A Spectral Analysis of Heterogeneous Graph Recommenders for Person-Job Fit

Haoyi Xiu[1,*], Shuya Nagayasu[1] and Norihiro Hizawa[1]

[1]Leverages Co., Ltd., 2-24-12 Shibuya, Shibuya Scramble Square 24F/25F, Shibuya-ku, Tokyo, 150-6190, Japan

Abstract

Person-job fit (PJF), the task of matching candidates to suitable positions, is a central problem in recruitment. Heterogeneous-graph-based recommenders have recently emerged as a promising direction for PJF, as they naturally alleviate cold-start and flexibly integrate the diverse signals available on recruitment platforms. Yet, to the best of our knowledge, no prior work has examined what such recommenders actually learn and the role the graph plays. To this end, we build an inductive heterogeneous graph neural network (GNN) recommender for PJF, train it on two real-world datasets across commonly used graph convolutions and large language model (LLM) text embeddings, and apply a spectral analysis to the trained models. The analysis shows that, first, the learned representations are highly collapsed. They occupy far fewer dimensions (a low effective rank) than their nominal size. Second, naively fusing content and structure yields a more collapsed representation than a structure-only model, for which we show that the decoupled modeling can provide a robust remedy. Third, message passing injects popularity signal into both seeker and job embeddings. While strongly predictive of mutual match, it disproportionately boosts seeker-side personalization, and projecting it out of the representation can sometimes even improve job-side personalization, demonstrating a personalization trade-off of a reciprocal matching market.

Keywords

Person-job fit, Heterogeneous graph neural networks, Spectral analysis, Effective rank, Popularity bias, Reciprocal recommendation

1. Introduction

Person-job fit (PJF) recommender systems match job seekers to suitable positions in an efficient and data-driven way [1, 2]. Unlike conventional recommendation, PJF is *reciprocal*: a match succeeds only when both sides agree. A recommendation is therefore realized through a funnel of two directed decisions. The seeker applies to a position, and the employer in turn passes the seeker [3, 4]. The domain is also rich in text, since both sides describe their capabilities, preferences, and requirements in natural language. Cold start is common as well [5]: new seekers and postings arrive continuously, and they must be served before they accumulate any interaction history.

These properties have made heterogeneous graph recommenders an emerging approach for PJF [6]. Casting the market as a graph over seekers, jobs, and their attributes has two advantages. First, message passing propagates evidence to sparsely observed nodes, which mitigates cold start. Second, a single model can fuse behavioral interactions, structured attributes, and free text [7, 8], where frozen large language model (LLM) embeddings supply the textual semantics. The standard design choice is to feed the content features *into* message passing as raw node inputs, so that structure and content are optimized together in one encoder [8, 9, 10]. This choice is largely inherited rather than examined.

Despite their growing adoption, graph PJF recommenders are validated almost only by their downstream metrics. As a result, what the trained model learns, and what the graph actually contributes, remain poorly understood. At the same time, a spectral literature has shown that learned representations tend to collapse onto far fewer dimensions than allotted [11], and that such collapse governs the behavior of collaborative filtering [12, 13, 14]. This spectral lens has not yet been applied to PJF, where the task is arguably harder: it is reciprocal, it runs on a heterogeneous graph, and it carries rich text features. We therefore ask two questions. First, **does pushing rich text embeddings through message passing help the representation, or harm it?** Second, when the graph helps, **does it help both sides of the market, or only one?**

We answer both questions with a controlled study. We build an inductive, entire-space multi-task (ESMM) [15] heterogeneous graph recommender. It has two directed heads that predict the seeker's and the job's preferences at the same time, and it scores the mutual match as the product of the two. We train it on two real-world datasets. We then compare four variants that hold everything fixed except *where content enters*. Finally, we analyze the trained representations with a spectral lens, and we study the capacity of the model and the role of the graph in performance and personalization. The analysis yields several findings. First, representations collapse consistently along the pipeline. Second, fusing content and structure naively hurts. Third, the graph's main contribution is a popularity signal, which is both predictive and a source of personalization trade-offs between the two sides. We believe these findings offer useful directions for future PJF recommender research.

Our contributions are:

- We construct an inductive, ESMM heterogeneous graph recommender for reciprocal PJF. We run systematic experiments on two real-world recruitment datasets, two common graph convolutions, and four LLM encoders, in order to study what the model learns and the role of the graph in it.
- We run a thorough spectral analysis. It shows that representations consistently collapse along the pipeline, that fusing content and structure naively hurts, and that the graph's main contribution is a popularity signal. This signal is both predictive and a source of personalization trade-offs between the two sides.
- We propose a simple and robust variant that keeps content out of message passing. It achieves strong

RecSys in HR '26: The 6th Workshop on Recommender Systems for Human Resources, in conjunction with the 20th ACM Conference on Recommender Systems, September 28–October 2, 2026, Minneapolis, MN, United States.
*Corresponding author.
hiroki.shu@leverages.jp (H. Xiu); shuya.nagayasu@leverages.jp (S. Nagayasu); norihiro.hizawa@leverages.jp (N. Hizawa)
0000-0002-2092-3879 (H. Xiu); 0000-0001-7033-8734 (S. Nagayasu); 0000-0002-2325-7170 (N. Hizawa)

performance while retaining much more capacity than the other variants.

Our code is publicly available to facilitate further research[1].

2. Literature review

2.1. Person-job fit systems

Neural network-based PJF is often modeled as a text-matching problem [1]. Le et al. [16] then formulate it as a two-sided matching problem in which each side is trained with triplet-based ranking. In addition, ideas such as profiling memory [17] and multi-view co-teaching [18] are introduced to effectively integrate user-job interactions and text semantics. The recent text-centric line has since adopted modern encoders. For instance, ConFit and ConFit v2 [19, 20] use contrastive resume–job matching with data augmentation and hard-negative mining. Du et al. [21] utilize LLMs and generative adversarial networks to refine the representation. MIRROR [4] uses transformers to extract features from different stages of the recruitment process for more precise matching. In addition, the joint modeling of multiple behaviors such as clicks and applications is also explored [22, 23].

Closest to this work is the line that models the recruitment market as a graph. DPGNN [9] builds a transductive graph over seekers and jobs and shows that modeling both active and passive intentions is effective for reciprocal matching. CSAGNN [7] and LiGNN [24] expand the graph further by incorporating professional connections and cross-service graphs to maximize the information used to build node representations. However, a transductive approach is not suitable for this type of marketplace, as new seekers and jobs keep arriving and leaving, which makes the cold-start problem acute. In contrast, inductive graph representation learning [25] can tackle this problem well. For instance, TIMBRE [10] builds a heterogeneous graph by incorporating nodes such as seekers, jobs, and skills into the graph topology to facilitate message passing. LinkSAGE [6] builds a billion-scale industrial graph and learns the inductive representation using GraphSAGE, which is shown to be effective across domains. Text-graph co-training [18] has also emerged as a promising direction, especially in the era of LLMs and text-rich PJF [8].

