MOIRE: OBJECTIVE-SPECIFIC ADAPTATION AND RESIDUAL READOUT FOR MULTI-OBJECTIVE SELF-SUPERVISED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-objective training in self-supervised learning (SSL) combines several pretraining objectives to learn representations for downstream tasks. However, training these objectives together requires coordinating their losses through a shared backbone, where updates for one objective can impair predictions for others. We therefore introduce two complementary diagnostic protocols to quantify cross-objective damage under isolated and coordinated updates, and use them to examine existing loss-weighting and gradient-coordination methods from multi-task learning. Our measurements show that these methods only modestly reduce cross-objective damage compared with a fixed-weight sum of losses. Moreover, even effective coordination of self-supervised objectives need not improve downstream performance, because pretraining objectives may emphasize different information from what downstream tasks and evaluation metrics reward. To address both challenges, we propose MOIRE (Multi-Objective Interaction and Residual Evidence). MOIRE gives each objective a separate adaptation path, reducing interference with the shared representation. It retains the resulting objective-specific changes as residual evidence. This evidence can then be combined with the shared representation to better align the learned representation with downstream tasks. Mechanistic experiments show that MOIRE has the lowest relative cross-objective damage among the compared methods. Its downstream performance is similarly strong: across eight datasets spanning graph representation learning and sequential recommendation, it achieves the best average rank in both domains.

1 INTRODUCTION

Self-supervised learning (SSL) often combines multiple pretraining objectives, with each objective capturing a different aspect of the data. Together, these objectives aim to learn representations that transfer well to downstream tasks. When these objectives share a backbone, however, their losses must be converted into a single training update. Multi-task learning (MTL) offers two main ways to coordinate this process: by weighting objective losses before aggregation, or by modifying their gradients to balance learning progress or reduce conflict. These strategies have also been adopted in multi-objective self-supervised learning. For example, S^3-Rec jointly trains four objectives for sequential recommendation using a fixed weighted sum of losses—referred to here as linear scalarization (LS)—whereas ParetoGNN uses the multiple gradient descent algorithm (MGDA) to coordinate several graph self-supervised objectives (Zhou et al., 2020; Ju et al., 2023). Earlier multi-task self-supervised learning has likewise examined how several pretext tasks can be organized around a shared representation (Doersch & Zisserman, 2017). Together, these studies establish practical ways to optimize multiple self-supervised objectives. They leave a further representation-level question open: when one objective updates the shared backbone, what happens to the predictive relationships that other objectives have already established with that representation?

We therefore examine how one objective's update changes another objective's loss under a fixed readout of the shared backbone, building on directed task-affinity measurements (Fifty et al., 2021). To measure this effect under an isolated source, we start from a jointly pretrained checkpoint and continue training with one objective at a time. A frozen auditor then evaluates each target objective from the shared backbone representation. On Chameleon (Fig. 1), continuing the Link objective

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

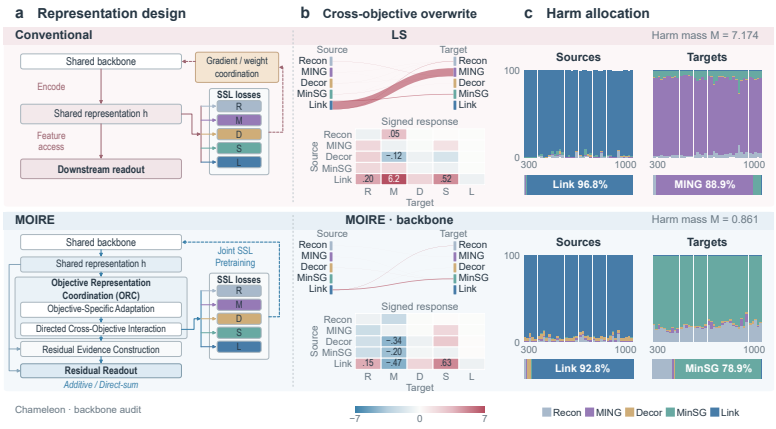


Figure 1: Representation design and cross-objective damage on Chameleon node classification. (a) Model designs; (b) directed changes in frozen-readout loss; (c) damage by objective.

under LS substantially increases the frozen-auditor loss of MING (a local–global discrimination objective), revealing a pronounced directional overwrite of an existing readout. We also introduce a matched coordinated-update protocol that compares full updates with matched leave-one-source updates. Both protocols are detailed in Section 2.

Reducing cross-objective damage, however, does not guarantee better downstream performance. SSL objectives guide pretraining, but downstream tasks and evaluation metrics may reward different information. Even if the objectives are well coordinated, improvements in their losses may therefore fail to improve a downstream metric. This raises a second question: can information learned for individual objectives remain available after pretraining, so that a downstream reader can choose how much of it to use?

These two challenges motivate MOIRE (Multi-Objective Interaction and Residual Evidence). MOIRE builds a separate state for each objective during joint pretraining, reducing reliance on changes to the shared representation. A lightweight interaction module can share information across these states. MOIRE then reuses information retained in these objective-specific paths as residual signals alongside the shared representation to better align what SSL learns with downstream tasks. MOIRE therefore offers a new perspective on MTL for multi-objective SSL: it coordinates objective-specific representations without introducing another loss-weighting or gradient-modification rule and retains their changes for downstream use.

We then evaluate MOIRE under source-only continuation and the coordinated-update protocol and on downstream tasks in graph learning and sequential recommendation. We also examine how the strength and relative proportions of the residual blocks, together with the choice of readout form, affect how well the learned information serves the chosen downstream metric. MOIRE ranks among the least damaging methods under both protocols and achieves the strongest cross-dataset downstream ranking. In summary, our contributions are as follows:

(1) Building on directed task-affinity measurements (Fifty et al., 2021), we use source-only continuation and introduce a matched coordinated-update protocol. Together, they quantify how updates for one self-supervised objective affect established readouts of others.

(2) We introduce MOIRE, which gives each objective a separate adaptation path to reduce interference with the shared representation. MOIRE retains the resulting objective-specific changes as residual evidence, which can be selectively reused to better align downstream prediction with the target evaluation metric.

(3) Across eight datasets and ten baselines, MOIRE substantially reduces cross-objective damage and achieves the best domain-average mean rank in both graph learning and sequential recommendation.

2 MOTIVATION: CROSS-OBJECTIVE DAMAGE AND DOWNSTREAM ALIGNMENT

Multi-objective SSL faces two related questions. During pretraining, does progress on one objective disrupt what another objective can read from the shared representation? After pretraining, can the learned information be used in a way that better matches downstream evaluation? We examine the first question with controlled updates and use the second to motivate downstream readout alignment.

2.1 CROSS-OBJECTIVE DAMAGE DURING SSL

Let $h = f_\theta(x)$ denote the shared representation, and let s and t denote a source and a target SSL objective. Starting from a jointly pretrained checkpoint, we freeze an auditor that predicts t from h. We then update the model with source objective s and measure the resulting change in the auditor loss.

A positive signed response indicates damage: the fixed target readout has become less accurate after the source update. Because this response is directional, the effect of s on t need not equal the effect of t on s. For example, on Chameleon (Fig. 1), continuing Link under linear scalarization (LS) produces a $+6.20$ response for MING, whereas the corresponding response from the MOIRE backbone is -0.47. Thus, the signed response measures how stable an established target readout remains after an update involving another objective.

We measure this response under two protocols. In source-only continuation, we start from a jointly pretrained checkpoint, update the model for five steps using only the source objective, and compare the frozen target auditor with its value at the starting checkpoint. In the coordinated-update protocol, we compare a full joint-training update with a matched leave-one-source update. We average the signed response over five such matched pairs. The first protocol measures what a source can change in isolation; the second measures its marginal effect during coordinated training. These measurements motivate separate objective-specific states that can still exchange useful information.

2.2 SSL OBJECTIVES AND DOWNSTREAM EVALUATION

SSL objectives guide representation learning during pretraining, but downstream tasks are evaluated by different criteria. An update that improves an SSL objective therefore need not improve a downstream metric. Figure 2 illustrates this mismatch schematically: SSL training may favor representation changes that are not aligned with the directions preferred by a downstream reader.

More locally, let $\Delta h_{\mathrm{SSL}}(x)$ denote the representation change induced by an SSL update and let L_{down} be a differentiable surrogate for a downstream objective. Its approximate change is

$$\Delta L_{\mathrm{down}} \approx \mathbb{E}_x \left[\nabla_h L_{\mathrm{down}}(h(x))^\top \Delta h_{\mathrm{SSL}}(x) \right].$$

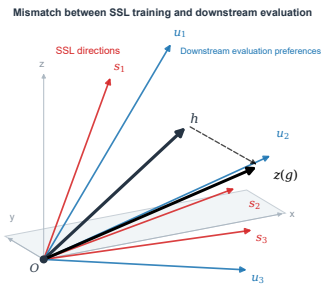


Figure 2: SSL update directions (red) and downstream preferences (blue), shown schematically.

Reducing an SSL loss does not determine the sign of this expression. This motivates preserving objective-specific information after pretraining and allowing the downstream criterion to determine how it is used. MOIRE implements this idea by retaining objective-specific representation changes as residual evidence and selecting their downstream use with validation data, without repeating SSL pretraining. Section 4.2 evaluates both the recovery of affected readouts and whether the retained residual evidence can better align the downstream readout with a chosen evaluation metric.

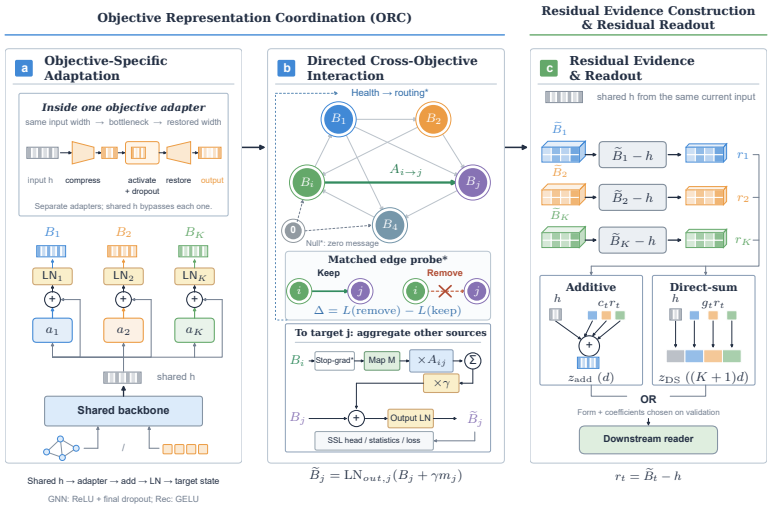


Figure 3: Objective-specific coordination and residual readout in MOIRE. (a) Objective adapters; (b) directed exchange; (c) residual construction.

