image generation (Du and Mordatch, 2019b). Apart from the appealing similarity to models used in statistical mechanics since more than a hundred years, they present some advantages in comparison to other model classes like *Generative Adversarial Networks* (Goodfellow et al., 2014) or *Variational Autoencoders* (Kingma and Welling, 2013). The most important ones related to the present work are their relative uniformity and simplicity and their composionality (both also mentioned in (Du and Mordatch, 2019b)). By uniformity and simplicity we refer to the fact that due to the generic formulation using a single energy function, tasks like training, sampling and analysis can often be formulated generically, independent of the exact parametrization of the energy. By composionality, we refer to the idea that EBMs can be combined easily by summing their respective energies, leading to a so called *product of experts* model (Hinton, 2002). While the lack of normalization grants a large degree of freedom when specifying the model, training and sampling become harder. However, many training methods like *contrastive divergence* (Carreira-Perpinan and Hinton, 2005) or *pseudolikelihoods* (see SM Section A) can be adapted for EBMs, and sampling can be done using standard MCMC algorithms like *Metropolis-Hastings* (Metropolis et al., 1953).

In this paper, we leverage the compositionality of EBMs in a physics-inspired machine learning approach where we combine the advantages of a simple model with a black-box neural network model to *help* with the more complex patterns in the data. This approach seems sensible in cases where we suspect or know that a simple model is able to capture most of the variability in the data, but that it might fail to capture some additional aspects or even gets confused by them.

We implement this idea defining a new energy function

$$E(s) = E_{pw}(s) + E_{nn}(s), \tag{1}$$

where $E_{pw}$ is a pairwise model and $E_{nn}$ is a neural network that maps a configuration $s$ to a real number. We then look at cases where the data generating process contains a simple part, corresponding to another pairwise model, and a more complicated part, corresponding to higher-order interactions. The expectation is that the neural network picks up these higher-order interactions and thus helps the pairwise model in matching the pairwise interactions of the generative process.

We will focus on the so-called inverse problem of statistical mechanics, that is reconstructing the pairwise couplings of a generative model containing also some unknown higher-order interaction terms.

## 2. Methods

### 2.1. Pairwise Models and Energy-Based Models

We consider a probability distribution $p(s)$ over all possible configurations of $N$ binary variables, $\{-1, +1\}^N$. Any such distribution with support over the whole space can be written in the form

$$p(s) = \exp(-E(s))/Z, \tag{2}$$

where $E : s \to \mathcal{R}$ is the so called energy function and $Z$ is a normalization constant called the partition function. Denoting, with $\mathcal{I}$ the power set of $\{1, \ldots, N\}$, the energy can be uniquely expressed by the expansion

$$E(s) = -\sum_{I \in \mathcal{I}} \xi_I \prod_{i \in I} s_i, \tag{3}$$

where $\xi_I \in \mathcal{R}$ is the interaction coefficient for the term containing the variables specified by $I$. Such expansions are known in theoretical computer science and Boolean algebra as *Fourier expansions*, and the corresponding parameters $\xi_I$ are called *Fourier coefficients* (O'Donnell, 2014). Determining specific coefficients from a black-box function $E$ can be done efficiently through sampling techniques (see Section 2.2) and coefficients larger than a given threshold can be determined using the *Goldreich-Levin* algorithm (O'Donnell, 2014). This is useful in our setting, since these techniques also apply when the energy $E$ is parametrized using an arbitrary neural network.

The class of models where $\xi_I = 0$ if $|I| > 2$ are called pairwise models, defined by the energy

$$E_{pw}(s) = -\sum_i h_i s_i - \sum_{i<j} J_{ij} s_i s_j. \tag{4}$$

The coefficients $h_i$ are called external fields and the coefficients $J_{ij}$ are called couplings. Such models have a long history in statistical physics and have been exported to various fields. In a typical application, the model is fitted to a dataset $D = \{s^m\}_{m=1}^M$ consisting of $M$ configurations sampled from the system under study, and can be afterwards either used as a generative model or insights about the system can be gained from examining the fitted parameters $J$ and $h$.

### 2.2. Hybrid Models and Extraction of Coefficients

If we assume the existence of a generating distribution $p_G(s)$ that includes important interactions involving more than two variables, the pairwise distribution might fail to describe the variability in the dataset (see Section 4) and the inferred couplings and fields might not correspond to the ones in the generating distribution. If such a case is suspected, one is tempted to use a more complicated function to describe the energy $E$. Given the flexibility of neural networks in approximating arbitrary input-output dependencies, a promising choice could be a multi-layer perceptron with $L$ layers,

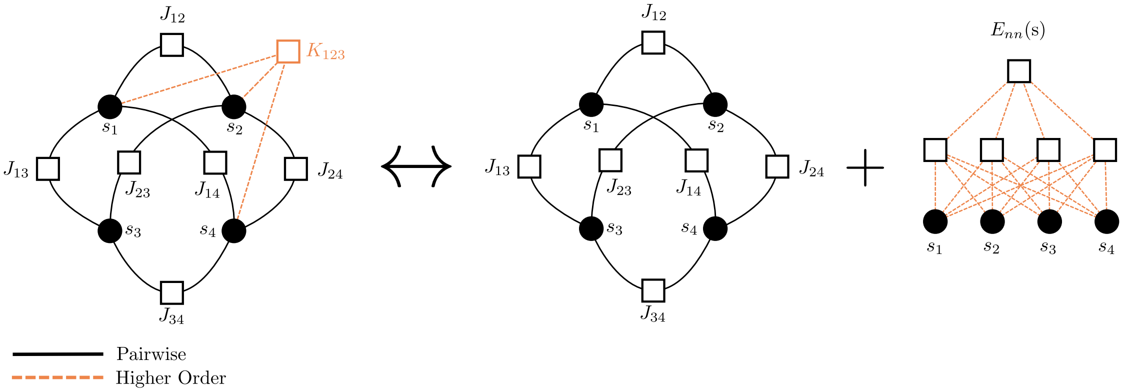

Figure 1: Representation of the basic idea of this work: Given a generative distribution that contains a strong pairwise part but also higher-order interactions, we fit an energy-based model including a pairwise part and a neural network. The expectation is that the neural network captures the higher-order interactions, while the pairwise parts match up after training.

where the operations in each layer are a matrix multiplication followed by the addition of a bias and the application of a possibly non-linear activation function (Goodfellow et al., 2016).

In this work, we propose a combination of the two types of models which we call *hybrid models*. They are of the form

$$E(s) = E_{pw}(s) + E_{nn}(s) = -\sum_{i<j} J_{ij} s_i s_j + E_{nn}(s). \tag{5}$$

For simplicity and since we want to focus on the more complex problem of reconstructing the couplings, we do not explicitly consider external fields $h_i$ in this work, although they could be easily accounted for. $E_{nn}(s)$ is a multi-layer perceptron network with one layer of hidden neurons with $\tanh$ activations. While we could also test networks with more than one layer, there is evidence that in similar settings the most important characteristic is still the size of the first hidden layer, while the depth is of minor importance (Morningstar and Melko, 2017). Since adding depth would also add the problem of finding the optimal architecture, we restrict ourselves to a single layer in this work, and also leave the exploration of other methods like self-attention (Vaswani et al., 2017) or autoregressive architectures (Wu et al., 2019) for further research.

One interpretation of these models is that we model the pairwise terms in the Fourier expansion Eq. (3) explicitly, while we use a neural network for describing all other interactions (see Fig. 1). When implementing these models, both parts are kept explicitly and trained together. The objective function used (see below) is agnostic of the details of the energy function, and gradients for the parameters of both parts can be obtained by simple back propagation.

For the neural network part $E_{nn}$, the general Fourier expansion in Eq. (3) can contain in principle interactions of all orders. We can formally invert the Fourier expansion in Eq. (3) to get a mapping from the energy $E$ to the Fourier coefficients $\xi_I$ for all interactions $I$:

$$\xi_I = -\mathbb{E}_s \left[ E(s) \prod_{i \in I} s_i \right], \tag{6}$$

where the expectation is according to the uniform distribution over all possible $2^N$ configurations. If $N$ is sufficiently small, this expectation can be calculated exactly. For larger $N$, it can be approximated by using samples drawn from a uniform distribution. Such an estimator for the interaction coefficients is unbiased and its approximation error scales as the inverse of the square root of the number of samples used. Since we do not limit the capacity of the neural network, $E_{nn}(s)$ can also contain significant pairwise interactions. Therefore, we may have $\mathbb{E}_s[E(s)s_i s_j] \not\approx -J_{ij}$. We show below that this can be indeed observed in specific situations and approach the problem as follows: we reconstruct the couplings from $E = E_{pw} + E_{nn}$ using Eq. (6). We refer to these effective couplings as *reconstructed* couplings

$$\hat{J}_{ij} = -\mathbb{E}_s\left[E(s)\, s_i s_j\right], \tag{7}$$

as opposed to the *explicit* couplings $J$ in the trained model (5). The reconstruction is performed only at the end of the training, and approximated for large systems with $10^6$ Monte Carlo samples in our experiments. For the system sizes considered in this work, the time spent in the reconstruction step is negligible in comparison to the training. As an alternative, in SM Section B, we show that it can be also done during training, which effectively limits the pairwise interactions in the hybrid model to the pairwise part.

