Learning Regularized Positional Encoding for Molecular Prediction

Anonymous Author(s)
Affiliation
Address
email

Abstract

Machine learning has become a promising approach for molecular modeling. Positional quantities, such as interatomic distances and bond angles, play a crucial role in molecule physics. The existing works rely on careful manual design of their representation. To model the complex nonlinearity in predicting molecular properties in an more end-to-end approach, we propose to encode the positional quantities with a learnable embedding that is continuous and differentiable. A regularization technique is employed to encourage embedding smoothness along the physical dimension. We experiment with a variety of molecular property and force field prediction tasks. Improved performance is observed for three different model architectures after plugging in the proposed positional encoding method. In addition, the learned positional encoding allows easier physics-based interpretation. We observe that tasks of similar physics have the similar learned positional encoding.[1]

1 Introduction

The prediction of molecular properties is crucial for the discovery of molecules with desired properties, with applications in drug, material, and energy industries. *Ab initio* quantum chemical calculations, such as these based on density functional theory (DFT), give accurate predictions for the chemical properties with a high computational cost. Semi-empirical methods [1] are faster but much less accurate. Machine learning based molecular predictor is a promising alternative solution to balance accuracy and efficiency. In recent years scientists have started leveraging machine learning methods in molecular prediction tasks [2, 3].

An effective machine learning model should represent the key physical quantities properly. Positional quantities, such as interatomic distances and bond angles, are among the most fundamental physical quantities for ML-based molecular predictors. According to Born-Oppenheimer approximation, a molecular system is uniquely determined by the nucleus positions and charges. Almost all molecular properties are thus a function of the positions and types of the atoms. Moreover, interatomic distances and bond angles are invariant to transformations such as rotations and translations. These symmetric properties make them preferred inputs to build machine learning models to exploit the symmetry of physical problems [4, 5, 6].

The design of an effective positional quantity representation faces challenges from the complex physics in molecules. First, the interactions in molecules are often nonlinear or even non-monotonic with respect to the positional quantities. Lennard-Jones potential, for instance, is a function of a repulsive term and an attractive term. Second, multiple nonlinear effects co-exist and they scales differently with positional quantities. For example, Lennard-Jones potential involves two polynomial

[1] The code will be open-sourced on GitHub.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

terms, while Morse potential contains two exponential terms. Ideally, a good representation of positional quantities should help the machine learning models to effectively model these multi-modal non-linearity.

Existing methods of positional quantities representations can be grouped into three categories. (*i*) Directly using original scalar form as the model inputs [6]. Although neural networks can learn any function of the inputs according to the universal approximation theorem, it is practically challenging given the multiple complex physics in molecules mentioned above. (*ii*) Using a manually defined set of simple transformations, such as the reciprocal of interatomic distances used by Wang et.al. [7]. Such representations maybe more effective than directly using original scalar forms, but they are only ideal for limited potential functions. For example, Lennard-Jones potential contains polynomial terms of reciprocal of interatomic distances but Morse potential do not. The generalizability of such a method is limited. (*iii*) Using the representation of positional quantities under a set of manually picked basis functions , such as the Gaussian kernels [8, 9] or Bessel functions [10]. This method also has limited generalizability and the efficiency depends on the choice of the basis functions.

Motivated by the success of learnable positional encoding in natural language processing (NLP) [11, 12], we propose to use learnable positional encoding in molecular properties prediction.Compared to the existing methods, the learnable position embedding is not limited to a manually picked function format. Empirically we demonstrate that our method improve the accuracy on multiple molecular property and force field tasks compared to state-of-the-art models. Our method works as a plug-in module, combined with any models with positional quantities as inputs. Moreover, we propose a smoothness regularization technique, which filters out unnecessary embedding fluctuations and makes physics-based interpretation possible. We demonstrate that the learned embedding reflects the dependence of different tasks on short and long-range interactions, and observe that tasks of similar physical nature result in similar positional encoding.

Our contribution is two-folds.

1. We propose a learnable positional encoding method that is continuous and differentiable. The method can be used to represent interatomic distance, bond angles and other positional quantities with a series of nonlinear transformations. Experiments show that new state-of-the-art results are achieved with this technique.

2. We propose a regularization method for the proposed positional encoding. Based on physical intuition, this method encourages the embedding to form a meaningful manifold for easier visualization and interpretation from a physical point of view. The regularization reduces overfitting and can further increase model accuracy for certain tasks.

