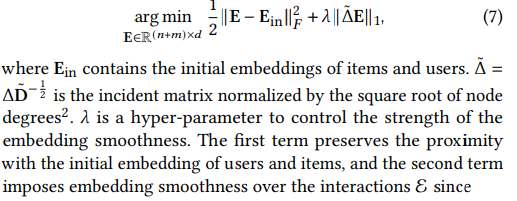
The key of recommender systems is to predict whether users are likely to interact with items based on the historical interactions [5, 12, 20, 56], including clicks, add-to-cart, purchases, etc. As a widely used solution, collaborative filtering (CF) techniques are developed to model historical user-item interactions, assuming that users who behave similarly are likely to have similar preferences towards items.

To better exploit the user-item interactions as well as the high-order connectivity therein, graph collaborative filtering models such as NGCF [46] and LightGCN [20] propose to explicitly propagate the user embedding e 𝑘 𝑢 and item embedding e 𝑘 𝑖 according to the user-item interaction through the propagation.

**Despite their success, prior manner of modeling user-item relationships is insufficient to discover the heterogeneous reliability of interactions among instances in recommender systems. The key reason is that most existing deep recommender systems uniformly treat all the interactions. Therefore, it is desired to design a new collaborative filtering method that adaptively propagates the embedding in recommender systems, which can lead to more accurate and robust recommendations.**

Graph Trend filtering Networks (GTN) to capture and learn the adaptive importance of the interactions in recommender systems.

Proposed the following embedding smoothness objective for user-item graph in recommendations:

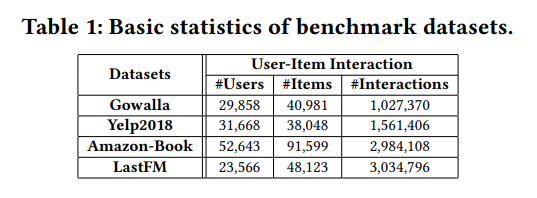


Therefore, if an interaction happens between a user and an item that has significantly different embeddings, such interaction might be unreliable or might not reflect the actual preference of the user, so that it should be down weighted.

**Experiment**

**1. Main**

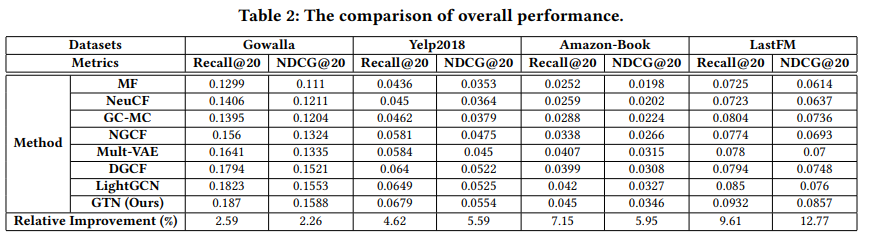
**Datasets**: Gowalla, Yelp2018, Amazon-Book, and LastFM



**Baseline models**: MF, NeuCF, GC-MC, Milt-VAE, NGCF, DGCF, LightGCN

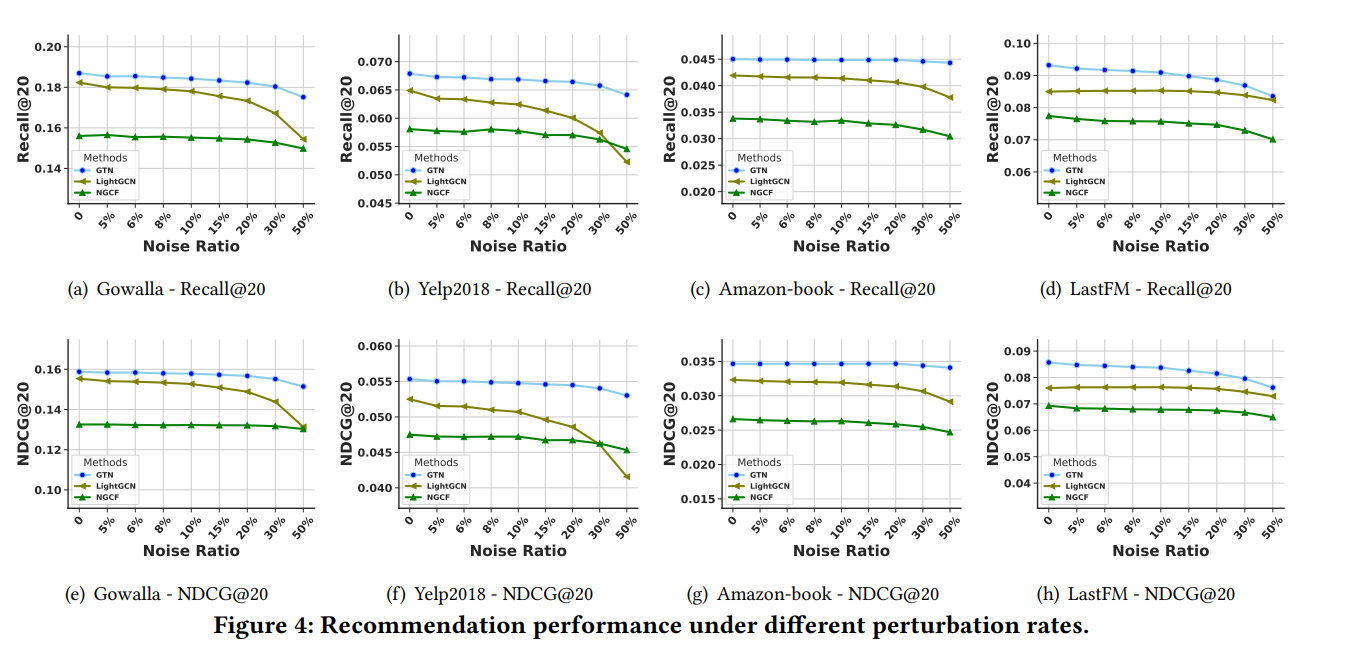
**Evaluation Metrics**: Recall@𝐾 and Normalized Discounted Cumulative Gain (NDCG@𝐾). By default, we set the value of 𝐾 as 20. Note that higher values of Recall@𝐾 and NDCG@𝐾 indicate better performance for recommendations. We report the average metrics for all users in the test set.