3 MOIRE: Objective Representation Coordination and Residual Readout

3.1 Setting and Overview

Let $\mathcal{T} = \{1, \ldots, K\}$ denote the SSL objectives. For an input view $x^{(v)}$, the shared backbone produces $h^{(v)} = f_\theta(x^{(v)}) \in \mathbb{R}^d$. MOIRE constructs objective-specific states from the same $h^{(v)}$, coordinates them during joint SSL pretraining, and exposes their changes relative to h as residual evidence for downstream readout.

3.2 Objective Representation Coordination

Objective-Specific Adaptation. Each SSL objective receives its own adaptation path:

$$B_t = \mathrm{LN}_t(h + a_t(h)),$$

where a_t is a two-layer bottleneck adapter and LN_t is objective-specific. The residual structure lets each objective modify the representation supplied to its SSL head without requiring every adjustment to be absorbed by the shared backbone. This provides an objective-specific space in which to fit the pretraining signal while retaining the shared representation h.

Directed Interaction. Objective-specific states can optionally exchange information before SSL supervision. For target j, we aggregate transformed states from the other objectives:

$$m_j = \sum_{i \neq j} A_{i \to j} M_{i \to j}(\mathrm{sg}(B_i)), \qquad \widetilde{B}_j = \mathrm{LN}_j^{\mathrm{out}}(B_j + \gamma m_j),$$

where $A_{i \to j}$ are normalized directed weights and $\mathrm{sg}(\cdot)$ stops gradients through source states. This interaction allows a target to reuse information learned by other objectives while keeping its main adaptation path objective-specific.

Joint SSL pretraining. Each objective supervises its own post-interaction state on the input views required by that objective. Writing q_t for its head or representation statistic, we optimize

$$L_t = \ell_t\Big(q_t\big(\{\widetilde{B}_t^{(v)}\}_{v \in V_t}\big), y_t\Big), \qquad \mathcal{L}_{\mathrm{SSL}} = \sum_{t=1}^{K} \lambda_t L_t,$$

where V_t contains the required views, y_t denotes the corresponding SSL supervision, and λ_t weights the training losses. Joint optimization learns the shared backbone, the objective-specific paths, the

interaction parameters, and the SSL heads. This produces a shared representation together with states shaped by individual objectives and their directed exchange. The next section carries these states into downstream prediction through their residuals $r_t = \widetilde{B}_t - h$, allowing the downstream criterion to determine how much of each learned change to use.

3.3 Residual Evidence Construction and Residual Readout

Section 2.2 raises an alignment question: SSL objectives may emphasize a different mix of information from what a downstream evaluation metric rewards. Figure 3c shows how MOIRE addresses this mismatch after pretraining by reusing the objective-specific paths learned in Stage 1. For each downstream input, MOIRE computes the shared representation h and the post-interaction state $\widetilde{B}_t$ at the same node or sequence position, then defines

$$r_t = \widetilde{B}_t - h.$$

Each residual captures the change produced by Stage 1's objective-specific adaptation, normalization, and interaction path. It is recomputed for the current downstream input, while h remains available to the reader.

Residual Readout offers two alternative interfaces to this evidence, Additive and Direct-sum (DS):

$$z_{\mathrm{add}} = h + \sum_{t=1}^{K} c_t r_t \in \mathbb{R}^d, \qquad z_{\mathrm{DS}} = [h; g_1 r_1; \ldots; g_K r_K] \in \mathbb{R}^{(K+1)d}.$$

Additive combines the scaled residuals with h in one d-dimensional representation. Direct-sum gives h and each scaled residual a separate input block. The effective coefficients c_t and g_t include any scaling used by the corresponding implementation; the two forms need not use numerically identical coefficients. For each downstream setting, the readout form and its coefficients are chosen using the designated validation metric and fixed before test evaluation. Because the pretrained model is reused, selecting the readout form and coefficients requires validation-time search but no additional SSL pretraining. This flexibility is limited to evidence retained during pretraining: the readout can reweight or recombine existing objective-specific states, but it cannot recover information that those states never captured. Section 4.2 tests both the recovery of affected readouts and how changing these coefficients shifts downstream metric outcomes. Residual Readout is therefore an additional downstream mechanism rather than the sole source of MOIRE's gains. Section 5.4 isolates this contribution: even without Residual Readout, the MOIRE backbone retains the lowest domain-average mean rank in both graph learning and sequential recommendation.

4 Mechanistic Evaluation

4.1 Cross-objective damage under two protocols

Figure 1 showed a representative reduction in cross-objective damage under MOIRE relative to LS on Chameleon. We next asked whether this advantage extends across other coordination methods and to sequence recommendation. We tracked both protocols across eight checkpoints in each training trajectory, comparing MOIRE with eight baselines on Chameleon and ML-1M through fixed target-specific readouts of the shared backbone h (Fig. 4). Source-only continuation captures damage after five updates from one objective; the coordinated-update protocol averages five matched pairs of full and leave-one-source updates during coordinated training.

On Chameleon, Link-to-MING accounted for most source-only damage in all eight comparison methods, but MOIRE substantially mitigated damage along this direction (Fig. 4b). Across all directions, MOIRE had the lowest mean HarmMass (the sum of positive damage across directed objective pairs) among the nine methods, with reductions relative to LS of 88.0% on Chameleon and 67.7% on ML-1M (Fig. 4a,d).

In the coordinated-update protocol, MOIRE also had the lowest mean harmful marginal influence among the nine methods. Relative to LS, this influence was reduced by 68.5% on Chameleon and 75.4% on ML-1M (Fig. 4c,f). Together, the two protocols show lower cross-objective damage in MOIRE's learned backbone before residual readout.

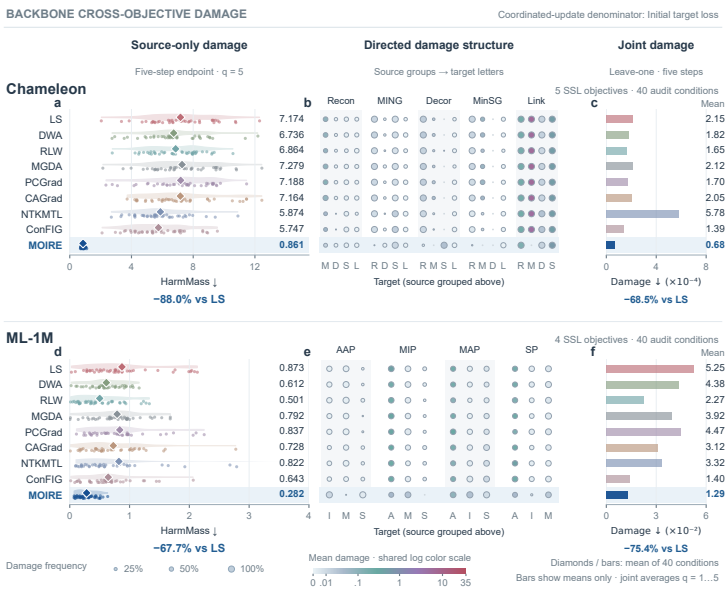


Figure 4: Cross-objective damage on Chameleon and ML-1M. (a,d) Total damage; (b,e) source-to-target damage; (c,f) damage from coordinated updates.

4.2 Residual readout preserves evidence and supports downstream alignment

MOIRE's objective-specific adaptation and interaction paths are trained alongside the shared backbone. We first asked whether their learned states remained useful when interference weakened fixed readouts of the shared representation, h. Starting from the same source-only damaged checkpoints, we froze the backbone and compared its readout alone with Additive and Direct-sum readouts of the residuals (Fig. 5a). Harmful-mass recovery (HMR) measures the share of observed damage regained through each readout.

Both residual readouts recovered part of the damaged predictions in all four settings. Additive recovered 32.03–49.81% of harmful mass, and Direct-sum recovered 43.26–56.10% (Fig. 5a). The recovery across two graph and two recommendation datasets shows that the learned objective-specific changes remain available to a downstream reader even when a fixed readout of h has deteriorated.

We next asked whether the residual blocks that remained readable could also help close the gap between SSL training and downstream evaluation. SSL objectives shape the representation before the downstream criterion is known, so the information they emphasize may differ from what that criterion rewards. As motivated by the conceptual mismatch in Fig. 2 and implemented by the residual readout in Fig. 3, MOIRE retains changes learned by the objective-specific adapters and uses validation feedback to choose how they enter the downstream readout. We tested this downstream alignment on Chameleon by varying the residual gates after SSL pretraining and measuring the resulting test scores (Fig. 5b,c).

The gate scan traced distinct paths through link and node score space, and improving one metric sometimes came at the expense of another. On Chameleon, these responses show that the SSL-trained representation does not serve every downstream criterion equally. The stars mark configurations chosen separately for individual validation metrics and show where those choices fall in test score space. Thus, recovery shows that objective-specific evidence remains available after backbone interference; the gate experiment shows that validation can use the pretrained residual blocks to adjust the readout toward a chosen downstream criterion without another SSL training run. Section 5 evaluates whether this downstream alignment contributes to performance across datasets.

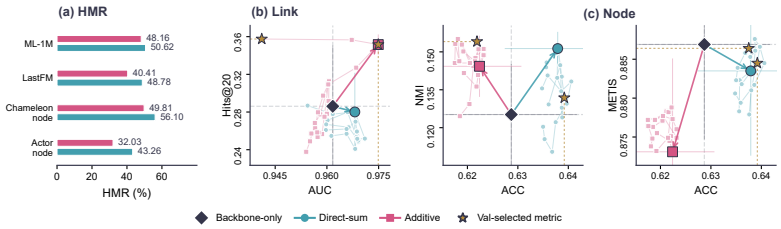


Figure 5: Residual readout on Chameleon. (a) Recovery of harmful backbone changes; (b,c) link and node test scores.

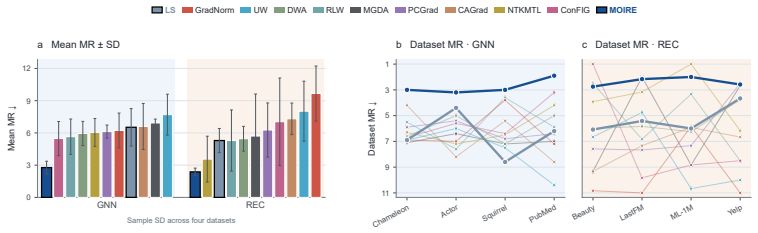


Figure 6: Mean ranks across GNN and recommendation datasets. (a) Domain means and standard deviations; (b,c) dataset ranks. Lower is better.