We use the same reconstruction method for extracting coefficients in models consisting *only* of the neural network, without the explicit pairwise part, to understand whether using an explicit pairwise term in model (5) brings any advantage. While we do this here only for comparison and use only simple multi-layer perceptrons (MLP), we note that it would be an interesting avenue of research to use more advanced neural network models and see if the extracted couplings can be used in applications where pairwise models are typically used.

### 2.3. Training Procedure

The difficulties in evaluating the normalization constant in energy-based models make density evaluation intractable, and efficient sampling becomes problematic as well. Many techniques have been proposed for the challenging task of training EBMs, the most commonly used ones being contrastive divergence with Langevin dynamics (Hinton, 2002; Du and Mordatch, 2019a), noise-contrastive estimation (Gutmann and Hyvärinen, 2010), and score matching (Hyvärinen, 2005). In this work, we use pseudolikelihood maximization to train the parameters of the model given the data (Besag, 1977). This method is very popular for the training of pairwise models (Aurell and Ekeberg, 2012; Ekeberg et al., 2013; Decelle and Ricci-Tersenghi, 2014) and is furthermore very similar to the method of training for state-of-the-art neural network models summarily called *self-supervised learning*, which transforms the task of unsupervised learning of unlabeled data into a supervised learning task by training the model to predict an artificially masked part of the data from the unmasked part. This technique is for example used when training the self-attention based *Bert* models (Devlin et al., 2018). We also note that the use of a related estimator called interaction screening has been proposed by (Jayakumar et al., 2020) in a similar context with neural networks.

Given a single mini-batch $\{s^b\}_{b=1}^B$ with $B$ training configurations, we use the negative pseudo-likelihood loss function

$$\mathcal{L} = -\frac{1}{B}\sum_{b=1}^{B}\sum_{i=1}^{N}\log p(s_i^b|s_{/i}^b), \tag{8}$$

where the quantity $p(s_i^b|s_{/i}^b)$ corresponds to the probability of observing $s_i^b$ given the other variables in $s^b$, excluding $s_i^b$. This loss function can be calculated for a generic energy model over configurations using $2N$ forward passes. For a pairwise model instead, we can use more efficient calculation schemes. For implementation details see Appendix A. It is worth mentioning that the interaction screening approach of Ref. (Vuffray et al., 2020) provides an alternative with well understood sample complexity guarantees to the pseudolikelihood framework used here.

We train the models by standard stochastic gradient descent with batch size $B = 1024$ and a learning rate of $0.02$. We did not find consistent improvements for the hybrid models when applying an $L_2$ regularization and do not apply it in this work. We did find, however, a slight improvement for models containing only the pairwise energy $E_{pw}$, as explained in detail below. We trained all models for 250 epochs. The models were implemented using PyTorch (Paszke et al., 2019). The loss function depends only on energy differences (see Appendix A). After having implemented the energies of the pairwise and the neural network part, the automatic differentiation function of PyTorch can be used to calculate the gradients.

## 3. Experimental Setting: Generating Distributions

In this work, the experimental setting is given by a data generating distribution $p_G(s) \propto \exp(-E_G(s))$ over the configurations $\{-1,+1\}^N$, where $E_G(s)$ contains a pairwise part and an additional number of higher-order interactions:

$$E_G(s) = E^G_{pw}(s) + E^G_{ho}(s) = -\sum_{i<j} J^G_{ij} s_i s_j - \sqrt{\gamma} \sum_{I \in \mathcal{I}_G} \xi^G_I \prod_{i \in I} s_i. \tag{9}$$

$\mathcal{I}_G$ is a set of sets of indices determining the higher-order interactions of the generator. Since we are interested in the effect of additional higher-order interactions, we restrict ourselves to cases where $|I| \geq 3$. In order to model the situation where a pairwise distribution is probably a good approximation, we will keep these higher-order interactions sparse and choose only a small subset of the $2^N$ possible interactions, mostly only $N$. The factor $\gamma$, which we call higher-order strength, is used to weight the two terms against each other (see below). The specific interacting sets $I \in \mathcal{I}_G$ are independently and randomly chosen either as only triplets or as interactions of order 3 to 10, according to the different settings we present in the following sections.

The interaction parameters $\xi^G_I$ and the couplings $J^G_{ij}$ are independently sampled from Gaussian distributions. In order to ensure that none of the two parts of the generator completely dominates the distribution, we fine tune their relative strength for each sample as follows. For a system size of $N$, we generate Gaussian i.i.d couplings for the pairwise part of the generator, $J^G_{ij} \sim \mathcal{N}(0, 1/N)$. We call $\sigma^2_{G,pw}$ the variance of the induced pairwise energy across uniformly distributed configurations, $\sigma^2_{G,pw} = \mathrm{Var}[E^G_{pw}] = \sum_{i<j} (J^G_{ij})^2$. Next, we generate i.i.d. parameters $\hat{\xi}^G_I \sim \mathcal{N}(0,1)$, compute the induced higher-order energy variance across uniformly sampled configurations, $\sigma^2_{G,ho} = \sum_I (\hat{\xi}^G_I)^2$, and finally set $\xi^G_I = (\sigma_{G,pw}/\sigma_{G,ho})\, \hat{\xi}^G_I$. We can then use $\gamma$ to set the ratio between the two variances: $\mathrm{Var}[E^G_{ho}] \,/\, \mathrm{Var}[E^G_{pw}] = \gamma$. We note that this procedure is not meant to balance the two terms perfectly for $\gamma = 1$, but rather to give a well-defined starting point for the exploration of different values of $\gamma$. The idea of this work is to explore situations in which a pairwise model describes the variability in the generator well, but not perfectly. We therefore evaluate different values of $\gamma$ in terms of how it affects the training of a purely pairwise model on data from the generator and use this metric to decide which values of $\gamma$ are interesting.

We generate configurations independently sampled from the generator as follows. For $N < 20$, it is feasible to calculate the probabilities involved exactly. We therefore calculate the energies for all possible sequences, exponentiate and normalize them, and then sample sequences using a standard numeric library (Harris et al., 2020). For larger $N$, we resort to the standard Metroplis-Hastings algorithm, which we parallelized on the GPU by running the energy evaluations on all sequences as one batch. We used $N \cdot 10^4$ MC update steps for sampling.

# 4. Results

## 4.1. Reconstructing Pairwise Interactions with Neural Networks

We analyse the effect that additional higher-order interactions in the generating process might have on the reconstruction of the pairwise couplings by training the same models on data from generators with different higher-order strength $\gamma$. We call the criterion that we adopt to measure the reconstruction performance the reconstruction error $\epsilon$. It is a relative measure of the deviation of the inferred couplings $\hat{J}_{ij}$ from the true ones $J_{ij}^G$:

$$\epsilon = \sqrt{\frac{\sum_{i<j}\left(J_{ij}^G - \hat{J}_{ij}\right)^2}{\sum_{i<j}\left(J_{ij}^G\right)^2}}. \tag{10}$$

We expect that the additional interactions will have little to no effect for small values of $\gamma$ in the generative model (9). In this case, we can expect that training a purely pairwise model will lead to satisfactory results. When increasing $\gamma$, however, the generating distribution deviates significantly from a pairwise model, and an increase in the reconstruction error can be expected using a purely pairwise model.

For the experiments in this section, the generators contained $N$ uniformly sampled triplet interactions ($|I| = 3\ \forall I \in \mathcal{I}_G$). Other details of the data generation process are given in Sec. 3, while the training procedure is the one outlined in Sec. 2.3. We generated $M = 5 \cdot 10^4$ training configurations for system size $N = 64$, and $M = 10^4$ for $N = 16$. The neural network part of the hybrid model, $E_{nn}$, was an MLP with one hidden layer of 128 units and $tanh$ activations. For the hybrid model, we evaluated both the explicit couplings in $E_{pw}$ and the reconstructed couplings obtained at the end of the training from Eq. (7).

