A comparison of multi-task approaches on molecular property prediction

Chao Han, Hao Wang, Jianbao Zhu, Qi Liu, and Wenguang Zhu

1School of Physical Sciences, University of Science and Technology of China, Hefei 230026, China
2Anhui Province Key Lab. of Big Data Analysis and Application, School of Computer Science and Technology, University of Science and Technology of China, Hefei 230026, China

With the bloom of deep learning algorithms, various models have been widely utilized in quantum chemistry calculation to design new molecules and explore molecular properties. However, limited studies focus on multi-task molecular property prediction, which offers more efficient ways to simultaneously learn different but related properties by leveraging the inter-task relationship. In this work, we apply the hard parameter sharing framework and advanced loss weighting methods to multi-task molecular property prediction. Based on the performance comparison between single-task baseline and multi-task models on several task sets, we find that the prediction accuracy largely depends on the inter-task relationship, and hard parameter sharing improves the performance when the correlation becomes complex. In addition, we show that proper loss weighting methods help achieve more balanced multi-task optimization and enhance the prediction accuracy. Our additional experiments on varying amount of training data further validate the multi-task advantages and show that multi-task models with proper loss weighting methods can achieve more accurate prediction of molecular properties with much less computational cost.

Key words: Deep learning, Multi-task learning, Molecular property prediction, Uncertainty weighting

I. INTRODUCTION

Molecular modeling has become a crucial part of computational chemistry and materials research over the last few decades. Ab initio calculation has been utilized to investigate various properties of molecules and perform atomic-scale dynamic simulation, which provides the convenience to bypass the unrealizable amount of experiments to explore large chemical spaces, thus significantly accelerate the discovery of novel molecules with desired properties. However, the computational cost of these algorithms increases rapidly when interested systems become larger or when the higher accuracy is required. For example, the widely used density functional theory (DFT) methods have the computational scaling of $O(N_b^3)$ where $N_b$ is the number of basis sets, while more accurate standard CCSD(T)/CBS requires much more expensive cost at the scaling of $O(N_b^7)$. These limitations lead to great difficulties for computational molecular research, especially for investigating properties of larger molecules or exploring vast spaces to find new stable candidates.

With the fast development of machine learning (ML) algorithms and availability of large molecule datasets [1–4], ML has been introduced into quantum chemistry and significantly accelerates computational molecular exploration at high accuracy. Data-driven algorithms take advantage of various molecule datasets to generate new candidates or predict atomistic and electronic properties, which avoid additional quantum chemical calculation and remarkably reduce computational cost. Previous works using traditional ML methods usually need to manually construct molecular fingerprints to represent molecular components and structure [5, 6], which calls for extensive trails and may lead to harmful bias. With the bloom of deep learning (DL) algorithms in recent years, the application of DL to molecular research has attracted more and more attention, for molecular representations are automatically learned by neural networks. Consisting of interacting atoms, molecules can be naturally modeled as graphs, where nodes represent atoms and edges represent atomic interaction. Therefore, graph-based methods including graph neural networks (GNNs) have been implemented to learn molecular representation from their structures and applied to many areas including molecule generation [7, 8], property prediction [9–19], dynamic simulation [20–23], etc.
As an important downstream application of molecular representation learning, molecular property prediction plays a crucial role in new molecule discovery, inverse molecular design and dynamic simulation. Although existing models can extract molecular embeddings from its spatial information and achieve precise results for property prediction, it is common that separate models are built to predict different but related properties. Training models in this single-task way not only costs far more parameters, but also loses the ability to share knowledge among related molecular properties. For example, as representations of atomization energy under different pressure and temperature, targets of internal energy, enthalpy, and free energy are closely correlated, but they are often treated as different tasks and learned by separate models. Therefore, it is necessary to develop multi-task models to encourage inductive transfer and learn molecular properties simultaneously in a more efficient way.

