# TOGETHER IS ***WAY*** BETTER
## WITH GRAPH NEURAL NETWORKS

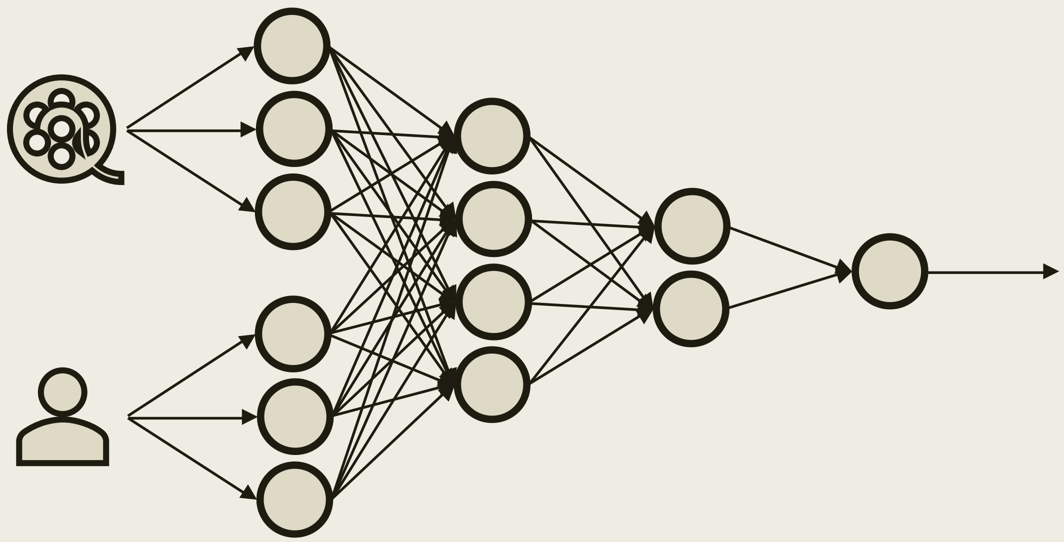
De Marinis Pasquale p.demarinis6@studenti.uniba.it
Loconte Lorenzo l.loconte5@studenti.uniba.it

# Objective

- Introduction of **Graph Neural Networks** (GNNs) in existing deep recommender systems architectures
- Research questions:
  - *How GNNs perform in contrast with Knowledge Graph Embeddings (KGE) models for learning collaborative features ?*
  - *How GNNs can be integrated in both collaborative and content-based hybrid deep recommender systems ?*

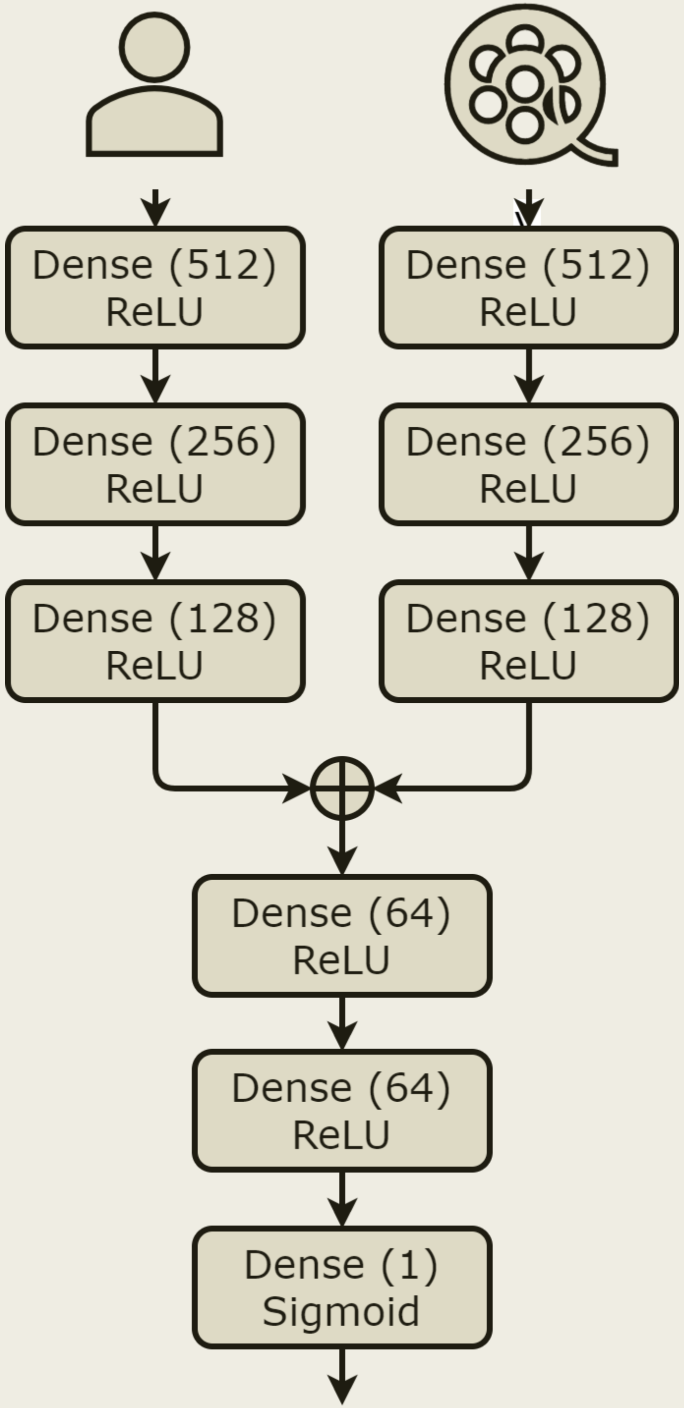
# Previous Works
## Deep Amar

- **Hybrid** architectures for recommendations based on Neural Networks
- Take user ***u*** and item ***i*** and return a relevance score ***s(u,i)***
- Usage of Knowledge Graph Embeddings and Word Embeddings