2 Related works

2.1 Positional encoding

There are generally two types of positional encoding, fixed and learned. Fixed positional encoding represent positions with manually picked basis functions. In the initial version of transformer [13], the discrete token positions are represented by a series of sine and cosine functions of a range of fixed frequencies. Gaussian kernels are employed to represent some continuous physical quantities such as interatomic distances in works of molecule modeling [8, 9]. DimeNet [10, 14] and GemNet [15] and SphereNet [16] instead use Spherical Bessel functions and spherical harmonics. Learned positional encoding represent each position with a learned vector. Although initially the fixed positional encoding was proposed for transformers [13], the learnable positional encoding has become more popular technique in natural language processing (NLP) community, as in BERT [11] and GPT models [17, 18, 12]. Besides transformers, learnable positional encoding is used with convolutional neural networks [19].

2.2 Molecular modeling

Convolution neural networks are equivariant to translations and this enables them to generalize better on image problems. This characteristic inspires researcher to use them in molecular properties prediction, where translation equivalence is a useful inductive bias [20, 4, 5]. Graph neural networks are another popular family of models used in modeling molecules. [21, 6, 22, 23, 14, 10].

3 Method

In this section, we introduce our method to encode positional quantities such as interatomic distance or bond angle, to a fixed-length learnable vector, $h(x) \in \mathbb{R}^s$, where s is the embedding size.

3.1 Continuous and Differentiable Positional Encoding

In contrast to the application in NLP, where the positional embedding are designed for discrete tokens, the positional quantities are continuous for molecule modeling. To represent these *continuous* physical quantities with embeddings on *discrete* points, we divide the space of x into n_{bin} bins, and obtain the embedding of a arbitrary x by interpolating the embeddings of the nearest bin centers.

A simple implementation is via linear interpolation

$$h(x) = (1-t)h(\lfloor x \rfloor) + th(\lceil x \rceil), \quad t \equiv \frac{x - \lfloor x \rfloor}{\lceil x \rceil - \lfloor x \rfloor},$$

where $\lfloor x \rfloor$ is the largest bin center smaller than x, and $\lceil x \rceil$ is the smallest bin center larger than x.

The linear interpolation method is simple but it makes $h(x)$ not differentiable with respect to x, as the derivative $g(x) \equiv \frac{dh(x)}{dx} \in \mathbb{R}^s$ is not continuous at the bin centers. $g(x)$ is required if the trained model is used in calculation that involving the derivative with respect to x. For example, the potential force acting on atoms is the derivative of potential energy with respect to their positions. To resolve this issue, we consider higher order interpolation. We choose the Cubic Hermite spline as follows,

$$h(x) = c_1 h(\lfloor x \rfloor) + c_2 h(\lceil x \rceil) + c_3 g(\lfloor x \rfloor) + c_4 g(\lceil x \rceil),$$

where the interpolation coefficients are

$$c_1 = 2t^3 - 3t^2 + 1,\ c_2 = 1 - c_1,\ c_3 = t^3 - 2t^2 + t,\ c_4 = t^3 - t^2.$$

The positional encoding $h(x)$ defined in this way is continuous and differentiable with respect to x.

This implementation involves two sets of learnable parameters, $\{h(x_i)\}$ and $\{g(x_i)\}$, where $i = 1, 2, ..., n_{\text{bin}}$ is the bin index, and $\{x_i\}$ are the bin centers. The total number of additional trainable parameters is $2sn_{\text{bin}}$. The proposed encoding method provides a simple way to build a universal approximation of any continuous functions of x. The accuracy of the approximation can be easily increased by adding the number of bins.

3.2 Smoothness Regularization

As the positional quantities are continuous, we assume the embedding should be locally smooth. The difference between the embedding of two adjacent bins should not be large. Following this intuition we consider a smoothness loss, defined as the average relative change of a embedding, $h(x_i)$, compared to the next bin, $h(x_{i+1})$.

$$\mathcal{L}_{\text{smooth}} = \frac{\sum_{i=1}^{n-1} ||h(x_{i+1}) - h(x_i)||}{\sum_{i=1}^{n-1} ||h(x_i)||}.$$

where $||.||$ is 2-norm. During training, $\mathcal{L}_{\text{smooth}}$ is combined with the original training loss $\mathcal{L}_{\text{orig}}$ as the new training loss

$$\mathcal{L} = \mathcal{L}_{\text{orig}} + \lambda \mathcal{L}_{\text{smooth}},$$

where λ is a hyperparameter to control the contribution of the smoothness loss.

$\mathcal{L}_{\text{smooth}}$ acts as a regularization term as it reduces the flexibility of the positional encoding. This may help the model generalize in case of small amount of data. Even if a bin is never directly trained during training (i.e., no x fall near this bin during training), its parameters are still updated because its difference with neighbor bins is regularized by the smoothness loss. We will show that smoothness regularization can improve model accuracy and provide physical interpretability in the next section.