**Parameter Settings**: For embedding size 𝑑, we tested the value of {16, 32, 64, 128, 256, 512}. The batch size and learning rate were searched in {128, 512, 1024, 2048} and {0.0005, 0.001, 0.005, 0.01, 0.05, 0.1}, respectively. Note that we adopt the default hyper-parameters as suggested by the corresponding papers for all baselines, and we closely follow the settings of the NGCF and LightGCN3 for a fair comparison.

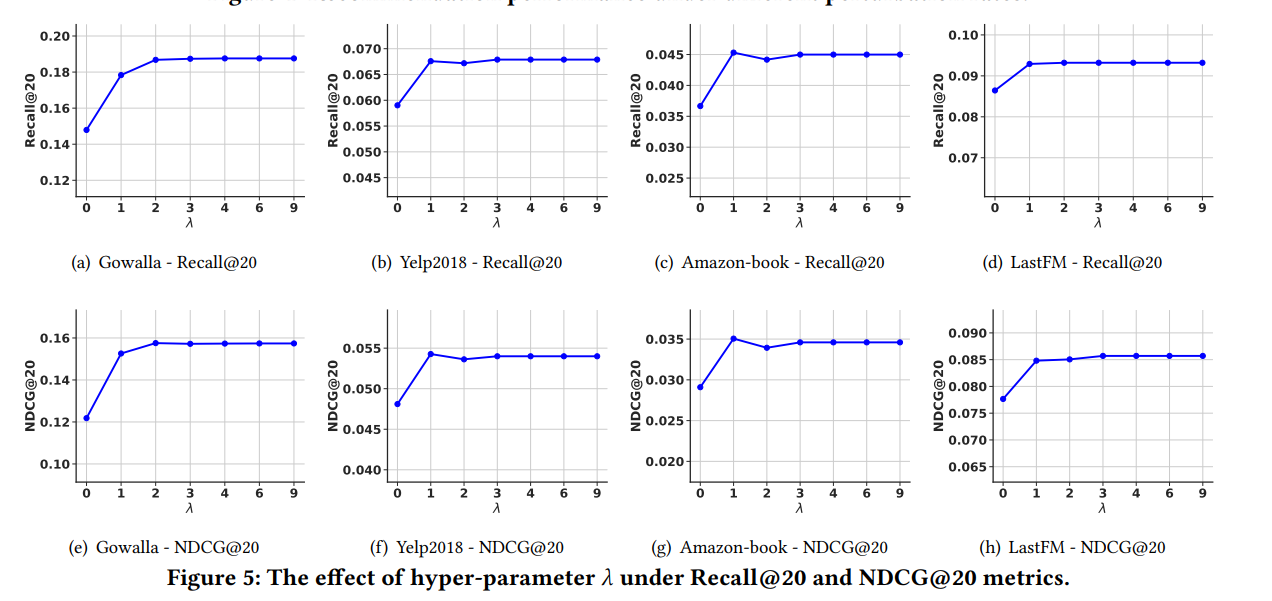
**Performance Comparison:**

**2. Robustness**

Specifically, we perturb the test data by randomly injecting a certain ratio of edges and evaluate the recommendation performance of the models trained on clean data. In addition, we vary the ratio of noisy edges in the range of {5%, 6%, 8%, 10%, 15%, 20%, 30%, 50%}



**3. Different Lambda parameter**



**4. Different Nums of Layers**