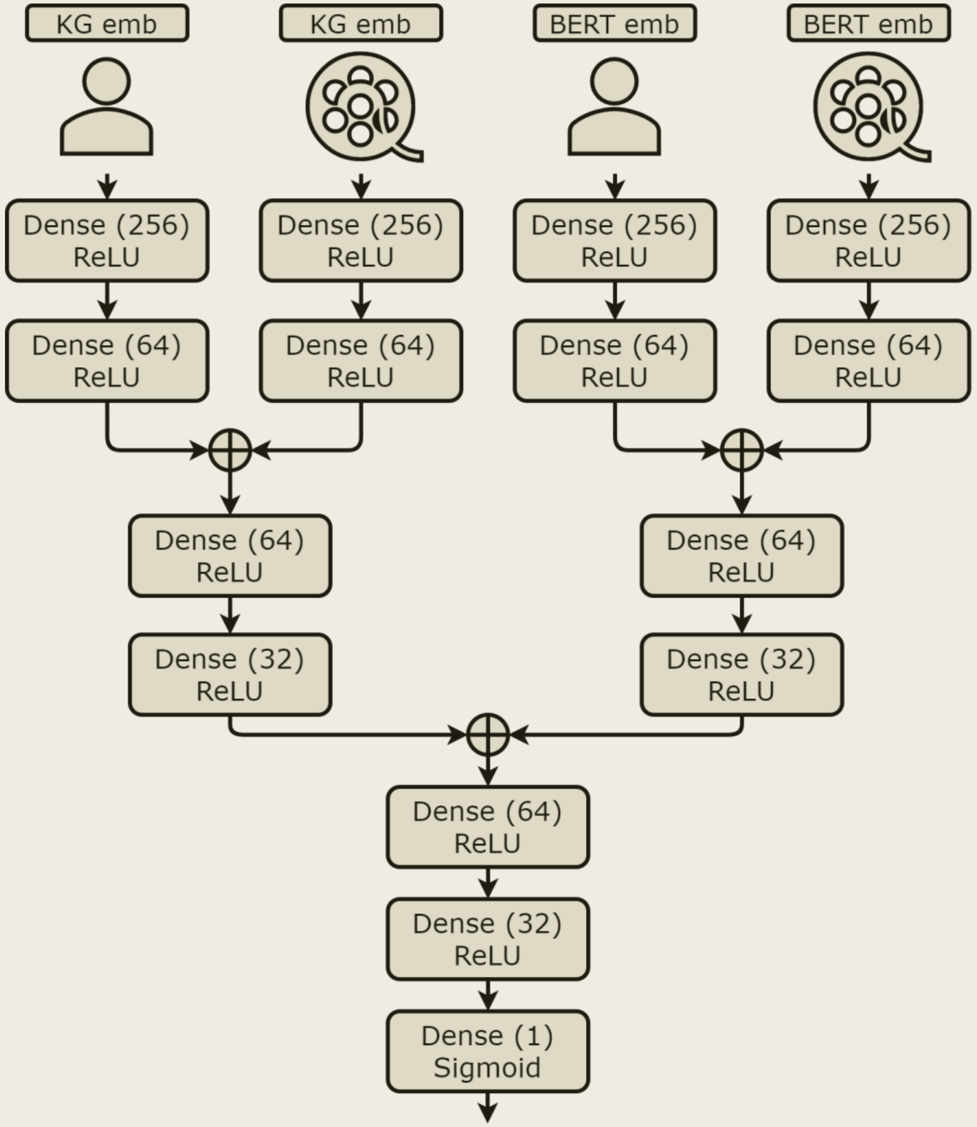


# Previous Works
Deep Amar Revisited - BASIC

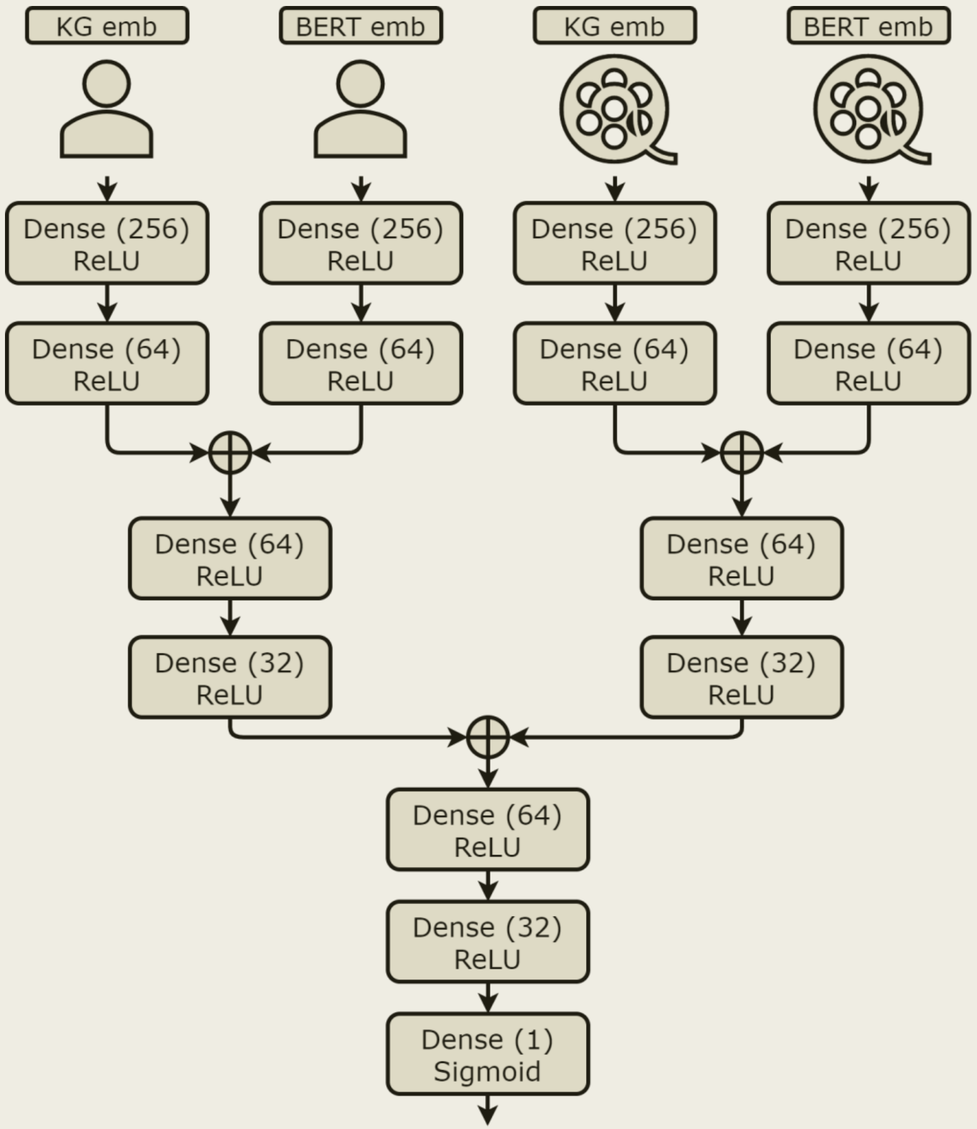
- User and Movie features as inputs
  - *KG embeddings (e.g. TransH)*
  - *Word embeddings (e.g. BERT)*



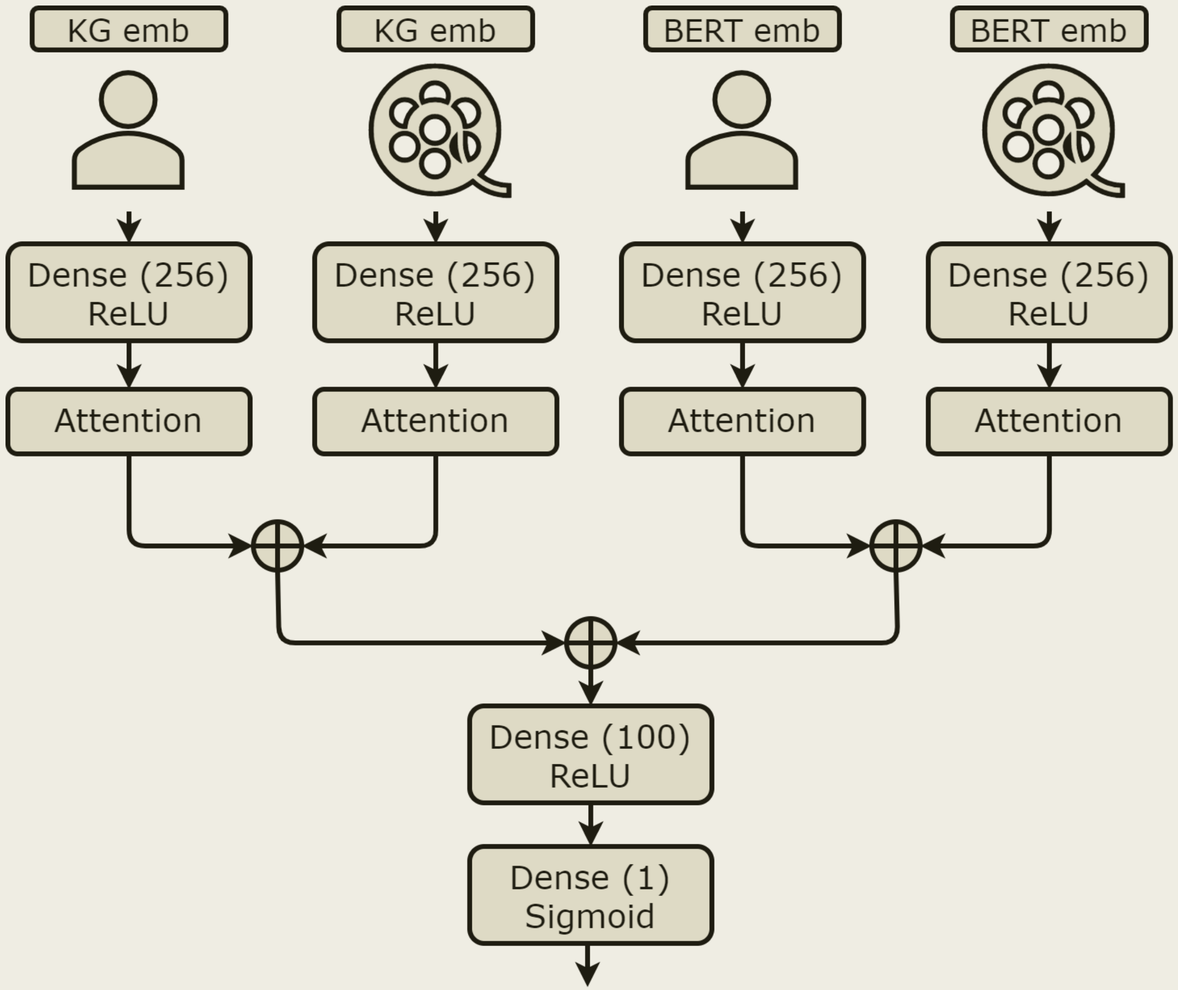
# PREVIOUS WORKS
DEEP AMAR
REVISITED
MIXED
FEATURE BASED