In this work, we build a heterogeneous graph largely following works such as LinkSAGE [6], because it can handle cold-start and integrate various information seamlessly. On top of that, we comprehensively analyze the content-structure interaction, where content is encoded with a *frozen LLM*, motivated by the rich text available, which has yet to be explored thoroughly in PJF research.

2.2. Spectral analysis of recommender systems

A separate line of work analyzes recommenders through the *spectrum* of their learned representations. Its measuring instrument is the effective rank [26]. Its central usage is to diagnose dimensional collapse [11], where some of the capacity of the representation is unused, resulting in suboptimal performance. Most recently, embedding collapse has been identified as the bottleneck when scaling recommendation models [12]. Loveland et al. [13] find that embeddings possessing higher rank are actually beneficial for improving embedding quality. Further, Peng et al. [14] demonstrate that directly optimizing the spectrum to force user and item embeddings to span the entire space is beneficial. As such, inspecting the spectrum of the learned representation can reveal various insights into the model's properties. Spectral analysis has also been used to understand popularity bias in recommender systems [27, 28], where popularity tends to dominate the representation, harming personalization.

However, no prior work has applied spectral analysis specifically to the PJF graph recommender, whose characteristics, such as reciprocity and text richness, differ significantly from the conventional collaborative filtering designed for one-sided marketplaces.

3. Method

3.1. Preliminaries

Heterogeneous graph. We cast PJF as link prediction over a *directed*, *heterogeneous* interaction graph $G = (V, E, \phi, \psi)$, where $\phi : V \to A$ assigns each node a type and $\psi : E \to R$ assigns each edge a relation, with $|A| + |R| > 2$. The node types $A = \{\text{seeker}, \text{job}, \text{attribute}\}$ comprise the two principal nodes, job seekers (seeker, set M) and postings (job, set J), together with the *attribute* nodes that describe them, such as skill and title.

The relation set R partitions into two families, inducing the edge partition $E = E_{\text{int}} \cup E_{\text{attr}}$ via ψ. *Interaction edges* E_{int} record behavioral events between seekers and jobs along the recruitment funnel: a seeker expresses interest in a job (like; on Tech only), a seeker applies to a job, and a job passes a seeker. *Attribute edges* E_{attr} connect a node to its descriptive attributes, e.g., a seeker has a skill or title and a job requires a skill or title. Because the graph is directed, every relation $r = (\text{src}, \text{rel}, \text{dst})$, where src, rel, dst refer to source, relation, and destination, is paired with a reverse relation $r^{-1} = (\text{dst}, \text{rel}^{-1}, \text{src})$, so that messages propagate in both directions while edge direction remains distinct.

Graph neural networks. A graph neural network (GNN) learns node representations by iterative *message passing*. At layer ℓ, each node updates its state by aggregating transformed messages from its neighbors,

$$h_i^{(\ell+1)} = \text{UPD}\Big(h_i^{(\ell)}, \bigoplus_{j \in N(i)} \text{MSG}\big(h_i^{(\ell)}, h_j^{(\ell)}\big)\Big), \qquad (1)$$

where $N(i)$ denotes the neighbors of node i and $\bigoplus$ is a permutation-invariant aggregator; stacking L layers propagates information across L-hop neighborhoods. Using no per-node identity parameters keeps the encoder inductive.

A single, shared transformation is ill-suited to a heterogeneous graph, whose relations carry qualitatively different semantics. We therefore adopt a *relational* heterogeneous GNN in the R-GCN framework [29], which extends the message passing above with *relation-specific* parameters:

$$h_i^{(\ell+1)} = \rho\Big(W_0^{(\ell)} h_i^{(\ell)} + \sum_{r \in R} \sum_{j \in N_r(i)} \frac{1}{c_{i,r}} W_r^{(\ell)} h_j^{(\ell)}\Big), \qquad (2)$$

where $N_r(i)$ are the neighbors of i under relation r, $W_r^{(\ell)}$ a per-relation weight, $c_{i,r}$ a normalization constant, $W_0^{(\ell)}$ a

[1] https://github.com/lvgs-research/spectral_pjf

self-loop transform, and ρ a nonlinearity. This form corresponds to the mean-aggregation (GraphSAGE) operator; the attention operator (GATv2) replaces the fixed $1/c_{i,r}$ with a learned, input-dependent coefficient $\alpha_{ij}^{(r)}$. Every relation in R thus carries its own message function, and the resulting per-relation representations are combined at each node. This relational GNN is our encoder backbone throughout; the specific per-relation operator is a design choice we study empirically (Section 4.2).

Effective rank. Our analysis quantifies how many directions a learned representation actually uses through its *effective rank* [26]. For a representation matrix $Z \in \mathbb{R}^{n\times d}$ (a stack of n node or edge embeddings), let $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_q \geq 0$ with $q = \min(n, d)$ be its singular values from the singular value decomposition (SVD), and form the normalized spectrum $p_k = \sigma_k / \sum_{j=1}^{q} \sigma_j$. The effective rank is defined as the exponential of the Shannon entropy of this distribution,

$$\mathrm{erank}(Z) = \exp\Big(-\sum_{k=1}^{q} p_k \log p_k \Big). \tag{3}$$

Taking values in $[1, q]$, it equals 1 when the spectrum collapses onto a single direction (maximal anisotropy) and q when all singular values are equal (isotropy). Unlike the algebraic rank, it is continuous and insensitive to negligible singular values, and hence describes the ranks that are actually doing anything. This fact makes it a natural measure of representational collapse and of how content and structure combine in the learned embeddings. In all analyses Z is column-mean-centered before the SVD.

3.2. Multi-task inductive link prediction

PJF is *reciprocal*, meaning that a successful match requires interest from both sides. We therefore frame it as *multi-task* [15] *inductive link prediction* and predict two directed link types jointly. Specifically, we jointly model the real-world recruitment process that describes whether a seeker applies to a job (seeker → applies → job) and whether the job then passes that seeker (job → pass → seeker). For an exposed seeker–job pair (u, v), we write $y_{uv}^{a} \in \{0, 1\}$ for the apply outcome and $y_{uv}^{p} \in \{0, 1\}$ for the pass outcome. Note that we model the *candidate-driven recruitment process* where only seekers can initiate, and enterprises respond. Therefore, $y_{uv}^{p} := 0$ when $y_{uv}^{a} = 0$, and y^{p} enters the objective only through the product $y^{a}y^{p}$. The task is to learn a scorer for both. The problem is *inductive*: the model uses no per-node identity embeddings and is evaluated on seekers and jobs disjoint from those seen during training. Therefore, it must generalize to unseen (cold-start) nodes from features and structure alone. We instantiate the scorer as a GNN encoder feeding a multi-layer perceptron (MLP) decoder, trained with an ESMM objective. Figure 1 gives an overview of the system and of the four encoder variants compared in our experiments.

GNN encoder. The encoder takes each node's *content* features, which are frozen text embeddings of the seeker/job profile text concatenated with tabular attributes, as its raw input representation x_i, and produces a contextualized embedding z_i by L rounds of the relational message passing,

$$z_i = \mathrm{GNN}_\theta\big(x_i,\ G\big), \tag{4}$$

so that z_i blends a node's own content and its typed neighborhood along interaction and attribute edges. As no ID embeddings are used, z_i is well defined for nodes never seen in training.

