Appendix

Operating platform and hyperparameters of the models

Under the MNIST dataset, the hardware platform on which the models run is the Ubuntu operating system, including 8 NVIDIA A100 GPUs; the model code is based on the Python 3.8.6 language and developed by the Pytorch 2.1.0 framework; the hyperparameters of the models are set to: *learning rate* is *[0.001-0.005]*, *training batch* is *50*, *epoch* is *20*, *input size* is *784*, *hidden size* is *784*, *output size* is *10*, *dropout rate* is *0.1*, and the number of copies of the weight matrix k in Eq. (4) is *1*.

Under the CIFAR-10 dataset, the hardware platform on which the models run is the Ubuntu operating system, including 8 NVIDIA A100 GPUs; the model code is based on the Python 3.8.6 language and developed by the Pytorch 2.1.0 framework; the hyperparameters of the models are set to: *learning rate* is *[0.005-0.0005]*, *training batch* is *50*, *epoch* is *20*, *input size* is *3072*, *hidden size* is *3072*, *output size* is *10*, *dropout rate* is *0.1*, and the number of copies of the weight matrix k in Eq. (4) is *1*.

Under the IMDb dataset, the hardware platform on which the models run is the Ubuntu operating system, including 8 NVIDIA A100 GPUs; the model code is based on the Python 3.8.6 language and developed by the Pytorch 2.1.0 framework; the hyperparameters of the models are set to: *learning rate* is *[0.00001-0.00002]*, *training batch* is *128*, *epoch* is *5*, *input size* is *768*, *hidden size* is *768*, *output size* is *2*, *dropout rate* is *0.1*, and the number of copies of the weight matrix k in Eq. (4) is *1*.

Under the NewsGroups dataset, the hardware platform on which the models run is the Ubuntu operating system, including 8 NVIDIA A100 GPUs; the model code is based on the Python 3.8.6 language and developed by the Pytorch 2.1.0 framework; the hyperparameters of the models are set to: *learning rate* is *[0.001-0.005]*, *epoch* is *500*, *input size* is *5000*, *hidden size* is *5000*, *output size* is *20*, *dropout rate* is *0.1*, and the number of copies of the weight matrix k in Eq. (4) is *1*.

Analysis of model training efficiency under MNIST dataset

Under the MNIST dataset, the training losses of the models are shown in Fig. 1. It can be seen from the loss diagram that the classic MLP converges faster, while QiMLP is relatively slower. This phenomenon is in line with expectations, because it is more difficult for QiMLP to learn strong correlations among features, so the convergence speed of the models will be relatively slower. In addition, it can be seen from this diagram that the convergence effect of QiMLP of the two-body quantum system is better than that of the classical MLP.

Under the MNIST dataset, the training accuracy of the models is shown in Fig. 2. It can be seen from the accuracy diagram that the classic MLP achieves the best results faster, while QiMLP is relatively slower. This phenomenon is expected, because it is more difficult for QiMLP to learn strong correlations among features, so it is more difficult to achieve

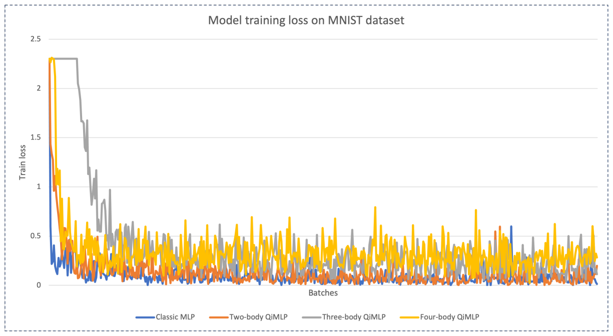

Figure 1: Training losses of the models under the MNIST dataset.

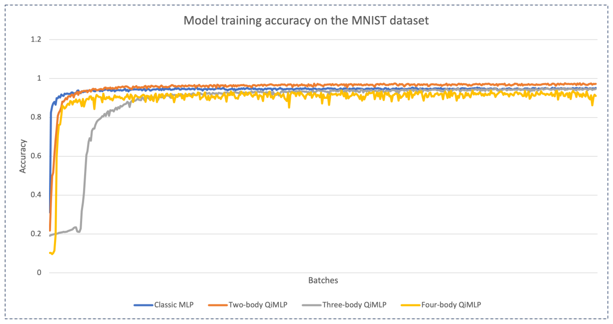

Figure 2: The training accuracy of the models under the MNIST dataset.