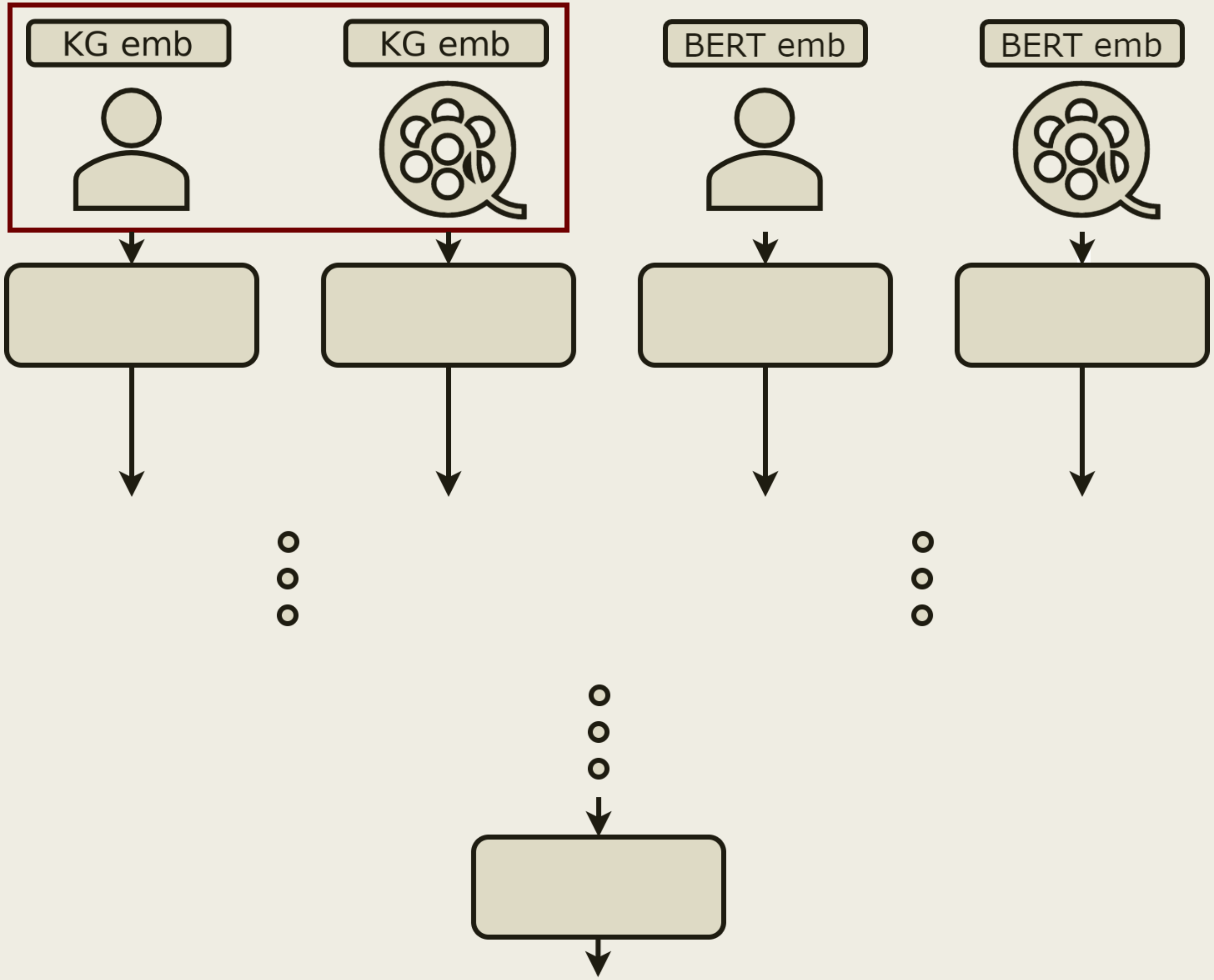


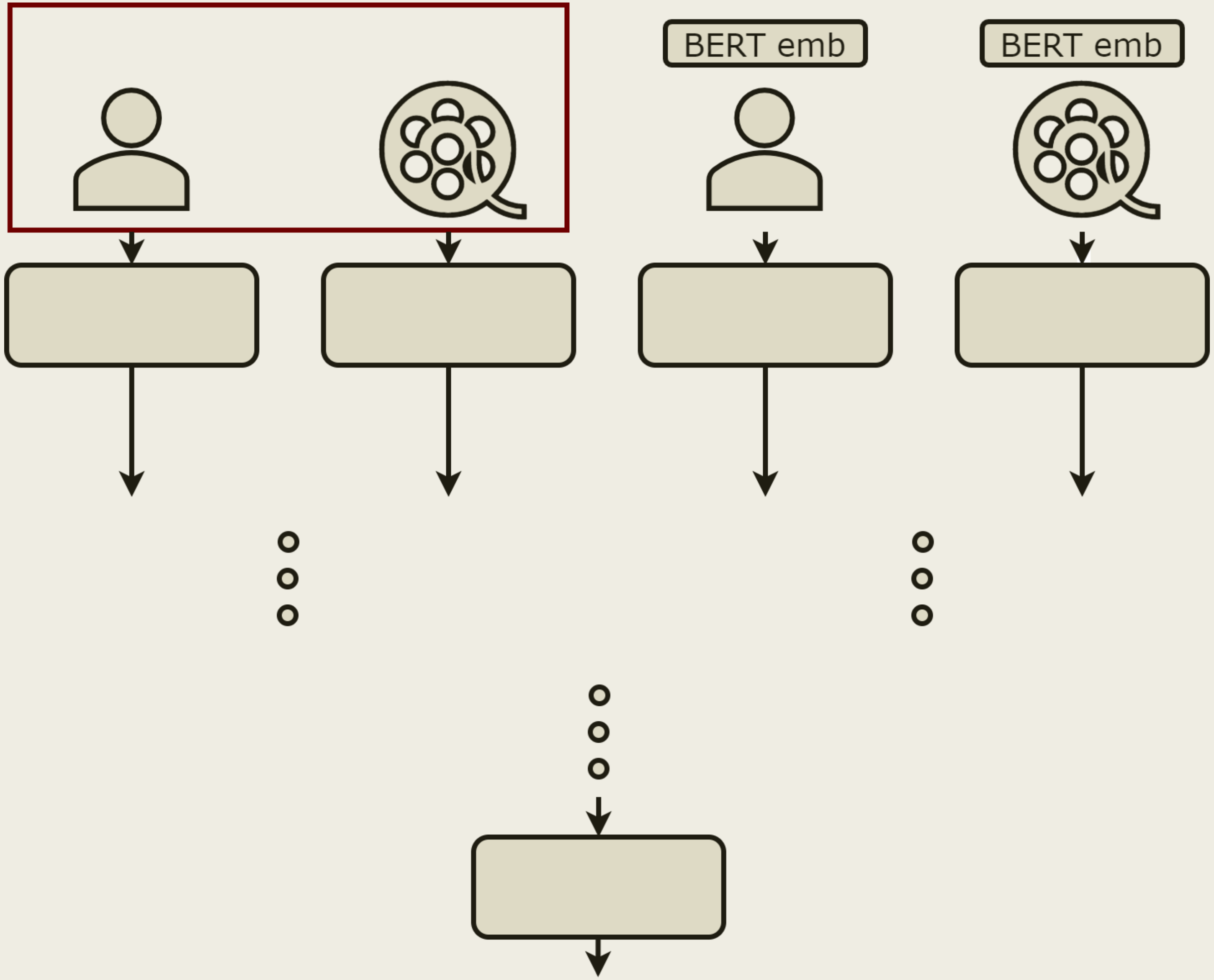
# PREVIOUS WORKS
DEEP AMAR
REVISITED
MIXED
ENTITY BASED