5 Downstream Evaluation

5.1 Experimental Setup

Baselines. We compare MOIRE with ten commonly used multi-task optimization baselines: Linear Scalarization (LS), Random Loss Weighting (RLW), Dynamic Weight Average (DWA), Uncertainty Weighting (UW), Multiple Gradient Descent Algorithm (MGDA), PCGrad, CAGrad, GradNorm, NTKMTL, and ConFIG (Liu et al., 2025). LS optimizes a fixed weighted sum of the self-supervised losses and serves as our primary reference.

Datasets. We evaluate multi-objective SSL in two domains. For sequential recommendation, we jointly optimize the four S^3-Rec objectives (Zhou et al., 2020) on Beauty, LastFM, ML-1M, and Yelp. For graph representation learning, we jointly optimize the five ParetoGNN objectives (Ju et al., 2023) on Chameleon, Actor, Squirrel, and PubMed.

Metrics. We report Mean Rank (MR) and mean score gain relative to LS. MR averages ranks across the downstream metrics of each dataset (lower is better), while the LS-relative gain reports the percentage change in their mean score (higher is better). Domain-level values equally average the four datasets. Individual metrics are reported in the Appendix.

Training and evaluation. Within each domain, all methods use the same backbone, SSL objectives, training schedule, and evaluation protocol. We average graph results over 10 random seeds and recommendation results over 5.

5.2 Dataset-Dependent Gains from Existing MTL Methods

The dataset-level ranks expose a weakness of existing MTL optimizers in multi-objective SSL (Fig. 6b,c). The two panels follow the same methods across four graph and four recommendation datasets, respectively. Their changing order shows that a favorable rank on one dataset does not reliably carry over to another within the same domain, even though all four datasets within each domain use the same SSL pretraining losses. To determine whether these rank changes corresponded to gains over fixed-weight LS, we compared each method with LS on every downstream measure and counted a dataset as a win when more measures improved than declined (Table 1; score gains are shown in Fig. 7). Across the eight datasets, the nine competing optimizers won an average of only 3.44 datasets and 19.33 of the 44 measures; none won all eight. MOIRE, in contrast, won the majority

Table 1: Dataset-level mean ranks (MR) and wins over LS across graph (GNN) and recommendation datasets. MR averages metric-wise ranks among 11 methods; lower is better. GNN and Rec are equal-dataset means. Metrics counts improved metrics out of 44; Dataset gives dataset wins/ties/losses. The dataset columns are Chameleon, Actor, Squirrel, PubMed, Beauty, LastFM, ML-1M and Yelp. Best results are shown in bold and second-best results are underlined; ties share the same formatting.

Method	Cha	Act	Squ	Pub	GNN	Bea	LFM	ML1M	Yelp	Rec	Metrics	Dataset
LS	6.90	4.40	8.60	6.20	6.53	6.08	5.42	6.00	3.67	5.29	–	–
GradNorm	6.90	7.00	3.80	7.20	6.23	10.83	11.00	5.83	11.00	9.67	14/44	2/1/5
UW	6.90	6.00	7.50	10.40	7.70	6.67	4.75	10.67	10.00	8.02	15/44	2/2/4
DWA	6.60	5.00	7.20	5.00	5.95	6.00	5.83	6.25	3.75	5.46	22/44	3/2/3
RLW	5.50	7.60	3.60	5.90	5.65	2.50	6.83	3.33	8.50	5.29	25/44	5/0/3
MGDA	7.10	6.40	7.20	7.00	6.93	9.33	**2.00**	8.83	2.67	5.71	18/44	4/0/4
PCGrad	5.90	5.40	6.80	6.40	6.13	7.58	7.67	7.33	**2.50**	6.27	16/44	4/0/4
CAGrad	4.20	8.20	5.40	8.60	6.60	9.33	7.33	5.92	6.67	7.31	17/44	3/0/5
NTKMTL	6.30	7.20	6.50	4.20	6.05	3.92	3.17	**1.00**	6.17	3.56	28/44	5/0/3
ConFIG	6.70	5.60	6.40	3.20	5.48	**1.00**	9.83	8.83	8.50	7.04	19/44	3/0/5
MOIRE	**3.00**	**3.20**	**3.00**	**1.90**	**2.78**	2.75	2.17	2.00	2.58	**2.38**	**38/44**	**8/0/0**

of measures on every dataset (8/8) and improved 38/44 measures overall. Thus, methods developed largely for supervised MTL yielded dataset-dependent gains in this SSL setting and often struggled to surpass even fixed-weight LS, whereas MOIRE maintained its advantage across both domains.

5.3 Overall Downstream Advantage

The consistency across datasets translated into the lowest domain-level mean rank in both settings (Fig. 6a; Table 1). On the graph benchmarks, MOIRE reached an MR of 2.78, compared with 5.48 for the next-best method, ConFIG, and 6.53 for LS. Its 2.70-rank lead over ConFIG was larger than the 2.23-rank range from ConFIG to the worst competing optimizer. On recommendation, MOIRE achieved an MR of 2.38, ahead of NTKMTL (3.56) and LS (5.29).

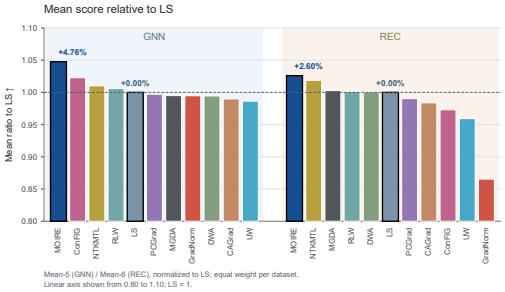


Figure 7: Domain mean scores relative to LS, as defined in Section 5.1.

The score-based comparison in Fig. 7 shows that this rank advantage also corresponds to larger downstream gains. Relative to LS, MOIRE increased the dataset-averaged mean score by 4.76% on graphs and 2.60% on recommendation; the best competing gains were 2.27% (ConFIG) and 1.88% (NTKMTL), respectively. Most competing optimizers produced only small gains or reduced the mean score relative to LS. MOIRE therefore combined the strongest overall rank with the largest score improvement in both domains.

5.4 Ablation and Cost

Figure 8 compares full MOIRE with a backbone-only variant that retains MOIRE's pretraining procedure but uses only the shared representation h for downstream prediction, without Residual Readout. Even without Residual Readout, this backbone-only variant retains the lowest domain-average mean rank among 11 methods across four datasets in each of the two domains. Adding Residual Readout substantially widens this lead. Complete ablation results are reported in the Appendix.

Figure 9 compares recorded runtime with the baselines. The readout form and residual coefficients are selected using validation data, without repeating SSL pretraining. This selection took approximately 35% of SSL pretraining time in graph learning and 43% in recommendation. The selection results are reported in the Appendix.

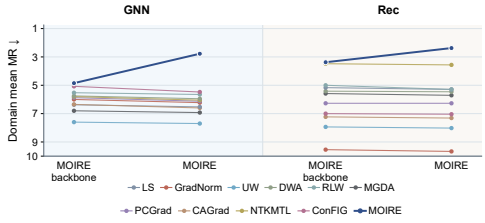


Figure 8: MOIRE backbone and full-model mean ranks across GNN and recommendation domains. Each variant is ranked against the same ten comparison methods; lower is better.

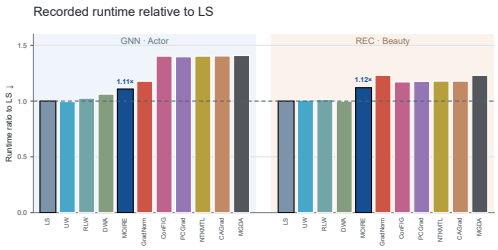


Figure 9: Runtime of comparison methods on Actor and Beauty.

6 Related Work

Multi-Task Learning. Methods for coordinating tasks in a shared network are commonly divided into loss-oriented and gradient-oriented approaches. Loss-oriented methods assign fixed or adaptive weights to task losses, often to address differences in loss scale or learning progress; examples include linear scalarization (LS), uncertainty weighting, dynamic weight averaging, and random loss weighting (Kendall et al., 2018; Liu et al., 2019; Lin et al., 2022). Gradient-oriented methods use task gradients to balance their contributions or find an update direction that reduces conflict, including gradient normalization, Pareto-based descent, gradient projection, and conflict-averse updates (Chen et al., 2018; Sener & Koltun, 2018; Yu et al., 2020; Liu et al., 2021). NTKMTL additionally analyzes task convergence rates to guide task balancing (Qin et al., 2025). The methods above have been benchmarked mainly on supervised multi-task prediction, with some also tested in multi-task reinforcement learning (Sener & Koltun, 2018; Yu et al., 2020; Liu et al., 2021; Qin et al., 2025).

Multi-Objective Self-Supervised Learning. In sequential recommendation, S^3-Rec combines four self-supervised objectives derived from item, attribute, and sequence relationships to enrich pretraining for recommendation (Zhou et al., 2020). In graph learning, ParetoGNN combines pretext tasks spanning reconstruction, decorrelation, and mutual-information objectives, using MGDA to improve performance across different downstream tasks (Ju et al., 2023). Related graph studies have also found that pretext tasks differ in their downstream value across datasets and have examined relationships among the pretext tasks themselves (Jin et al., 2022; Fang et al., 2024).

Graph Multi-Objective SSL. The graph studies above, together with methods for pretext selection and scheduling (Fan et al., 2024; Grover et al., 2026), develop their approaches within graph learning and evaluate gains on downstream measures chosen for that setting. MOIRE instead asks how a common MTL framework can coordinate SSL objective sets that differ in both number and type, with the goal of improving the full range of downstream measures consistently across graph learning and sequential recommendation. To study this question, we compare ten general-purpose MTL coordinators under matched SSL protocols and analyze both directed damage to established readouts and the downstream reuse of objective-specific residual evidence.

7 Discussion and Conclusion

MOIRE addresses cross-objective damage through objective-specific adaptation and directed exchange, then retains the resulting residual evidence for downstream use. The intervention audits show lower backbone damage, and residual recovery shows that objective-specific changes remain readable after backbone interference. Validation-based Residual Readout adapts access to this evidence without repeating SSL pretraining. Across the eight studied datasets, the MOIRE backbone has the best domain-average rank in both graph learning and sequential recommendation; residual readout further improves performance. These findings are bounded by the prescribed interventions and evaluated datasets.

AI USE STATEMENT

Generative AI tools were used to assist and polish writing, support retrieval and discovery of related literature, and assist research ideation or execution. AI assistance was also used for submission-format checks and preparation of the disclosure and availability statements. The authors take responsibility for the final manuscript, including its claims, references, code, and any AI-assisted content.