Multi-task learning (MTL) has been used successfully across many applications of deep learning, from computer vision [24] and natural language processing [25] to drug discovery [26]. Various multi-task approaches can be categorized into three groups [27]: multi-task architecture design, multi-task optimization, and task relationship learning. Previous works have utilized multi-task architecture adaption including stacking single-target method to design models for molecular property prediction [28]. Compared with architecture adaption dependent on specific base model, few molecular-related studies focus on using multi-task optimization methods. Besides, though inter-task correlation is crucial to the performance of multi-task models, multi-task attempts on molecular property prediction [10–14] often simultaneously learn all the properties in the dataset, where the influence of task choices has not been fully investigated.

In this work, we construct multi-task models for molecular property prediction and conduct comprehensive experiments to compare single-task learning performance with multi-task approaches. We utilize hard parameter sharing architecture [31] and various loss weighting methods to achieve better prediction of the properties from the benchmark dataset QM9 [2]. The performance comparison on different task sets shows that the inter-task relationship largely influence the prediction accuracy of multi-task models, while hard parameter sharing provides significant improvement when learning tasks with complex relationship. Among different loss weighting methods, uncertainty weighting improves the performance with little additional computational cost. Experiments on varying amount of training data show that multi-task learning outperforms single-task baseline more significantly with less training samples. In addition to the comparison on prediction accuracy, We also illustrate the advantage of multi-task models as a parameter-efficient and time-saving choice.

II. METHODOLOGY

A. Model architecture

We choose state-of-the-art message passing neural networks as base models, which are the most prevalent GNN architectures in various applications for molecular property prediction. Taking three-dimensional spatial information of atoms in molecules as input, message passing algorithms can learn the embeddings of atoms according to their spatial relationship and finally output the prediction of graph-level molecular properties [10–14]. Specifically, we utilize the recent SphereNet model [30] which provides comprehensive embedding from 3D geometric structure of molecules. It systematically considers distance, angle, and torsion information in its spherical message passing scheme and achieves significant performance improvement compared with previous GNNs. FIG. 1 illustrates the general architecture of SphereNet model. Embedding and interaction blocks encode atom types $Z$ and 3D spatial information $(d, \theta, \phi)$ into edge embeddings, which are further accumulated into atomic embeddings in output layers. The atomic contributions are summed up to finally provide a graph-level prediction.

As these models predict scalar properties only based on molecular components and geometric structures, they are capable of learning various targets at a reasonable accuracy. When the targets are potentially correlated, multi-task approaches provide more efficient and accurate models with a better understanding of the inter-task relationship. To formally describe the multi-task problem, we can define a molecular graph as $G = (V, E)$, where $V$ is the set of nodes and $E$ is the set of edges. Nodes in a molecular graph represents atoms, while edges represents the chemical interaction between two atoms. Given the molecule set represented as $N$ graphs $\{G^{(1)}, ..., G^{(N)}\}$ and their mul-
FIG. 1. The architecture of single-task SphereNet. Taking molecular spatial information and atomic number as input, it predicts molecular property A as a single task.

FIG. 2. Multi-task SphereNet with hard parameter sharing framework. With task-specific output modules, it can predict multiple properties simultaneously.

multiple corresponding properties \{Y^{(1)},...,Y^{(N)}\} where \(Y^{(i)} = \{y_{1}^{(i)},...,y_{K}^{(i)}\}\) represents \(K\) molecular properties of molecule \(i\), the goal is to build one model using \(\{G^{(1)},...,G^{(N)}\}\) as input to predict corresponding \(\{Y^{(1)},...,Y^{(N)}\}\) instead of building separate models for each property.

To adapt the original SphereNet into the multi-task model, we choose the widely-used hard parameter sharing architecture [31]. Previous attempts for multi-task molecular property prediction have used the most modest adaption, where only the final layers’ output dimension is extended from 1 to the amount of predicted tasks. This approach often leads to significantly worse performance [10, 29], because sharing too many model parameters causes negative transfer, where conflicting task gradients emerge during training. A more advanced multi-task learning framework is hard parameter sharing, where the single-task model is separated into two parts: sharing module and task-specific module. The shared trunk of hidden layers learns a common representation, while the task-specific output layers maintain the ability to adjust parts of the model’s parameters to fit certain tasks.

