iteratively. The process can be expressed as:

$$\boldsymbol{\mu}_k = \frac{1}{|C_k|} \sum_{x_v=k, \boldsymbol{b}_v \in C_k} \boldsymbol{b}_v, \tag{3}$$

where $|C_k|$ is the number of micro nodes within C_k, and x_v is the macro node index that v is assigned to. Further, the optimization objection of the behavior pattern grouping is:

$$\begin{aligned} &\min_{\substack{x_1,\dots,x_{m+n} \\ \boldsymbol{\mu}_1,\dots,\boldsymbol{\mu}_K}} J\left(x_1, \dots, x_{m+n}; \boldsymbol{\mu}_1, .., \boldsymbol{\mu}_K\right) \\ &\triangleq \sum_{k=1}^{K} \sum_{x_v=k, \boldsymbol{b}_v \in C_k} \sqrt{(\boldsymbol{b}_v - \boldsymbol{\mu}_k)(\boldsymbol{b}_v - \boldsymbol{\mu}_k)^\top}. \end{aligned} \tag{4}$$

where J is the objection function of behavior pattern grouping. As shown in Figure 2, the micro nodes with similar behavior patterns will be composed of a macro node. Note that each macro node v will also be assigned a trainable embedding $\widetilde{\boldsymbol{E}}_v \in \mathbb{R}^d$.

3.1.2 ***Organizing Macro Edges.*** Macro edges depict relationships between two macro nodes within a specific user/item subgraph, signifying the behavioral patterns within that subgraph. It's important to note that macro edges have a distinct design compared to micro edges. The micro edges present connections between fixed micro user nodes and micro item nodes. Since micro nodes remain constant, the micro edges are also fixed. In contrast, macro edges capture the connection strength between two macro nodes in a subgraph, which is tailored to each user and item subgraph.

In Figure 2, the user v is depicted as having two 1-hop macro nodes, each with macro edge weights of 4 and 3, respectively. Moving to the second hop, the user extends to three macro nodes, and these macro edges represent the connections between the 1-hop macro nodes and the 2-hop macro node. Formally, we use $\widetilde{C} = \{C_1, C_2, \dots, C_{\widetilde{n}+\widetilde{m}}\}$ to represent the entire set of macro nodes in the MAG. We employ $\widetilde{\mathcal{R}}^{(k)}_{v;p,q}$ to denote the macro edge for any user/item node v with its k^{th}-hop neighbors, where $C^{(k-1)}_{v;p}$ represents the macro node in $(k-1)^{th}$-hop macro neighbors $\widetilde{\mathcal{N}}^{(k-1)}_v$, and $C^{(k)}_{v;q}$ represents the macro node in k^{th}-hop macro neighbors $\widetilde{\mathcal{N}}^{(k)}_v$. Thus the weight of macro edges can be computed as:

$$\widetilde{\mathcal{R}}^{(k)}_{v;p,q} = \sum_{a \in C^{(k-1)}_{v;p}, b \in C^{(k)}_{v;q}} r_{ab}, \tag{5}$$

where $C^{(k-1)}_{v;p} = C_{v;p} \cap \mathcal{N}^{(k-1)}_v$ represents the macro nodes related to node v within its $(k-1)^{th}$-hop neighbors and $C^{(k)}_{v;q} = C_{v;q} \cap \mathcal{N}^{(k)}_v$ represents the macro nodes related to node v within its k^{th}-hop neighbors. In § 3.3, we will introduce how to get online updating macro edges on billion-scale recommender systems. Finally, after transforming micro recommendation graphs into macro recommendation graphs, MAGs have significantly fewer nodes and edges by extracting behavior patterns explicitly into macro nodes.