3.3 Implementation

The regularized positional encoding is proposed as a plug-in for existing models. When a model uses positional quantities as their input, one may consider replacing these quantities by the proposed embedding, as illustrated in Figure **??**. In this work, we consider three backbond models: EGNN [6], DimeNet++ [14], and a Transformer[13] based model[2]. The experiments are conducted on a Nvidia Tesla V100 GPU.

4 Experiments and discussion

4.1 Molecular force fields

	Aspirin	Benzene	Ethanol	Malonaldehyde	Naphthalene	Salicylic	Toluene	Uracil
sGDML	29.5	2.6	14.3	17.8	4.8	12.1	6.1	10.4
FCHL19	20.7	-	5.9	10.6	6.5	9.6	8.8	4.6
DimeNet	21.6	-	10.0	16.6	9.3	16.2	9.4	13.1
SphereNet	18.6	-	9.0	14.7	7.7	15.6	6.7	11.6
NequIP	15.1	2.3	9.0	14.6	4.2	10.3	4.4	7.5
PaiNN	14.7	-	9.7	14.9	**3.3**	8.5	**4.1**	**6.0**
Transformer	$49.7_{(0.9)}$	$36.8_{(0.6)}$	$51.0_{(1.3)}$	$53.0_{(1.5)}$	$50.4_{(1.5)}$	$53.2_{(1.6)}$	$47.6_{(1.5)}$	$54.5_{(1.9)}$
+PosEnc	$24.6_{(0.8)}$	$3.3_{(0.2)}$	$13.8_{(0.3)}$	$24.2_{(0.7)}$	$11.6_{(0.5)}$	$22.5_{(0.9)}$	$15.2_{(0.5)}$	$17.7_{(0.8)}$
+Smooth	$12.3_{(0.5)}$	$1.9_{(0.2)}$	$6.2_{(0.5)}$	$13.8_{(0.8)}$	$5.4_{(0.4)}$	$8.3_{(0.7)}$	$5.7_{(0.2)}$	$10.1_{(0.2)}$
EGNN	$51.9_{(1.1)}$	$36.9_{(0.4)}$	$50.1_{(1.3)}$	$53.5_{(1.8)}$	$50.5_{(1.2)}$	$53.3_{(0.9)}$	$47.6_{(0.4)}$	$54.6_{(0.6)}$
+PosEnc	$9.3_{(0.8)}$	$\mathbf{1.5}_{(0.2)}$	$4.9_{(0.3)}$	$8.8_{(0.7)}$	$4.9_{(0.4)}$	$7.8_{(0.8)}$	$7.7_{(0.5)}$	$7.4_{(0.6)}$
+Smooth	$\mathbf{6.7}_{(0.5)}$	$\mathbf{1.3}_{(0.2)}$	$\mathbf{3.5}_{(0.3)}$	$\mathbf{7.0}_{(0.8)}$	$\mathbf{3.0}_{(0.3)}$	$\mathbf{6.5}_{(0.5)}$	$\mathbf{4.1}_{(0.4)}$	$\mathbf{6.2}_{(0.4)}$

Table 1: Test error (MAE in meV/Å) on MD17 DFT dataset. Gray rows are our methods. **Bold** text indicates the best error. The numbers in brackets are the standard deviation.

We experiment with a set of molecular force field datasets, MD17 [24]. The task is to predict the force acting on each atom given the 3D geometry (configuration) of the molecule. Following [25], we use 1000 training and test configurations.

Surprisingly, as illustrated in Table 1, both EGNN and the Transformer-based model shows a large error, although these two architectures previously achieve success in a various tasks [6, 26]. We hypothesise the data characteristics of the current task makes the relatively simple representation of the interatomic distance r by EGNN and the Transformer-based model performing not well on this task. Molecular force fields are highly sensitive to the change of 3D geometries. The 3D geometries of the same molecule in MD17 dataset only slightly different from each other, while the forces acting on atoms change significantly across molecular configurations.

We then use the proposed positional encoding method (+PosEnc in Table 1) to represent the interatomic distance. This significantly reduce the test error for both EGNN and the transformer-based model. When we employ the proposed smoothness regularization technique (+Smooth in Table 1), the test error further decreases.

We compare the results with a few recent works, sGDML [27], FCHL19 [28], DimeNet [10], SphereNet [29], NequIP [25], and PaiNN [30]. They either use kernels or carefully designed spatial basis functions to represent the positional quantities. We show that with the proposed learnable positional encoding technique, similar or better results on force fields can be achieved, as illustrated in Table 1.