MLP decoder. The decoder scores an ordered pair (u, v) by passing the concatenation of the two node embeddings and a vector m_{uv} of *content-match* features, e.g., the number of skills shared by the seeker and the job, and the cosine similarity of their text embeddings, through two *independent* task-head MLPs applied to the same input $h_{uv} = [\, z_u \,\|\, z_v \,\|\, m_{uv} \,]$,

$$s_{uv}^{a} = \mathrm{MLP}_{\omega_a}(h_{uv}), \qquad s_{uv}^{p} = \mathrm{MLP}_{\omega_p}(h_{uv}), \tag{5}$$

where θ and $\omega = \{\omega_a, \omega_p\}$ are the trainable parameters of the encoder and decoder. We prefer an MLP to the usual inner-product decoder for two reasons. First, it is strictly more expressive than the inner product $z_u^{\top} z_v$. Second, it can encode asymmetry, which is precisely what a directed, reciprocal task demands.

Entire-space multi-task objective. The pass outcome is observed only *after* an application, so a pass head trained on applied pairs alone would suffer sample-selection bias and severe data sparsity. Following the ESMM paradigm [15], we instead supervise over the full exposure space $\Omega \subseteq M \times J$ of exposed pairs. With $p_{uv}^{a} = \sigma(s_{uv}^{a})$ the apply probability and $p_{uv}^{p|a} = \sigma(s_{uv}^{p})$ the conditional pass probability, where $\sigma(\cdot)$ is the sigmoid function, the end-to-end match probability factorizes along the funnel as

$$p_{uv}^{a\wedge p} = P(\mathrm{apply} \wedge \mathrm{pass} \mid u, v) = p_{uv}^{a} \cdot p_{uv}^{p|a}, \tag{6}$$

and we minimize the joint cross-entropy of the two entire-space tasks,

$$\mathscr{L} = \frac{1}{|\Omega|} \sum_{(u,v)\in\Omega} \Big[\mathrm{BCE}\big(y_{uv}^{a},\ p_{uv}^{a}\big) + \mathrm{BCE}\big(y_{uv}^{p},\ p_{uv}^{a\wedge p}\big) \Big], \tag{7}$$

where BCE denotes the binary cross-entropy. Because both terms range over all exposed pairs, the conditional pass head is learned without restricting the loss to the apply-selected subpopulation. Beyond mitigating selection bias and data sparsity, this decomposition mirrors the real recruitment funnel (exposure → apply → pass). Note that we do not claim that ESMM removes selection bias entirely, as its entire-space estimate is itself known to remain biased [30], which currently is an active area of research.

Per-head calibration. Multiplying two stage probabilities is meaningful only when each is well-calibrated, so we attach a LiRank-style calibration layer [31] to every head: a co-trained, monotone (isotonic) piecewise-linear map applied to the head's logit before the sigmoid. Being monotone it is rank-preserving, which keeps the funnel product $p_{uv}^{a} \cdot p_{uv}^{p|a}$ a faithful estimate of the joint match probability. The layer is identity-initialised, so it departs from the plain decoder only as training warrants.

Temporal constraint. When a dataset provides action timestamps, we additionally impose a strict point-in-time (PIT) constraint that mirrors deployment. Specifically, scoring an interaction anchored at time t may use *only* nodes

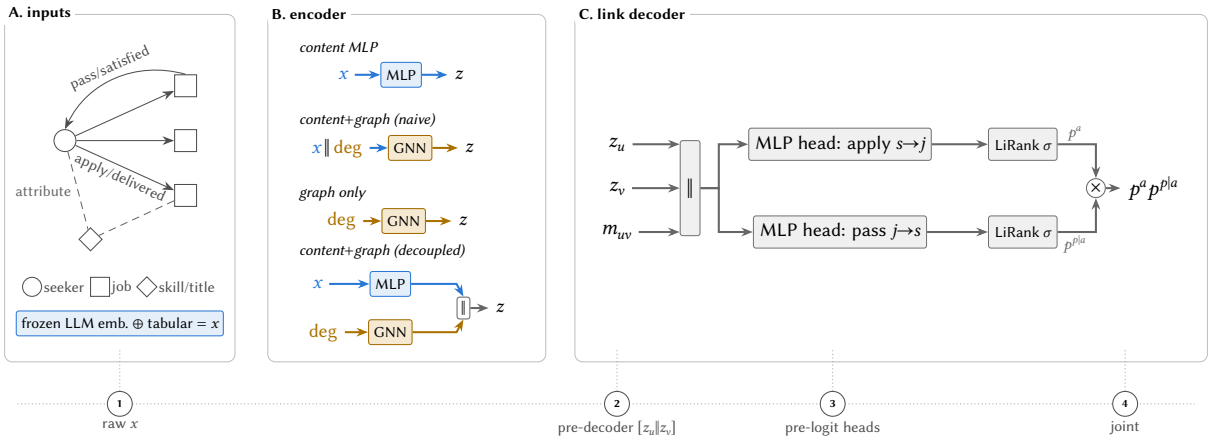

Figure 1: Overview of the constructed graph recommender. **A:** a directed heterogeneous graph over seekers, jobs, and attributes (interaction edges solid, attribute edges dashed). The content feature x includes frozen LLM text embeddings and tabular features, which is basically the remaining attributes excluding the skill and title. The structural features deg consist of interaction and attribute degree counts. **B:** the four encoder variants that model content and structure differently. **C:** The link decoder, which takes the pair of node embeddings and match features to predict seeker-side, job-side, and mutual scores. Circles 1–4 mark the representation stages traced in the spectral analysis.

and edges that exist strictly before t. Writing $\tau(e)$ for the event time of an interaction edge and $\beta(\mathrm{src}(e))$ for the birth time of a node $\mathrm{src}(e)$ ($\mathrm{src}(e)$ denotes the source node of edge e), the visible edge set is

$$E_{<t} = \{e \in E_{\mathrm{int}} : \tau(e) < t\} \cup \{e \in E_{\mathrm{attr}} : \beta(\mathrm{src}(e)) < t\}, \tag{8}$$

i.e. interaction edges are gated by their event time and profile edges by the birth of the focal node they describe (an attribute is visible once its focal node exists), while the focal nodes, that is, seekers and jobs, being scored retain their own profile edges. The strict inequality ensures that neither future information nor the target edge itself leaks into its own prediction. The encoder then operates on the subgraph $G_{<t}$ induced by $E_{<t}$, i.e., $z_i = \mathrm{GNN}_\theta(x_i, G_{<t})$ for pairs anchored at t. On datasets without timestamps this constraint is simply ignored.

4. Setup

4.1. Graph construction

We evaluate on two real-world person-job-fit datasets. The first, denoted **Tech**, is a large proprietary dataset from an industrial recruitment platform. It logs the full recruitment funnel together with seeker profiles, job descriptions, and timestamped actions. Note that the dataset covers only part of the platform's entire seeker and job population. The second is the public **Tianchi** Zhaopin dataset [32], a widely used benchmark of seeker–job interactions. The two differ in scale, in label semantics (screening *pass* on Tech vs. *satisfied* on Tianchi), and, crucially, in the availability of action timestamps (present on Tech, absent on Tianchi), which lets us test whether our findings hold across settings.