We compare the reconstruction based on the hybrid model against two other methods: The first is the commonly used regularized pseudolikelihood inference, which amounts to training only the pairwise part $E_{pw}$ of the hybrid model. In this setting, the possibilities of model mismatch and overfitting are often addressed by adding an $L_2$ regularization, which we therefore also add in our experiments for this model type. We found that a relatively low regularization strength $\lambda = 0.01$ lead indeed to a slight improvement for a large range of $\gamma$ and used this value for all our experiments.

The second model we compare against is the energy-based model containing only the neural network part $E_{nn}$.

In Fig. 2 we show the error in the inferred couplings with respect to the couplings in the generator. While for all models the reconstruction degrades as $\gamma$ grows, the hybrid approach performs substantially better than models containing only the pairwise part $E_{pw}$ or only the neural network part $E_{nn}$. The explicit and the reconstructed couplings for the hybrid model yield similar result, meaning that the learned $E_{nn}(s)$ function is approximately orthogonal to the pairwise family in this experiment. It is interesting to note that the neural network with 128 hidden neurons is insufficient to reconstruct the couplings. This confirms the idea that the explicit pairwise model is useful in training. However, we will later show that using networks with much larger capacity, the MLP only model can approach the hybrid model performance in some of the settings explored.

## 4.2. Specificity of the Inferred Interactions

Using the same experimental setting as in the previous section, we investigate in detail how closely the trained hybrid model matches the generator.

In Fig. 3 (left) we compare the reconstructed interaction parameters from the hybrid model through Eqs. (6) and (7) to the corresponding ones in the generator. The interaction parameters that

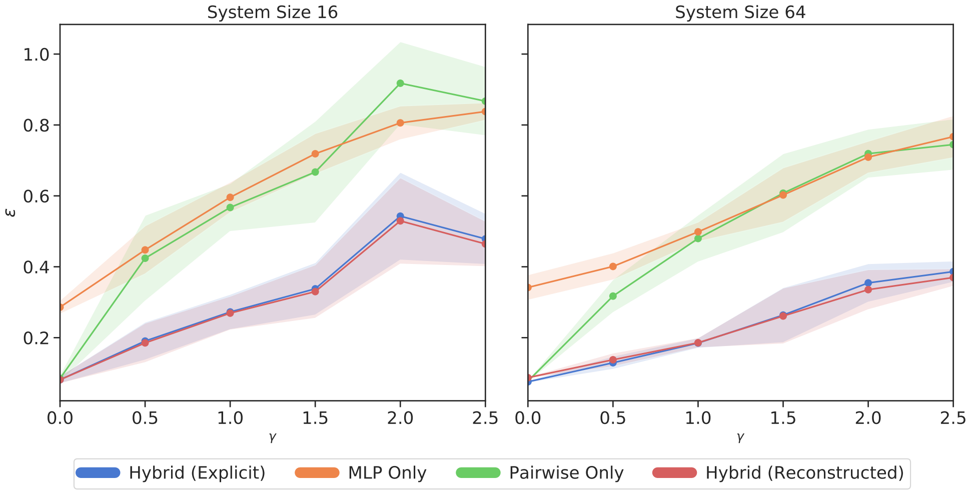

Figure 2: Reconstruction error for different system sizes $N$ and different models as a function of higher-order strength $\gamma$ in the data generator. The data is generated by a pairwise model with $N$ additional interactions involving only 3 variables (see Eq. (9)). For every combination of $\gamma$ and $N$ we sampled 5 generators and used them to create training sets. Shown are means and standard deviations over these training sets. The reconstructed couplings for the Hybrid and the MLP only model are calculated using Eq. (7). Both the hybrid and the MLP only model had a single layer of 128 hidden neurons.

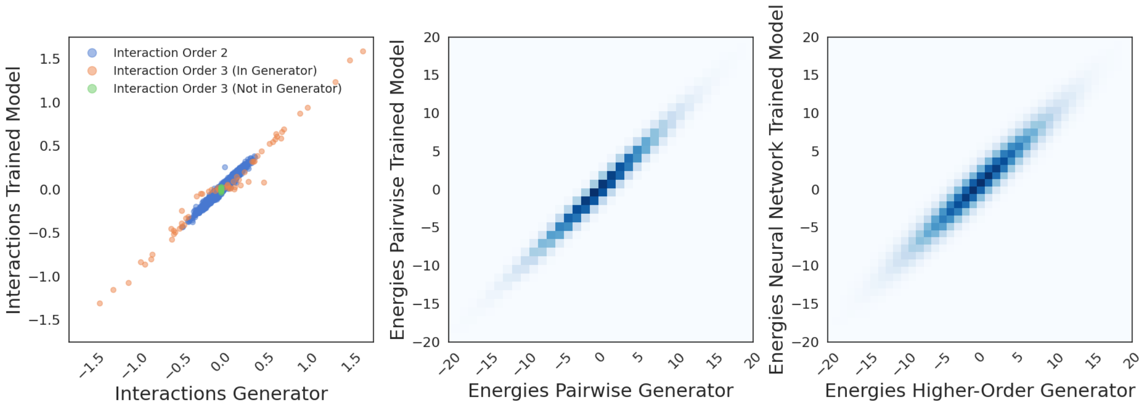

Figure 3: Inferred versus true interactions for system size $N = 64$. The generator includes $N$ triplet interactions and $\gamma$ is set to $1.0$. (Left) Blue points refer to pairwise interactions, orange points to all $64$ triplet interactions present in the generator and green to $64$ random triplet interactions not present in the generator. (Center and Right) Relation of the energies between the submodels of the generator (pairwise and higher-order) and the trained model (pairwise and neural network). The color intensity is proportional to the density of points. The hybrid model contained a single hidden layer with $128$ hidden neurons. All interactions were estimated using Eq. (6) using $10^6$ samples.

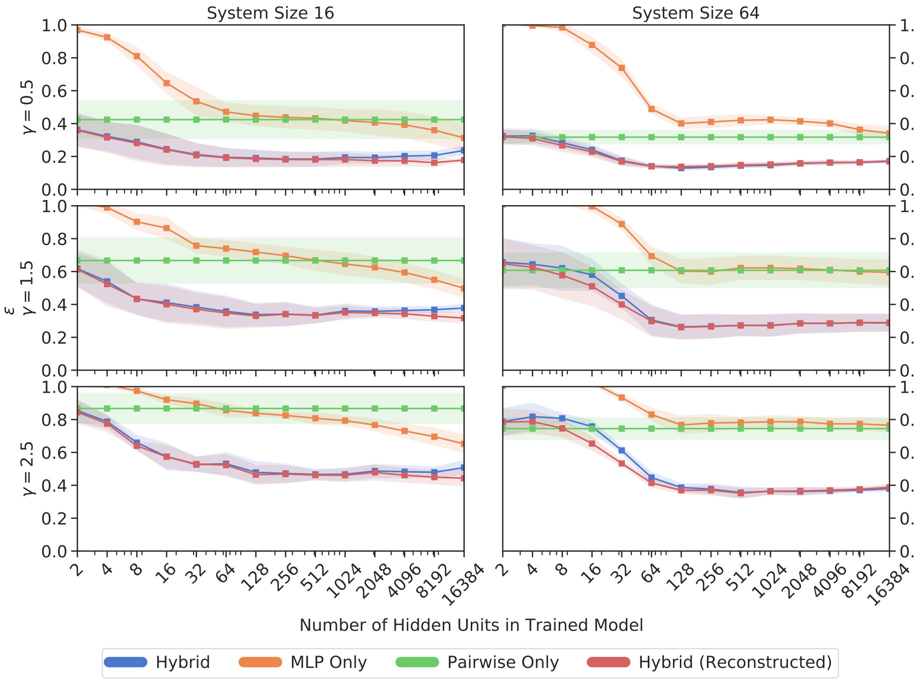

Figure 4: Reconstruction error of couplings in presence of $N$ triplet interactions in the generator, for varying number of hidden neurons in the trained model and different values of $\gamma$. We used $M = 10^4$ training samples for $N = 16$ and $M = 5 \cdot 10^4$ training samples for $N = 64$. For every combination of $\gamma$ and $N$ we sampled 5 independent generators. Shown are means and standard deviation over training sets created from these 5 generators.

we estimate are all pairwise interactions, the $N$ triplet interactions that are present in the generator and $N$ random triplet interactions not present in the generator. Pairwise interactions are well fitted, as well as the strongest triplet interactions in the generator. Some weaker triplet interactions in the generator are underestimated instead. The triplet interactions not contained in the generator are close to 0 in the hybrid model. These results indicate that the hybrid model does not only learn an effective model of the generator, but extracts the true interactions in the underlying system.