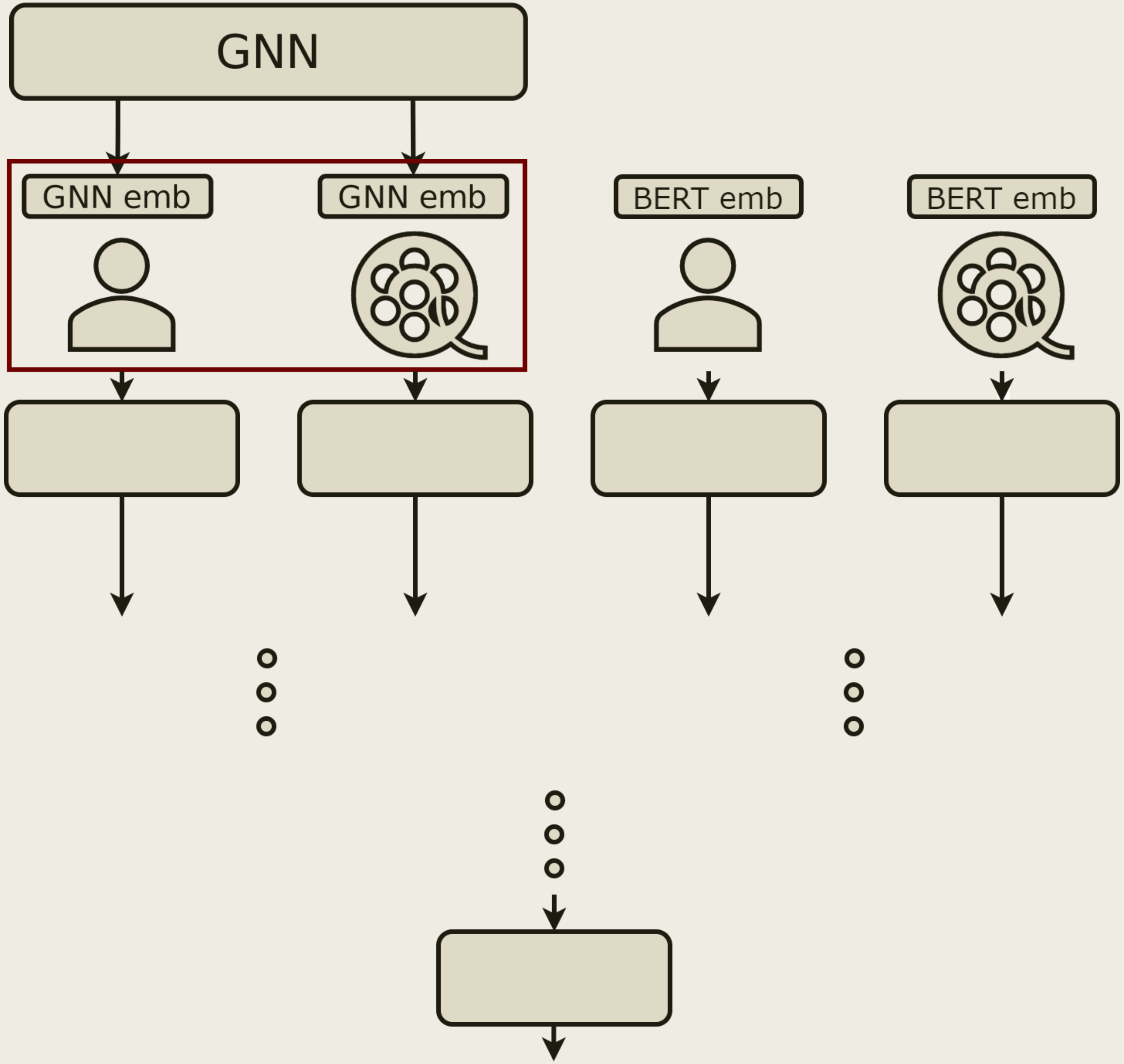


# PREVIOUS WORKS
DEEP AMAR
REVISITED
EXTENDED




KG emb
KG emb
BERT emb
BERT emb

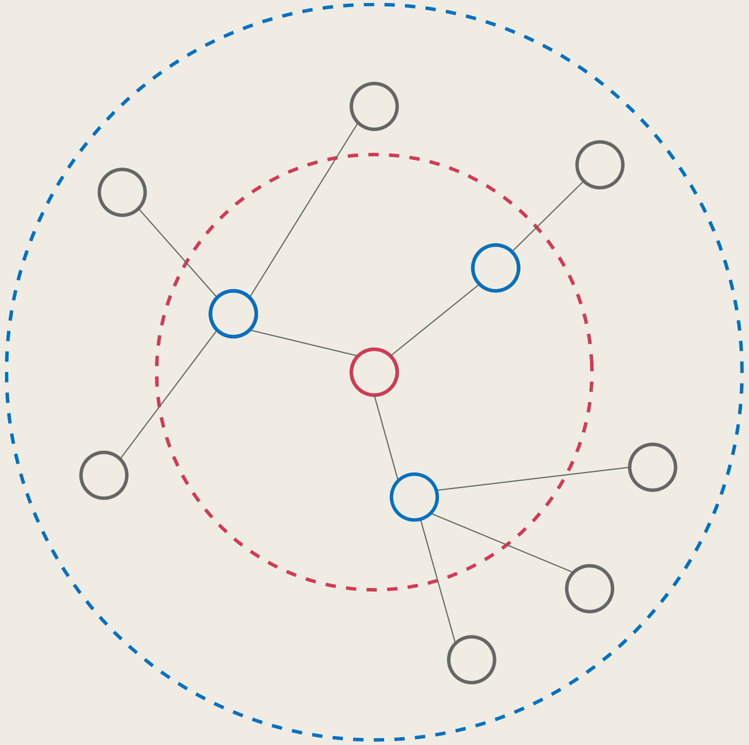

BERT emb
BERT emb


GNN
GNN emb
GNN emb
BERT emb
BERT emb

# Graph Neural Networks

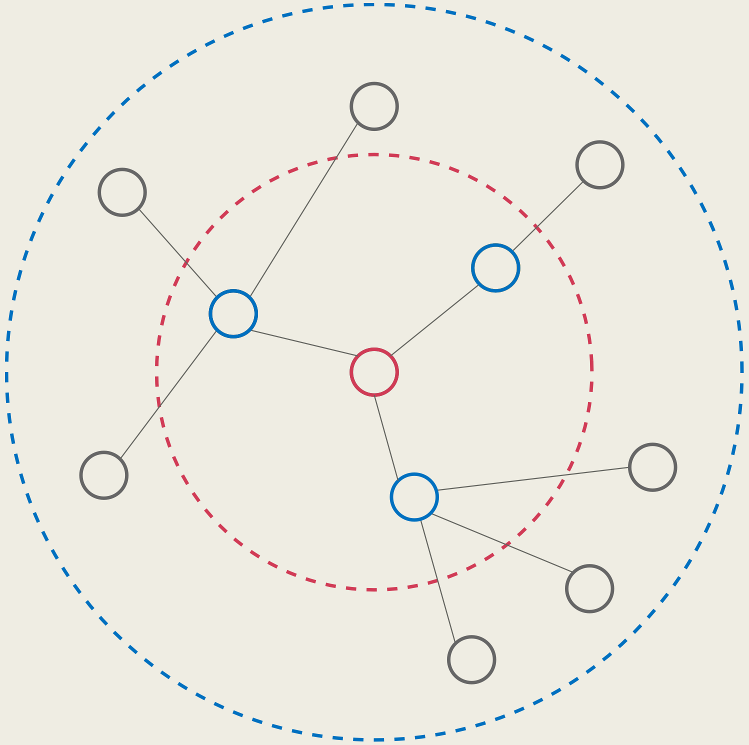
- Neural Network which directly operates on **graph data**
- Neighborhood aggregation as most important operation
- Able to catch higher order interactions
  - *Stacking multiple layers*

$$\mathrm{H}^{(l+1)} = F(\mathrm{H}^{l}, \mathrm{X})$$



# Graph Neural Networks

- Graph Convolutional Networks (GCNs)
- GraphSage (SAmple and aggreGatE)
- Graph Attention Networks (GATs)
- Gated Graph Neural Networks
  (for sequential recommendation)
- … and several others!

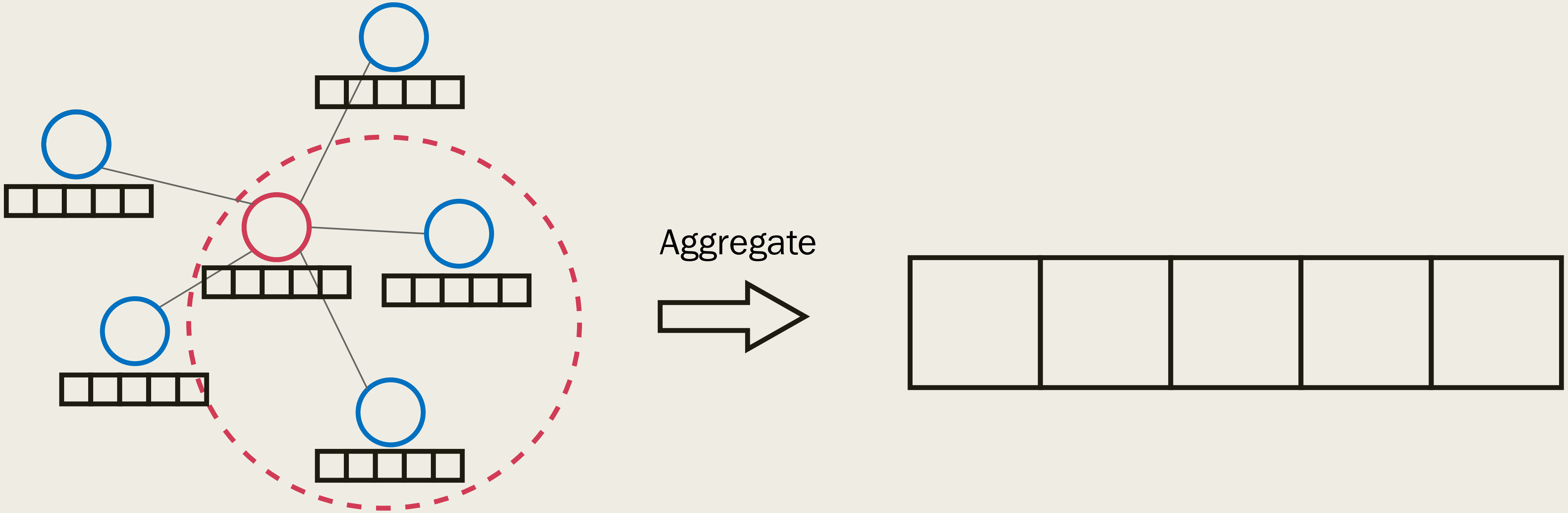


# Graph Convolutional Networks (GCNs)

- Preprocess the **adjacency matrix** to be a symmetrically normalized Laplacian matrix
- Neighbors' features are weighted equally
- A non-linear transformation using a weight matrix is then applied
- LightGCN: the same as for GCN, but without the non-linear transformation
  → less parameters and more efficient

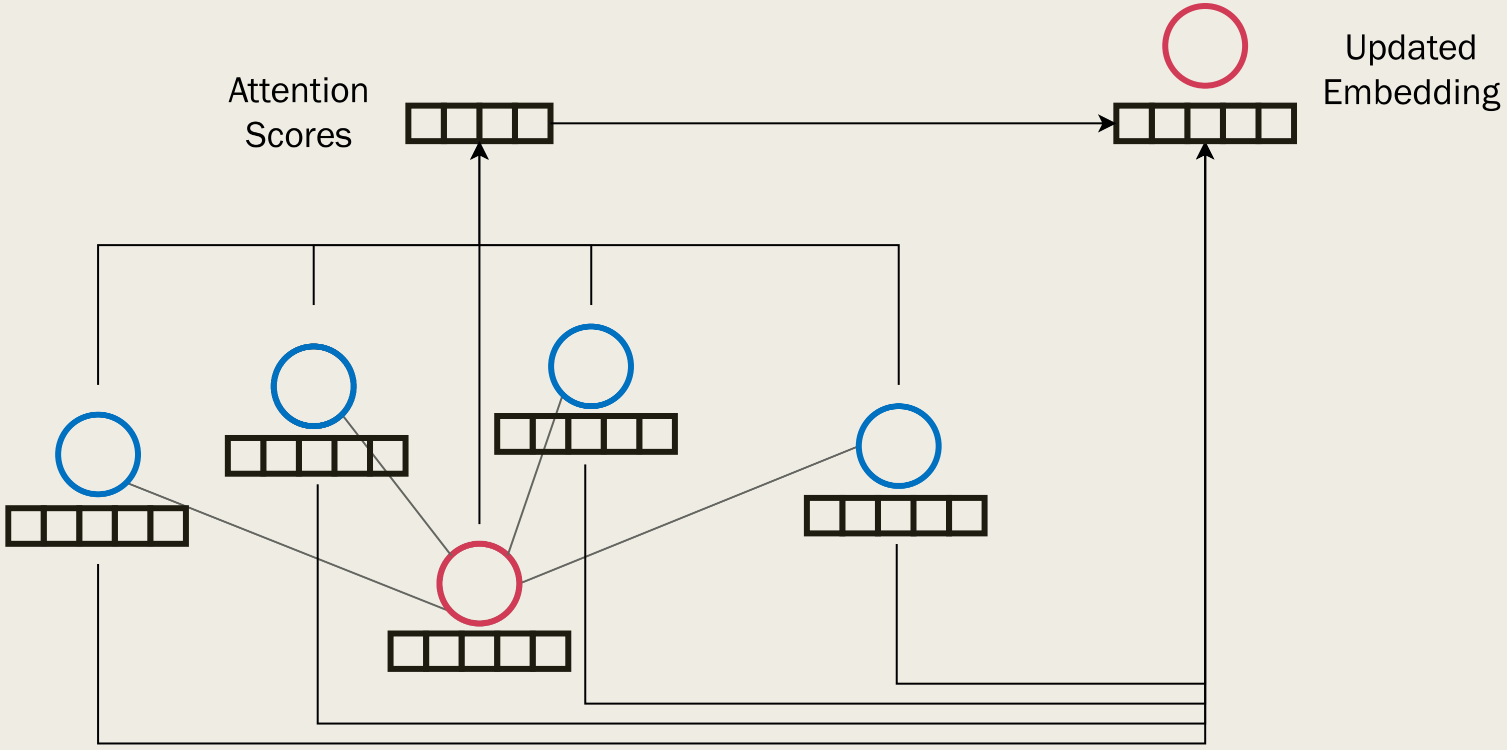
# Graph SAGE (SAmple and aggreGatE)

- Sample neighbors
- Aggregates (mean, sum, pooling)
- Multiply with Weight Matrix
- Activation Function