From each dataset we build the directed, heterogeneous graph by largely following the construction protocol popularized by industrial graph recommenders [24, 6]. Seekers and jobs become focal nodes, and skills and titles (and, on Tech, companies) become attribute nodes shared across focal nodes. Behavioral events become *interaction* edges along the funnel. Specifically, light interest (like; Tech only), apply, and pass. Profile fields become *attribute* edges (has/requires a skill/title). Each relation is complemented by a reverse relation, and the skill and title vocabularies are canonicalized so that attribute nodes are shared consistently across focal nodes. On Tech, every edge carries a timestamp (interaction edges the event time; attribute edges the birth time of the focal node they describe), which the PIT constraint of Section 3.2 uses to build leak-free snapshots. On Tianchi, which lacks action timestamps, the graph is static and the temporal constraint is omitted. Node inputs are the raw content features: frozen text embeddings of the seeker/job text (we compare several LLM encoders) concatenated with standardized tabular attributes. Both datasets provide explicit positive and negative outcomes for the scored pairs, and both are used as supervision for link prediction in this study. No random negative sampling is involved. Table 1 reports the node and edge statistics of the two resulting

Table 1
Node and edge statistics of the two person-job-fit graphs. The light-interest relation like is unique to Tech; Tianchi's browse events define the exposed candidate pool and are not a message-passing relation. The second-stage outcome relation is pass on Tech and satisfied on Tianchi; Tianchi has no company nodes. Exposure is a property of the logged recommendation rows (the scored pairs), not an edge type.

		Tech	Tianchi
Nodes	seekers	13,000+	4,500
	jobs	37,000+	91,479
	skills	200+	1,618
	titles	70+	580
	companies	3,000+	—
Interaction edges	like	150,000+	—
	apply/delivered	180,000+	61,293
	pass / satisfied	69,000+	28,594
Attribute edges	seeker–skill	100,000+	160,159
	seeker–title	12,000+	2,766
	job–skill	390,000+	1,276,927
	job–title	52,000+	80,295
	job–company	37,000+	—

graphs.

Train–validation split. On Tech the split is seeker-disjoint over the global timeline, in which 70% of seekers form the training set, 20% validation, and 10% test, so no seeker appears in more than one split, that is, all seekers are cold-start by construction, which reflects the real-world recruitment process. Within each split, supervision is organized chronologically into PIT chunks, and test pairs are scored against exact one-day snapshots. In other words, a seeker–job pair can see all interactions that happened a day before its own interaction. Following standard practices in the link prediction and recommender systems literature, on Tianchi, which lacks timestamps, we instead hold out seekers as cold cohorts (15% for validation, 15% for test), where cold seekers are defined as those who never appear in the train set. The remaining warm seekers have their pairs split 60/20/20 across train, validation, and test. To prevent leakage through the static Tianchi graph, a seeker's supervision outcome edges are hidden from the message-passing graph of the split in which they are scored. Warm seekers' validation and test pairs are held out in full, with their training edges left in as context. The training split and the cold validation/test cohorts instead keep 65% of each seeker's outcome edges as context and hold out the remaining 35% as targets, so cold seekers, who have no training edges, retain a partial neighborhood at inference. We note that, as how the two datasets are split differs, the conclusions may be confounded with the splitting strategy, mainly due to the lack of accurate timestamps in the Tianchi dataset.

4.2. Graph convolutions and LLM embeddings

We hold the encoder–decoder pipeline fixed and vary two components: the graph-convolution operator and the frozen text encoder. For the convolution we compare two widely used choices. **GraphSAGE** [25] aggregates a node's neighborhood with a permutation-invariant mean and combines it with the node's own state through learned linear maps. **GATv2** [33] instead uses dynamic attention, weighting each neighbor by a learned, input-dependent score before aggregation. Both are homogeneous operators; we lift each to our heterogeneous graph in the R-GCN style of Section 3.1, instantiating one operator per relation type (including the untied reverses) and summing the per-relation messages at each node. This isolates the effect of the aggregation operator from that of the heterogeneous relational wiring, which is shared across both baselines.

For the content features we compare four frozen LLM text-embedding models: the Qwen3-Embedding family at three scales (0.6B, 4B, and 8B) [34], which lets us test whether the findings depend on embedding-model capacity, and multilingual-e5-large [35], a widely used encoder from a different model family. Each encodes the seeker and job texts once; the embeddings are kept frozen throughout training.

4.3. Compared variants

To isolate how content and graph structure interact, we compare four variants that hold the decoder, objective, and training protocol fixed and vary only where the content enters (Figure 1, part B):

Content MLP no message passing at all. The frozen text embeddings and tabular features feed the MLP decoder directly. This is the content-only reference.

Content+graph (naive) the standard heterogeneous GNN recipe. Content features are used as the raw node representations *inside* message passing, so structure and content are optimized jointly through the same encoder.

Graph only content features are withheld from the encoder. Nodes carry only degree-based structural features, so the variant measures what message passing alone provides.

Content+graph (decoupled) content and structure are encoded in *separate* channels and fused late by concatenation at the decoder, so each one is optimized without interfering with the other.

4.4. Metrics

We report two complementary ranking metrics, both computed on *real* positive and negative pairs from the logged exposure pool (no sampled random negatives, which are known to inflate and distort ranking scores [36]).

The first metric is *AUC*, the area under the ROC curve over all scored pairs, measuring how well the model separates successful from unsuccessful outcomes across the whole population. Therefore, it rewards any globally discriminative signal, which includes popularity, so it is our headline discrimination metric but says little about ranking quality *within* a user or a job.

To counter this, we use *GAUC* as the second metric. The grouped AUC reports AUC of pairs grouped by seeker (seeker-side GAUC) or by job (job-side GAUC), and the unweighted mean over groups is reported (groups lacking both a positive and a negative are skipped). Because grouping removes between-group effects such as job popularity, through GAUC we can analyze *personalization*, which says whether the model orders the candidates of one seeker, or the applicants of one job, better than chance. Reporting it on both sides makes the two directions of this reciprocal task separately measurable. Additional grouped ranking metrics (MAP@k and Hit@k on both sides) are reported in Appendix A.

4.5. Implementation details

All variants share one training protocol. The encoder uses hidden width 128 and two message-passing layers with dropout probability set to 0.2. The decoupled variant's two channels are fused into a 256-dimensional pair representation, and GATv2 uses a single attention head with a residual connection. Models are trained with Adam (learning rate 5×10^{-3}), with early stopping on validation AUC (Tech: up to 12 epochs, patience 5; Tianchi: up to 250 epochs, patience 50; both tuned from prior experiments). Frozen text embeddings are used as-is (1,024–4,096 dimensions, depending on the encoder). All reported numbers average five seeds (0–4); each run fits on a single 24 GB GPU. Please refer to the released code for further details.