ETHICS STATEMENT

This work studies multi-objective self-supervised representation learning using existing graph and sequential-recommendation benchmarks. Recommendation applications can raise privacy and fairness concerns; improved predictive performance alone does not establish fairness or suitability for deployment. Any downstream use should respect the original datasets' terms and evaluate privacy, bias, and application-specific risks.

REPRODUCIBILITY STATEMENT

Section 3 describes the model and residual readout, Section 4 describes the diagnostic protocols, and Section 5.1 specifies the benchmark datasets, baselines, evaluation metrics, and numbers of random seeds. Additional experimental results and implementation details are provided in the appendix. If the paper is accepted, we will publicly release the source code.

REFERENCES

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 794–803, 2018.

Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2051–2060, 2017.

Tianyu Fan, Lirong Wu, Yufei Huang, Haitao Lin, Cheng Tan, Zhangyang Gao, and Stan Z. Li. Decoupling weighing and selecting for integrating multiple graph pre-training tasks. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2403.01400.

Taoran Fang, Wei Chow, Yifei Sun, Kaiqiao Han, Lvbin Ma, and Yang Yang. Exploring correlations of self-supervised tasks for graphs. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 12957–12972, 2024. URL https://proceedings.mlr.press/v235/fang24b.html.

Christopher Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently identifying task groupings for multi-task learning. In *Advances in Neural Information Processing Systems*, 2021.

Karish Grover, Theodore Vasiloudis, Han Xie, Sixing Lu, Xiang Song, and Christos Faloutsos. Feedback control for multi-objective graph self-supervision. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.05036.

Wei Jin, Xiaorui Liu, Xiangyu Zhao, Yao Ma, Neil Shah, and Jiliang Tang. Automated self-supervised learning for graphs. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=rFbR4Fv-D6-.

Mingxuan Ju, Tong Zhao, Qianlong Wen, Wenhao Yu, Neil Shah, Yanfang Ye, and Chuxu Zhang. Multi-task self-supervised graph neural networks enable stronger task generalization. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1tHAZRqftM.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7482–7491, 2018.

Baijiong Lin, Feiyang Ye, Yu Zhang, and Ivor W. Tsang. Reasonable effectiveness of random weighting: A litmus test for multi-task learning. *Transactions on Machine Learning Research*, 2022. URL https://openreview.net/forum?id=jjtFD8A1Wx.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. In *Advances in Neural Information Processing Systems*, 2021.

Qiang Liu, Mengyu Chu, and Nils Thuerey. ConFIG: Towards conflict-free training of physics informed neural networks. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/94e85561a342de88b559b72c9b29f638-Abstract-Conference.html.

Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1871–1880, 2019.

Xiaohan Qin, Xiaoxing Wang, Ning Liao, and Junchi Yan. NTKMTL: Mitigating task imbalance in multi-task learning from neural tangent kernel perspective. In *Advances in Neural Information Processing Systems*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/4522de4178bddb36b49aa26efad537cf-Abstract-Conference.html.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, 2018.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, 2020.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S^3-Rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, pp. 1893–1902, 2020. doi: 10.1145/3340531.3411954. URL https://arxiv.org/abs/2008.07873.

Supplementary Material for MOIRE

A Full Downstream Results

This appendix reports the complete downstream evaluation underlying the summary comparisons. It presents the raw metric means and standard deviations first, followed by metric-wise comparisons with LS and dataset-level aggregate scores. Graph neural network (GNN) results cover Chameleon, Actor, Squirrel, and PubMed; recommendation (Rec) results cover Beauty, LastFM, ML-1M, and Yelp. All displayed metrics are oriented so that higher is better. The source tables use the main-result rows of the GNN and Rec workbooks.

A.1 Complete metric-wise results

Tables A.1–A.8 report the complete metric-wise downstream means and standard deviations for the graph and recommendation datasets.

A.1.1 Graph representation learning

Shared note for Tables A.1–A.4. Entries are mean ± standard deviation for ACC, AUC, Hits@20, METIS-ACC, and NMI. Scores are multiplied by 100 and rounded to two decimal places. Higher means are better. For each metric, the highest mean is bold and the second-highest distinct mean is underlined.

Table A.1: Raw downstream GNN results on Chameleon.

Model	ACC	AUC	Hits@20	METIS-ACC	NMI
LS	62.45 ± 1.01	95.98 ± 0.17	24.14 ± 3.97	88.41 ± 1.20	14.19 ± 1.70
GradNorm	62.74 ± 1.21	95.63 ± 0.10	24.49 ± 4.13	88.51 ± 1.25	14.00 ± 2.35
UW	62.49 ± 1.05	95.98 ± 0.17	24.06 ± 4.04	88.62 ± 1.44	13.29 ± 1.69
DWA	62.49 ± 1.22	95.93 ± 0.17	24.12 ± 3.97	88.55 ± 1.17	14.10 ± 1.60
RLW	62.71 ± 1.12	95.91 ± 0.17	24.46 ± 3.27	88.55 ± 1.20	14.38 ± 1.74
MGDA	62.48 ± 1.33	95.96 ± 0.16	23.56 ± 3.76	88.51 ± 1.13	14.34 ± 1.70
PCGrad	62.51 ± 1.30	96.00 ± 0.15	24.06 ± 3.70	88.64 ± 1.15	13.57 ± 1.80
CAGrad	62.82 ± 1.27	95.89 ± 0.16	24.70 ± 4.71	88.56 ± 1.10	14.39 ± 1.54
NTKMTL	62.38 ± 1.41	95.73 ± 0.11	32.11 ± 4.38	89.35 ± 1.11	13.59 ± 1.55
ConFIG	62.38 ± 1.17	95.67 ± 0.11	31.84 ± 4.43	**89.36** ± 1.19	13.49 ± 1.70
MOIRE	**63.79** ± 1.05	**97.49** ± 0.08	**35.16** ± 1.08	88.35 ± 1.08	**15.04** ± 0.51

Table A.2: Raw downstream GNN results on Actor.

Model	ACC	AUC	Hits@20	METIS-ACC	NMI
LS	25.20 ± 0.65	81.29 ± 0.03	16.73 ± 0.17	**59.05** ± 1.27	0.43 ± 0.21
GradNorm	25.46 ± 0.60	78.08 ± 0.05	14.63 ± 0.39	53.82 ± 0.84	1.10 ± 0.19
UW	25.28 ± 0.55	80.92 ± 0.02	13.94 ± 0.13	57.32 ± 0.66	0.84 ± 0.10
DWA	25.50 ± 0.50	80.34 ± 0.08	16.77 ± 0.50	57.42 ± 0.53	0.77 ± 0.10
RLW	25.07 ± 0.56	80.06 ± 0.04	15.75 ± 0.20	58.47 ± 1.06	0.36 ± 0.10
MGDA	25.04 ± 0.54	81.22 ± 0.02	15.15 ± 0.20	58.79 ± 1.03	0.42 ± 0.15
PCGrad	25.14 ± 0.86	**81.78** ± 0.02	15.88 ± 0.27	58.94 ± 0.61	0.32 ± 0.22
CAGrad	25.39 ± 0.58	76.46 ± 0.09	13.22 ± 0.35	56.92 ± 0.91	0.85 ± 0.17
NTKMTL	25.22 ± 0.57	80.49 ± 0.02	13.86 ± 0.14	56.91 ± 0.58	1.06 ± 0.15
ConFIG	25.52 ± 0.47	80.71 ± 0.02	12.53 ± 0.07	57.16 ± 0.61	1.14 ± 0.07
MOIRE	**26.59** ± 0.73	79.01 ± 0.57	**17.68** ± 1.75	58.73 ± 0.66	**1.68** ± 0.06

Table A.3: Raw downstream GNN results on Squirrel.

Model	ACC	AUC	Hits@20	METIS-ACC	NMI
LS	44.72 ± 1.21	92.83 ± 0.05	8.48 ± 1.31	80.16 ± 0.77	2.80 ± 0.81
GradNorm	45.33 ± 0.67	93.85 ± 0.09	9.39 ± 0.86	**80.75** ± 0.51	3.20 ± 0.33
UW	44.91 ± 0.96	93.29 ± 0.15	8.91 ± 1.37	80.43 ± 0.78	2.73 ± 0.29
DWA	45.84 ± 1.16	93.03 ± 0.16	4.29 ± 2.57	80.02 ± 0.73	3.15 ± 0.62
RLW	45.54 ± 1.02	93.62 ± 0.23	13.98 ± 4.31	80.62 ± 0.52	2.84 ± 1.02
MGDA	45.12 ± 0.83	93.04 ± 0.07	6.97 ± 1.72	80.57 ± 0.82	2.75 ± 0.06
PCGrad	45.51 ± 1.08	93.15 ± 0.21	5.54 ± 1.67	80.23 ± 0.75	2.94 ± 0.38
CAGrad	45.41 ± 0.76	93.56 ± 0.20	10.68 ± 0.79	79.91 ± 0.67	3.26 ± 0.33
NTKMTL	44.01 ± 0.80	94.14 ± 0.05	13.00 ± 2.31	80.43 ± 0.64	2.31 ± 0.26
ConFIG	43.88 ± 1.01	93.52 ± 0.03	20.81 ± 0.47	80.45 ± 0.67	2.71 ± 0.20
MOIRE	**48.71** ± 0.93	**95.76** ± 0.08	**22.04** ± 3.05	79.41 ± 0.72	**3.85** ± 0.28

Table A.4: Raw downstream GNN results on PubMed.

Model	ACC	AUC	Hits@20	METIS-ACC	NMI
LS	86.66 ± 0.21	95.82 ± 0.10	20.85 ± 3.27	78.09 ± 0.22	26.50 ± 3.92
GradNorm	86.59 ± 0.19	95.39 ± 0.05	28.04 ± 2.32	77.20 ± 0.20	25.35 ± 4.89
UW	86.18 ± 0.19	95.44 ± 0.12	20.04 ± 3.00	77.40 ± 0.28	18.53 ± 5.03
DWA	86.70 ± 0.19	95.90 ± 0.07	22.19 ± 3.58	78.14 ± 0.16	24.80 ± 2.76
RLW	86.58 ± 0.22	**95.97** ± 0.06	21.97 ± 3.25	77.42 ± 0.30	27.15 ± 5.24
MGDA	86.61 ± 0.26	95.84 ± 0.08	21.88 ± 3.41	77.77 ± 0.29	24.15 ± 3.19
PCGrad	86.54 ± 0.19	95.89 ± 0.09	23.93 ± 4.16	78.21 ± 0.18	23.45 ± 5.10
CAGrad	86.57 ± 0.21	95.29 ± 0.06	23.78 ± 2.56	77.92 ± 0.30	22.01 ± 1.45
NTKMTL	86.78 ± 0.28	95.68 ± 0.05	28.18 ± 2.05	**78.88** ± 0.27	24.17 ± 5.68
ConFIG	86.74 ± 0.25	95.69 ± 0.03	**28.43** ± 1.57	78.60 ± 0.31	31.00 ± 7.19
MOIRE	**86.79** ± 0.23	**95.97** ± 0.14	27.71 ± 4.63	78.68 ± 0.38	**31.77** ± 0.01

A.1.2 SEQUENTIAL RECOMMENDATION

Shared note for Tables A.5–A.8. Entries are mean ± standard deviation for HIT@5/10/20 and NDCG@5/10/20. Scores are multiplied by 100 and rounded to three decimal places. Higher means are better. For each metric, the highest mean is bold and the second-highest distinct mean is underlined.