In Fig. 3 (center and right) we show that the energies calculated from the pairwise part in the generator are strongly correlated with the energies from the pairwise part in the trained model and the energies calculated from the trained neural network are strongly correlated with the energies coming from the higher-order interactions in the generator.

See also supplementary Fig. 11 for the same experiment on a smaller system.

## 4.3. Varying Network and Sample Sizes

In order to evaluate the impact of the neural architecture used in the hybrid model (5), we repeat the experiments with different sizes for the hidden layer of the MLP. As in the previous section, we keep the higher-order interactions in the generator restricted to $N$ triplets, where $N$ is the system size. We vary the number of hidden neurons between 2 and 16384 in powers of 2. The results in Fig. 4 indicate that size of the neural network has only a small effect on the error above a certain threshold (around 128 in this specific case). While using a pure pairwise model for training leads

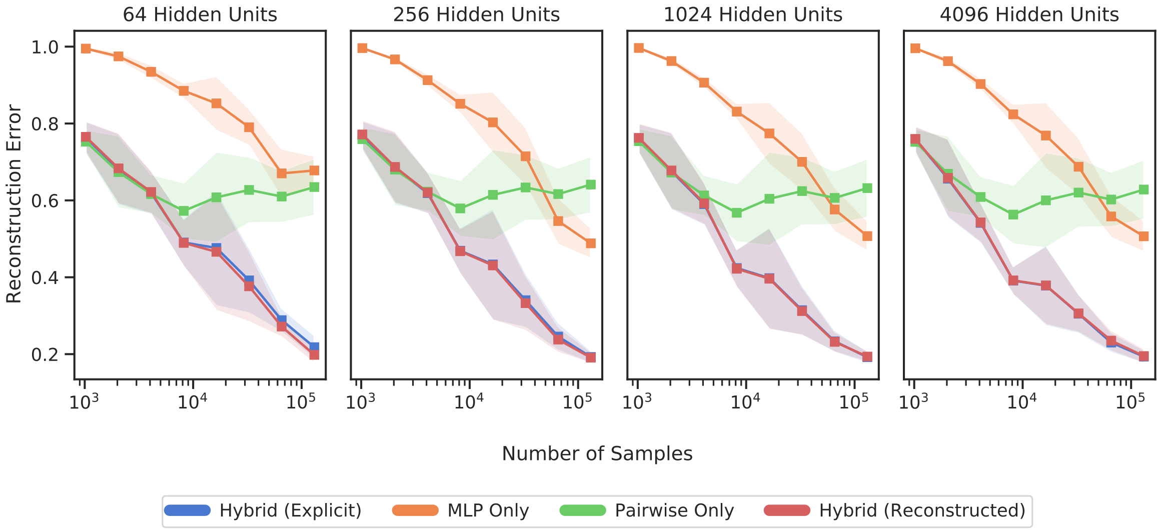

Figure 5: Reconstruction error of couplings in presence of $64$ triplet interactions in the generator as a function of the sample size and for $\gamma = 1.5$. The system size is $N = 64$. For every sample size 5 independent generators were sampled and used to create training set of the corresponding size. The sample sizes used were powers of two from $2^{10}$ to $2^{17}$.

to a quickly increasing reconstruction error (as already visible in Fig. 2), the addition of a single layer neural network with even a small number of hidden neurons (on the order of the system size $N$) leads to a significantly better reconstruction of the pairwise couplings in the generator.

Varying the number of hidden neurons allows us also to test the hypothesis that a sufficiently large neural network on its own is enough for inferring the pairwise couplings. In this setting, the models containing only an MLP approach the performance of the hybrid model only for $N = 16$ and for very wide networks, while a large gap remains at $N = 64$. We note that where models based only on a neural network perform well in terms of the reconstruction error, the hybrid model obtains comparable results with two orders of magnitude less parameters. It is also to be said, however, that this comparison is not completely fair since the hybrid model contains an inductive prior by design, which the pure neural network model lacks. Still, we take this observation as evidence that adding a pairwise part in the trained model is sensible if the generating distribution is expected to contain a significant pairwise part.

In Fig. 5 instead, we fix the network size and evaluate the reconstruction error for increasing sample size. In the small sample size regime, the reconstruction is similarly bad for both the pairwise-only and the hybrid model. Increasing the sample size, the reconstruction error improves but quickly hits a plateau for the pairwise models, while the hybrid models keeps improving. The models based on an MLP only have considerably worse performance across the whole range explored.

## 4.4. Varying the Interaction Orders in the Generator

In the preceding sections we restricted ourselves to triplet interactions in the generator. In order to probe the limits of our approach, we repeat the experiments with generators that contain $N$ higher-order interactions up to order $10$, leaving all other characteristics like training set size and

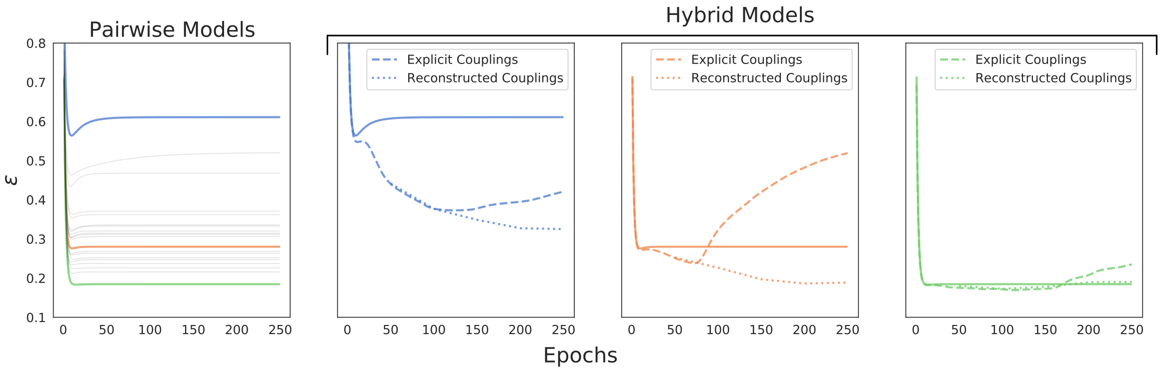

Figure 6: Reconstruction error of couplings in presence of higher-order (3 to 10) interactions in the generator as a function of the training epoch. The higher-order interaction strength $\gamma$ was set to 1.5, the system size is $N = 64$, and we used $M = 5 \cdot 10^4$ training samples. (Left) Reconstruction error for 20 independent generators using only a pairwise model for training. Training sets corresponding to the colored lines are also used in the 3 right panels. (Right 3 panels) Reconstruction error given by pairwise only models (solid lines), and by hybrid models using either explicit or reconstructed couplings. The hybrid models contained a single hidden layer with 256 hidden units. Lines with the same colors correspond to the same generator and training set.

training approach the same. The order of each interaction is chosen from a uniform distribution between 3 and 10, and the variables involved in each interaction are a random subset of all variables.

We note that this is a very ambitious test: the idea is that the neural network picks up the higher-order interactions in the generator, which are of the type $\xi \prod_{i=1}^{I} s_i$, where $I$ is the interaction order and $\xi$ the corresponding parameter. This means that we try to fit a combination of overlapping sparse parity problems of up to 10 inputs. While constructing a solution to a single instance of such a problem is easy using a single hidden layer with continuous weights (see e.g. (Franco and Cannas, 2001)), parity functions are generally considered among the hardest functions to learn from data (Tesauro and Janssens, 1988). While we might be able to alleviate this problem by adding more layers to the neural network, we consider this to be out of scope for the current work and note that in a realistic application the size of the underlying interactions is often not known. Even in this hard case, however, one could expect that the neural network gives a contribution to the quality of training by fitting at least some of the variability due to the higher-order interactions.

While also in this setting we report generally better performance of the hybrid approach over the pairwise only and neural network only approaches, the gain is not as large as in the case of triplet interactions of the last sections (see supplementary Fig. 10). Moreover, in this setting the explicit couplings of the hybrid model significantly deviate from the couplings reconstructed using Eq. 7 at the end of training, as can be seen in Fig. 6. While the additional reconstruction step is computationally cheap, these observations suggest that additional constraints for keeping the pairwise interactions in the neural network small might lead to further improvements. In Appendix B we present a rough way of doing this and speculate about more sophisticated approaches.