Table 2
Performance of the content-placement variants (Section 4.3) across graph convolutions and frozen text encoders (5-seed mean). On Tech the split is seeker-disjoint, so no separate seeker-cold slice is defined by construction. The content MLP does not message-pass, so its rows are identical under both convolutions and vary only with the encoder. **bold** = best, underline = second best per column *within each block*; * = the best value is significantly better than the second best (two-sided paired t-test over the five seeds, $p < 0.05$).

	Tech					Tianchi				
Model	AUC	Seeker cold AUC	Job cold AUC	Seeker GAUC	Job GAUC	AUC	Seeker cold AUC	Job cold AUC	Seeker GAUC	Job GAUC
SAGE · Qwen3-0.6B (main)										
content MLP	.673	—	.651	.632	**.573***	.685	.671	.654	.640	.526
content+graph (naive)	.683	—	.674	.648	.551	.730	.720	.653	.687	.529
graph only	**.689**	—	**.676**	**.655**	.549	.727	.713	.644	.680	.546
content+graph (decoupled)	.688	—	.663	.654	.556	**.746***	**.739***	**.678***	**.699***	**.552**
GATv2 · Qwen3-0.6B										
content MLP	.673	—	.651	.632	**.573***	.685	.671	.654	.640	.526
content+graph (naive)	.682	—	.668	.646	.550	.733	.725	.656	.689	.535
graph only	**.689**	—	**.678**	.654	.548	.726	.718	.643	.678	.540
content+graph (decoupled)	.689	—	.673	**.655**	.555	**.744***	**.733***	**.675***	**.698***	**.559**
SAGE · Qwen3-4B										
content MLP	.675	—	.658	.638	**.572**	.692	.685	.656	.654	.545
content+graph (naive)	.684	—	.674	.649	.553	.735	.727	.660	.693	.523
graph only	.689	—	**.682**	.656	.550	.727	.717	.645	.681	.542
content+graph (decoupled)	**.692***	—	.665	**.657**	.567	**.747***	**.736**	**.679***	**.701***	**.569***
SAGE · Qwen3-8B										
content MLP	.681	—	.668	.642	**.583***	.693	.689	.660	.655	.538
content+graph (naive)	.689	—	**.681**	.654	.563	.734	.725	.660	.691	.530
graph only	.689	—	.680	.656	.551	.728	.716	.645	.681	.544
content+graph (decoupled)	**.694***	—	.669	**.656**	.564	**.749***	**.743***	**.683***	**.708***	**.556**
SAGE · multilingual-e5-large										
content MLP	.668	—	.643	.624	**.570***	.692	.686	.658	.652	.542
content+graph (naive)	.688	—	.671	.651	.555	.718	.700	.639	.677	.531
graph only	.687	—	.672	.652	.546	.728	.719	.645	.682	.543
content+graph (decoupled)	**.692***	—	**.675**	**.655**	.556	**.736***	**.723**	**.661**	**.693***	**.549**

5. Results

Table 2 reports ranking performance for the controlled content-placement family of Section 4.3, across both graph convolutions and all four frozen text encoders, on both datasets. Unless noted otherwise, we describe the result of the main SAGE · Qwen3-0.6B block. Introducing the graph improves the global AUC and the seeker-side GAUC on both datasets (content MLP → graph variants), and content+graph (decoupled) attains the best joint AUC on Tianchi. The job-side GAUC, by contrast, stays notably lower than the seeker-side in every variant. On Tech it is highest for the content MLP, and on Tianchi the decoupled variant is the best. We conjecture that the systematically lower job-side ranking may be partly dataset-induced, as in both datasets the job side is observed only *reacting* to applicants and never initiates an action of its own, resulting in significant data sparsity.

Representational collapse along the pipeline. Figure 2 traces the effective rank of the representation of the scored test pairs through the pipeline on both datasets (SAGE, Qwen3-0.6B). Based on the figure, we can see that, first, every trained variant is *highly collapsed* (see [14] for a detailed description of representation collapse). The pre-decoder embeddings compress the content-only effective rank by one to two orders of magnitude. Interestingly, the graph-only variant starts from low-rank degree features and expands instead. The decoder then partially recovers the rank. Second, the naive content+graph variant carries the lowest rank *among the graph variants* at every learned stage on both datasets. Moreover, on Tianchi it even falls below the content MLP, despite consuming strictly more input signal. Therefore, pushing content through message passing may actively hurt the model capacity, suggesting suboptimal model design. The decoupled content+graph variant maintains the highest learned-stage rank among the graph variants on both datasets, consistent with its ranking performance in Table 2. Further, the decoupled modeling achieves gains consistently from the encoder scaling, which is disabled for the graph only variant by construction. Therefore, preserving the most spectral capacity while ranking at or near the top in every setting, decoupled modeling is the robust default baseline we recommend for future research. We note that the aforementioned findings are consistent with prior spectral analyses of collaborative filtering, which report highly collapsed ranks and show that recovering rank empirically improves performance [12, 13, 14].

Graph encodes one-sided popularity. We define the *popularity* of a seeker (job) as its count of positive interactions, that is, the sum of its interaction-edge degrees (like, apply, pass on Tech, and delivered, satisfied on Tianchi). Correlations use log-scaled counts. Table 3 decomposes the pre-decoder edge embedding by side on both datasets.

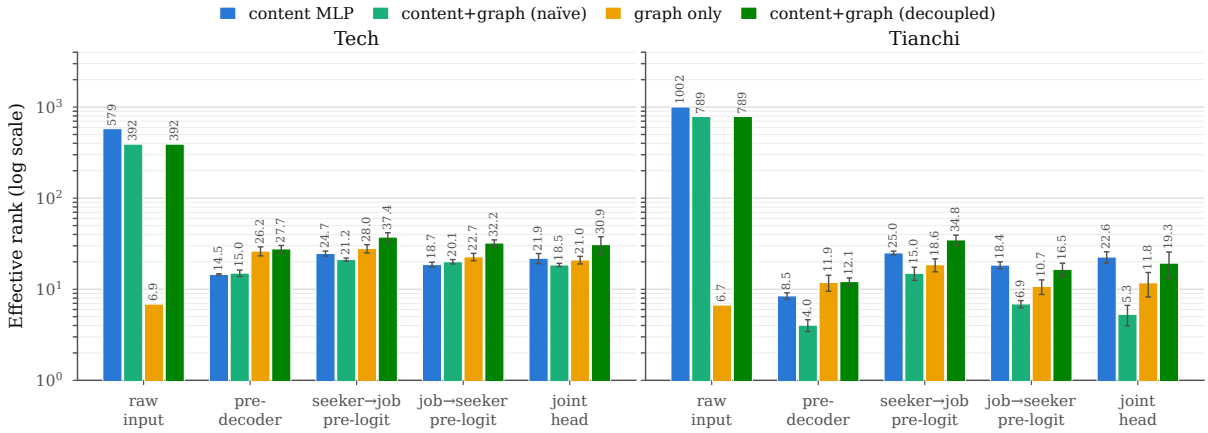

Figure 2: Effective rank (log scale) of the scored-pair representation at each pipeline stage on Tech (left) and Tianchi (right) (SAGE, frozen Qwen3-0.6B). Value labels give the mean.