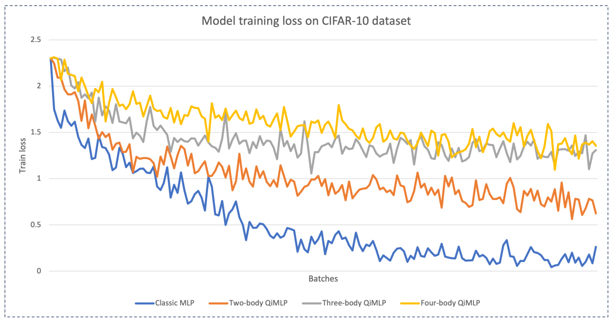

Figure 3: Training losses of the models under the CIFAR-10 dataset.

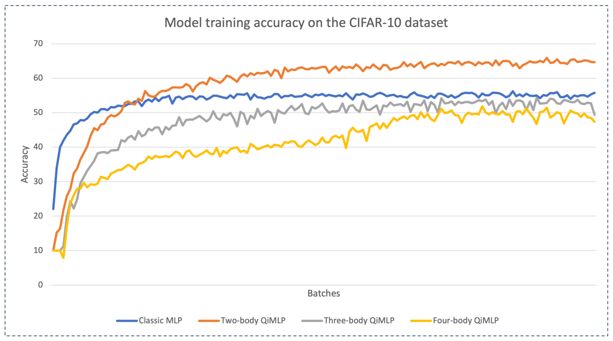

Figure 4: The training accuracy of the models under the CIFAR-10 dataset.

	Classic MLP	Two-body QiMLP	Three-body QiMLP	Four-body QiMLP
Used time	296 s	312 s	335 s	350 s
GPU memory	915 MiB	558 MiB	387 MiB	157 MiB

Table 1: Model training time and GPU memory usage under MNIST dataset

	Classic MLP	Two-body QiMLP	Three-body QiMLP	Four-body QiMLP
Used time	319 s	350 s	362 s	377 s
GPU memory	1524 MiB	929 MiB	728 MiB	512 MiB

Table 2: Model training time and GPU memory usage under CIFAR-10 dataset

the best results. In addition, it can be seen from this diagram that the convergence speed of QiMLP of the two-body quantum system is better than that of the classical MLP, and the convergence speed of QiMLP of the three-body quantum system is similar to that of the classical MLP.

Under the MNIST dataset, the usage time and GPU memory of the models trained for the same number of epochs is shown in Tab. 1. As can be seen from the table, the classical MLP takes the least time compared to QiMLP on the same dataset and the same epoch. The reason for this phenomenon is that it is more difficult for QiMLP to learn strong correlations among features. Fortunately, QiMLP takes up less GPU memory.

Analysis of model training efficiency under CIFAR-10 dataset

Under the CIFAR-10 dataset, the training losses of the models are shown in Fig. 3. It can be seen from this diagram that the classic MLP has the best convergence effect, while QiMLP has a relatively poor convergence effect. This phenomenon shows that QiMLP needs to learn strong correlations and has fewer parameters, so the convergence effect will be worse. However, combined with the accuracy diagram, it can be seen that the QiMLP can achieve better performance, indicating that its learning ability is stronger.

Under the CIFAR-10 dataset, the training accuracy of the models is shown in Fig. 4. It can be seen from this diagram that the classical MLP converges faster, while QiMLP converges relatively slowly. However, the final performance obtained by QiMLP based on the two-body quantum system is better than the classical MLP; the final performance obtained by QiMLP based on the three-body quantum system is similar to the classic MLP. This phenomenon shows that QiMLP is effective and has better learning capabilities.

Under the CIFAR-10 dataset, the usage time and GPU memory of the models trained for the same number of epochs is shown in Tab. 2. It can be seen from this table that the training time of QiMLP will be longer than that of classical MLP under the same data and the same training epoch. The reason for this phenomenon is that it takes more time for QiMLP to learn more complex relationships among features. Notably, QiMLP uses less GPU memory.