Table A.5: Raw downstream Rec results on Beauty.

Model	HIT@5	NDCG@5	HIT@10	NDCG@10	HIT@20	NDCG@20
LS	3.547 ± 0.158	2.239 ± 0.112	5.858 ± 0.152	2.981 ± 0.118	9.218 ± 0.242	3.825 ± 0.132
GradNorm	3.390 ± 0.280	2.169 ± 0.218	5.306 ± 0.403	2.785 ± 0.258	8.180 ± 0.656	3.506 ± 0.321
UW	3.599 ± 0.286	2.313 ± 0.203	5.778 ± 0.376	3.015 ± 0.230	8.914 ± 0.671	3.802 ± 0.305
DWA	3.570 ± 0.130	2.237 ± 0.109	5.946 ± 0.065	3.001 ± 0.080	9.216 ± 0.090	3.822 ± 0.079
RLW	3.647 ± 0.286	2.342 ± 0.199	5.961 ± 0.267	3.083 ± 0.183	9.340 ± 0.394	3.932 ± 0.213
MGDA	3.421 ± 0.090	2.147 ± 0.027	5.765 ± 0.170	2.901 ± 0.045	9.103 ± 0.397	3.741 ± 0.100
PCGrad	3.523 ± 0.199	2.239 ± 0.134	5.815 ± 0.231	2.974 ± 0.145	9.095 ± 0.244	3.797 ± 0.147
CAGrad	3.509 ± 0.254	2.200 ± 0.148	5.726 ± 0.366	2.910 ± 0.181	9.014 ± 0.324	3.736 ± 0.172
NTKMTL	3.637 ± 0.185	2.315 ± 0.115	5.925 ± 0.305	3.050 ± 0.146	9.252 ± 0.309	3.884 ± 0.146
ConFIG	**3.788** ± 0.091	**2.434** ± 0.047	**6.112** ± 0.048	**3.179** ± 0.037	**9.491** ± 0.064	**4.028** ± 0.044
MOIRE	3.700 ± 0.167	2.314 ± 0.109	5.988 ± 0.265	3.050 ± 0.140	9.362 ± 0.370	3.900 ± 0.164

Table A.6: Raw downstream Rec results on LastFM.

Model	HIT@5	NDCG@5	HIT@10	NDCG@10	HIT@20	NDCG@20
LS	4.844 ± 0.199	3.173 ± 0.069	7.872 ± 0.675	4.136 ± 0.216	12.312 ± 0.427	5.262 ± 0.143
GradNorm	3.706 ± 0.423	2.531 ± 0.236	5.376 ± 0.694	3.066 ± 0.275	8.330 ± 0.666	3.812 ± 0.245
UW	4.844 ± 0.300	3.276 ± 0.153	7.560 ± 0.280	4.146 ± 0.158	12.514 ± 0.201	5.383 ± 0.120
DWA	4.661 ± 0.255	3.069 ± 0.209	7.761 ± 0.294	4.062 ± 0.138	12.624 ± 0.221	5.287 ± 0.122
RLW	4.752 ± 0.447	3.023 ± 0.199	7.596 ± 0.386	3.920 ± 0.215	12.679 ± 0.375	5.202 ± 0.152
MGDA	5.211 ± 0.271	3.381 ± 0.205	8.073 ± 0.425	4.292 ± 0.173	**12.789** ± 0.694	**5.479** ± 0.126
PCGrad	4.697 ± 0.219	3.057 ± 0.074	7.651 ± 0.484	4.001 ± 0.146	11.927 ± 0.405	5.078 ± 0.124
CAGrad	4.917 ± 0.308	3.182 ± 0.175	7.541 ± 0.447	4.027 ± 0.258	11.486 ± 0.512	5.018 ± 0.231
NTKMTL	5.101 ± 0.461	3.338 ± 0.177	**8.275** ± 0.397	**4.357** ± 0.205	12.092 ± 0.346	5.316 ± 0.181
ConFIG	4.294 ± 0.219	2.675 ± 0.113	7.229 ± 0.327	3.615 ± 0.160	11.670 ± 0.521	4.729 ± 0.166
MOIRE	**5.303** ± 0.491	**3.416** ± 0.240	8.239 ± 0.574	4.350 ± 0.176	12.495 ± 0.278	5.422 ± 0.141

Table A.7: Raw downstream Rec results on ML-1M.

Model	HIT@5	NDCG@5	HIT@10	NDCG@10	HIT@20	NDCG@20
LS	11.841 ± 0.786	7.729 ± 0.405	19.162 ± 0.871	10.086 ± 0.426	30.007 ± 0.873	12.811 ± 0.421
GradNorm	11.589 ± 0.709	7.557 ± 0.456	19.616 ± 0.883	10.131 ± 0.524	30.235 ± 1.287	12.803 ± 0.623
UW	10.666 ± 1.038	6.824 ± 0.696	17.702 ± 1.424	9.075 ± 0.818	27.715 ± 1.979	11.594 ± 0.953
DWA	11.844 ± 0.633	7.698 ± 0.402	19.281 ± 0.496	10.084 ± 0.352	29.977 ± 0.833	12.769 ± 0.436
RLW	12.192 ± 0.416	7.925 ± 0.298	19.530 ± 0.389	10.280 ± 0.297	30.487 ± 0.431	13.036 ± 0.287
MGDA	11.722 ± 0.798	7.678 ± 0.419	18.864 ± 1.010	9.966 ± 0.484	29.288 ± 1.631	12.592 ± 0.642
PCGrad	11.874 ± 0.717	7.693 ± 0.469	19.126 ± 0.987	10.025 ± 0.548	29.868 ± 1.787	12.728 ± 0.747
CAGrad	11.844 ± 0.517	7.630 ± 0.287	19.603 ± 0.702	10.119 ± 0.355	30.013 ± 0.842	12.738 ± 0.392
NTKMTL	**12.609** ± 0.797	**8.194** ± 0.555	**20.192** ± 0.584	**10.624** ± 0.491	**30.881** ± 0.862	**13.315** ± 0.559
ConFIG	11.351 ± 0.269	6.430 ± 0.224	19.199 ± 0.378	8.948 ± 0.246	30.321 ± 0.317	11.743 ± 0.261
MOIRE	12.417 ± 0.395	7.998 ± 0.224	19.685 ± 0.656	10.328 ± 0.300	30.546 ± 0.728	13.065 ± 0.269

Table A.8: Raw downstream Rec results on Yelp.

Model	HIT@5	NDCG@5	HIT@10	NDCG@10	HIT@20	NDCG@20
LS	**1.954** ± 0.035	1.222 ± 0.027	3.282 ± 0.042	1.649 ± 0.027	5.408 ± 0.119	2.182 ± 0.027
GradNorm	1.597 ± 0.072	0.994 ± 0.051	2.733 ± 0.074	1.357 ± 0.033	4.530 ± 0.106	1.807 ± 0.031
UW	1.792 ± 0.067	1.113 ± 0.047	3.030 ± 0.069	1.509 ± 0.045	5.128 ± 0.098	2.034 ± 0.049
DWA	1.922 ± 0.056	1.206 ± 0.040	3.295 ± 0.113	1.647 ± 0.040	5.473 ± 0.144	2.193 ± 0.053
RLW	1.864 ± 0.072	1.183 ± 0.052	3.173 ± 0.070	1.603 ± 0.044	5.320 ± 0.203	2.140 ± 0.073
MGDA	1.936 ± 0.066	1.221 ± 0.039	3.320 ± 0.081	**1.664** ± 0.037	5.453 ± 0.124	2.199 ± 0.037
PCGrad	1.941 ± 0.045	**1.231** ± 0.032	3.268 ± 0.080	1.657 ± 0.046	5.462 ± 0.177	**2.207** ± 0.067
CAGrad	1.919 ± 0.095	1.195 ± 0.055	3.260 ± 0.065	1.625 ± 0.035	5.390 ± 0.123	2.160 ± 0.034
NTKMTL	1.911 ± 0.071	1.193 ± 0.045	3.265 ± 0.137	1.626 ± 0.056	5.414 ± 0.214	2.164 ± 0.062
ConFIG	1.870 ± 0.087	1.153 ± 0.047	3.213 ± 0.109	1.584 ± 0.052	5.330 ± 0.144	2.115 ± 0.045
MOIRE	1.922 ± 0.060	1.198 ± 0.026	**3.368** ± 0.068	1.662 ± 0.029	**5.520** ± 0.074	2.204 ± 0.032

Tables A.1–A.8 provide the complete downstream results for each metric.

A.2 Metric-wise comparisons with LS

Table A.9: Metric-wise wins over LS across GNN and Rec datasets. Each dataset cell counts the metrics for which the method's displayed mean exceeds the LS mean: five metrics per GNN dataset and six per Rec dataset, for 44 metrics in total. A metric tie adds no win to either method. "Metric wins" counts wins across all 44 metrics. A dataset win, tie, or loss compares the number of metric wins with the number of metric losses on that dataset; "Dataset W/T/L" reports these outcomes across the eight datasets. The comparison uses GNN means after multiplication by 100 and rounding to two decimal places, and Rec means after multiplication by 100 and rounding to three decimal places. These counts are distinct from mean rank (MR) in Tables A.10–A.11.