The graph-only and decoupled variants roughly double the per-focal node effective rank of the content MLP on both sides; the naive variant does not (on the Tianchi job side its rank even falls). The energy concentration ($\sigma_1^2 / \sum_{j=1}^{q} \sigma_j^2$, the share of the centered spectral energy carried by the top principal component) shows all variants remain dominated by a single direction (PC1 carrying 51–97% of the variance). Correlating the top two principal coordinates with popularity reveals what those directions encode. On Tianchi the graph writes popularity straight into the dominant axis. The seeker-side popularity $|r|$ rises from .07 (content MLP) to .51–.57 for the graph-only and decoupled variants. PC2 stays low ($\leq .16$), and the job side reaches .70 (decoupled). This echoes prior findings that recommender representations memorize item popularity in their principal spectrum [27]. On Tech the signal is present but shared across PCs. Seeker-side correlations are not measurable because the split is seeker-disjoint, i.e., all seekers are cold by definition. On the job side, PC1 alignment is weak (.17–.26), but PC2 alignment is stronger for the graph-only (.41) and decoupled (.34) variants. This popularity signal is strongly predictive for ranking *candidates for a seeker*, but provides weaker benefits for the job side according to the job-side GAUC in Table 2. We conjecture that the strong injection of the job popularity signal, as evidenced by the generally higher pop. $|r|$ on the job side, makes the job representation discriminative with respect to a seeker. In contrast, the seeker-side representation generally shows lower alignment to the popularity signal provided by the graph, resulting in low discriminativeness of seekers given a job. Therefore, the popularity signal is one-sided, favoring seeker-side personalization.

Removing popularity. We ask: if the graph's main contribution is a popularity signal, what happens when it is taken away? Table 4 reports the ablation for the decoupled content+graph variant: we project the first (and/or second) principal direction out of the trained pre-decoder embeddings on each side, re-apply the *trained* decoder heads, and re-evaluate. As shown in Table 4, on Tianchi, where popularity is concentrated in PC1, removing that single axis is catastrophic for the seeker direction while the effect of removing PC2 is relatively small. On Tech, removing PC1 costs less. Interestingly, it even *improves* the job-side GAUC: removing PC1 gives a +.019 gain, and removing both PC1 and PC2 further recovers +.030, surpassing even the best job GAUC achieved by the content-only baseline (Table 2). Strikingly, the cold-job GAUC benefits even more. Therefore, the popularity-aligned directions induced by the graph actively benefit seeker-side personalization while hurting the job side, showcasing the reciprocal market trade-offs.

Robustness across convolutions and encoders. Table 2 covers five settings: both convolutions (SAGE, GATv2) at the main encoder, and four text encoders (Qwen3-0.6B/4B/8B, multilingual-e5-large) under SAGE. The observed pattern is stable in every block. The decoupled content+graph variant sweeps most of the reported metrics and attains the best or near-best performance. The content MLP retains the best job-side GAUC on Tech. The spectral ordering is equally stable, in which the naive variant has the lowest pre-decoder effective rank among the graph variants, and the decoupled variant the highest, in every convolution and encoder cell.

6. Conclusions

We asked what an inductive heterogeneous graph recommender for PJF actually learns, and answered with a spectral, representation-level study across two real-world datasets, four content-placement variants, two graph convolutions, and four LLM text encoders. Three findings emerged. First, every trained variant is severely collapsed. Second, feeding content *into* message passing is rank-destructive, which yields the lowest effective rank among the graph variants. We show that this can be fixed by decoupling content and structure, which preserves the most spectral directions and ranks strongest. Third, the graph's dominant contribution is a popularity signal, which is strongly predictive. On the other hand, this signal disproportionately favors seeker-side personalization, while projecting it out improves job-side personalization in some cases, showcasing the reciprocal trade-offs of matching markets.

Future work. Three directions follow naturally. First, moving beyond late concatenation, i.e., the decoupled variant proposed by this study, toward a principled fusion of content and graph structure that lets the two modalities interact without the rank destruction of naive coupling.

Table 3
Per-side spectral diagnostics of the pre-decoder embeddings on both datasets (SAGE, Qwen3-0.6B). erank = effective rank of the unique seeker/job embedding. PC1 energy = fraction of the centered spectral energy (variance) in the top principal component. pop. $|r|$ = |Pearson| correlation between the first (PC1) or second (PC2) principal coordinate and the focal node's popularity. On Tech the split is seeker-disjoint, so every test seeker has zero historical interaction degree and the seeker-side correlations are undefined (—).

	Seeker				Job			
			pop. $\|r\|$				pop. $\|r\|$	
Variant	erank	PC1 energy	PC1	PC2	erank	PC1 energy	PC1	PC2
Tech								
content MLP	7.0	.65	—	—	8.2	.53	.26	.11
content+graph (naive)	6.9	.82	—	—	8.4	.54	.23	.21
graph only	13.1	.62	—	—	13.9	.52	.17	.41
content+graph (decoupled)	12.9	.55	—	—	16.0	.51	.20	.34
Tianchi								
content MLP	3.9	.97	.07	.02	5.0	.80	.26	.15
content+graph (naive)	4.2	.85	.14	.16	3.3	.97	.59	.39
graph only	6.7	.88	.51	.13	6.7	.84	.59	.44
content+graph (decoupled)	7.1	.81	.57	.09	7.3	.86	.70	.33

Table 4
Effect of projecting out the first (PC1), second (PC2), or both (PC1+2) principal directions from the pre-decoder embeddings of the content+graph (decoupled) variant. Decreases are shown in dark red, increases in dark green.

Dataset		AUC	Seeker GAUC	Job GAUC	Cold-job GAUC
Tech	baseline	.688	.654	.556	.525
	−PC1	.633 (−.056)	.598 (−.056)	.575 (+.019)	.569 (+.045)
	−PC2	.684 (−.004)	.647 (−.007)	.561 (+.005)	.539 (+.014)
	−PC1+2	.616 (−.073)	.580 (−.074)	.586 (+.030)	.593 (+.068)
Tianchi	baseline	.746	.699	.552	.544
	−PC1	.585 (−.160)	.530 (−.169)	.525 (−.026)	.544 (±.000)
	−PC2	.735 (−.011)	.692 (−.006)	.549 (−.003)	.551 (+.007)
	−PC1+2	.571 (−.174)	.527 (−.171)	.525 (−.027)	.542 (−.002)

Second, performing thorough analysis on the popularity signal and handling it for both sides explicitly. Isolating it as a modeled component would open the door to fairer and more controllable recommendation, for example, tempering popularity on the job side while retaining it where it genuinely ranks. Third, jointly training the LLM embedding model with the graph model, rather than freezing the text encoder. Whether end-to-end adaptation of the content representation alleviates or worsens the content–structure incompatibility is an open question worth investigating for a more accurate mutual match.