We apply the hard parameter sharing method to the SphereNet model, which is illustrated in FIG. 2. While the embedding and interaction blocks remain the same as the single-task model, output modules in each layer are extended according to the amount of multiple tasks. To compare the multi-task models’ performance with the single-task baseline, we use the same model parameter setting as the DIG implement [32] of SphereNet model. For the optimization in all experiments, we set the learning rate beginning at 0.0005 with a step-decay ratio of 0.5 for every 50 steps.

B. Loss weighting methods

Apart from manually adapting original single-task architectures for multi-task scenarios, recently researchers also find it critical to design appropriate optimization strategies for MTL to avoid negative transfer when gradients conflict occurs. MTL optimization also can be considered as a broader version of soft parameter sharing methods [31], which regularize separate but related model parameters in a fuse-style pattern instead of split-style hard parameter sharing. One of the most common MTL optimization methods is to weight individual loss functions for each task. To combine different losses together for jointly training, the simplest way is to sum them together. However, this often leads to imbalanced training for the data distribution and learning difficulties can vary a lot among different targets. As a result, the training process will be dominated by certain tasks. To avoid this, task-specific weights are used to adjust the sum of loss functions. Many works treat these weights as a set of hyperparameter and use experimentally decided weighted sum of losses [33], while several dynamically weighting methods are proposed to automatically combine the losses [34–36]. Though loss weighting methods have been widely utilized in other multi-task learning applications including natural language processing [37] and computer vision [38], few of
them have been applied to multi-task molecular property prediction. In this work, we apply several loss weighting methods to balance multi-task optimization and comprehensively compare their performances under different situations.

**Uniform weighting loss**: This is the most straightforward way to produce a combined loss, where all the losses are summed together. Using $L_k$ to represent the loss of task $k$, uniform weighing loss for $K$ tasks is $L = \sum_{k=1}^{K} L_k$. We apply this as a baseline weighting method.

**Uncertainty loss** [34]: This principled approach to weight losses considers the homoscedastic uncertainty of each task. The multi-task network is treated as a probabilistic model, and a set of learnable parameter $\{\sigma_k\}$ is used to signify the task-specific uncertainty. The combined loss function can be written as:

$$L = \sum_{k=1}^{K} \left( \frac{1}{2 \sigma_k^2} L_k + \log \sigma_k \right)$$

Learnable parameters $\{\sigma_k\}$ are used to adjust the weight for each task, while the log terms balance the former term to avoid infinite growth of $\{\sigma_k\}$ when training. To get a more stable training process, the network is trained to predict the log variance, $s_k := \log \sigma_k^2$.

**Revised uncertainty loss** [35]: A revised version of uncertainty weighted loss is further proposed to enforce positive regularization values. The original term $\log(\sigma_k)$ is adapted to $\log(1 + \sigma_k^2)$ and the final loss is:

$$L = \sum_{k=1}^{K} \left( \frac{1}{2 \sigma_k^2} L_k + \log(1 + \sigma_k^2) \right)$$

Experiments show that $\{\sigma_k\}$ is robust to the initialization and can quickly converge to a similar optima [34]. Therefore, we choose to initialize the $\{\sigma_k\}$ in both methods with a value of $1/K$ as suggested in the original paper [35].

**Dynamic weight averaging (DWA)** [36]: Other than weighting task-specific loss according to uncertainty, many works focus on weighting by learning speed. Inspired by GradNorm [39], dynamic weighting averaging method learns to dynamically average task weighting by considering the rate of change of loss for each task. The averaged loss at iteration $t$ can be written as:

$$L(t) = \sum_{k=1}^{K} \lambda_k(t) L_k(t)$$

where $\lambda_k(t)$ is decided by the relative descending rate of loss:

$$\lambda_k(t) = \frac{K e^{w_k(t-1)/T}}{\sum_j e^{w_j(t-1)/T}}, w_k(t-1) = \frac{L_k(t-1)}{L_k(t-2)}$$

The hyperparameter temperature $T$ controls the softness of task weighting. We adopt the original setting from the paper, including setting $T = 2$ and initializing $w_k(t) = 1$ for $t = 1, 2$.