Method	Chameleon	Actor	Squirrel	PubMed	Beauty	LastFM	ML-1M	Yelp	Metric wins	Dataset W/T/L
GradNorm	3	2	5	1	0	0	3	0	14/44	2/1/5
UW	2	2	4	0	3	4	0	0	15/44	2/2/4
DWA	2	3	3	4	3	2	2	3	22/44	3/2/3
RLW	4	0	5	3	6	1	6	0	25/44	5/0/3
MGDA	3	0	3	2	0	6	0	4	18/44	4/0/4
PCGrad	3	1	4	3	0	0	1	4	16/44	4/0/4
CAGrad	4	2	4	1	0	2	4	0	17/44	3/0/5
NTKMTL	2	2	3	3	6	5	6	1	28/44	5/0/3
ConFIG	2	2	3	4	6	0	2	0	19/44	3/0/5
MOIRE	**4**	**3**	**4**	**5**	**6**	**6**	**6**	**4**	**38/44**	**8/0/0**

MOIRE exceeds LS on 38 of 44 displayed metrics and wins all eight dataset-level comparisons. Across the nine external methods, the source summary reports an average of 19.33 metric wins and 3.44 dataset wins; RLW and NTKMTL have the most dataset wins among them (five each). At the displayed precision, the source identifies three metric ties: one each for UW on Chameleon and LastFM, and one for PCGrad on Beauty.

A.3 Mean rank and relative gain

Following prior MTL evaluations (Navon et al., 2022; Ban & Ji, 2024), we summarize performance across downstream metrics using mean rank (MR). Within each dataset, methods are ranked by their mean score across repeated runs on each metric, with ties assigned average ranks. We average these metric-wise ranks within each dataset and weight datasets equally when computing domain-level MR.

Gain compares a method's arithmetic mean of the displayed metric means with that of LS on the same dataset: $(\bar{x}_{\text{method}}/\bar{x}_{\text{LS}} - 1) \times 100\%$. The four-dataset columns give an equally weighted arithmetic mean of the dataset gains within each domain. Lower MR and higher gain are better. Gains summarize metrics on their reported scales, so the per-metric tables and win counts remain important for interpreting them.

Table A.10: GNN dataset-level MR and gain relative to LS. MR is computed from ACC, AUC, Hits@20, METIS-ACC, and NMI using the displayed two-decimal means after multiplication by 100. Gain is calculated from the arithmetic mean of these rounded metric scores for each method divided by the corresponding LS mean, then expressed as a percentage. Four-dataset means weight Chameleon, Actor, Squirrel, and PubMed equally; MR values are rounded to two decimal places only after ranking and averaging, and gains are averaged before final two-decimal display. Negative gain denotes a lower mean score than LS. Dataset cells give MR / gain relative to LS. For each column, the lowest MR and highest gain are bold; the second-best values are underlined.

Model	Chameleon	Actor	Squirrel	PubMed	4-dataset mean
LS	6.90 / 0.00%	4.40 / 0.00%	8.60 / 0.00%	6.20 / 0.00%	6.53 / 0.00%
GradNorm	6.90 / +0.07%	7.00 / -5.26%	3.80 / +1.54%	7.20 / +1.51%	6.23 / -0.53%
UW	6.90 / -0.26%	6.20 / -2.41%	7.30 / +0.56%	10.40 / -3.35%	7.70 / -1.37%
DWA	6.60 / +0.01%	5.00 / -1.04%	7.20 / -1.16%	5.00 / -0.06%	5.95 / -0.56%
RLW	5.30 / +0.29%	7.60 / -1.64%	3.60 / +3.32%	5.90 / +0.38%	5.60 / +0.59%
MGDA	7.30 / -0.11%	6.40 / -1.14%	7.20 / -0.24%	7.20 / -0.54%	7.03 / -0.51%
PCGrad	5.70 / -0.14%	5.40 / -0.35%	6.80 / -0.71%	6.40 / +0.03%	6.08 / -0.29%
CAGrad	4.20 / +0.42%	8.00 / -5.40%	5.40 / +1.67%	8.60 / -0.76%	6.55 / -1.02%
NTKMTL	6.30 / +2.80%	7.20 / -2.82%	6.50 / +2.14%	4.00 / +1.87%	6.00 / +1.00%
ConFIG	6.90 / +2.65%	5.60 / -3.09%	6.60 / +5.41%	3.20 / +4.07%	5.58 / +2.26%
MOIRE	**3.00 / +5.14%**	**3.20 / +0.54%**	**3.00 / +9.07%**	**1.90 / +4.22%**	**2.78 / +4.74%**

Table A.11: Rec dataset-level MR and gain relative to LS. MR is computed from HIT@5, NDCG@5, HIT@10, NDCG@10, HIT@20, and NDCG@20 using the displayed three-decimal means after multiplication by 100. Gain follows the same within-dataset arithmetic-mean ratio to LS as Table A.10. Four-dataset means weight Beauty, LastFM, ML-1M, and Yelp equally; MR values are rounded to two decimal places only after ranking and averaging, and gains are averaged before final two-decimal display. Negative gain denotes a lower mean score than LS. Dataset cells give MR / gain relative to LS. For each column, the lowest MR and highest gain are bold; the second-best values are underlined.

Model	Beauty	LastFM	ML-1M	Yelp	4-dataset mean
LS	6.08 / 0.00%	5.42 / 0.00%	6.00 / 0.00%	3.67 / 0.00%	5.29 / 0.00%
GradNorm	10.83 / -8.43%	11.00 / -28.67%	5.83 / +0.32%	11.00 / -17.07%	9.67 / -13.46%
UW	6.67 / -0.89%	4.75 / +0.33%	10.67 / -8.80%	10.00 / -6.95%	8.02 / -4.08%
DWA	6.00 / +0.45%	5.83 / -0.36%	6.25 / +0.02%	3.75 / +0.25%	5.46 / +0.09%
RLW	2.50 / +2.30%	6.83 / -1.14%	3.33 / +1.98%	8.50 / -2.64%	5.29 / +0.13%
MGDA	9.33 / -2.13%	**2.00 / +4.32%**	8.83 / -1.67%	2.67 / +0.61%	5.71 / +0.28%
PCGrad	7.58 / -0.81%	7.67 / -3.16%	7.33 / -0.35%	**2.50** / +0.44%	6.27 / -0.97%
CAGrad	9.33 / -2.07%	7.33 / -3.80%	5.92 / +0.34%	6.67 / -0.94%	7.31 / -1.62%
NTKMTL	3.92 / +1.43%	3.17 / +2.34%	**1.00 / +4.56%**	6.17 / -0.79%	3.56 / +1.88%
ConFIG	**1.00 / +4.93%**	9.83 / -9.01%	8.83 / -3.98%	8.50 / -2.75%	7.04 / -2.70%
MOIRE	2.75 / +2.33%	2.17 / **+4.32%**	2.00 / +2.62%	2.58 / **+1.13%**	**2.38 / +2.60%**

Across the four GNN datasets, MOIRE has the lowest mean MR (2.78) and the largest mean gain over LS (+4.74%). It also has the lowest mean MR (2.38) and the largest mean gain (+2.60%) across the four Rec datasets. These aggregate advantages coexist with metric-level exceptions in Tables A.1–A.8.

A.4 Ablation results

The ablations compare the validation-selected residual readout with the MOIRE backbone, pretraining with and without Directed Cross-Objective Interaction, and MOIRE with LS at a matched parameter count. The MOIRE full results are the rows already reported in Tables A.1–A.8. All downstream means and standard deviations in this appendix are displayed as original score $\times 100$: GNN values use two decimal places and Rec values use three, with conventional half-up rounding applied separately to means and standard deviations. MR is calculated by ranking these displayed metric means among all 11 methods with averaged ranks for ties, then averaging the metric and dataset ranks; only the final MR values are rounded to two decimal places.

A.4.1 Backbone and residual readout

The backbone-only variant retains MOIRE's pretraining procedure but uses only the shared representation h for downstream prediction, without Residual Readout. The validation-selected residual readout enlarged its aggregate advantage. In Table A.12, the full mean rank (MR) is the original MOIRE ranking, while the backbone MR is obtained by replacing MOIRE's full scores with its backbone scores and reranking all 11 methods on each metric. Lower MR is better. The four-dataset GNN mean moved from 4.83 for the backbone to 2.78 for the full readout; the Rec mean

moved from 3.38 to 2.38. Tables A.13–A.14 give the backbone measurements; the corresponding full-model measurements appear in Tables A.1–A.8.

Table A.12: MOIRE mean rank with the full and backbone readouts. Full MR values use the 11-method ranking of the displayed metric means in Tables A.1–A.8. For the backbone MR, the rounded backbone scores in Tables A.13–A.14 replace the MOIRE row and all 11 methods are reranked on each metric, with averaged ranks for ties. Four-dataset means weight datasets equally. Every MR value is rounded half up to two decimal places only after its metric ranks and, where applicable, dataset means have been averaged.

Domain	Dataset	Full MR	Backbone MR
GNN	Chameleon	3.00	3.80
GNN	Actor	3.20	3.80
GNN	Squirrel	3.00	6.40
GNN	PubMed	1.90	5.30
GNN	Four-dataset mean	2.78	4.83
Rec	Beauty	2.75	2.75
Rec	LastFM	2.17	4.33
Rec	ML-1M	2.00	3.83
Rec	Yelp	2.58	2.58
Rec	Four-dataset mean	2.38	3.38

Table A.13: MOIRE GNN backbone downstream scores. Backbone source means and standard deviations were each multiplied by 100 and rounded half up to two decimal places, ordered as ACC, AUC, Hits@20, METIS-ACC, and NMI. Backbone MR is from Table A.12. Full-model scores in Tables A.1–A.4 use the same display scale.

Dataset	ACC	AUC	Hits@20	METIS-ACC	NMI	Backbone MR
Chameleon	62.87 ± 1.30	96.17 ± 0.09	28.61 ± 6.45	88.69 ± 1.22	12.52 ± 2.00	3.80
Actor	26.17 ± 0.75	80.40 ± 0.19	14.77 ± 1.20	59.22 ± 0.74	1.02 ± 0.03	3.80
Squirrel	45.14 ± 1.27	92.71 ± 0.03	15.92 ± 1.45	79.37 ± 0.79	3.21 ± 0.50	6.40
PubMed	86.85 ± 0.18	95.38 ± 0.20	23.42 ± 4.44	77.77 ± 0.32	30.68 ± 0.04	5.30

Table A.14: MOIRE Rec backbone downstream scores where backbone differs from the selected full result. Backbone source means and standard deviations were each multiplied by 100 and rounded half up to three decimal places. Backbone MR is from Table A.12. Full results in Tables A.6–A.7 use the same display scale. The source records Beauty and Yelp backbone results as identical to their validation-selected full results in Tables A.5 and A.8; their unchanged MR values are in Table A.12.