## 5. Discussion

In this work we have shown that adding neural networks to pairwise models can improve the quality of reconstruction of pairwise interactions if the distribution underlying the data generating process contains additional higher-order interactions, as it typically occurs in natural data. While both the explicitly pairwise part and the neural network part of the hybrid model may contribute to the reconstructed couplings in general, we showed that in certain settings the neural network and the pairwise model specialize in fitting the separate parts of the generating model.

There are many directions for future investigations. Systematic exploration of the neural architecture employed, which we did not pursue at great length in this work, could yield significant improvements. Different training methods for energy based models could be applied, possibly speeding up simulations or giving more robust predictions. We also did not check the quality of the trained models when used as generative distributions, which might be an important factor when applying similar methods for example to protein design. In addition, constraining the neural network to account only for higher-order interactions in a more sophisticated way might lead to further improvements.

To the best of our knowledge, this work is the first one that solves the inverse problem by using the couplings reconstructed from a neural network. This leads to another line of possible research, were the training of possibly very large and complex generative models without explicit pairwise couplings is followed by a reconstruction step. In principle, common architectures like GANs or autoregressive networks could be adapted at little additional computational cost.

The next immediate step, however, would be to screen the current application domains of pairwise models and translate the improvements observed in the well-controlled settings in this work to real-world data. While we show some preliminary results on homologous protein data in Appendix D, we note that many of these fields present idiosyncrasies, for example in the data characteristics, the expected topology of the underlying interactions or the additional tricks in training or preprocessing that are important for achieving good results when using purely pairwise models. We therefore expect that some additional adaptions to our method are necessary in these cases. Given, however, that in most or all applications pairwise models are used as effective models and one would expect higher-order interactions to play a role in almost all complicated real-world scenarios, we believe that our work presents a very promising perspective. More in general, we consider energy-based models a promising tool for knowledge integration in machine learning.

## References

Erik Aurell and Magnus Ekeberg. Inverse ising inference using all the data. *Physical review letters*, 108(9): 090201, 2012.

Helen Berman, Kim Henrick, Haruki Nakamura, and John L Markley. The worldwide protein data bank (wwpdb): ensuring a single, uniform archive of pdb data. *Nucleic acids research*, 35(suppl_1):D301–D303, 2007.

Julian Besag. Efficiency of pseudolikelihood estimation for simple gaussian fields. *Biometrika*, pages 616–618, 1977.

Miguel A Carreira-Perpinan and Geoffrey E Hinton. On contrastive divergence learning. In *Aistats*, volume 10, pages 33–40. Citeseer, 2005.

Simona Cocco, Christoph Feinauer, Matteo Figliuzzi, Rémi Monasson, and Martin Weigt. Inverse statistical physics of protein sequences: A key issues review. *Reports on Progress in Physics*, 81(3):1–18, 2018. ISSN 00344885. doi: 10.1088/1361-6633/aa9965.

Aurélien Decelle and Federico Ricci-Tersenghi. Pseudolikelihood decimation algorithm improving the inference of the interaction network in a general class of ising models. *Phys. Rev. Lett.*, 112:070603, Feb 2014. doi: 10.1103/PhysRevLett.112.070603. URL https://link.aps.org/doi/10.1103/PhysRevLett.112.070603.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 3608–3618. Curran Associates, Inc., 2019a. URL https://proceedings.neurips.cc/paper/2019/file/378a063b8fdb1db941e34f4bde584c7d-Paper.pdf.

Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019b.

Magnus Ekeberg, Cecilia Lövkvist, Yueheng Lan, Martin Weigt, and Erik Aurell. Improved contact prediction in proteins: using pseudolikelihoods to infer potts models. *Physical Review E*, 87(1):012707, 2013.

Christoph Feinauer, Marcin J. Skwark, Andrea Pagnani, and Erik Aurell. Improving Contact Prediction along Three Dimensions. *PLoS Computational Biology*, 10(10), 2014. ISSN 15537358. doi: 10.1371/journal.pcbi.1003847.

Daniel S Fisher and David A Huse. Ordered phase of short-range ising spin-glasses. *Physical review letters*, 56(15):1601, 1986.

Leonardo Franco and Sergio A Cannas. Generalization properties of modular networks: implementing the parity function. *IEEE transactions on neural networks*, 12(6):1306–1313, 2001.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press Cambridge, 2016.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304, 2010.

Gavin Hall and William Bialek. The statistical mechanics of twitter communities. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(9):093406, 2019.

Charles R. Harris, K. Jarrod Millman, St'efan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fern'andez del R'ıo, Mark Wiebe, Pearu Peterson, Pierre G'erard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(Apr):695–709, 2005.

Aapo Hyvärinen. Consistency of pseudolikelihood estimation of fully visible boltzmann machines. *Neural Computation*, 18(10):2283–2292, 2006.

Abhijith Jayakumar, Andrey Lokhov, Sidhant Misra, and Marc Vuffray. Learning of discrete graphical models with neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5610–5620. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/3cc697419ea18cc98d525999665cb94a-Paper.pdf.

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

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Debora S Marks, Thomas A Hopf, and Chris Sander. Protein structure prediction from sequence variation. *Nature biotechnology*, 30(11):1072–1080, 2012.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6): 1087–1092, 1953.

Jaina Mistry, Sara Chuguransky, Lowri Williams, Matloob Qureshi, Gustavo A Salazar, Erik LL Sonnhammer, Silvio CE Tosatto, Lisanna Paladin, Shriya Raj, Lorna J Richardson, et al. Pfam: The protein families database in 2021. *Nucleic Acids Research*, 49(D1):D412–D419, 2021.

Faruck Morcos, Andrea Pagnani, Bryan Lunt, Arianna Bertolino, Debora S Marks, Chris Sander, Riccardo Zecchina, José N Onuchic, Terence Hwa, and Martin Weigt. Direct-coupling analysis of residue coevolution captures native contacts across many protein families. *Proceedings of the National Academy of Sciences of the United States of America*, 108(49):E1293–301, dec 2011. ISSN 1091-6490. doi: 10.1073/pnas.1111471108. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3241805&tool=pmcentrez&rendertype=abstract.

Alan Morningstar and Roger G Melko. Deep learning the ising model near criticality. *The Journal of Machine Learning Research*, 18(1):5975–5991, 2017.

Alexander Mozeika, Onur Dikmen, and Joonas Piili. Consistent inference of a general model using the pseudolikelihood method. *Physical Review E*, 90(1):010101, 2014.

Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

Jian Peng and Jinbo Xu. Raptorx: exploiting structure information for protein alignment by statistical inference. *Proteins: Structure, Function, and Bioinformatics*, 79(S10):161–171, 2011.

Rama Ranganathan, Kun Ping Lu, Tony Hunter, and Joseph P Noel. Structural and functional analysis of the mitotic rotamase pin1 suggests substrate recognition is phosphorylation dependent. *Cell*, 89(6):875–886, 1997.

Adam J Riesselman, Jung-Eun Shin, Aaron W Kollasch, Conor McMahon, Elana Simon, Chris Sander, Aashish Manglik, Andrew C Kruse, and Debora S Marks. Accelerating protein design using autoregressive generative models. *bioRxiv*, page 757252, 2019.

Alexander Rives, Siddharth Goyal, Joshua Meier, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *bioRxiv*, page 622803, 2019.

Yasser Roudi, Joanna Tyrcha, and John Hertz. Ising model for neural data: model quality and approximate methods for extracting functional connectivity. *Physical Review E*, 79(5):051915, 2009.

Michael Schmidt and Kay Hamacher. hodca: higher order direct-coupling analysis. *BMC bioinformatics*, 19 (1):1–5, 2018.

Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021.

Didier Sornette. Physics and financial economics (1776–2014): puzzles, ising and agent-based models. *Reports on progress in physics*, 77(6):062001, 2014.

Dietrich Stauffer. Social applications of two-dimensional ising models. *American Journal of Physics*, 76(4): 470–473, 2008.

Gerald Tesauro and Bob Janssens. Scaling relationships in back-propagation learning. *Complex Systems*, 2(1): 39–44, 1988.

Gašper Tkačik, Olivier Marre, Dario Amodei, Elad Schneidman, William Bialek, and Michael J Berry II. Searching for collective behavior in a large network of sensory neurons. *PLoS Comput Biol*, 10(1):e1003408, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Laura Von Rueden, Sebastian Mayer, Jochen Garcke, Christian Bauckhage, and Jannis Schuecker. Informed machine learning–towards a taxonomy of explicit integration of knowledge into machine learning. *Learning*, 18:19–20, 2019.

Marc Vuffray, Sidhant Misra, and Andrey Lokhov. Efficient learning of discrete graphical models. *Advances in Neural Information Processing Systems*, 33, 2020.