**C. Datasets**

We apply our multi-task approaches to the publicly available dataset QM9 [2], which is widely used as an important benchmark dataset for molecular property prediction. QM9 consists of over 130000 organic molecules with up to 9 heavy (non-Hydrogen) atoms including C, O, N, and F. DFT calculation is used to find a reasonable low energy structure and atomic positions after geometry relaxation are provided for each molecule. In addition, 13 fundamental properties are calculated at the theoretical level of DFT/B3LYP/6-31G(2df, p). These properties can be roughly grouped into four categories [10]:

**Atomization energies**: Atomization energy at 0 K, $U_0$ (eV), atomization energy at room temperature, $U$ (eV), enthalpy of atomization at room temperature, $H$ (eV), free energy of atomization, $G$ (eV), and heat capacity at room temperature, $C_v$ (cal/mol K).

**Fundamental vibrations**: Highest fundamental vibrational frequency $\omega_1$ ($\text{cm}^{-1}$) *, and zero point vibrational energy (ZPVE) (eV).

**Electronic states**: The highest occupied molecular orbital (HOMO) $\varepsilon_{\text{HOMO}}$ (eV), the lowest unoccupied molecular orbital (LUMO) $\varepsilon_{\text{LUMO}}$ (eV), and electron energy gap $\Delta \varepsilon$ (eV). The energy gap is defined as $\Delta \varepsilon = \varepsilon_{\text{LUMO}} - \varepsilon_{\text{HOMO}}$, and the prediction of the energy gap is also calculated as the difference between predictions of $\varepsilon_{\text{LUMO}}$ and $\varepsilon_{\text{HOMO}}$.

**Electronic spatial distribution**: Electronic spatial extent $\langle R^2 \rangle$ ($\text{Å}^2$), norm of the dipole moment $\mu$ (D), and the norm of the static polarizability $\alpha$ (D).

*As there are different versions of QM9 dataset, we use the dataset from DIG [32] package where the data of $\omega_1$ is not provided. Therefore, we keep the same with many other studies on QM9 and have not taken $\omega_1$ into consideration.
To predict graph-level molecular properties, most GNNs often express the target $\hat{y}$ as a sum of atom-wise contributions, i.e., $\hat{y} = \sum_{i=1}^{n} p(v_i)$, where $v_i$ is the embedding of atom $i$ learned from message passing blocks and $p$ is usually a fully-connected network to reduce the dimension of atomic embedding. This output manner is based on the assumption that desired molecular property can be interpreted as the sum of atomic contributions. While some targets like $U_0$ naturally obey this assumption, others may need task-specific modification on the output block according to prior knowledge. We utilize two kinds of target modifications introduced by SchNetPack [40] on the prediction of $\mu$ and $\langle R^2 \rangle$ respectively. For dipole moment $\mu$, we modify the formula as $\mu = |\sum_{i=1}^{n} q(v_i)(r_i - r_0)|$, where $(r_i - r_0)$ denotes the position vector from the molecular center of mass to the atom $i$, and $q(v_i)$ can be interpreted as latent atomic charges. Following this, the prediction of electronic spatial extent is modified similarly as $\langle R^2 \rangle = \sum_{i=1}^{n} p(v_i)(r_i - r_0)^2$. We apply these two modifications to both single-task models and multi-task ones.

D. Evaluation metrics

We use Mean Absolute Error (MAE) to evaluate the performance on multiple targets respectively, which is a standard metric used by many other studies on QM9 dataset. Besides, we utilize MAE divided by standard deviation of each target to balance all targets and use the average $\text{std}_{\text{MAE}}$ to reflect models’ general performance:

$$\text{std}_{\text{MAE}} = \frac{1}{K} \sum_{k=1}^{K} \left( \frac{1}{N} \sum_{i=1}^{N} \frac{|\hat{y}_k(i) - y_k(i)|}{\sigma_k} \right)$$

where $\hat{y}_k(i)$ and $y_k(i)$ are model’s prediction and target value for the property $k$ of the molecule sample $i$, respectively. $\sigma_k$ is the standard deviation of property $k$’s value for $N$ samples in the dataset.