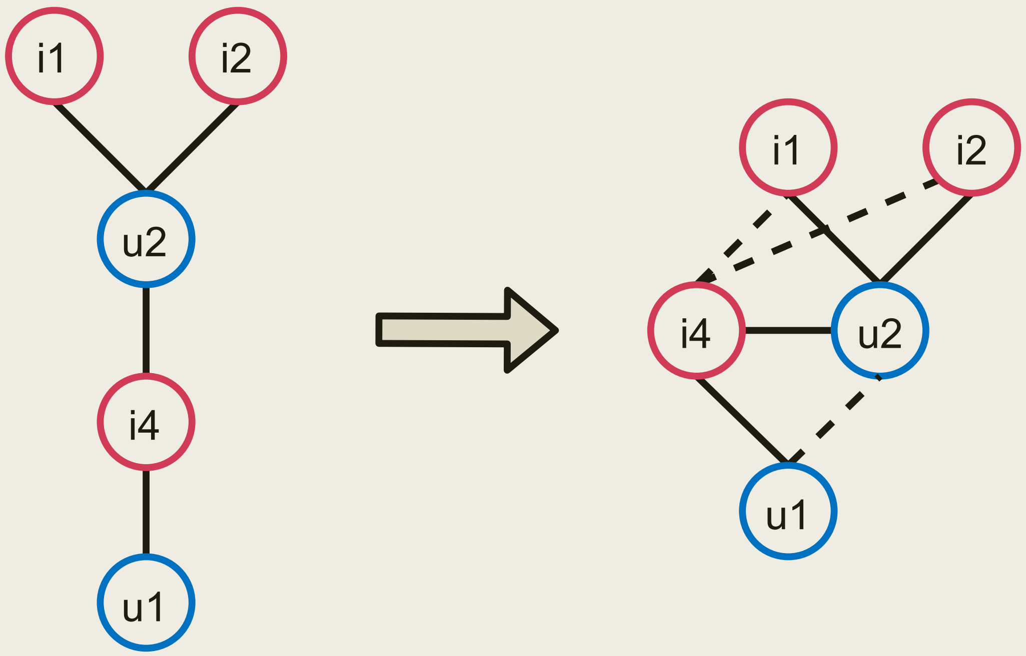
# Graph Attention Network (GAT)

- The neighbors' features are weighted differently, by using an attention mechanism
- The **aggregated features** are then passed through a non-linear transformation



# Deoscillated Graph Collaborative Filtering (DGCF)

- Try to avoid the «Oscillation problem»
  - *Cross-hop matrix*
  - *Laplacian normalization*
    - High-Pass Filter
- BPRLoss
  - *Maximizes distance between positives and negatives item relevance scores*
- Locality-Adaptive Weights
  - *Weights each node*



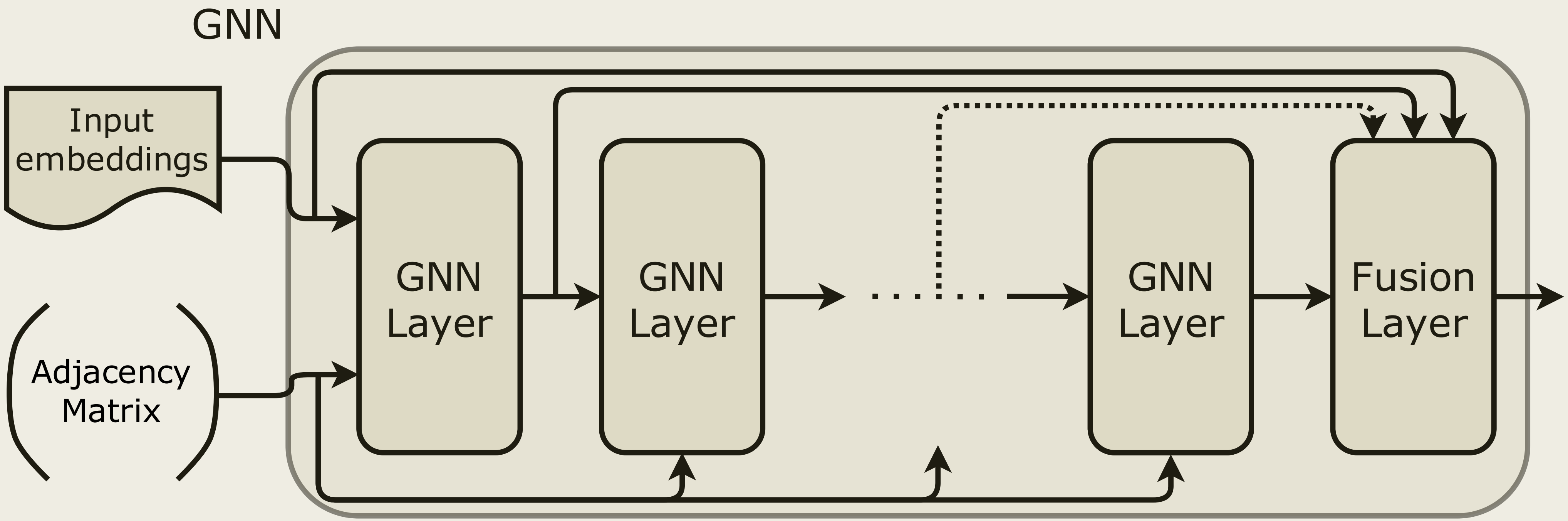

Adding cross-hop connections example

# The Oversmoothing Problem

- With a relatively high number of GNN layers, nodes have approximately the same higher order neighbors in common
- The learned embeddings of nodes will be very similar, hence not permitting to effectively differentiate the nodes
- A simple solution is to limit the number of GNN layers

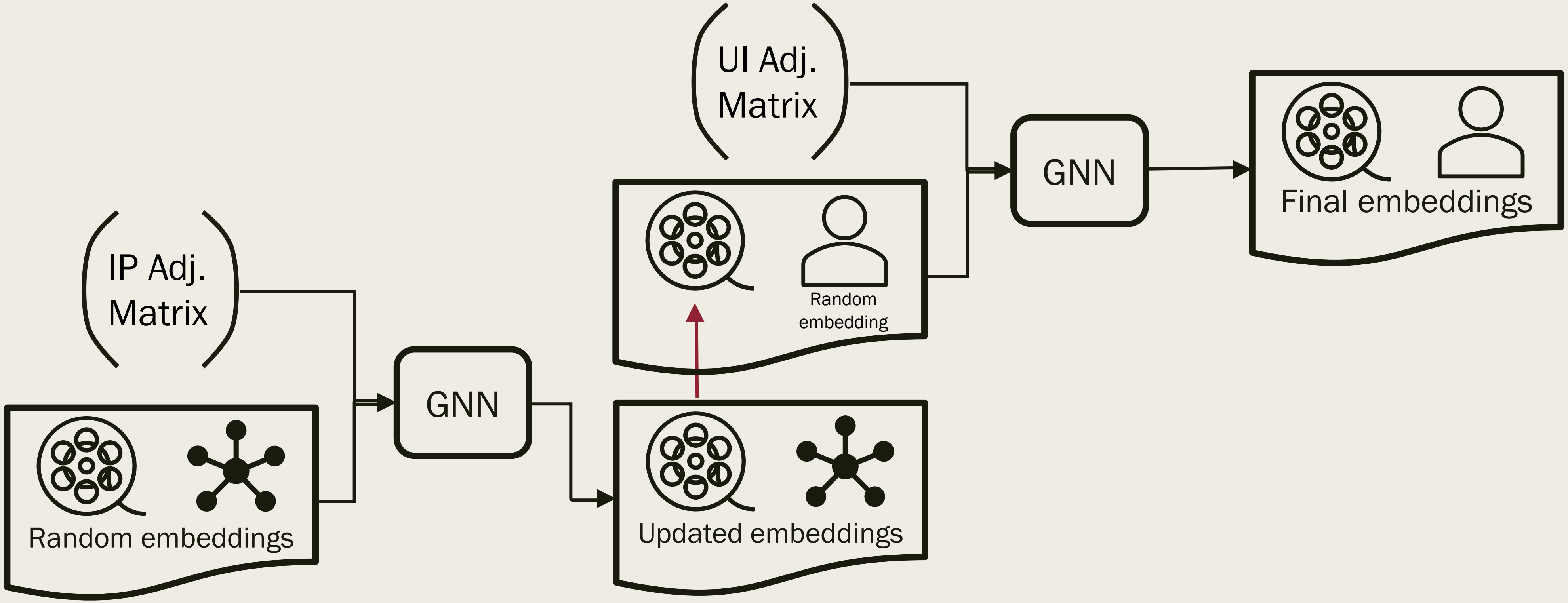
# General GNN architecture

- Input embeddings: given or random
- Fusion layer: concatenation / mean / sum / etc...