Limitations. First, both datasets capture only one direction of initiation: the seeker acts (apply on Tech, delivered on Tianchi) and the job side merely reacts (pass or satisfied). There is no active outreach from the job side and no passive or implicit signal on the seeker side, so the job→seeker head is trained on purely reactive labels. This one-sided funnel limits how far we can probe reciprocity. A dataset with genuine two-way initiation would enable stronger conclusions. Second, the mutual-match target we predict is a mid-funnel proxy, not the ultimate hire. Mutual interest does not guarantee an offer, an acceptance, or a completed placement, so our conclusions about the role of the graph concern the ranking of mutual interest and need not transfer directly to the final hiring decision. Third, we hold the graph construction fixed and vary only where content enters and the choice of convolution and encoder. We do not study alternative graph-structure designs, such as different edge sets, higher-order or hyperedge connections, or learned or pruned topologies. Because message passing depends strongly on structure, these choices could substantially change both the learned geometry and the reciprocity behavior. Fourth, while the relationship between effective rank and ranking performance is established in one-sided collaborative filtering [14], we do not fully characterize it in the reciprocal setting.

Declaration on Generative AI

During the preparation of this work, the author(s) used Claude (Anthropic) in order to: assist with drafting and editing the manuscript text, and assist with coding. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] C. Zhu, H. Zhu, H. Xiong, C. Ma, F. Xie, P. Ding, P. Li, Person-Job Fit: Adapting the right talent for the right job with joint representation learning, ACM Transactions on Management Information Systems (TMIS) 9 (2018) 12:1–12:17. doi:10.1145/3234465.

[2] Y. Mashayekhi, N. Li, B. Kang, J. Lijffijt, T. D. Bie, A challenge-based survey of E-recruitment recommendation systems, ACM Computing Surveys 56 (2024) 1–33. doi:10.1145/3659942.

[3] X. Hu, Y. Cheng, Z. Zheng, Y. Wang, X. Chi, H. Zhu, BOSS: A bilateral occupational-suitability-aware recommender system for online recruitment, in: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2023, pp. 4146–4155. doi:10.1145/3580305.3599783.

[4] Z. Zheng, X. Hu, S. Gao, H. Zhu, H. Xiong, MIRROR: A multi-view reciprocal recommender system for online recruitment, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), 2024, pp. 543–552. doi:10.1145/3626772.3657776.

[5] Éric Behar, J. Romero, A. Bouzeghoub, K. Wegrzyn-Wolska, Tackling cold start for job recommendation

with heterogeneous graphs, in: Proceedings of the 3rd Workshop on Recommender Systems for Human Resources (RecSys in HR 2023), CEUR Workshop Proceedings Vol-3490, volume 3490, 2023.

[6] P. Liu, H. Wei, X. Hou, J. Shen, S. He, K. Q. Shen, Z. Chen, F. Borisyuk, D. Hewlett, L. Wu, S. Veeraraghavan, A. Tsun, C. Jiang, W. Zhang, LinkSAGE: Optimizing job matching using Graph Neural Networks, in: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1 (KDD), 2025, pp. 2448–2457. doi:10.1145/3690624.3709396.

[7] H. Chen, L. Du, Y. Lu, Q. Fu, X. Chen, S. Han, Y. Kang, G. Lu, Z. Li, Professional Network Matters: Connections empower Person-Job Fit, in: Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM), 2024. doi:10.1145/3616855.3635852.

[8] P. Liu, R. Arora, X. Shi, B. Le, Q. Shen, J. Shen, C. Jiang, N. Zhiltsov, P. Bannur, Y. Zhu, L. Dong, H. Wei, Q. Guo, L. Simon, L. Hong, W. Zhang, A scalable and efficient signal integration system for job matching, in: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD), 2025. doi:10.1145/3711896.3737185.

[9] C. Yang, Y. Hou, Y. Song, T. Zhang, J.-R. Wen, W. X. Zhao, Modeling Two-Way selection preference for Person-Job Fit, in: Proceedings of the 16th ACM Conference on Recommender Systems (RecSys), 2022. doi:10.1145/3523227.3546752.

[10] E. Behar, J. Romero, A. Bouzeghoub, K. Wegrzyn-Wolska, Timbre: Efficient job recommendation on heterogeneous graphs for professional recruiters, in: Proceedings of the 23rd IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), 2024, pp. 8–15. doi:10.1109/WI-IAT62293.2024.00010.

[11] L. Jing, P. Vincent, Y. LeCun, Y. Tian, Understanding dimensional collapse in Contrastive Self-supervised Learning, in: International Conference on Learning Representations (ICLR), 2022. ArXiv:2110.09348.

[12] X. Guo, J. Pan, X. Wang, B. Chen, J. Jiang, M. Long, On the Embedding Collapse when scaling up Recommendation Models, in: Proceedings of the 41st International Conference on Machine Learning (PMLR), volume 235, 2024, pp. 16891–16909. ArXiv:2310.04400.

[13] D. Loveland, X. Wu, T. Zhao, D. Koutra, N. Shah, M. Ju, Understanding and scaling Collaborative Filtering optimization from the perspective of Matrix Rank, in: Proceedings of the ACM Web Conference 2025 (WWW '25), 2025, pp. 436–449. doi:10.1145/3696410.3714904.

[14] S. Peng, K. Sugiyama, X. Liu, T. Mine, Balancing embedding spectrum for recommendation, ACM Transactions on Recommender Systems 3 (2025) 56:1–56:25. doi:10.1145/3718488.

[15] X. Ma, L. Zhao, G. Huang, Z. Wang, Z. Hu, X. Zhu, K. Gai, Entire Space Multi-Task Model: An effective approach for estimating post-click conversion rate, in: Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), 2018, pp. 1137–1140. doi:10.1145/3209978.3210104.

[16] R. Le, W. Hu, Y. Song, T. Zhang, D. Zhao, R. Yan, Towards effective and interpretable Person-Job Fitting, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM), 2019, pp. 1883–1892. doi:10.1145/3357384.3357949.

[17] R. Yan, R. Le, Y. Song, T. Zhang, X. Zhang, D. Zhao, Interview choice reveals your preference on the market: To improve job-resume matching through profiling memories, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2019, pp. 914–922. doi:10.1145/3292500.3330963.

[18] S. Bian, X. Chen, W. X. Zhao, K. Zhou, Y. Hou, Y. Song, T. Zhang, J.-R. Wen, Learning to match jobs with resumes from sparse interaction data using Multi-View Co-Teaching Network, in: Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM), 2020, pp. 65–74. doi:10.1145/3340531.3411929.

[19] X. Yu, J. Zhang, Z. Yu, ConFit: Improving resume-job matching using data augmentation and contrastive learning, in: Proceedings of the 18th ACM Conference on Recommender Systems (RecSys), 2024, pp. 601–611. doi:10.1145/3640457.3688108.

[20] X. Yu, R. Xu, C. Xue, J. Zhang, X. Ma, Z. Yu, ConFit v2: Improving resume-job matching using hypothetical resume embedding and runner-up hard-negative mining, in: Findings of the Association for Computational Linguistics: ACL 2025, 2025, pp. 12775–12790. doi:10.18653/v1/2025.findings-acl.661.

[21] Y. Du, D. Luo, R. Yan, X. Wang, H. Liu, H. Zhu, Y. Song, J. Zhang, Enhancing job recommendation through LLM-Based generative adversarial networks, in: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), volume 38, 2024, pp. 8363–8371. doi:10.1609/aaai.v38i8.28678.