III. RESULTS AND DISCUSSION

A. Inter-task correlations

FIG. 3 shows the inter-task relationship of properties from QM9 dataset represented by Pearson correlation coefficient. Four properties in atomization energy group including $U_0$, $U$, $G$, and $H$ have the highest coefficient above 99%, indicating their strong linear correlation. A larger task group including $U_0$, $U$, $G$, $H$, ZPVE, $C_v$, and $\alpha$ also obtain relatively high absolute Pearson correlation coefficient above 75%. On the contrary, other properties including electronic states ($\varepsilon_{\text{HOMO}}$, $\varepsilon_{\text{LUMO}}$, $\Delta \varepsilon$), $\mu$ and $\langle R^2 \rangle$ are barely linear-related. An exception in this category is the strong linear correlation between $\varepsilon_{\text{LUMO}}$ and $\Delta \varepsilon$, which may come from the definition of $\Delta \varepsilon$ as the difference between $\varepsilon_{\text{HOMO}}$ and $\varepsilon_{\text{LUMO}}$. In spite of the weak linear relationship indicated by Pearson correlation coefficient, these properties can obtain a more complex relationship. For example, $\varepsilon_{\text{HOMO}}$ and $\varepsilon_{\text{LUMO}}$ both represent electronic state information and can be closely but not linearly related.

![FIG. 3. Inter-task Pearson correlation coefficients of properties in QM9 dataset. A higher absolute value indicates a stronger linear correlation, while the sign of coefficients indicates whether properties are correlated positively or negatively.](image)

B. Learning properties of atomization energies

To find out when multi-task approaches perform better than single-task baseline and which multi-task method obtain a better performance when learning related molecular properties, we design three different task groups and compare the performance of different models in the following subsections. For experiments on each task group, we train three types of models: single-task learning (STL) baseline, adapted single-
TABLE I. Performance comparison on learning properties of atomization energies. Task-specific MAE and std_MAE on test dataset are both shown in the table. The best results are shown in bold.

<table>
<thead>
<tr>
<th>Property</th>
<th>$U_0$</th>
<th>$U$</th>
<th>$G$</th>
<th>$H$</th>
<th>std_MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>STL</td>
<td>31.8</td>
<td>32.9</td>
<td>33.7</td>
<td>33.4</td>
<td>0.324</td>
</tr>
<tr>
<td>Adapted STL</td>
<td>31.0</td>
<td>31.3</td>
<td>31.0</td>
<td>31.3</td>
<td>0.306</td>
</tr>
<tr>
<td>Uniform</td>
<td>34.3</td>
<td>34.6</td>
<td>34.1</td>
<td>34.6</td>
<td>0.338</td>
</tr>
<tr>
<td>Uncertainty</td>
<td>34.1</td>
<td>34.3</td>
<td>34.0</td>
<td>34.3</td>
<td>0.336</td>
</tr>
<tr>
<td>Revised uncertainty</td>
<td>34.1</td>
<td>34.5</td>
<td>34.0</td>
<td>34.4</td>
<td>0.337</td>
</tr>
<tr>
<td>DWA</td>
<td>34.1</td>
<td>34.3</td>
<td>34.1</td>
<td>34.4</td>
<td>0.337</td>
</tr>
</tbody>
</table>

As mentioned in section II A, adapted single-task model keeps its main trunk the same as original model but extend the output dimension of the last layer. We apply uniform loss weighting method on adapted single-task model and hard parameter sharing model to offer the baseline performance. Other different dynamic loss weighting methods are applied to hard parameter sharing model and compared under each task group. Experiments in previous works [38] suggest that the advantage of multi-task learning will become more significant when the model is fed with smaller training dataset, so we randomly split the QM9 dataset with 10% training data, 10% validation data and 80% test data for most experiments. We also investigate the relationship between the split ratio of the dataset and models’ performance in section III D.