3.2 Macro Graph Neural Network

3.2.1 ***Macro Weight Modeling.*** The overall framework of our proposed MacGNN is shown in Figure 3. To better identify the target user/item preferences over a certain macro node, we design

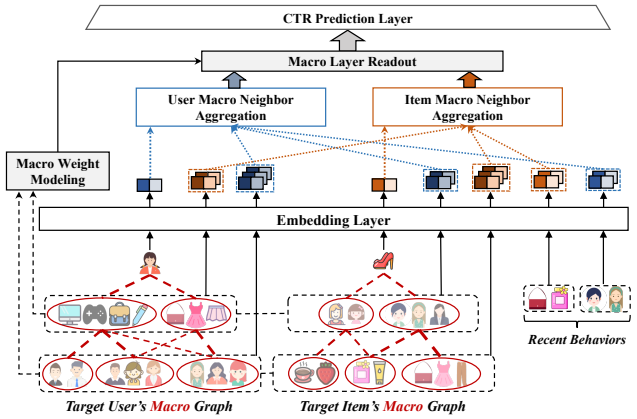

Figure 3: The model architecture of the proposed MacGNN.

the macro weight modeling for macro neighbors according to the weights of connected macro edges.

In order to avoid the excessive gap between the macro edge weights of hot nodes and cold nodes and conduct modeling flexibly, we equip the macro weight modeling with logarithmic smoothing and temperature-based softmax activation. Formally, take the target user/item v as an example, given a macro node q in its k^{th}-hop neighborhood, the macro weight $w^{(k)}_{v;q}$ of q toward the target user/item v is calculated as:

$$s^{(k)}_{v;q} = \log\left(\sum_{p \in \widetilde{\mathcal{N}}^{(k-1)}_v} \widetilde{\mathcal{R}}^{(k)}_{v;p,q} + 1\right), \quad w^{(k)}_{v;q} = \frac{exp\left(s^{(k)}_{v;q}/\tau\right)}{\sum_{j \in \widetilde{\mathcal{N}}^{(k)}_v} exp\left(s^{(k)}_{v;j}/\tau\right)}, \tag{6}$$

where τ is a temperature coefficient hyper-parameter [7]. These modeled weights represent the importance of these macro neighboring nodes in the target user/item's historical interactions.

3.2.2 ***Marco Neighbor Aggregation & Layer Readout.*** To mine the macro relationships effectively and efficiently, we first design a macro neighbor aggregation architecture rather than a time-consuming recursive graph convolution. Then, we propose the macro layer readout to aggregate the macro information of the target user and item.

Macro Neighbor Aggregation. Due to the different semantics of users and items, we utilized two separate macro neighbor aggregation modules without parameter sharing for user-type macro nodes and item-type macro nodes, respectively.

For user-type target nodes and their k^{th}-hop user-type macro neighbors, the aggregation function MNA_u can be defined as:

$$\begin{aligned} &MNA_u(u, p \in \widetilde{\mathcal{N}}^{(k)}_u, \boldsymbol{E}_u, \widetilde{\boldsymbol{E}}_p, \widetilde{\mathcal{N}}^{(k)}_u; \boldsymbol{Q}_u, \boldsymbol{K}_u, \boldsymbol{V}_u) \\ &\triangleq \sum_{p \in \widetilde{\mathcal{N}}^{(k)}_u} \sigma\left(\langle \boldsymbol{Q}_u \cdot \widetilde{\boldsymbol{E}}_p, \boldsymbol{K}_u \cdot \boldsymbol{E}_u \rangle\right) \cdot \boldsymbol{V}_u \cdot \widetilde{\boldsymbol{E}}_p, \end{aligned} \tag{7}$$

where $\boldsymbol{Q}_u, \boldsymbol{K}_u, \boldsymbol{V}_u \in \mathbb{R}^{d \times d'}$ are trainable self-attention matrics for user-type nodes, $\langle \cdot \rangle$ is the inner product function, and $\sigma(\cdot)$ is the softmax activation function. Specifically, given the given target user u and a user-type macro node p in its k^{th}-hop neighborhood (such