4.2 Molecular properties

We then experiment with the molecular property prediction tasks using the QM9 [31] dataset. This dataset consists the DFT calculation results of 134 k stable small organic molecules. The

[2]It uses self attention mechanism to model the interaction among atoms. The interatomic distance is sent to a multi-layer perceptron (MLP) as the bias vector for the multi-head attention

Task	α	$\Delta\epsilon$	ϵ_{HOMO}	ϵ_{LUMO}	μ	C_v	G	H	R^2	U	U_0	ZPVE
Unit	bohr^3	meV	meV	meV	D	cal/mol K	meV	meV	bohr^3	meV	meV	meV
NMP	.092	69	43	38	.030	.040	19	17	.180	20	20	1.50
SchNet	.235	63	41	34	.033	.033	14	14	.073	19	14	1.70
Cormorant	.085	61	34	38	.038	.026	20	21	.961	21	22	2.03
L1Net.	.088	68	46	35	.043	.031	14	14	.354	14	13	1.56
LieConv.	.084	49	30	25	.032	.038	22	24	.800	19	19	2.28
DimeNet++	.048	44	27	21	.032	.023	7	7	.334	7	7	1.18
+MLP	.059	49	30	25	.035	.027	11	11	.286	11	10	1.54
+PosEnc	.048	44	26	**20**	.030	.023	7	7	**.260**	7	7	1.18
+Smooth	.048	44	**24**	**20**	**.028**	.023	7	7	**.260**	7	7	1.18
EGNN	.071	48	29	25	.029	.031	12	12	.106	12	11	1.55
+MLP	.068	45	35	24	.027	.029	10	10	.122	11	10	1.50
+PosEnc	.063	44	**26**	**22**	.027	**.028**	10	10	**.086**	10	**9**	**1.49**
+Smooth	**.062**	**43**	27	23	**.023**	**.028**	**9**	**9**	.091	**9**	**9**	1.50

Table 2: Test error on QM9 dataset. Gray rows are our methods. **Bold** text indicates the best error of a given backbone model.

geometries minimal in energy with a variety of corresponding chemical properties are included: isotropic polarizability (α), energy of HOMO (ϵ_{HOMO}), energy of LUMO (ϵ_{HOMO}), energy gap ($\Delta\epsilon = \epsilon_{\text{HOMO}} - \epsilon_{\text{HOMO}}$), dipole moment ($\mu$), heat capacity ($Cv$), free energy ($G$), enthalpy ($H$), electronic spatial extent (R^2), internal energy at 298.15 K (U) and 0 K (U_0) , and zero point vibrational energy (ZPVE). The results are compared with NMP [21], SchNet [20], Cormorant [32], L1Net [4], and LieConv [5].

Compared with the original backbone models, using the proposed positional encoding method (PosEnc) can improve the test error for both EGNN and DimeNet++, as listed in Table 2. The smoothness regularization technique (Smooth) can further increase the accuracy for several tasks. We also conducted experiments using a 2-layer MLP with ReLU activation. For DimeNet++, replacing the manually-designed representation by an MLP generally increases test error compared to the original version. For EGNN, MLP increases error for ϵ_{HOMO} and R^2 and reduces error for the other tasks compared to the original version. The improvement however is not as significant as PosEnc.

4.3 Regularized Embedding

The smoothness regularization helps the learned embedding to be more smooth for easier interpretation. The first 10 dimensions of the learned embedding are visualized in Figure 1 as a function of the normalized distance x as

$$\hat{x} \equiv \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}}.$$

Without the smoothness regularization, the learned embedding appear to have no obvious patterns. The difference between the embeddings of two adjunct bins are large. In contrast, the embedding learned with smoothness regularization appear to be more smooth and show a simpler pattern. The embedding are generally nonlinear and non-monotonic. For the embedding learned for this particular task, U_0, we observe that the turning points concentrate at the small distance, and the embedding do not change much at the large distance.

We conduct PCA analysis and visualize the learned embedding on a 2D space in Figure 2. The embedding learned without regularization does not show a clear manifold, while the embedding learned with regularization forms a low-dimensional manifold. This 2-D manifold is not a straight line, consistent with the non-linearity and non-monotonicity observed above.

We further investigate the learned distance embedding in four aspects: non-linearity, non-monotonicity, diversity, and smoothness.

- The non-linearity is measured based on Pearson's correlation, ρ_{Pearson}, between the distance and the embedding. Non-linearity $= 1 - \frac{1}{ls}\sum_{i=1}^{l}\sum_{j=1}^{s}\rho^2_{\text{Pearson}}(h_{ij}, x)$, where l is the number of model layers and h_{ij} is the j-th dimension of the embedding in the i-th layer.