First, we choose $U_0$, $U$, $G$, and $H$ as the atomization energy task group where inter-task linear correlation is very strong (>99%). Both task-specific MAEs and overall evaluation metric std_MAE on the test dataset are shown in TABLE I. Adapted SphereNet model outperforms single-task models on all 4 targets with the relative reduction on std_MAE at 5.6%. As these tasks are strongly linear-correlated, simultaneously learning them leads to an implicit data augmentation, where the data-dependent noise has lower influence and multi-task model can generalize better. Therefore, with the highest sharing ratio of model parameters, adapted STL model benefits from multi-task learning thus obtains the best task-specific and overall performance.

On the contrary, hard parameter sharing models obtain a higher deviation than both adapted STL model and single-task baseline. This is caused by introducing a large amount of additional parameters in output layers. Though separate output modules will enhance the task-specific adjustability of multi-task model, the targets’ linear correlation can be well modelled by a linear recombination in the last output layer of the adapted single-task model. Reversely, the large vector dimension of node embeddings leads to redundant parameters in additional output layers of hard parameter sharing models, which will increase the risk of over-fitting.

Besides, the performance of different loss weighting methods is similar with uniform weighting baseline on hard parameter sharing models. As loss weighting methods aim at providing balanced optimizations for each task, the enhancement of dynamic weighting methods will become significant if the data distribution or learning difficulty of each task varies a lot. However, the targets in atomization energy group have the similar data distribution, thus advanced optimization methods show no improvement compared with uniform weighting method.

C. Learning electronic properties

In addition to performance comparison on learning linearly-correlated tasks, we conduct experiments to simultaneously learn tasks with more complex relationship. To achieve this, we choose $\varepsilon_{\text{HOMO}}$, $\varepsilon_{\text{LUMO}}$, $\Delta \varepsilon$, and $\langle R^2 \rangle$ as an electronic property group. As most applications of GNNs on molecular property prediction are based on the classical molecular modeling where electronic effect is implicitly represented in the atomic interaction, some electronic properties cannot perfectly fit into this paradigm which lead to a higher relative prediction error compared with other properties. Intertask relationship analysis in section III A also shows that electronic properties do not obtain a strong linear correlation. However, these properties might have a more complex connection due to their close relationship with molecular electronic structure.

We show the performance comparison when learning electronic properties in Table II. First, adapted single-task model has the worst overall performance (std_MAE at 9.10%) and also the highest MAE on $\varepsilon_{\text{HOMO}}$, $\varepsilon_{\text{LUMO}}$, and $\Delta \varepsilon$ prediction among all models. This corresponds to reported results in previous works [10, 29], which may be caused by gradient conflict when sharing too many model parameters. All the hard parameter shar-
TABLE II. Performance comparison on learning 4 electronic properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>$\langle R^2 \rangle$</th>
<th>$\varepsilon_{\text{HOMO}}$</th>
<th>$\varepsilon_{\text{LUMO}}$</th>
<th>$\Delta \varepsilon$</th>
<th>std-MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit</td>
<td>$a_0^2$</td>
<td>meV</td>
<td>meV</td>
<td>meV</td>
<td>%</td>
</tr>
<tr>
<td>STL</td>
<td>0.486</td>
<td>95.2</td>
<td>67.6</td>
<td>119</td>
<td>7.66</td>
</tr>
<tr>
<td>Adapted STL</td>
<td>0.511</td>
<td>104</td>
<td>98.2</td>
<td>142</td>
<td>9.10</td>
</tr>
<tr>
<td>Uniform</td>
<td>0.470</td>
<td>97.9</td>
<td>83.7</td>
<td>130</td>
<td>8.29</td>
</tr>
<tr>
<td>Uncertainty</td>
<td>0.686</td>
<td>89.8</td>
<td>71.6</td>
<td>117</td>
<td>7.50</td>
</tr>
<tr>
<td>Revised uncertainty</td>
<td>0.581</td>
<td>90.2</td>
<td>73.9</td>
<td>118</td>
<td>7.56</td>
</tr>
<tr>
<td>DWA</td>
<td>0.468</td>
<td>97.7</td>
<td>84.3</td>
<td>130</td>
<td>8.30</td>
</tr>
</tbody>
</table>

Adapted single-task model has the worst performance among all attempts, and hard parameter sharing framework is proved effective to enhance the performance of adapted single-task model, reducing the std-MAE from 3.76% to 3.38%. The uncertainty weighting method still performs the best, with 14.3% relative enhancement over adapted STL model and 4.7% relative enhancement over hard parameter sharing model using uniform weighting loss. The DWA method still has no effect and obtains the similar result as the uniform weighting loss baseline. As this result may be caused by a large value of hyperparameter $T$ where the task weighting becomes too soft, we have also tried smaller values of $T$ but get even higher prediction error. The reason leading to this still needs further investigation.