Dataset	HIT@5	NDCG@5	HIT@10	NDCG@10	HIT@20	NDCG@20	Backbone MR
LastFM	5.156 ± 0.480	3.172 ± 0.191	8.278 ± 0.484	4.164 ± 0.111	12.018 ± 0.673	5.102 ± 0.299	4.33
ML-1M	11.996 ± 0.355	7.774 ± 0.246	19.330 ± 0.497	10.122 ± 0.272	30.180 ± 0.628	12.854 ± 0.315	3.83

The validation procedure selected the backbone-only readout on Beauty and Yelp. Residual alternatives shifted some metrics slightly upward and others slightly downward without improving the overall validation choice. This is a selection boundary of the readout: when the backbone already serves the downstream task well, the backbone-only readout can be the best available choice. The pattern is consistent with the representation–downstream mismatch motivation in Fig. 2, but this ablation does not directly measure alignment on these datasets. The aggregate MR changes show the substantial overall contribution of residual access without requiring an improvement on every metric.

A.4.2 Directed interaction ablation

Removing directed interaction yielded task- and dataset-dependent differences. On Actor, the backbone without interaction exceeded the backbone with interaction on all five recorded GNN means (Table A.15); on Chameleon the directions were mixed.

Table A.15: GNN backbone scores with and without Directed Cross-Objective Interaction. For both configurations, source means and standard deviations were each multiplied by 100 and rounded half up to two decimal places. With-interaction rows repeat the corresponding backbone data in Table A.13. Without interaction rows are available only for Chameleon and Actor. The source also reports HMEAN4 and HMEAN5 means without standard deviations; those aggregates are omitted because this comparison uses the five downstream metrics. No 11-method MR was reported for without-interaction rows.

Dataset	Interaction	ACC	AUC	Hits@20	METIS-ACC	NMI
Chameleon	Disabled	63.93 ± 0.81	96.66 ± 0.21	27.96 ± 3.48	88.31 ± 0.86	14.83 ± 1.14
Chameleon	Enabled	62.87 ± 1.30	96.17 ± 0.09	28.61 ± 6.45	88.69 ± 1.22	12.52 ± 2.00
Actor	Disabled	26.49 ± 0.44	80.52 ± 0.08	16.76 ± 0.49	60.65 ± 1.08	1.75 ± 0.11
Actor	Enabled	26.17 ± 0.75	80.40 ± 0.19	14.77 ± 1.20	59.22 ± 0.74	1.02 ± 0.03

In recommendation, the backbone with interaction was higher on four of six recorded LastFM means and two of six ML-1M means (Table A.16). The source supplies no separate without-interaction rows for Beauty or Yelp. Thus, these results do not establish a uniform downstream improvement from the interaction configuration.

Table A.16: Rec backbone scores with and without Directed Cross-Objective Interaction. Source means and standard deviations were each multiplied by 100 and rounded half up to three decimal places. With-interaction rows repeat the corresponding backbone data in Table A.14. The comparison is limited to LastFM and ML-1M; no MR for the without-interaction variant against the 11-method main comparison was supplied.

Dataset	Interaction	HIT@5	NDCG@5	HIT@10	NDCG@10	HIT@20	NDCG@20
LastFM	Disabled	4.698 ± 0.256	3.150 ± 0.155	7.834 ± 0.354	4.154 ± 0.132	12.054 ± 0.460	5.222 ± 0.126
LastFM	Enabled	5.156 ± 0.480	3.172 ± 0.191	8.278 ± 0.484	4.164 ± 0.111	12.018 ± 0.673	5.102 ± 0.299
ML-1M	Disabled	11.904 ± 0.306	7.792 ± 0.228	19.506 ± 0.354	10.224 ± 0.241	30.110 ± 0.487	12.888 ± 0.271
ML-1M	Enabled	11.996 ± 0.355	7.774 ± 0.246	19.330 ± 0.497	10.122 ± 0.272	30.180 ± 0.628	12.854 ± 0.315

A.4.3 LS COMPARISON AT MATCHED PARAMETER COUNT

The parameter-matched LS experiment covered Chameleon, Actor, LastFM, and ML-1M. Tables A.17–A.18 report its means and standard deviations on the same score $\times 100$ scale as the MOIRE full rows of Tables A.1–A.2 and A.6–A.7. On Chameleon, MOIRE full was higher on four of five metrics, with matched LS higher on METIS-ACC. On Actor, MOIRE full was higher on ACC, Hits@20, and NMI, while matched LS was higher on AUC and METIS-ACC. On LastFM and ML-1M, MOIRE full was higher on all six displayed means.

Our comparison evaluates complete MTL methods under common domain-level training and evaluation protocols. MOIRE includes residual readout and its validation-based selection as part of the proposed method. The backbone-only ablation separately measures the contribution of this component, and its selection cost is reported alongside training cost.

Because downstream representation tuning reuses the pretrained checkpoint, it took approximately 35% of a MOIRE pretraining run in GNN and 43% in Rec. To ensure a consistent tuning budget, each compared model underwent one round of pretraining hyperparameter tuning.

Table A.17: Parameter-matched LS on GNN datasets. Source means and standard deviations were each multiplied by 100 and rounded half up to two decimal places, with metric order aligned to Tables A.1–A.2. MOIRE full uses the same display scale in Tables A.1–A.2.

Dataset	ACC	AUC	Hits@20	METIS-ACC	NMI
Chameleon	63.38 ± 1.11	95.79 ± 0.10	30.03 ± 3.20	89.29 ± 1.14	14.26 ± 1.48
Actor	25.38 ± 0.79	81.52 ± 0.11	15.63 ± 0.88	60.54 ± 0.80	0.23 ± 0.09

Table A.18: Parameter-matched LS on Rec datasets. Source means and standard deviations were each multiplied by 100 and rounded half up to three decimal places, with metric order aligned to Tables A.6–A.7. MOIRE full uses the same display scale there.

Dataset	HIT@5	NDCG@5	HIT@10	NDCG@10	HIT@20	NDCG@20
LastFM	4.587 ± 0.290	2.977 ± 0.084	7.486 ± 0.525	3.895 ± 0.103	12.055 ± 0.529	5.047 ± 0.139
ML-1M	11.285 ± 0.982	7.187 ± 0.644	18.884 ± 1.474	9.627 ± 0.793	29.791 ± 2.192	12.371 ± 0.972

A.5 COMPUTATIONAL COST

Tables A.19–A.21 report the measured run time and memory percentages for recommendation (Rec) and graph neural network (GNN) experiments. Method names follow the main text. The LS rows correspond to the baseline runs recorded as `s3rec` in Rec and `no_pareto` in GNN.

Table A.19: Rec run time and reported memory percentage. Time is the mean seconds per epoch over five measured epochs after one warm-up epoch. Memory values reproduce the recorded percentages. Dataset cells give time / reported memory percentage.

Model	LastFM	ML-1M	Yelp	Beauty
LS	18.63 / 5.10%	51.99 / 4.58%	375.43 / 6.04%	149.78 / 5.49%
MOIRE	21.92 / 8.46%	83.36 / 7.95%	412.07 / 9.43%	167.77 / 8.86%
MGDA	24.37 / 5.44%	122.18 / 4.91%	433.93 / 6.52%	184.27 / 5.90%
PCGrad	21.44 / 5.44%	90.01 / 4.91%	429.75 / 6.60%	176.20 / 5.90%
CAGrad	21.79 / 5.44%	91.93 / 4.91%	432.46 / 6.53%	176.54 / 5.90%
NTKMTL	21.55 / 5.44%	89.61 / 4.91%	427.42 / 6.53%	176.58 / 5.90%
ConFIG	21.28 / 7.17%	91.03 / 6.55%	434.61 / 15.02%	175.43 / 11.13%
DWA	17.95 / 5.10%	53.35 / 4.58%	377.22 / 6.04%	149.77 / 5.49%
RLW	18.50 / 5.10%	52.14 / 4.58%	378.16 / 6.04%	151.80 / 5.49%
UW	18.29 / 5.10%	52.91 / 4.58%	383.41 / 6.04%	151.09 / 5.49%
GradNorm	21.53 / 5.41%	92.92 / 4.89%	429.50 / 6.42%	184.33 / 5.83%

Table A.20: GNN node-task run time and reported memory percentage. Time is the mean seconds per step over 50 measured steps after 10 warm-up steps. Memory values reproduce the recorded percentages. Dataset cells give time / reported memory percentage.

Model	Squirrel	Actor	Chameleon	PubMed
LS	1.471 / 63.4%	0.385 / 16.0%	0.319 / 13.7%	1.253 / 71.0%
MGDA	2.016 / 63.4%	0.533 / 16.0%	0.424 / 13.6%	1.999 / 71.0%
MOIRE	1.470 / 65.7%	0.422 / 19.2%	0.354 / 14.9%	1.372 / 77.0%
PCGrad	1.980 / 63.6%	0.527 / 16.1%	0.414 / 13.9%	1.995 / 71.0%
CAGrad	2.001 / 63.6%	0.530 / 16.1%	0.417 / 13.8%	1.987 / 71.0%
DWA	1.437 / 63.4%	0.380 / 16.0%	0.319 / 13.7%	1.244 / 71.0%
RLW	1.430 / 63.4%	0.386 / 16.0%	0.319 / 13.7%	1.240 / 71.0%
NTKMTL	2.004 / 63.6%	0.530 / 16.1%	0.417 / 13.8%	1.989 / 71.0%
UW	1.435 / 63.4%	0.381 / 16.0%	0.320 / 13.7%	1.250 / 71.0%
GradNorm	2.154 / 75.5%	0.447 / 18.7%	0.353 / 15.7%	1.831 / 75.3%
ConFIG	2.025 / 63.6%	0.530 / 16.1%	0.418 / 13.8%	1.677 / 71.0%

Table A.21: GNN link-task run time and reported memory percentage. Time is the mean seconds per step over 50 measured steps after 10 warm-up steps. Memory values reproduce the recorded percentages. Dataset cells give time / reported memory percentage.