Michael Waechter, Kathrin Jaeger, Stephanie Weissgraeber, Sven Widmer, Michael Goesele, and Kay Hamacher. Information-theoretic analysis of molecular (co)evolution using graphics processing units. *ECMLS 2012 - 3rd International Emerging Computational Methods for the Life Sciences Workshop*, pages 49–58, 2012. doi: 10.1145/2483954.2483963.

Dian Wu, Lei Wang, and Pan Zhang. Solving statistical mechanics using variational autoregressive networks. *Physical review letters*, 122(8):080602, 2019.

# Supplemental Material

## Appendix A. Using Pseudolikelihoods for training EBMs

Pseudolikelihoods are often used as an alternative to an intractable or at least computationally expensive likelihood (Besag, 1977). It has been applied successfully to pairwise models (Hyvärinen, 2006; Aurell and Ekeberg, 2012; Ekeberg et al., 2013; Decelle and Ricci-Tersenghi, 2014). We show here how it can be applied to a generic Energy-Based Model, and add some considerations specific to pairwise models. We note that while maximum pseudolikelihood is a widely applied method for training simple Energy-Based Models, to the best of our knowledge this is the first time it has been used for training deep feed-forward neural networks.

We assume the data that we want to model to consist of configurations $(s_1, \dots, s_N)$ of categorical variables of length $N$ and we will use $q$ to denote the number of categories. A common method for fitting a probability distribution $p_\Theta(s)$ with parameters $\Theta$ to a training set of sequences $\{s^m\}_{m=1}^M$ is to find the $\Theta^*$ for which

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}} \sum_{m=1}^{M} \log p_\Theta(s^m), \tag{11}$$

which corresponds to a *maximum-likelihood* solution. For an Energy-Based Model (EBM) $p_\Theta(s) = \frac{e^{-E_\Theta(s)}}{Z_\Theta}$, where $E_\Theta(s)$ is the energy function, this would correspond to solving

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}} \frac{1}{M} \sum_{m=1}^{M} \left[-E_\Theta(s^m) - \log Z_\Theta\right], \tag{12}$$

for example by gradient descent methods. The general problem in this approach is that the normalization constant $Z_\Theta = \sum\limits_s e^{-E_\Theta(s)}$, where we sum over all possible configurations $s$, contains $q^N$ terms. This is intractable even for modest $N$ and in the case of binary variables, where $q = 2$. The idea of pseudolikelihoods is to replace the likelihood objective by

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}} \frac{1}{M} \sum_{i=1}^{N} \sum_{m=1}^{M} \log p_\Theta\left(s_i^m | s_{/i}^m\right), \tag{13}$$

where $p_\Theta\left(s_i^m | s_{/i}^m\right)$ is the probability of symbol $s_i^m$ in sequence $m$, given the other symbols. We therefore train the distribution by using it for predicting a *missing* symbol from the other symbols.

Other variations are possible, for example to discard the sum over $i$ and find a maximum set of $\Theta_i^*$ for every $i$ independently. We found the approach with the sum to be conceptually easier and in the applications known to us, the performance seems to be the same (Ekeberg et al., 2013). While it can be shown that this new objective has the same maximum as the original likelihood under certain conditions (Mozeika et al., 2014), this is for example not generally true if the training samples come from a different model class than $p_\Theta$, which is true in our case. In this work, we are interested in whether we can make training using this objective work in practice and refrain from further theoretical analysis.We note that we have not restricted the form of $E_\Theta$. In the models we analyse in this work, the energy is calculated by a sum of the energy of a pairwise model and a neural network.

Neglecting the sum over $i$ and $m$ for the time being, we can write the quantity $\log p_\Theta(s_i|s_{/i})$ for an EBM as

$$\log p_\Theta(s_i|s_{/i}) = \log \frac{p_\Theta(s)}{p_\Theta(s_{/i})} = \log \frac{p_\Theta(s)}{\sum_{\hat{s}_i=1}^{q} p_\Theta(\hat{s}_i, s_{/i})}, \tag{14}$$

where we used the notation $(\hat{s}_i, s_{/i})$ for the configuration $s$ after $s_i$ has been replaced with $\hat{s}_i$. Since the normalization constant $Z_\Theta$ appears in the both the numerator and denominator, it cancels and we are left with

$$\log p_\Theta(s_i|s_{/i}) = \log \frac{e^{-E_\Theta(s)}}{\sum_{\hat{s}_i=1}^{q} e^{-E_\Theta(\hat{s}_i, s_{/i})}} = -\log\left(1 + \sum_{\hat{s}_i \neq s_i} e^{E_\Theta(s) - E_\Theta(\hat{s}_i, s_{/i})}\right) \tag{15}$$

The sum in this expression can be computed efficiently, using $q$ evaluations of $E$. This means that including the sum over $i$ and replacing the sum over $m$ with a sum over a mini-batch of $B$ in a *stochastic gradient descent* (SGD) setting, we need $q \cdot N \cdot B$ evaluations of $E$ for a single gradient step, corresponding to $q \cdot N$ forward passes.

In the case of binary strings with $s_i \in \{\pm 1\}$ and a pairwise model $E_\Theta(s) = -\sum_{i<j} J_{ij} s_k s_j$ with parameters $\Theta \equiv J$, we can simplify further by noticing that

$$E(s) - E(\hat{s}_i, s_{/i}) = (\hat{s}_i - s_i) \sum_{j \neq i} J_{ij} s_j, \tag{16}$$

where we identified $J_{ij} = J_{ji}$ for convenience. Since in Eq. (15) we sum only over $\hat{s}_i \neq s_i$ and in this case $\hat{s}_i - s_i = -2s_i$, this leads to

$$\log p_\Theta(s_i|s_{/i}) = -\log\left(1 + e^{-2s_i F_i\left(J, s_{/i}\right)}\right), \tag{17}$$

where $F_i(J, s_{/i}) = \sum_{j \neq i} J_{ij} s_j$. This means that in this model class, we do not need to evaluate the full energy, which contains $\Theta(N^2)$ terms, but only the part of the energy involving the variable $s_i$, which contains only $\Theta(N)$ terms. The quantities $F_i\left(J, s_{/i}\right)$ can be obtained for a whole batch of sequences using matrix multiplication, which is very efficient on modern GPUs.

## Appendix B. Absorbing Pairwise Interactions from the Neural Network

During training, we did not enforce a division of labour between the two parts of the hybrid models, which means that the neural network is not discouraged in any way from fitting also pairwise interactions. While extracting the pairwise coefficients from the entire hybrid model and constructing an effective pairwise model is a way of solving this after training, it would be more satisfactory to include this also in the training procedure. The cleanest way of ensuring only higher-order interactions in the neural network would be to constrain the optimization of the neural network to the part of parameters space where it does not contain pairwise interactions. In practice, Eq. (6) could be used to create a regularization term penalizing all pairwise interactions:

$$\frac{1}{N^2} \sum_{ij} \left(\mathbb{E}_s\, s_i s_j\, E_{nn}(s)\right)^2 = \mathbb{E}_{s,s'}\, E_{nn}(s) E_{nn}(s')\, q^2(s, s'), \tag{18}$$

where the expectation is over uniformly sampled Ising configurations and $q(s, s') = \frac{1}{N} \sum_i s_i s'_i$ is the overlap between two configurations. This expression can be approximately evaluated by Monte

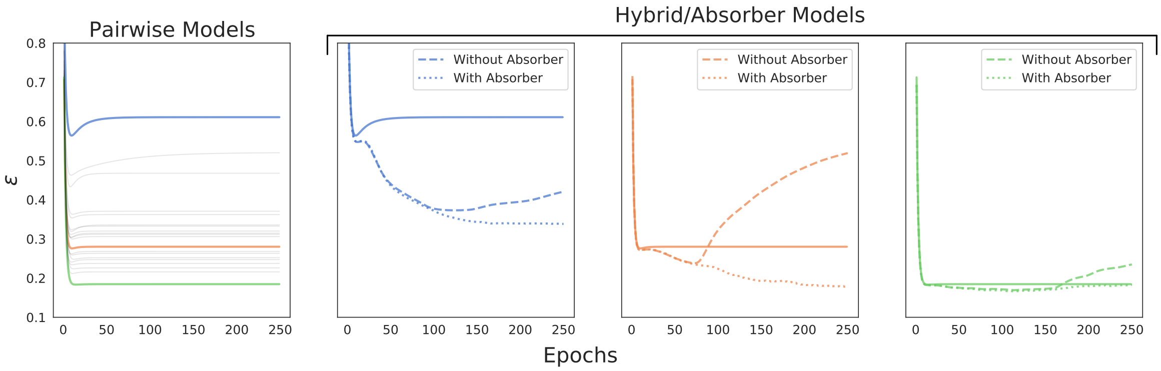

Figure 7: Reconstruction error with 64 higher-order (3 to 10) in generator for $\gamma = 1.5$ and trained with $M = 5 \cdot 10^4$ samples. The training samples in this figure are the same as in Fig. 6. Left panel: Reconstruction error for 20 independent generators using only a pairwise model for training. The training sets corresponding to the colored lines were further used as a training set for the hybrid and absorber models shown in the 3 right panels. Right 3 panels: Reconstruction error for trained models containing only pairwise terms (solid lines), reconstruction error for hybrid models (dashed lines) and reconstruction error for hybrid models with absorber terms (dotted lines). Lines with the same colors correspond to the same generator and training set.