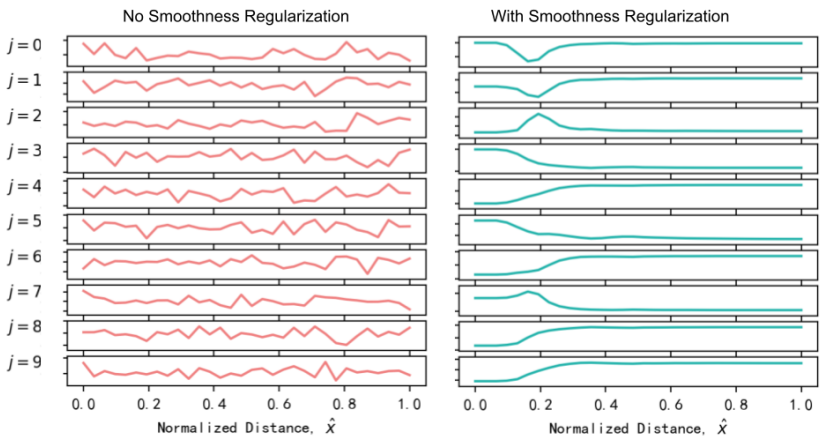

Figure 1: Values of the first 10 dimensions ($j = 1 \sim 10$) of the distance embedding learned on QM9 U_0 task with EGNN backbone.

- The non-monotonicity is quantified based on Spearman's correlation, ρ_{Spearman}, between the distance and the embedding. Non-monotonicity $= 1 - \frac{1}{ls}\sum_{i=1}^{l}\sum_{j=1}^{s}\rho^2_{\text{Spearman}}(h_{ij}, x)$.
- The diversity indicates the average similarity between different dimensions of the learned embedding. A high diversity indicate that the embedding consists of diverse transforms. Diversity $= 1 - \frac{2}{ls(s-1)}\sum_{i=1}^{l}\sum_{j=1}^{s}\sum_{k=i+1}^{s}\rho^2_{\text{Pearson}}(h_{ij}, h_{ik})$.
- The smoothness is defined based on the smoothness loss Smoothness $= 1 - \mathcal{L}_{\text{smooth}}$.

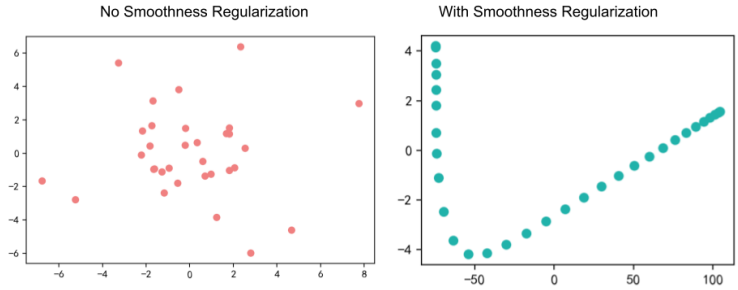

Figure 2: PCA visualization of the learned distance embedding on QM9 U_0 task with EGNN backbone model.

	Learned Distance Embedding												Polynomial
	α	$\Delta\epsilon$	ϵ_{HOMO}	ϵ_{LUMO}	μ	C_v	G	H	R^2	U	U_0	ZPVE	Baseline
Non-linearity	.27	.27	.19	.36	.21	.26	.30	.41	.19	.35	.41	.38	.08
Non-monotonicity	.19	.18	.13	.24	.15	.23	.22	.29	.18	.33	.33	.28	.00
Diversity	.30	.32	.24	.41	.26	.28	.32	.39	.17	.34	.37	.34	.09
Smoothness	.86	.92	.81	.68	.96	.79	.86	.90	.94	.98	.82	.82	.92

Table 3: Characteristics of the distance embedding learned with smoothness regularization and a polynomial embedding.

These metrics are calculated for the embedding learned with smoothness regularization with EGNN model on QM9 dataset. For comparison, we consider a toy embedding made of three polynomial functions as its three dimensions: $h_{\text{polynomial}}(x) = \{x, x^2, x^3\}$. As illustrated in Table 3, the learned embedding generally show higher non-monotocity and diversity compared to the polynomial embedding. The smoothness of the learned embedding are close to the polynomial embedding for some tasks. This indicates that, the proposed positional encoding method learn a embedding that contain

multiple nonlinear or non-monotonic transformations, yet remain smooth. More physical-based interpretation of the learned embeddings are relegated to the appendix.