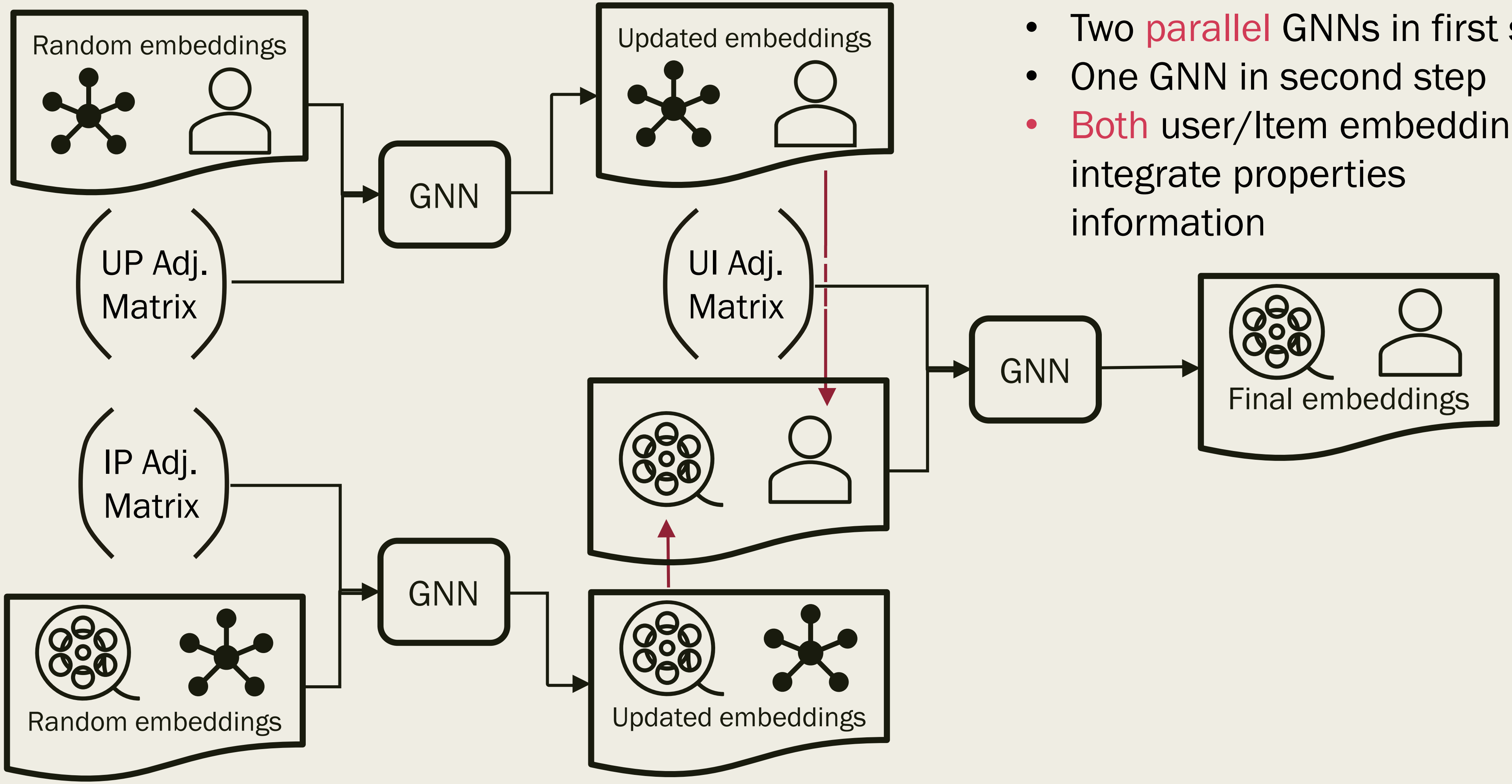


# Two-Step GNN

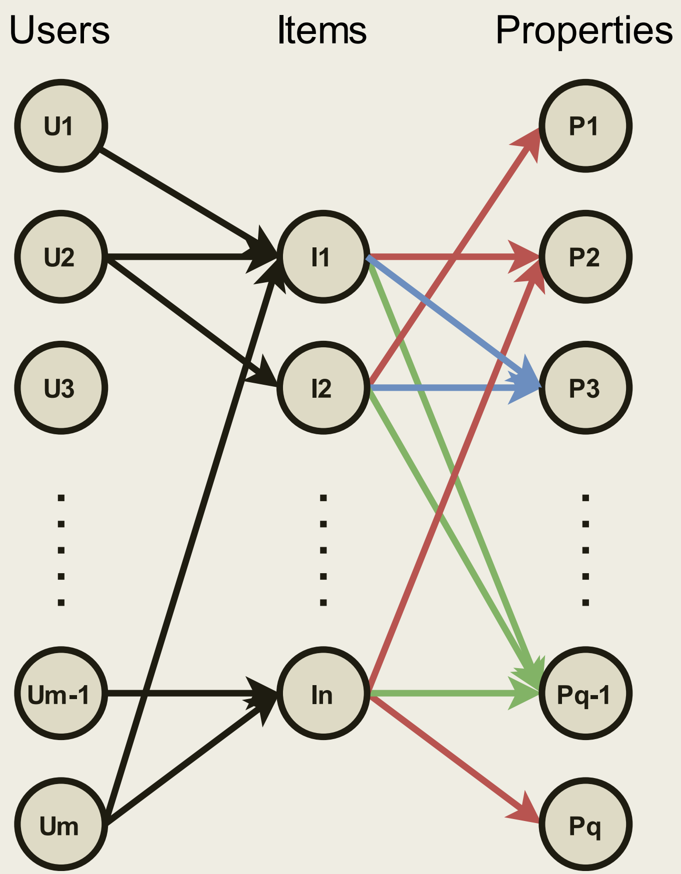
- Two sequential GNNs
- Item embeddings integrate properties information

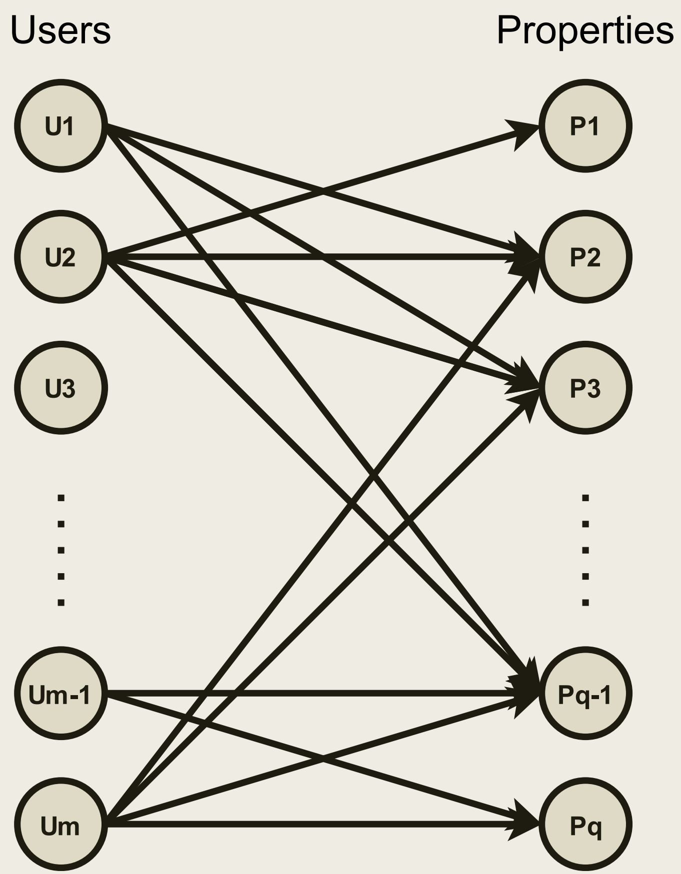


# Two-Way GNN



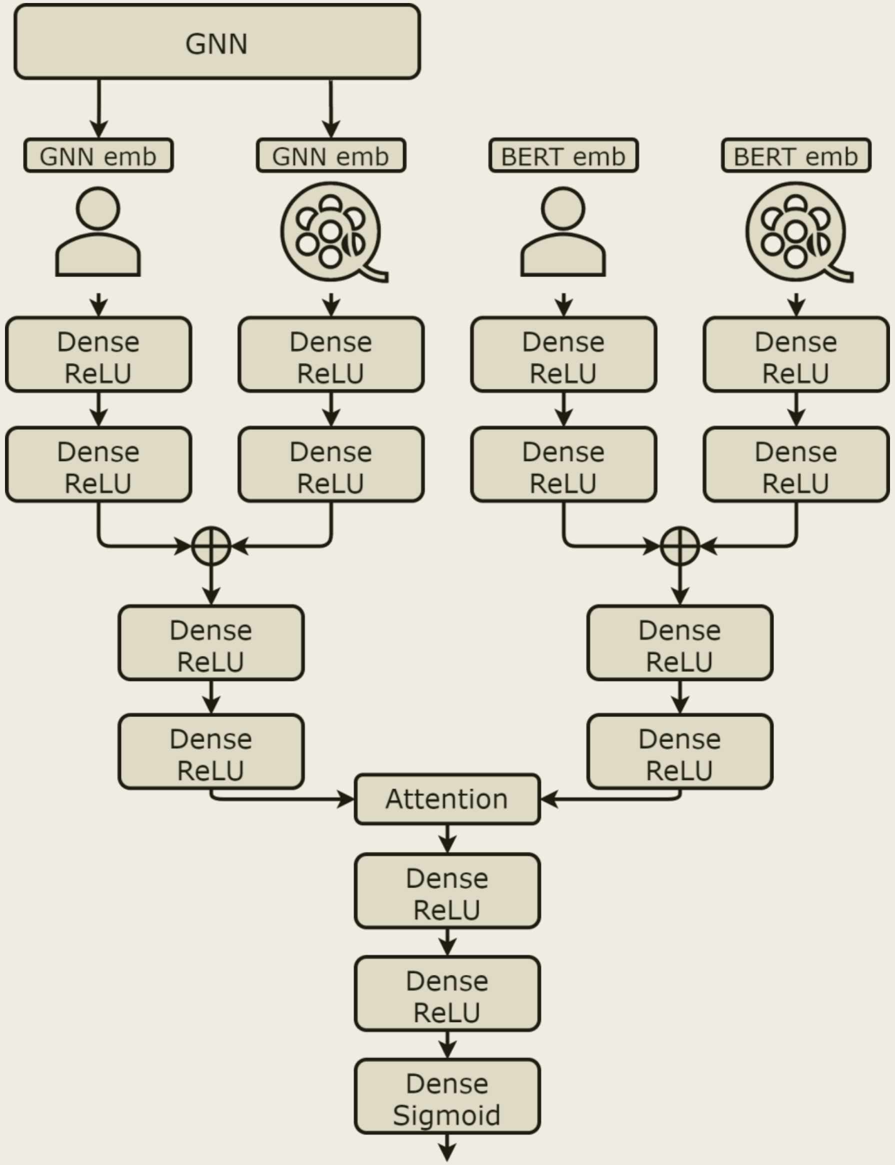

- Two parallel GNNs in first step
- One GNN in second step
- Both user/Item embeddings integrate properties information