Focusing on the results of electronic properties, we can also compare the performance of the same model utilizing uncertainty weighting loss but learning different targets. As shown in TABLE II and TABLE III, the model learning all 12 tasks obtains the same or better prediction accuracy on 4 electronic properties than the model learning only 4 tasks. Among these targets, the task-specific performance of $\varepsilon_{\text{HOMO}}, \varepsilon_{\text{LUMO}},$ and $\Delta \varepsilon$ remain the same level, but the $\langle R^2 \rangle$ prediction MAE is reduced from 0.686 to 0.524, which indicates learning $\langle R^2 \rangle$ benefits from its relationship with other properties in the larger group.

E. Influence of training data amount

To investigate the influence of training data amount on multi-task learning, we conduct experiments on 3 different dataset split strategies. While keeping 10% of the whole QM9 dataset as the validation dataset, we use 10%, 30%, and 50% samples as the training dataset respectively and the remaining as the test dataset. We show our comparison of std-MAE in FIG. 4. All the models perform better with more training data, while less enhancement is achieved when training data amount becomes larger, which indicates that the models’ performance may eventually converge with enough data. The performance ranking of multi-task models keeps the same for different data split strategies. Particularly, DWA method shows the similar performance with uniform weighting loss consistently, while both original and revised uncertainty weighting loss methods enhance the performance of hard parameter sharing model. Comparing uncertainty weighting loss model with STL base-
TABLE III. Performance comparison on learning all 12 properties from QM9.

<table>
<thead>
<tr>
<th>Property</th>
<th>$U_0$</th>
<th>$U$</th>
<th>$G$</th>
<th>$H$</th>
<th>$C_v$</th>
<th>ZPVE</th>
<th>$\alpha$</th>
<th>$\mu$</th>
<th>$\langle R^2 \rangle$</th>
<th>$\varepsilon_{\text{HOMO}}$</th>
<th>$\varepsilon_{\text{LUMO}}$</th>
<th>$\Delta \varepsilon$</th>
<th>std.MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit</td>
<td>meV</td>
<td>meV</td>
<td>meV</td>
<td>meV</td>
<td>cal/mol K</td>
<td>meV</td>
<td>meV</td>
<td>D</td>
<td>$\alpha^0$</td>
<td>meV</td>
<td>meV</td>
<td>meV</td>
<td>%</td>
</tr>
<tr>
<td>STL</td>
<td>31.8</td>
<td>32.9</td>
<td>33.7</td>
<td>33.4</td>
<td>0.0662</td>
<td>2.94</td>
<td>0.160</td>
<td>0.0600</td>
<td>0.486</td>
<td>95.2</td>
<td>67.6</td>
<td>119</td>
<td>3.32</td>
</tr>
<tr>
<td>Adapted STL</td>
<td>39.6</td>
<td>39.7</td>
<td>38.9</td>
<td>40.0</td>
<td>0.0673</td>
<td>11.1</td>
<td>0.161</td>
<td>0.0807</td>
<td>0.484</td>
<td>96.4</td>
<td>88.2</td>
<td>130</td>
<td>3.76</td>
</tr>
<tr>
<td>Uniform</td>
<td>35.0</td>
<td>34.4</td>
<td>34.2</td>
<td>34.3</td>
<td>0.0648</td>
<td>3.23</td>
<td>0.157</td>
<td>0.0743</td>
<td>0.424</td>
<td>87.9</td>
<td>79.0</td>
<td>120</td>
<td>3.38</td>
</tr>
<tr>
<td>Uncertainty</td>
<td>32.2</td>
<td><strong>31.4</strong></td>
<td><strong>31.2</strong></td>
<td><strong>31.8</strong></td>
<td><strong>0.0611</strong></td>
<td><strong>2.58</strong></td>
<td>0.162</td>
<td>0.0694</td>
<td>0.524</td>
<td><strong>85.0</strong></td>
<td>73.2</td>
<td><strong>114</strong></td>
<td><strong>3.22</strong></td>
</tr>
<tr>
<td>Revised uncertainty</td>
<td>33.0</td>
<td>32.6</td>
<td>32.5</td>
<td>34.6</td>
<td>0.0618</td>
<td>2.97</td>
<td><strong>0.155</strong></td>
<td>0.0700</td>
<td>0.482</td>
<td>85.8</td>
<td>74.0</td>
<td><strong>114</strong></td>
<td>3.25</td>
</tr>
<tr>
<td>DWA</td>
<td>36.0</td>
<td>34.6</td>
<td>35.6</td>
<td>34.8</td>
<td>0.0638</td>
<td>2.92</td>
<td>0.157</td>
<td>0.0757</td>
<td>0.429</td>
<td>88.5</td>
<td>78.2</td>
<td>119</td>
<td>3.38</td>
</tr>
</tbody>
</table>