4.4 Physics-based Interpretation

In this appendix, we present some physic-based interpretation of learned embedding in various tasks from Section 4.2. Different tasks show different characteristics, as illustrated in Table 3. U_0 and H show a high non-linearity, while μ and R^2 show a low non-linearity. In this section, we conduct a few case studies to investigate the connection between embedding characteristics and the physical nature of the corresponding task.

4.4.1 Short vs. Long-Distance Physics

The smoothness regularization encourage the embedding to only change with x when such change increases the accuracy. By analyzing how much the learned distance embedding changes with distance, we know which region of distance the model learned to focus on. We quantify the change by the normalized derivative.

$$\hat{g}(x) \equiv \frac{\text{abs}(g(x))}{\text{std}(g(x))}.$$

As illustrated in Figure 3, for both U_0 and H, $\hat{g}(x)$ is large at small $\hat{x}$, and then quickly drop to a small value close to zero as $\hat{x}$ increases. This indicates that the embedding values do not change much when the distance is greater than certain "cutoff" value. This is consistent with the physical nature of the problem. U_0 and H of the system considered in QM9 are generally dominated by short-range interactions. There is no significant long-range forces such as electrostatic force in these systems.

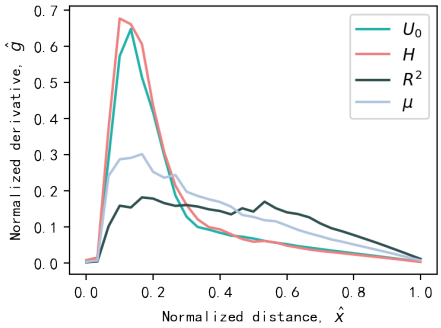

Figure 3: Distribution of derivative over distance for different tasks.

In contrast, for both μ and R^2, $\hat{g}(x)$ show a much flatter profile. This indicates that both long distance and short distance are important. This observation is consistent with the physical natural of the tasks. Dipole moment (μ) is roughly the charges timed by the distance. By definition it is a quantity sensitive to both short and long distance. Therefore the embedding learned for this task has different value at different distance, as shown by the non-zero $\hat{g}(x)$ over full $\hat{x}$ range. Electronic spatial extent (R^2) reports how far out the electronic density extends with significant probability. This roughly measure the molecule size and is obviously not just sensitive to short interatomic distance.

These case studies illustrate that the learned embedding reflect the physical dependence on short and long distance for different tasks.

4.4.2 Monotonic vs. Non-monotonic Physical Dependence

The non-monotonicity of the learned distance embedding illustrated in Table 3 is rooted in the physical nature of the tasks.

For physical quantity governed by non-monotonic dependence on distance, the corresponding learned distance embedding shows a high non-monotonicity. The internal energy is connected to the potential forces acting on the atoms. The force can be non-monotonic w.r.t. distance. Short distance is

often dominated by repulsive forces while attractive forces are more significant at larger distance. Consistent with this non-monotonic physics, The embeddings learned for U_0 and U_0 have the highest non-monotonicity score, 0.33, listed in Table 3.

In contrast, if there is no strong non-monotonic physical dependence on distance, the learned distance embedding has a low non-monotonicity score. For instance, R^2 roughly measure the molecule size, as mentioned above, and is monotonic with distance. The distance embedding learned for R^2 consequently shows the a relatively low non-monotonicity score, 0.18.

4.4.3 Similar Physics have Similar Distance Embedding

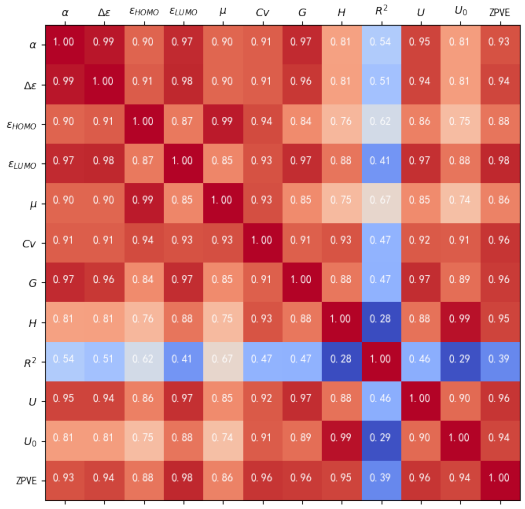

Figure 4: Pairwise similarity of the learned distance embedding.

The tasks can be closely related by their similar physical nature. Some relation is obvious. Both U and U_0 are internal energy, and there is a high correlation between their values in the QM9 dataset. Some physical connection is not straightforward. For example, the energy gap $\Delta\epsilon$ have been used to predict the polarizability α [33, 34], but such relation can not be simply observed in the original data. The Pearson correlation between α and $\Delta\epsilon$ is close to zero in QM9. Can the learned distance embedding capture such physical relation?