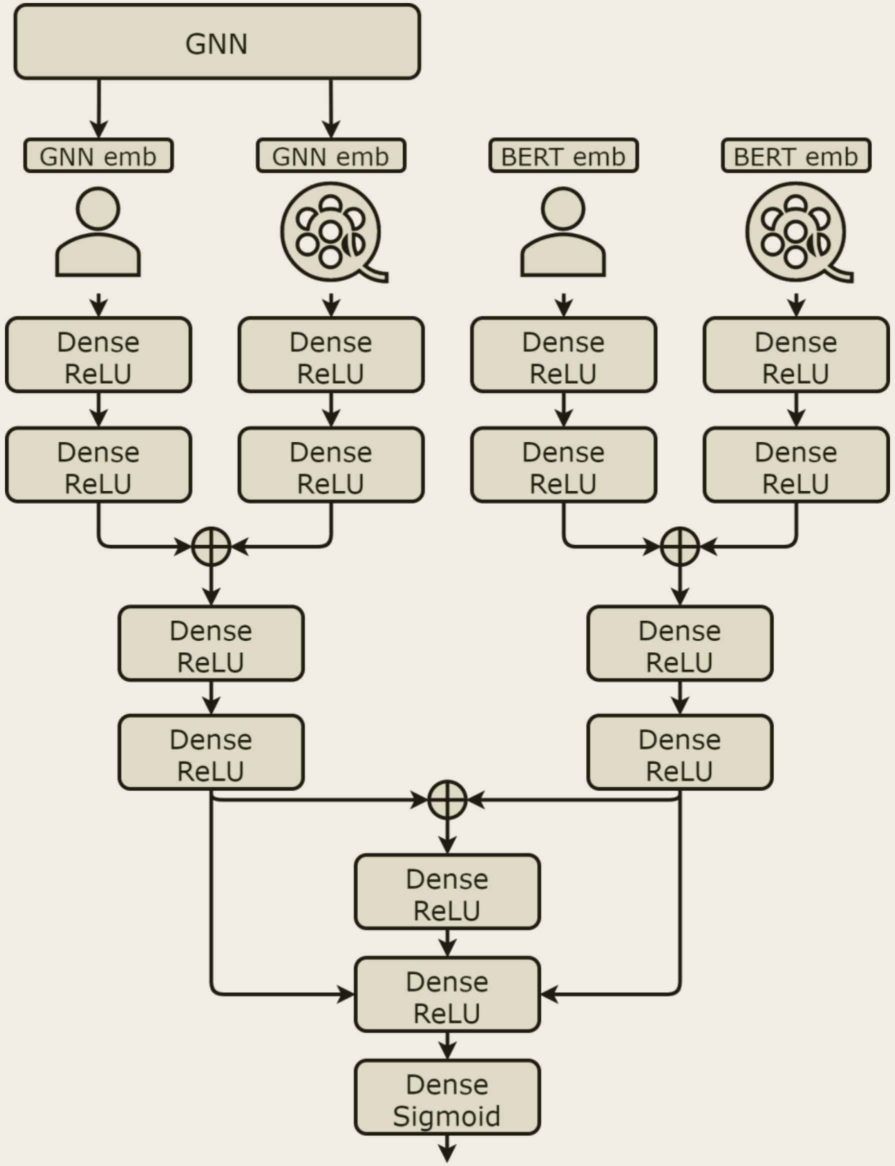

# User-Properties Graph

- Dereification of items
- Outdegree of Resulting graph is the product of User-Items and Item-Properties
  - *Adjacency matrix is less sparse*




# Hybrid Architecture Tweaks

- Attention layer instead of concatenation




# Hybrid Architecture Tweaks

- Residual connection of embeddings before concatenation

# Experiments

## The Dataset

- Movielens-1M with user-item **positive** and **negative** ratings
- Two item-properties relations settings:
  - **RS1** {*subject, director, starring, writer, language, editing, narrator*}
  - **RS2** {*subject, director, starring, writer, language, editing, cinematography, musicComposer, country, producer, basedOn*}
- The item-properties adjacency matrix is **way sparser** than the user-item one

# Experiments

## Grid Search

- **Basic** architecture with GNNs

| BasicRS | Reduce |
|---|---|
| GCN | Concatenate |
| GraphSage | Concatenate |
| GAT | Concatenate |
| LightGCN | Average |
| DGCF | Average |

X

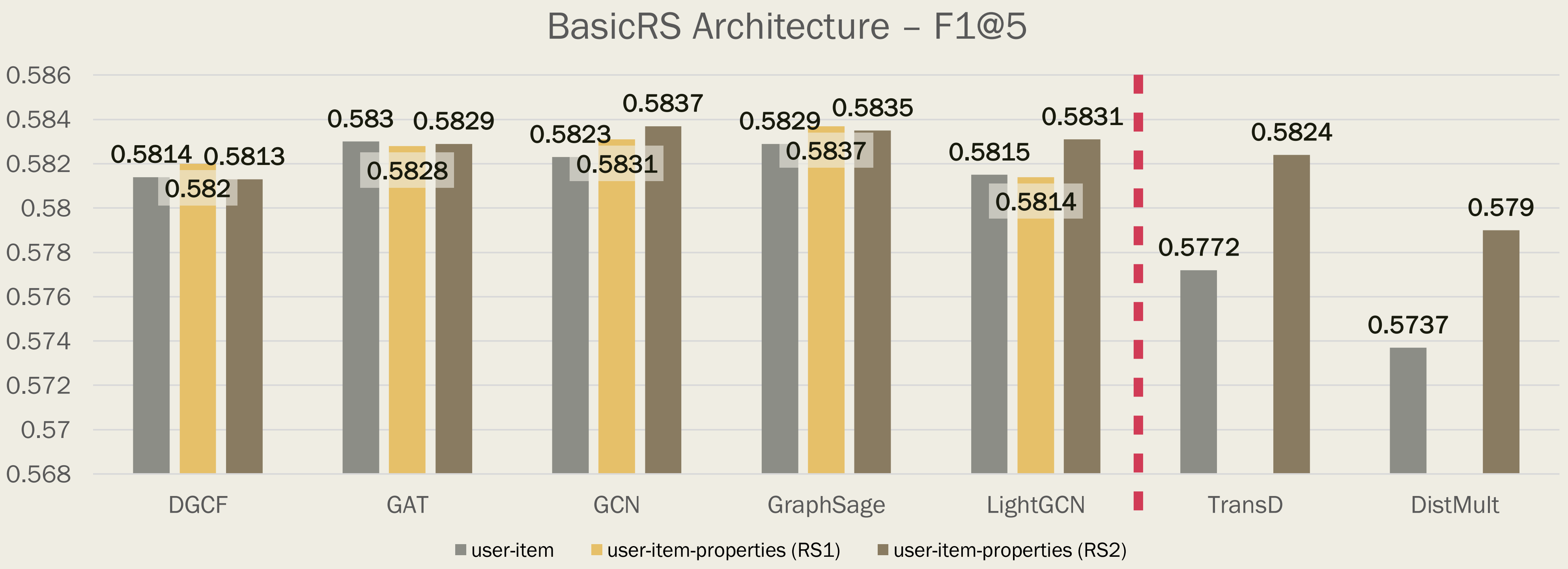
| Dense Units | Channels | # Layers |
|---|---|---|
| (24, 24) | 8 | 2 |
| (32, 32) | 8 | 3 |
| (48, 48) | 16 | 2 |
| (64, 64) | 16 | 3 |
| (96, 48) | 32 | 2 |
| (128, 64) | 32 | 3 |

X

| $L_2$ Reg. |
|---|
| $10^{-5}$ |
| $10^{-4}$ |
| $10^{-3}$ |

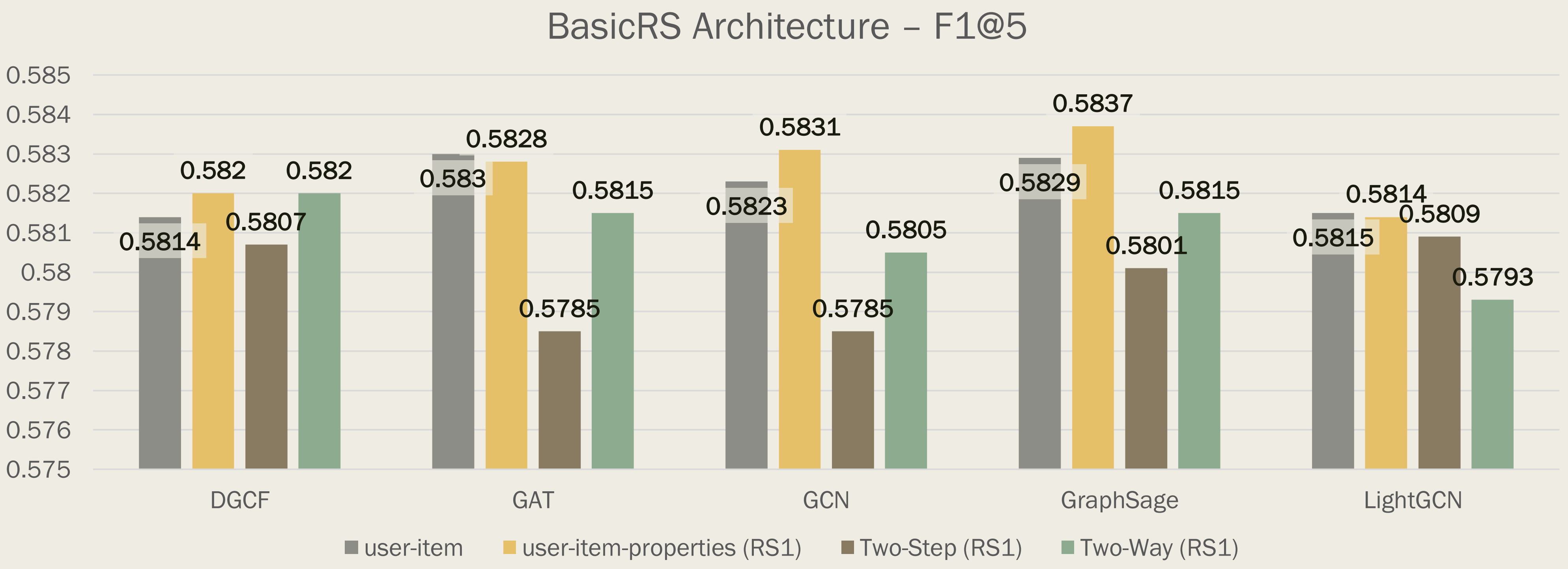
# Experiments

## Results - GNN / KGE comparison



# Experiments

## Results – UI GNN / UIP GNN / Two-Step / Two-Way comparison



# Experiments

## Grid Search

- **Feature-based Hybrid** architecture with GNNs

| HybridCBRS | Reduce |
|---|---|
| GCN | Concatenate |
| GraphSage | Concatenate |
| GAT | Concatenate |
| LightGCN | Average |
| DGCF | Average |