FIG. 4. Model performance with different amounts of training data. We use 10%, 30%, and 50% samples in QM9 as training dataset respectively and show models’ std.MAE on test dataset to evaluate their overall performance.

line, we find that the advantage of multi-task learning becomes more significant with less training data, which coincides to the experiments in previous works.

F. Multi-task efficiency study

In addition to the performance comparison of different models on various datasets, we evaluate the multi-task model efficiency over the single-task baseline. We choose the models trained on all 12 targets in QM9 and compare their model parameter amount and training time per epoch. For single-task models, parameter amount and training time for each target are summed up to provide a baseline.

We show the results in TABLE IV. Compared with the single-task baseline, multi-task models need fewer parameters and much less training time. In particular, the cheapest multi-task model is adapted STL, but its performance is limited by the maximum sharing ratio of parameters and can only be applied to the extreme situation where learned tasks are closely linear-correlated. Multi-task models with hard parameter sharing architecture need more parameters than adapted STL model because of additional output modules, but they still have lower computational cost than STL approach. Efficiency comparison between models using uniform weighting loss and uncertainty weighting loss shows that the latter enhances model’s overall performance with no significant additional parameters or training time. Compared with multi-task architecture adaption, proper loss weighting methods offer a more general enhancement with little negative impact on computational cost during training.

<table>
<thead>
<tr>
<th>Models</th>
<th>#Param.</th>
<th>Time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>STL</td>
<td>$2.28 \times 10^7$</td>
<td>660</td>
</tr>
<tr>
<td>Adapted STL</td>
<td>$1.91 \times 10^6$</td>
<td>64</td>
</tr>
<tr>
<td>Uniform</td>
<td>$1.34 \times 10^7$</td>
<td>123</td>
</tr>
<tr>
<td>Uncertainty</td>
<td>$1.34 \times 10^7$</td>
<td>122</td>
</tr>
</tbody>
</table>

TABLE IV. Efficiency comparison in terms of number of parameters and training time cost per epoch.

IV. CONCLUSION

In summary, we have applied both model architecture adaption and multi-task optimization methods to the base model for multi-task molecular property prediction. We investigate the influence of task choices on multi-task model performance and find that when inter-task correlation becomes complex or loosely related, hard parameter sharing framework helps main-
tain the model’s ability to adjust according to specific tasks and performs better than adapted STL model. Among various loss weighting methods, we find that uncertainty weighting loss can better improve the overall performance when learning targets from QM9. Utilizing hard parameter sharing framework with uncertainty weighting loss achieves better performance than single-task baseline while requires much less computational cost when learning properties with complex inter-task relationship.

V. ACKNOWLEDGMENTS

This work is supported by the National Key Research and Development Program of China (Grant No. 2017YFA0204904), the Fundamental Research Funds for the Central Universities, China (Grant No. WK3510000013), and the National Natural Science Foundation of China (Grant No. 61922073).