We hypothesize that if two tasks are governed by the similar set of physical laws, the embedding learned for these two tasks should share certain similarity. We quantify the similarity between the embedding for task a and task b based on Pearson correlation.

$$\text{Similarity} = \frac{1}{ls^2} \sum_{i=1}^{l} \sum_{j=1}^{s} \sum_{k=1}^{s} \rho^2_{\text{Pearson}}(h_{i,j}^{\text{task } a}, h_{i,k}^{\text{task } b}.)$$

The pairwise similarity is visualized in Figure 4. The energies U, U_0, H, G and ZPVE show a high similarity (≥ 0.88) between each other. Another group of physical quantities, α, μ, and $\Delta\epsilon$ are closely related to the reactivity of the molecules. The pairwise embedding similarity for this group is high (≥ 0.90). In comparison, the similarity cross these two groups are relatively low. This observation indicates that tasks of the similar physical nature have the similar embedding.

5 Conclusion

We propose and demonstrate a regularized positional encoding method for molecular properties prediction tasks. As a plug-in module, the proposed method improves the accuracy of three model architecures, over variety molecular property and force field prediction tasks. The novel smoothness regularization technique encourages the learned embedding to form a simpler low-dimensional manifold for easier physics-based interpretation. Key characteristics of the embedding are connected to the physical nature of the tasks. Tasks of similar physics have the similar learned embedding.

References

[1] James JP Stewart. Optimization of parameters for semiempirical methods v: Modification of nddo approximations and application to 70 elements. *Journal of Molecular modeling*, 13(12):1173–1213, 2007.

[2] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O Anatole Von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical review letters*, 108(5):058301, 2012.

[3] Grégoire Montavon, Matthias Rupp, Vivekanand Gobre, Alvaro Vazquez-Mayagoitia, Katja Hansen, Alexandre Tkatchenko, Klaus-Robert Müller, and O Anatole Von Lilienfeld. Machine learning of molecular electronic properties in chemical compound space. *New Journal of Physics*, 15(9):095003, 2013.

[4] Benjamin Kurt Miller, Mario Geiger, Tess E Smidt, and Frank Noé. Relevance of rotationally equivariant convolutions for predicting molecular properties. *arXiv preprint arXiv:2008.08461*, 2020.

[5] Marc Finzi, Samuel Stanton, Pavel Izmailov, and Andrew Gordon Wilson. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In *International Conference on Machine Learning*, pages 3165–3176. PMLR, 2020.

[6] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. *arXiv preprint arXiv:2102.09844*, 2021.

[7] Han Wang, Linfeng Zhang, Jiequn Han, and E Weinan. Deepmd-kit: A deep learning package for many-body potential energy representation and molecular dynamics. *Computer Physics Communications*, 228:178–184, 2018.

[8] Albert P Bartók, Sandip De, Carl Poelking, Noam Bernstein, James R Kermode, Gábor Csányi, and Michele Ceriotti. Machine learning unifies the modeling of materials and molecules. *Science advances*, 3(12):e1701816, 2017.

[9] Stefan Chmiela, Alexandre Tkatchenko, Huziel E Sauceda, Igor Poltavsky, Kristof T Schütt, and Klaus-Robert Müller. Machine learning of accurate energy-conserving molecular force fields. *Science advances*, 3(5):e1603015, 2017.

[10] Johannes Klicpera, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. *arXiv preprint arXiv:2003.03123*, 2020.

[11] 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.

[12] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[13] 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.

[14] Johannes Klicpera, Shankari Giri, Johannes T Margraf, and Stephan Günnemann. Fast and uncertainty-aware directional message passing for non-equilibrium molecules. *arXiv preprint arXiv:2011.14115*, 2020.

[15] Johannes Klicpera, Florian Becker, and Stephan Günnemann. Gemnet: Universal directional graph neural networks for molecules. *arXiv preprint arXiv:2106.08903*, 2021.

[16] Yi Liu, Limei Wang, Meng Liu, Xuan Zhang, Bora Oztekin, and Shuiwang Ji. Spherical message passing for 3d graph networks. *arXiv preprint arXiv:2102.05013*, 2021.

[17] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[18] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[19] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *International Conference on Machine Learning*, pages 1243–1252. PMLR, 2017.

[20] Kristof T Schütt, Pieter-Jan Kindermans, Huziel E Sauceda, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *arXiv preprint arXiv:1706.08566*, 2017.