[22] B. Fu, H. Liu, Y. Zhu, Y. Song, T. Zhang, Z. Wu, Beyond matching: Modeling two-sided multi-behavioral sequences for dynamic person-job fit, in: Proceedings of the 26th International Conference on Database Systems for Advanced Applications (DASFAA), 2021, pp. 359–375. doi:10.1007/978-3-030-73197-7_24.

[23] Z. Zheng, X. Hu, Z. Qiu, Y. Cheng, S. Gao, Y. Song, H. Zhu, H. Xiong, Bilateral multi-behavior modeling for reciprocal recommendation in online recruitment, IEEE Transactions on Knowledge and Data Engineering 36 (2024) 5681–5694. doi:10.1109/TKDE.2024.3397705.

[24] F. Borisyuk, S. He, Y. Ouyang, M. Ramezani, P. Du, X. Hou, C. Jiang, N. Pasumarthy, P. Bannur, B. Tiwana, P. Liu, S. Dangi, D. Sun, Z. Pei, X. Shi, S. Zhu, Q. Shen, K.-H. Lee, D. Stein, B. Li, H. Wei, A. Ghoting, S. Ghosh, LiGNN: Graph Neural Networks at LinkedIn, in: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2024. doi:10.1145/3637528.3671566.

[25] W. L. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Advances in Neural Information Processing Systems 30 (NeurIPS), 2017, pp. 1024–1034. ArXiv:1706.02216.

[26] O. Roy, M. Vetterli, The effective rank: A measure of effective dimensionality, in: Proceedings of the 15th European Signal Processing Conference (EUSIPCO), 2007, pp. 606–610.

[27] S. Lin, C. Gao, J. Chen, S. Zhou, B. Hu, Y. Feng, C. Chen, C. Wang, How do recommendation models amplify popularity bias? an analysis from the spectral per-

spective, in: Proceedings of the 18th ACM International Conference on Web Search and Data Mining (WSDM), 2025. doi:10.1145/3701551.3703579, arXiv:2404.12008.

[28] Y. Zhang, H. Zhu, Y. Chen, Z. Song, P. Koniusz, I. King, Mitigating the popularity bias of graph collaborative filtering: A dimensional collapse perspective, in: Advances in Neural Information Processing Systems 36 (NeurIPS), 2023.

[29] M. S. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, M. Welling, Modeling relational data with Graph Convolutional Networks, in: The Semantic Web — 15th International Conference (ESWC 2018), volume LNCS 10843, 2018, pp. 593–607. doi:10.1007/978-3-319-93417-4_38.

[30] H. Wang, T.-W. Chang, T. Liu, J. Huang, Z. Chen, C. Yu, R. Li, W. Chu, ESCM^2: Entire space counterfactual multi-task model for post-click conversion rate estimation, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), 2022, pp. 363–372. doi:10.1145/3477495.3531972.

[31] F. Borisyuk, M. Zhou, Q. Song, S. Zhu, B. Tiwana, G. Parameswaran, S. Dangi, L. Hertel, Q. Xiao, X. Hou, Y. Ouyang, A. Gupta, S. Singh, D. Liu, H. Cheng, L. Le, J. Hung, S. Keerthi, R. Wang, F. Zhang, M. Kothari, C. Zhu, D. Sun, Y. Dai, X. Luan, S. Zhu, Z. Wang, N. Daftary, Q. Shen, C. Jiang, H. Wei, M. Varshney, A. Ghoting, S. Ghosh, LiRank: Industrial large scale ranking models at LinkedIn, in: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2024, pp. 4804–4815. doi:10.1145/3637528.3671561.

[32] Tianchi, The second Alibaba big data intelligent cloud programming competition: Zhilian Zhaopin intelligent matching of people and jobs, 2019. URL: https://tianchi.aliyun.com/competition/entrance/231728/information.

[33] S. Brody, U. Alon, E. Yahav, How attentive are Graph Attention Networks?, in: International Conference on Learning Representations (ICLR), 2022. ArXiv:2105.14491.

[34] Y. Zhang, M. Li, D. Long, X. Zhang, H. Lin, B. Yang, P. Xie, A. Yang, D. Liu, J. Lin, F. Huang, J. Zhou, Qwen3 embedding: Advancing text embedding and reranking through foundation models, 2025. ArXiv:2506.05176.

[35] L. Wang, N. Yang, X. Huang, L. Yang, R. Majumder, F. Wei, Multilingual E5 text embeddings: A technical report, 2024. ArXiv:2402.05672.

[36] W. Krichene, S. Rendle, On sampled metrics for item recommendation, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2020, pp. 1748–1757. doi:10.1145/3394486.3403226.

Table 5
Extended grouped ranking metrics for the controlled content-placement family (SAGE, Qwen3-0.6B; 5-seed mean). Seeker side groups the exposed pairs by seeker, job side by job; MAP@k (M@k) and Hit@k (H@k) are computed within each group and averaged (unweighted) over groups. **Bold** = best, underline = second best per column *within each dataset block*. Leading zeros omitted.

	Seeker side							Job side						
Model	GAUC	M@5	M@10	M@20	H@5	H@10	H@20	GAUC	M@5	M@10	M@20	H@5	H@10	H@20
Tech														
content MLP	.632	.216	.232	.292	.716	.850	.949	**.573**	**.499**	**.529**	**.542**	**.890**	**.975**	**.997**
content+graph (naive)	.648	.231	.247	.307	.725	.859	**.951**	.551	.477	.507	.521	.881	.972	.997
graph only	**.655**	**.239**	**.253**	**.314**	.719	.858	.949	.549	.475	.507	.520	.879	.970	.996
content+graph (decoupled)	.654	.233	.251	.310	**.739**	**.873**	.950	.556	.483	.513	.526	.884	.972	.996
Tianchi														
content MLP	.640	.567	.606	.618	.913	.985	.998	.526	.823	.827	.827	**1.000**	**1.000**	**1.000**
content+graph (naive)	.687	.611	.647	.657	.939	**.991**	.999	.529	.823	.827	.827	.999	1.000	1.000
graph only	.680	.608	.644	.655	.933	.990	**1.000**	.546	.830	.835	.835	.999	1.000	1.000
content+graph (decoupled)	**.699**	**.623**	**.658**	**.668**	**.942**	.991	1.000	**.552**	**.832**	**.836**	**.837**	.999	1.000	1.000

A. Extended ranking metrics

Beyond the AUC and GAUC of Table 2, the evaluation emits the full grouped ranking suite on both sides of the market. Table 5 reports it for the main controlled family (SAGE, Qwen3-0.6B): per group (one seeker's exposed candidates, or one job's exposed applicants), mean average precision (MAP@k) and hit rate (Hit@k) at $k \in \{5, 10, 20\}$, alongside the GAUC. Groups lacking both a positive and a negative are skipped. The extended metrics agree with the GAUC picture: on the seeker side the graph variants dominate the content MLP on GAUC and MAP at every cutoff (Hit@20 saturates for all variants), while on the job side the content MLP leads every metric on Tech and the decoupled variant leads GAUC and MAP on Tianchi, where Hit@k is saturated throughout.