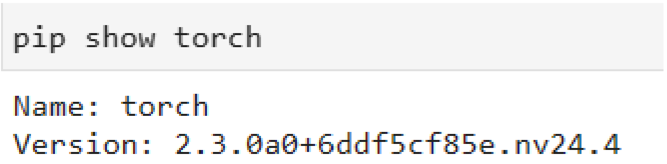
Assignment 3 Report

1. PyTorch Version



```
pip show torch
```

```
Name: torch
Version: 2.3.0a0+6ddf5cf85e.nv24.4
```

2. Implement the three methods

I created three classes to implement the methods mentioned in the paper: `GMF`, `MLP`, and `NeuMF`. Then, I defined the functions `train` and `evaluate` to train and test all three models.
For detailed code, please refer to `run_models.ipynb`.

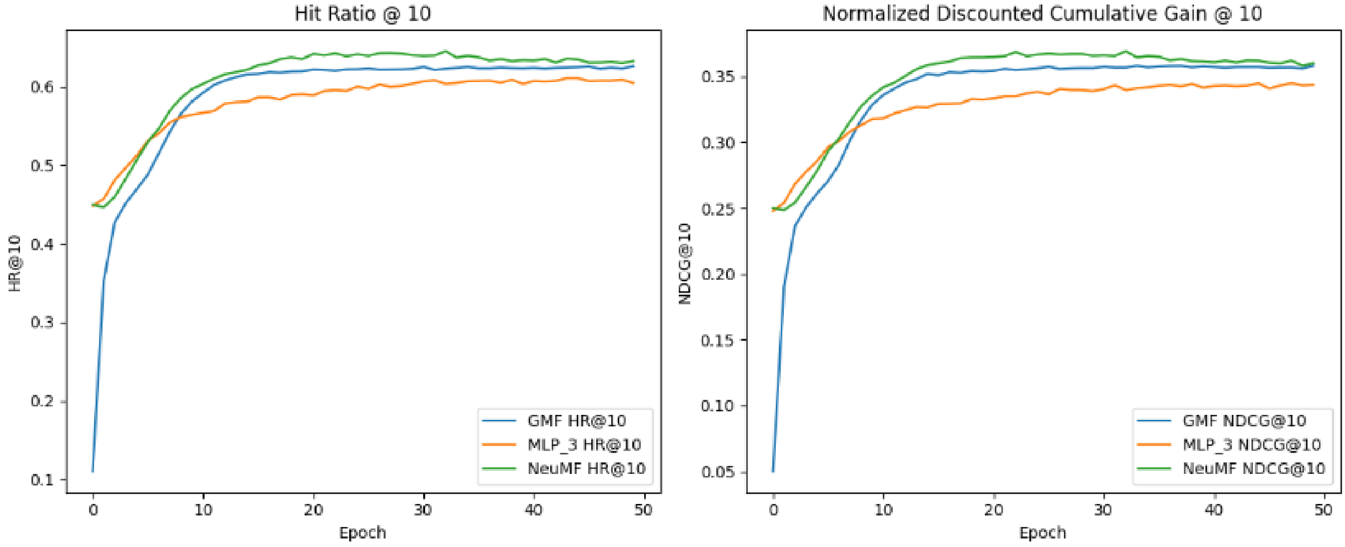
3. Training and Test Setting

I followed the settings introduced in Subsection 4.1 of the paper. Specifically:

- **Loss function:** Binary Cross Entropy
- **Number of negative instances:** Four negative instances per positive instance
- **Optimization method:** Adam with a learning rate of 0.001
- **Predictive factors:** Using factor=8
- **Network structure:** A tower pattern, where the bottom layer is the widest and each successive layer has half the number of neurons

4. Comparing the Three Methods

First, I must declare that the number of layers in the `MLP` model and the `NeuMF` model are the same, and all three models share the same predictive factor.



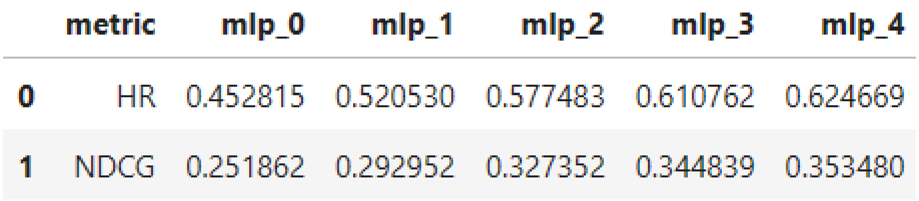
Analysis:

1. According to the plots, `NeuMF` has the best performance on both HR@10 and NDCG@10, followed by the `GMF`.

This is different from the outcome in the paper, where `GMF` underperforms compared to the other two models.

2. All models converge quickly: all three models reach relatively stable performance levels within approximately 10 epochs.
   After nearly 20 epochs, there is a downward trend in the performance of `NeuMF`, which may be caused by overfitting.

5. Reproduce the ablation study in Table 3



	metric	mlp_0	mlp_1	mlp_2	mlp_3	mlp_4
0	HR	0.452815	0.520530	0.577483	0.610762	0.624669
1	NDCG	0.251862	0.292952	0.327352	0.344839	0.353480

The scores obtained in the table are the highest scores during training. For instance, the `mlp_4` model achieves the best performance after 20 epochs and then its performance starts to decrease.

Analysis:

1. The results are similar to the paper: stacking more layers is beneficial to performance. This result is highly encouraging, indicating the effectiveness of using deep models for collaborative recommendation.
2. However, more layers can also lead to the possibility of overfitting.