Carlo sampling. While this approach seems promising, we did not pursue it in this exploratory analysis.

A different approach instead is to counter the pairwise interactions in the neural network by using an additional pairwise model. To this end, we define a new energy

$$E(s) = E_{pw}(s) + E_{nn}(s) - \hat{E}_{pw}(s). \tag{19}$$

Here, $E_{pw}$ and $E_{nn}$ are the same as in the hybrid models of the preceding sections. The new term $\hat{E}_{pw}$ is another pairwise model, but it is excluded from the gradient descent step and we set its couplings explicitly every $k$ epochs. The values of these couplings are the pairwise interactions extracted from $E_{nn}$ using Eq (6). The idea is to estimate the pairwise terms in the expansion of the neural network energy $E_{nn}$ and *absorb* these interactions in the additional $\hat{E}_{pw}$, which we therefore call an absorber model. After setting the couplings of this absorber, the last two terms on the right hand side of Eq. (19) should contain approximately no pairwise interactions, i.e.

$$\sum_s s_i s_j \left( E_{nn}(s) - \hat{E}_{pw}(s) \right) \approx 0 \tag{20}$$

for all $i, j$. This leaves the term $E_{pw}$ as the only one with significant pairwise interactions. While we could do this in principle after every epoch or even after every gradient step, this would make the computations unfeasibly slow since at every step we estimate the pairwise interactions in $E_{nn}$ using $10^6$ samples. We therefore restrict ourselves to doing the estimate less frequently, every $k = 5$ epochs in our experiments.

In Fig. 7 it can be seen that using these additional absorbers improves the training of the couplings significantly. We used the same training samples as for Fig 6 and also left the other training characteristics the same. The results are very similar to what would have been obtained by reconstructing the couplings at every step (compare also Fig. 6). While this also means that there was no strong improvement over approach of reconstructing the couplings after the training has ended, we think it still noteworthy that enforcing that the pairwise model $E_{pw}$ should be the one solely responsible for fitting pairwise interactions is possible during training.

## Appendix C. Relation between an MLP-based energy model and RBMs

In our hybrid energy model, combining a physics-inspired energy term and a black-box one, we decided to model the latter through a multi-layer perceptron with two fully connected layers of learnable weights. Within our framework, several alternatives choices can be made, among which the use of a restricted Boltzmann machine (RBM). RBMs have been used for a long time as generative models, although they have been generally replaced by more modern techniques: they are able to capture arbitrary-order interactions among the features and their bipartite structure makes them easy to train using efficient Monte Carlo sampling in contrastive divergence (Hinton, 2002). Learning by contrastive divergence becomes more demanding when inter-visible connections are present, therefore one of the main advantages of RBM over alternative energy models is lost in our hybrid framework. Nonetheless, training is still possible using the pseudo-likelihhod objective we propose, which is agnostic to the specifics of the energy modelling.

We remark that the energy function of an RBM defined on the visible variables $s$ once the hidden ones are traced out is equivalent to that given by an MLP with two layers, where the top one has weights all equal to one. In fact, calling $W$ the visible-to-hidden couplings and $b$ the fields on the hidden neurons, and assuming hidden neurons that take $0/1$ as values, we can trace them out and obtain

$$E(s) = \sum_k \mathrm{softplus}\left(\sum_i W_{ki} s_i + b_k\right), \tag{21}$$

where $\mathrm{softplus}(x) = \log(1 + e^x)$.

## Appendix D. Residue-residue contact prediction

In order to assess the applicability of the method to real-world data, we ran a number of preliminary experiments using a hybrid model on homologous protein sequences. In this case, the pairwise model is a 21-state Potts model and the inputs are protein sequences of length $N$, where at any position one of $q = 21$ different amino acids can occur. The mathematical form of the pairwise energy for a sequence $a = (a_1, \dots, a_N)$ is

$$E_{Potts}(a) = -\sum_{i=1}^{N} \sum_{j=i+1}^{N} J_{ij}(a_i, a_j) - \sum_{i=1}^{N} h_i(a_i), \tag{22}$$

where the couplings $J_{ij}(a, b)$ describe the contribution to the energy when finding amino acid $a$ at position $i$ and $b$ at position $j$, while the fields $h_i(a)$ describe the contribution to the energy of finding amino acid $a$ at position $i$. We refer to the review (Cocco et al., 2018) for further details. The couplings of the trained Potts model are interpreted as the strength of co-evolution between the positions in the protein. Since nearby residues in a protein tend to co-evolve, large couplings can be seen as evidence for protein contacts (Morcos et al., 2011). The training set is given by a set of $M$ evolutionary related sequences $\{a^m\}_{m=1}^{M}$, which have already been preprocessed to have the same length $N$. Protein structures from the *Protein Data Bank* (Berman et al., 2007) can be used for benchmarking. For the definition of protein contacts, we refer to (Morcos et al., 2011).

In order to build a hybrid model, we add to $E_{Potts}(a)$ an energy $E_{MLP}(a)$ based on a multilayer perceptron. The neural network takes a one-hot encoded version of an amino acid sequence as an input and returns a single number, interpreted as the energy. We compare the hybrid model, the pairwise Potts model and the model using only the neural network. The inference method follows the same line as described Section A of this Appendix, using pseudolikelihoods as the objective function. We used a $L2$ regularization of $0.01$ for $E_{Potts}$ and none for $E_{MLP}$. The learning rate was set to $0.01$. Since it is not standard to use small batch sizes when training a Potts model, we used full-batch gradient descent and ran the experiments for $1000$ epochs.

In this preliminary application, to be followed by a more extensive investigation, we did not optimize the training for this specific use case or run extensive hyper-parameter searches.

We use the same sequence reweighting technique, score calculation and average-product corrections as described in (Ekeberg et al., 2013).

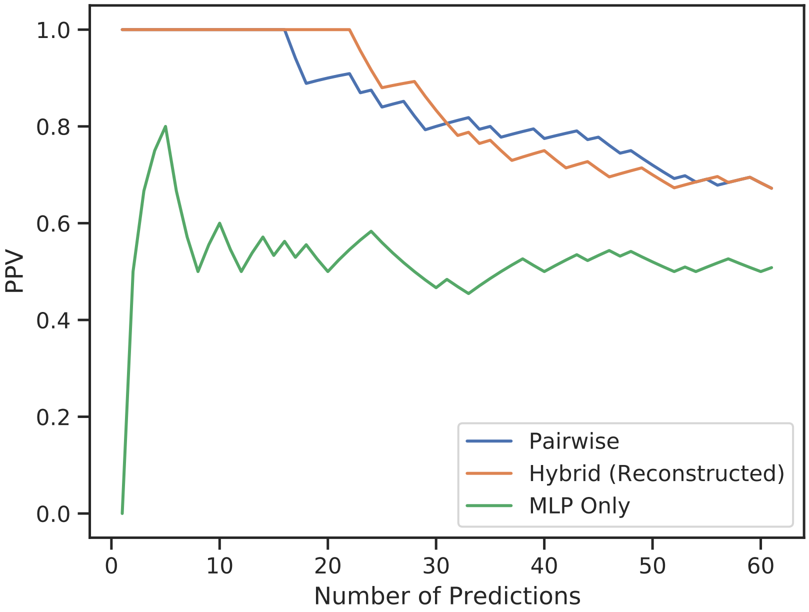

Figure 8: *Positive Predictive Values* with respect to contact prediction in the WW Domain. The PPV value (ordinate) at a number $N$ of predictions (abscissa) is the fraction of true positives within the top scoring $N$ predictions. The blue line corresponds to predictions derived from a model featuring only a pairwise term, the orange line to a hybrid model (a pairwise term with an MLP with 32 hidden units) and the green line to predictions when using only the MLP term.