X

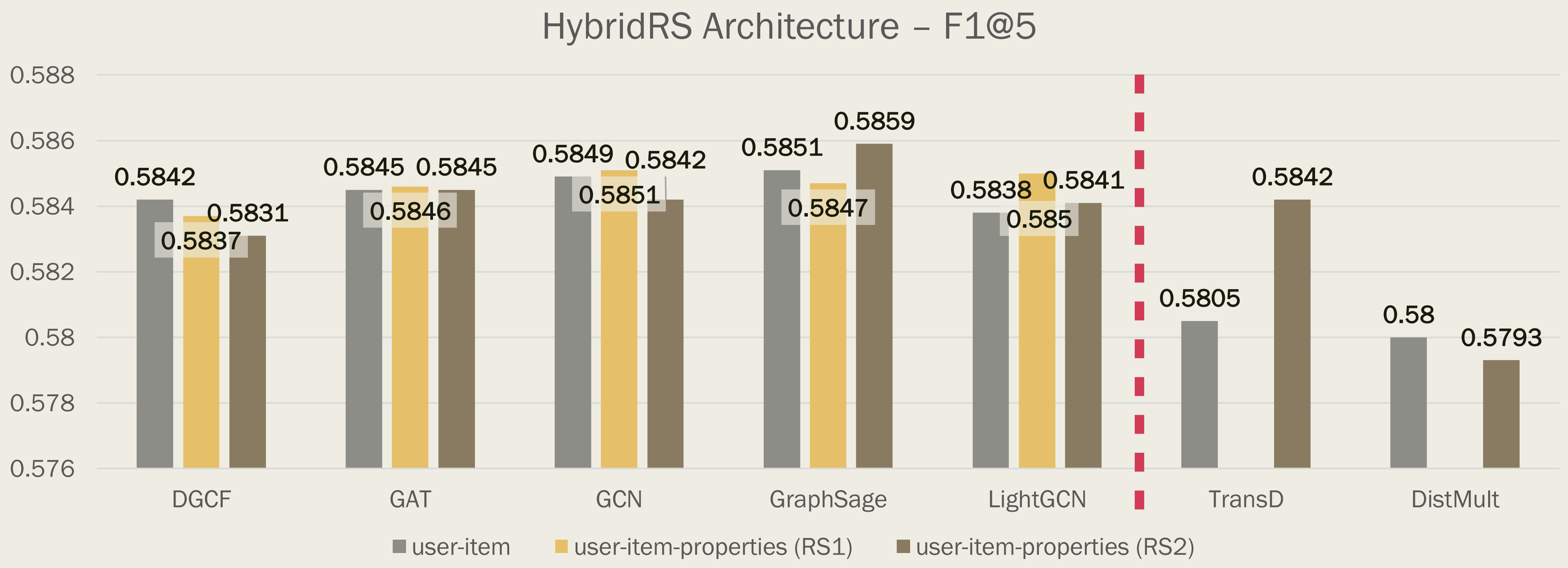
| Dense Units | Channels | # Layers |
|---|---|---|
| (24, 24) | 8 | 2 |
| (32, 32) | 8 | 3 |
| (48, 48) | 16 | 2 |
| (64, 64) | 16 | 3 |
| (96, 96) | 32 | 2 |
| (128, 128) | 32 | 3 |

X

| $L_2$ Reg. |
|---|
| $10^{-5}$ |
| $10^{-4}$ |
| $10^{-3}$ |

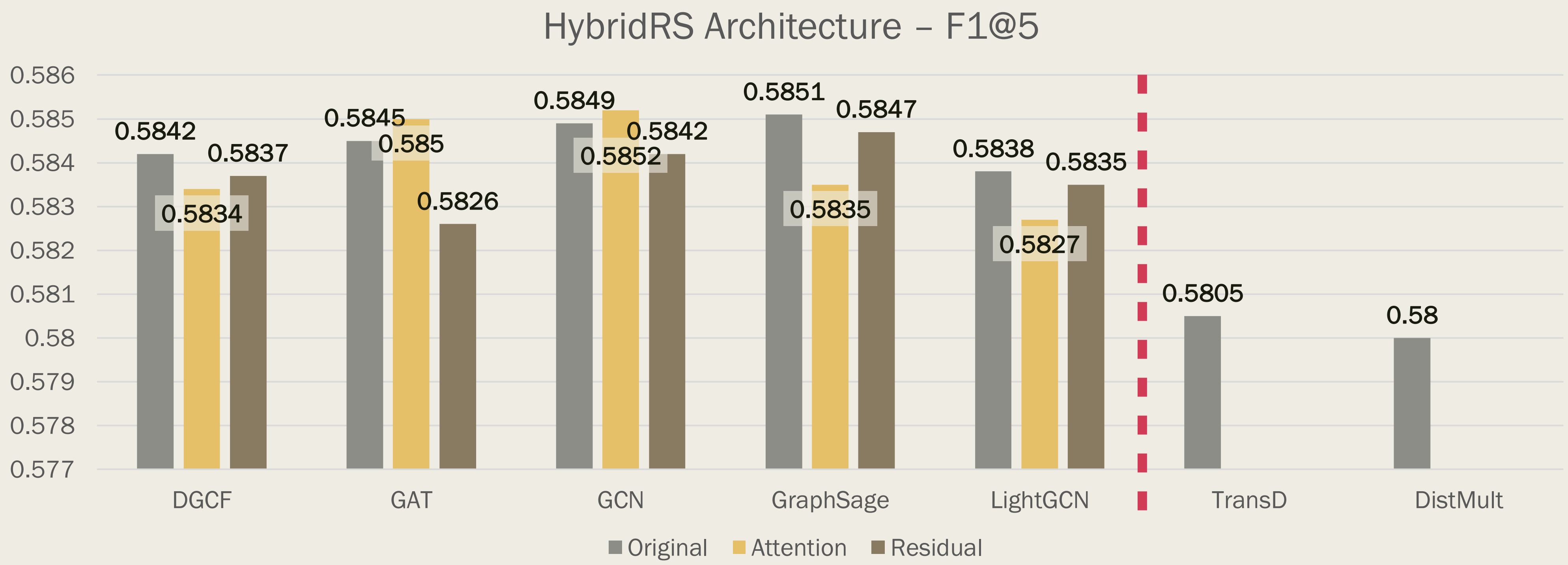
# Experiments

## Results - GNN / KGE comparison



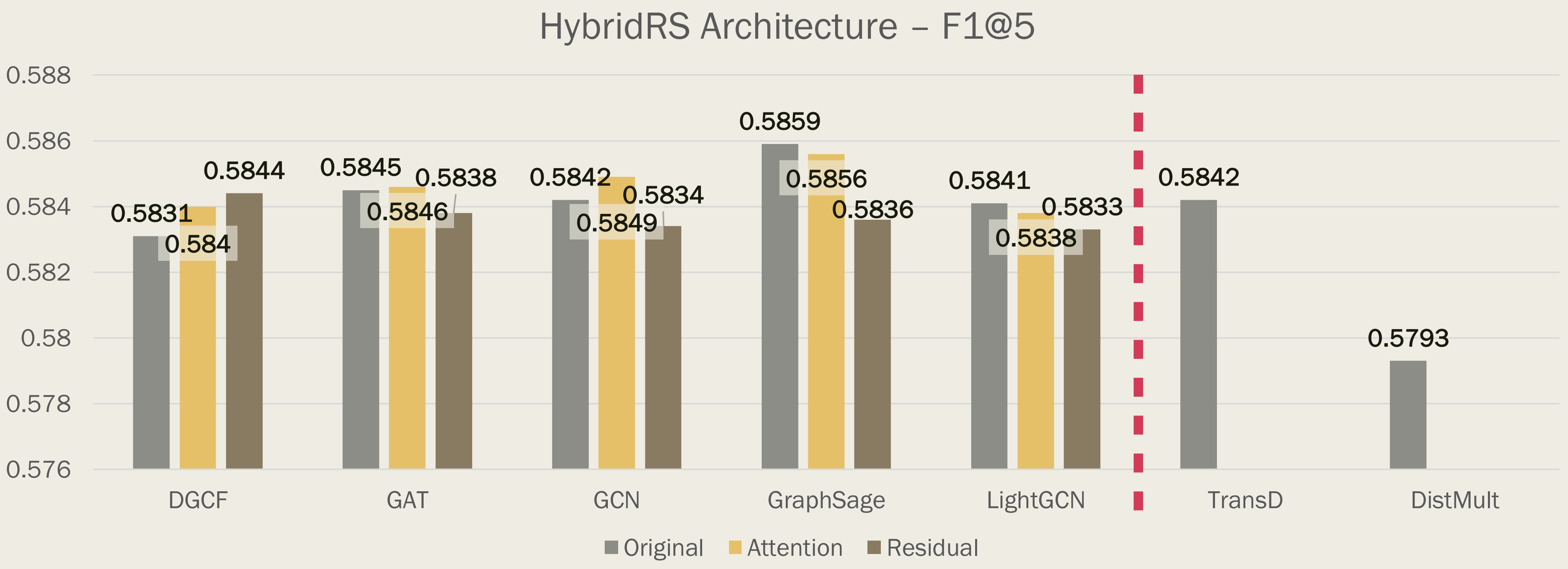
# Experiments

## Results – User-Item - Original / Tweaks comparison



# Experiments

## Results – User-Item-Properties (RS2) - Original / Tweaks comparison



# Conclusion

- Graph Neural Networks are good for **graph data** applied to recommendation tasks
- The learned embeddings are more **expressive**, with way less parameters
- It is possible to learn models in an end-to-end fashion

# Future Works

- Evaluate such models on more datasets with a richer set of properties
- Introduce a transformer-based model to learn items' content embeddings **jointly** with the rest of the model