[21] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[22] Fabian Fuchs, Daniel Worrall, Volker Fischer, and Max Welling. Se (3)-transformers: 3d roto-translation equivariant attention networks. *Advances in Neural Information Processing Systems*, 33:1970–1981, 2020.

[23] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. *arXiv preprint arXiv:2110.07875*, 2021.

[24] Stefan Chmiela, Alexandre Tkatchenko, Huziel E Sauceda, Igor Poltavsky, Kristof T Schütt, and Klaus-Robert Müller. Machine learning of accurate energy-conserving molecular force fields. *Science advances*, 3(5):e1603015, 2017.

[25] Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E Smidt, and Boris Kozinsky. E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature communications*, 13(1):1–11, 2022.

[26] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform bad for graph representation? *arXiv preprint arXiv:2106.05234*, 2021.

[27] Stefan Chmiela, Huziel E Sauceda, Igor Poltavsky, Klaus-Robert Müller, and Alexandre Tkatchenko. sgdml: Constructing accurate and data efficient molecular force fields using machine learning. *Computer Physics Communications*, 240:38–45, 2019.

[28] Anders S Christensen, Lars A Bratholm, Felix A Faber, and O Anatole von Lilienfeld. Fchl revisited: Faster and more accurate quantum machine learning. *The Journal of chemical physics*, 152(4):044107, 2020.

[29] Benjamin Coors, Alexandru Paul Condurache, and Andreas Geiger. Spherenet: Learning spherical representations for detection and classification in omnidirectional images. In *Proceedings of the European conference on computer vision (ECCV)*, pages 518–533, 2018.

[30] Kristof Schütt, Oliver Unke, and Michael Gastegger. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In *International Conference on Machine Learning*, pages 9377–9388. PMLR, 2021.

[31] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

[32] Brandon Anderson, Truong-Son Hy, and Risi Kondor. Cormorant: Covariant molecular neural networks. *arXiv preprint arXiv:1906.04015*, 2019.

[33] NM Ravindra and VK Srivastava. Variation of electronic polarizability with energy gap in compound semiconductors. *Infrared Physics*, 19(5):605–606, 1979.

[34] RR Reddy and Y Nazeer Ahammed. Relation between energy gap and electronic polarizability of ternary chalcopyrites. *Infrared physics & technology*, 37(4):505–507, 1996.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] see Section 3.3
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The data is public dataset. The instructions are listed in Section 3.3. The code will be open-sourced on GitHub
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] see Section 3.3
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Sizing the Embedding

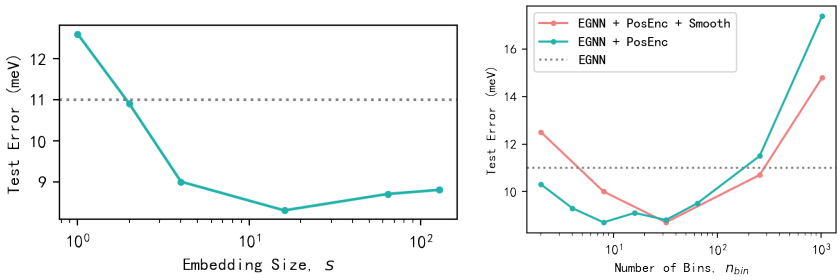

Figure 5: Left: Dependence of test loss on embedding size, tested on QM9 U_0 prediction task with EGNN as the backbone model. Right: Dependence of test loss on number of bins, tested on QM9 U_0 prediction task with EGNN as the backbone model.

The number of bins, n_{bin}, and the embedding size s are the two hyperparameters that determine the size of the embedding. They affect the accuracy of the proposed method.

As illustrated in the left panel of Figure 5, the loss is large with a small embedding size. Small embedding size limits the ability of the embedding to express multi-modal nonlinearity. As the embedding size increases, the test error decreases. This indicates multiple nonlinear transformations are necessary for the model to accurately predict the labels. As the embedding size further increases, the test error no longer improves. For the number of bins, there is a stronger non-monotonic relation between the test error and n_{bin}, as shown in the right panel of Figure 5. At a small n_{bin}, the bins are too sparse and cannot express a complex nonlinear function. This causes large loss. Adding regularization further reduce the flexibility of the embedding and increases the loss. At a large n_{bin}, the space between bins is too small, and some bins may be never trained during training. Therefore the test loss is large with these un-trained parameters. Adding the regularization helps the bins to remain smooth, so even if a bin is not trained directly, its embedding is updated due to the changes propagated from its adjacent bins. This reduces the effects of the untrained parameters issue, and reduces the loss compared to the case without regularization.