We ran the experiments on the WW domain dataset from *Pfam 33.1* (Mistry et al., 2021) with ID PF00397, and PDB structure 1PIN (Ranganathan et al., 1997) for assessment. The WW domain is a well-studied domain that is relatively small ($N = 31$) while having $\approx 11000$ unique samples, of which we used $90\%$ for training and $10\%$ as a test set for evaluating the objective function. In Fig. 8 we show the positive predictive value (PPV) for contact prediction using different models, for the first $2N$ predictions. We used 32 hidden units for the hybrid and MLP-only models in these plots,

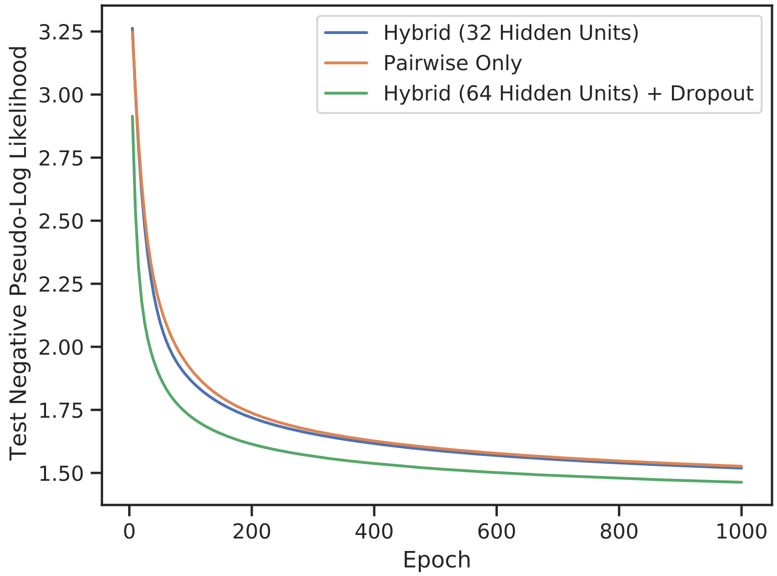

Figure 9: Negative Pseudo-Log Likelihood on the test set during training for three different models: A Potts model (orange), a hybrid model with 32 hidden units (blue) and a hybrid model with 64 hidden units and a weight dropout with probability parameter $p = 0.5$ (green).

which is very similar to the sequence length. As can be seen in the figure, the hybrid model produces the first false positive several positions later than the Potts model (position $22$ versus $16$). Both models arrive at the same PPV at $2N$ predictions. The model using only the neural network misses the first prediction and has a significantly lower PPV than the other two models for all number of predictions.

Monitoring the objective function on the test set, we saw only a very small improvement for the hybrid model. We therefore ran an experiment with $64$ hidden units and added weight-dropout on both layers with probability parameter $p = 0.5$. For this setting, we observed an improvement in the objective function. Since the negative pseudo-log likelihood is related to the probability for observing an amino acid at some position given the other amino acids in the sequences, it is directly related to the task of reconstructing a missing amino acid. An improvement in this metric can therefore be seen as an improvement in the generative properties of the model. Surprisingly, this improvement was not accompanied by an improvement in contact prediction (data not shown). We speculate that it might promising to explore use of different architectures, regularization, and alternative objective functions for this specific application. We noticed a tendency to overfit the dataset for larger networks. We stress that these experiments cannot be seen as more than a preliminary demonstration of feasibility. A structured exploration of hyper-parameter values and neural architectures is necessary to fully understand the potential of hybrid models when applied to homologous protein sequences.

## Appendix E. Additional Figures

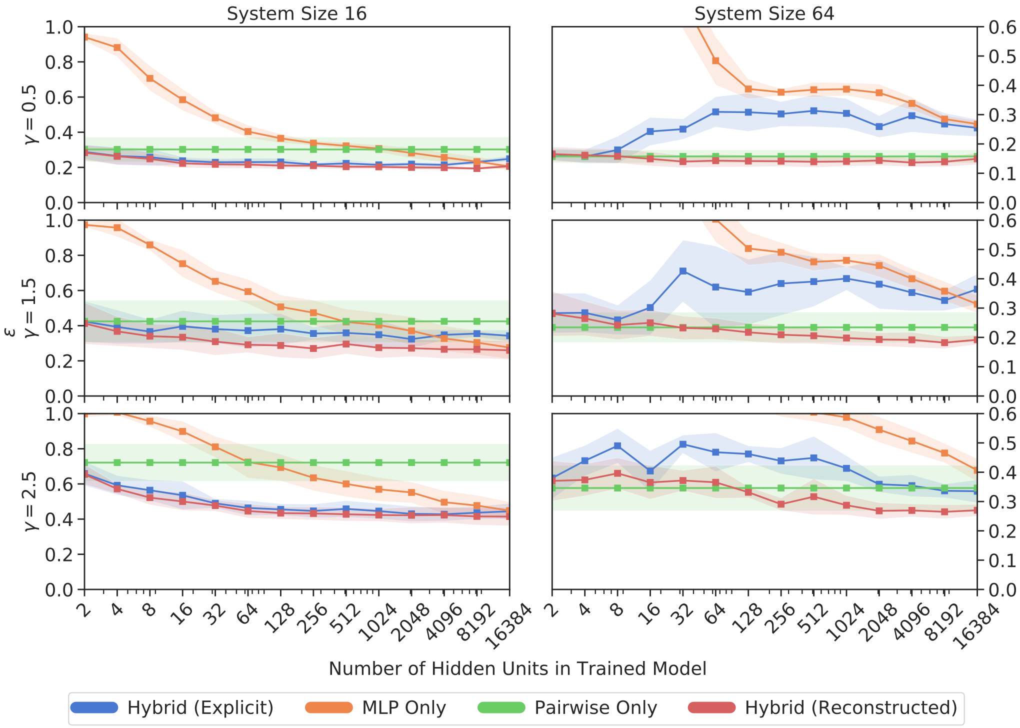

Figure 10: Pairwise reconstruction errors with higher-order (size 3 to 10) interactions in generator for varying number of hidden neurons in the trained model and 3 different values of $\gamma$. The number of these interactions is equal to $N$. Training was done with $M = 10^4$ samples for $N = 16$ and $M = 5 \cdot 10^4$ samples for $N = 64$. Shown are means and standard deviation over 5 runs. The reconstruction error is defined in Eq. (10). The blue line corresponds to reconstruction error calculated using the couplings of the pairwise part of the hybrid model, the red line to the pairwise interactions reconstructed from the complete hybrid model using Eq. (7)

.

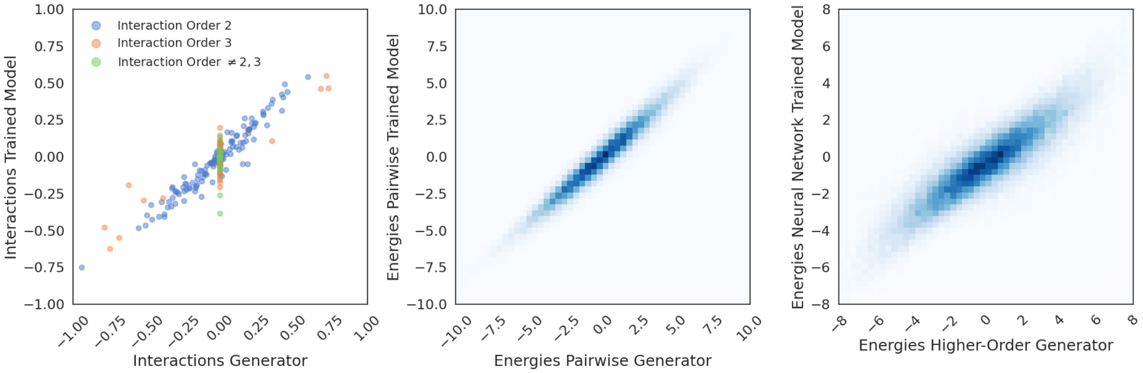

Figure 11: Inferred versus true interactions for system size $N = 16$. The generator included $N$ triplet interactions and $\gamma$ was set to $1.0$ and the hybrid model had a single hidden layer with $128$ hidden units. The left panel shows all interactions color coded by their order: blue points refer to pairwise interactions, orange points to triplet interactions and green to all other orders. The middle and right panel show the relation of the energies between the submodels of the generator (pairwise and higher-order) and the trained model (pairwise and neural network). The color is proportional to pairs of energy values that fall in the corresponding quadrant. All interactions were recovered using Eq. (6) using all possible sequences and should be exact. The energies in the middle and right panel also correspond to all possible sequences.