

395 A Implementation Details

396 A.1 Dataset Description

397 **QM9 Dataset** The QM9 dataset [41] is a significant resource in the field of quantum chemistry,
398 offering a single equilibrium conformation and 12 labels that include geometric, energetic, electronic,
399 and thermodynamic properties. For the purpose of performance evaluation, we select the following
400 properties: HOMO, LUMO, gap, alpha, C_v , mu, R^2 , and ZPVE.

401 **COMPAS-1D Dataset** The COMPAS-1D dataset is a part of the COMPAS Project, which is an
402 acronym for the computational Database of Polycyclic Aromatic Systems. The dataset is specifically
403 focused on data-condensed poly-benzenoid hydrocarbons, which are a type of polycyclic aromatic
404 hydrocarbons (PAHs) with a unique structure where the benzene rings are connected edge-to-edge.
405 The COMPAS-1D [43] contains 8,678 molecules and offers essential computational properties crucial
406 for comprehending the behavior of polycyclic aromatic hydrocarbons and other organic molecules
407 across various chemical and physical processes.

408 A.2 Hyperparameter Settings

409 In line with previous methods, we employ grid search to find the optimal hyper-parameters for tasks
410 within the QM9 and COMPAS-1D datasets. The specific hyper-parameters are detailed in Table 7. In
411 all experiments, we select the checkpoint with the lowest validation loss and report the corresponding
412 test set results based on that checkpoint. For the COMPAS-1D dataset, experiments were conducted
413 using a single A100 GPU, whereas for the QM9 dataset, the experiments were run on eight A100
414 GPUs.

Table 7: Hyper-paramters for fine-tuning on QM9 and COMPAS-1D Dataset

Hyperparameter	Value or description
Learning rate	[4e-5, 6e-5, 1e-4, 2e-4, 3e-4, 4e-4]
Batch size	[32, 64, 128]
Epochs	[40, 60, 80, 100, 200, 300]
Pooler dropout	[0.0, 0.1]
Warmup ratio	[0.0, 0.06, 0.1]

415 B Infrastructures

416 We utilize an efficient distributed PyTorch framework called Uni-Core [46], specifically designed for
417 swiftly developing high-performance PyTorch models [47], particularly those based on Transformer
418 architectures[48]. Given the variability in molecule lengths, padding inputs to match the maximum
419 molecular length is necessary during training. Consequently, the batch size for model training is
420 influenced by the longest molecule in each batch. However, since molecule lengths follow a long-
421 tail distribution (with the majority falling within a specific range), we employ dynamic batching
422 techniques to enhance GPU utilization. By adjusting batch sizes according to the maximum lengths
423 of different batches, we can significantly boost GPU utilization with minimal effort.

424 The time consumption of reading data from distributed storage is often overlooked. We employ a
425 singular, dedicated process on each computational node to asynchronously replicate the training
426 dataset of each epoch onto the host machine. This strategy effectively mitigates time overheads,
427 thereby obscuring the duration spent on data reading from distributed storage. To resume the
428 corruption due to the infra and other factors effectively, we save model weight and optimizer state for
429 every 1k step asynchronously. This means we will lose 1k step training resources in the worst case of
430 hardware instability or loss spike during training. Meanwhile, any checkpoints exceeding the most
431 recent ten files will be deleted to avoid consuming too much storage space.

432 **C Limitations**

433 The major limitation of our study pertains to the absence of an exploration of the optimal batch size
434 and learning rate. Our investigation primarily focuses on analyzing and delineating the power-law
435 relationships among validation loss, model size, dataset size, and computational resources. The
436 predictive accuracy of performance aligns well with the scaling curve, indicating that the current
437 optimal learning rate and batch size approximate the near-optimal values. However, existing research
438 suggests a progressive increase in the optimal batch size with augmented computing resources, while
439 the optimal learning rate tends to decrease gradually. It is necessary to note that as we further increase
440 the model's parameters, the final optimal values for learning rate and batch size may fall outside the
441 currently identified range. Consequently, investigating the scaling law for optimal batch size and
442 learning rate is also paramount.