Model	Squirrel	Actor	Chameleon	PubMed
LS	0.749 / 46.6%	0.308 / 13.3%	0.273 / 10.7%	3.658 / 47.8%
MGDA	0.987 / 46.6%	0.443 / 13.3%	0.360 / 10.6%	4.857 / 47.8%
MOIRE	0.782 / 49.0%	0.346 / 16.3%	0.303 / 11.9%	4.638 / 52.6%
PCGrad	0.977 / 46.8%	0.441 / 13.4%	0.354 / 10.8%	6.858 / 47.8%
CAGrad	0.984 / 46.8%	0.443 / 13.4%	0.355 / 10.8%	6.846 / 47.8%
DWA	0.743 / 46.6%	0.356 / 13.3%	0.271 / 10.7%	3.366 / 47.8%
RLW	0.748 / 46.6%	0.325 / 13.3%	0.272 / 10.7%	3.408 / 47.8%
NTKMTL	0.982 / 46.8%	0.442 / 13.4%	0.354 / 10.8%	6.947 / 47.8%
UW	0.748 / 46.6%	0.309 / 13.3%	0.274 / 10.7%	3.376 / 47.8%
GradNorm	0.852 / 55.0%	0.369 / 15.6%	0.296 / 12.0%	3.606 / 51.5%
ConFIG	1.001 / 46.8%	0.442 / 13.4%	0.355 / 10.8%	6.799 / 47.8%

B EXPERIMENTAL DETAILS

B.1 BENCHMARKS, OBJECTIVES, AND COMPARISON SET

We study graph representation learning on Chameleon, Actor, Squirrel, and PubMed, and sequential recommendation on Beauty, LastFM, ML-1M, and Yelp. The graph setting uses the five self-supervised objectives from the ParetoGNN pipeline: reconstruction (Recon), MING, decorrelation (Decor), MinSG, and link prediction (Link) (Ju et al., 2023). The recommendation setting uses the four S^3-Rec pretraining objectives: associated attribute prediction (AAP), masked item prediction (MIP), masked attribute prediction (MAP), and segment prediction (SP) (Zhou et al., 2020). ML-1M is an additional dataset evaluated under our common recommendation protocol; it was not part of the original S^3-Rec benchmark set.

Within each domain, the methods use the same input data, base encoder, self-supervised task definitions, and downstream evaluation pipeline. We compare MOIRE with linear scalarization (LS),

multiple gradient descent algorithm (MGDA), PCGrad, CAGrad, ConFIG, NTKMTL, dynamic weight average (DWA), random loss weighting (RLW), uncertainty weighting (UW), and GradNorm. MOIRE adds objective-specific adaptation and interaction paths to the common encoder. LS is the fixed-loss reference in both domains, but its task coefficients differ by domain: graph LS sums the five unscaled losses with coefficient one for each term, whereas recommendation LS uses the S^3-Rec coefficients $(0.2, 1, 1, 0.5)$ for AAP, MIP, MAP, and SP. Thus, the graph LS coefficients sum to five and the recommendation LS coefficients sum to 2.7.

B.2 Shared pretraining configurations

The graph encoder is a two-layer graph convolutional network with hidden widths 512 and 256, zero dropout, and a per-device batch size of 1,024. The archived configuration sets 1,000 optimization steps, AdamW with weight decay 10^{-5}, and a fixed learning rate. The learning rate is 5×10^{-5} on Chameleon and 10^{-3} on Actor, Squirrel, and PubMed. Chameleon and PubMed use PReLU in the encoder; Actor and Squirrel do not. For the MinSG and MING objectives, the batch multipliers are both one, and the decorrelation sample size is 1,500 on all four datasets. The learning-rate and base-model settings were adapted from the ParetoGNN implementation, while these three sampling settings were unified across our graph datasets rather than copied dataset by dataset from its scripts. Although a warmup-step value remains in the configuration, the selected scheduler is fixed and does not perform learning-rate warmup.

Software-version incompatibilities in our hardware environment prevented us from using the original DGL setup. We therefore implemented a PyTorch-based compatibility script that reproduces the graph-sampling and convolution operations required by the ParetoGNN pipeline. All compared graph models use this same script for SSL pretraining, ensuring a shared graph implementation and matched comparison conditions.

The recommendation encoder is a two-layer Transformer with two attention heads, a hidden width of 64, maximum sequence length 50, and attention and hidden dropout probabilities of 0.5. Item masking uses probability 0.2. The four objectives retain the S^3-Rec coefficients above. The archived configuration sets 30 pretraining epochs, batch size 200, Adam with learning rate 10^{-3}, weight decay zero, and $(\beta_1, \beta_2) = (0.9, 0.999)$. Pilot experiments showed that recommendation performance had reached a plateau on two datasets and was close to plateauing on the other two by epoch 30. To maintain a uniform budget across the four datasets, we used 30 pretraining epochs for every compared method.

B.3 Multi-task baseline implementations

For gradient-coordination baselines, the graph shared-parameter set is the trainable graph encoder. In recommendation, it comprises the item and position embeddings, LayerNorm, and the item encoder; task-specific parameters follow their corresponding task losses. The method-specific settings are listed in Table B.1. They are common across the four datasets within each domain unless a domain distinction is stated.

Table B.1: Method-specific settings in the archived multi-task baseline configurations. These settings supplement the domain-level data, encoder, optimizer, and training budgets in Section B.2.

Method	Effective setting
MGDA	L2-normalized shared-task gradients and a minimum-norm convex combination solved by Frank–Wolfe (at most 250 iterations; tolerance 10^{-5}).
PCGrad	Random order for successive projections of conflicting task gradients.
CAGrad	Conflict-aversion coefficient $c = 0.5$; SciPy solver; shared update rescaled to the loss-sum magnitude.
ConFIG	Equal task weighting, projection-based length model, and least-squares direction calculation.
NTKMTL	Exponent 0.5 in the gradient-Gram-based task weighting.
DWA	Softmax temperature 2.0; loss history is updated per optimization step in graphs and by epoch-mean loss in recommendation.
RLW	Softmax weights from standard-normal random logits, sampled per training batch, with loss-sum-matched scaling.
UW	Learned log-variance objective $\sum_t [L_t/(2e^{s_t}) + s_t/2]$; graph s_t starts at -0.5, whereas recommendation initialization corresponds to its four fixed task coefficients.
GradNorm	Asymmetry exponent 1.5 and separate task-weight learning rate 0.025; graph weights start equally, whereas recommendation weights start from $(0.2, 1, 1, 0.5)$.

The recommendation adaptations preserve the S^3-Rec task coefficients while allowing each baseline to apply its own rule. PCGrad, CAGrad, NTKMTL, and ConFIG operate on task gradients that already include these coefficients. MGDA, DWA, and RLW combine their method weights with the task coefficients and rescale the resulting effective coefficients to a total of 2.7. GradNorm is initialized from the same coefficients and learns relative weights under that total budget. UW is

initialized to their corresponding effective weights and subsequently follows its learned uncertainty objective. In graphs, MGDA instead produces convex coefficients summing to one, whereas LS uses five unit coefficients. The common comparison condition is therefore the domain-level training and evaluation protocol, not an identical effective loss scale for every optimizer.

B.4 MOIRE IMPLEMENTATION SETTINGS

Both domains use one bottleneck adapter per objective with compression ratio four. Directed exchange uses source-state messages, a stopped-gradient source branch, and a null-source edge. Table B.2 lists the feedback and interaction settings used in pretraining; Tables B.3–B4 list the downstream residual-readout settings.

Table B.2: Archived MOIRE pretraining settings. The feedback probe is run without a gradient update on its cached batch.

Setting	Graph	Recommendation
Adapter compression ratio	4	4
Initial interaction strength / cap	0.05 / 0.10	0.05 / 0.20
Probe start	After step 200	After epoch 1
Probe interval	Every 10 steps	Every epoch
Cached batches per probe	1	4
Absolute / relative threshold	10^{-6} / 0.01	10^{-6} / 0.001
Feedback EMA coefficient	0.9	0.6
Feedback gain α	1.0	2.0
Bias / score clipping	0.5 / 1.0	2.0 / 3.0
Stale-feedback decay	0.995	0.995
Blocking threshold / confidence	0.5 / 0.8	0.5 / 0.8

For a shared backbone state $h \in \mathbb{R}^d$ and post-interaction objective states $\widetilde{B}_t$ at the same node or sequence position, the residuals and downstream interfaces are

$$r_t = \widetilde{B}_t - h, \qquad z_{\text{add}} = h + \sum_{t=1}^{K} c_t r_t, \qquad z_{\text{DS}} = [h; g_1 r_1; \ldots; g_K r_K].$$

The selected readout form and coefficients determine how the downstream reader uses the shared and objective-specific states.

B.4.1 GRAPH INSTANTIATION

The graph convolutional backbone is trained on five objectives: feature reconstruction (Recon), local–global discrimination (MING), cross-view alignment and decorrelation (Decor), node contrast across views (MinSG), and edge prediction (Link). Each objective head reads its post-interaction state from the graph view required by that objective. Node and link evaluations use their corresponding pretrained graph checkpoints and export frozen features for each node. Additive features have width d; Direct-sum features have width $6d$. A node evaluator or learned link reader is then trained on the exported features.

For graph MOIRE, a separate downstream readout stage can train on the exported pretrained features. Its archived configuration specifies at most 200 epochs, batch size 4,096, Adam with learning rate 0.05 and zero weight decay. The supplied training-edge pool is partitioned internally into fit, development, and held-out subsets in approximately $0.8/0.1/0.1$ proportions; the default epoch-selection metric is development-set AUC.

This Stage-2 configuration applies to MOIRE's readout training. It adjusts the readout on subsets drawn from the supplied training-edge pool and exports node and link representations at the selected epoch. The subsequent link-prediction evaluation fits a separate edge reader to those exported representations under the protocol in Section B.5. That reader uses learning rate 0.01, weight decay 0.01, and the predefined train/validation/test edge files; its validation AUC selects the reader epoch, and the predefined test edges supply the reported AUC and Hits@20. Appendix E.2 gives the corresponding validation calculations for residual-readout candidates. The internal Stage-2 development and held-out subsets are not the predefined validation and test edge splits.

For the final graph residual readout, let $g_t^{(b)} = k_b \alpha_t^{(b)}$ for branch $b \in \{\text{node}, \text{link}\}$. In the task order (Recon, MING, Decor, MinSG, Link), the normalized proportions are $\alpha^{(\text{node})} = (0.1, 0.6, 0.1, 0.1, 0.1)$ and $\alpha^{(\text{link})} = (0.1, 0.1, 0.6, 0.1, 0.1)$. Table B.3 reports the scale k_b for each

readout form and the form selected for the final node and link evaluations. Graph validation scores are calculated as described in Appendix E.

Table B.3: Final graph residual-readout settings. The normalized task proportions above are shared across datasets; k scales their sum for the indicated branch and readout form.

Dataset	Direct-sum k_{link}	Direct-sum k_{node}	Additive k_{link}	Additive k_{node}	Selected link	Selected node
Chameleon	1.1	1.6	1.1	1.5	Additive	Direct-sum
Actor	0.6	1.6	0.6	1.5	Additive	Additive
Squirrel	0.3	1.6	0.4	1.5	Additive	Direct-sum
PubMed	0.5	1.6	0.3	1.5	Direct-sum	Direct-sum

B.4.2 Recommendation instantiation

The Transformer backbone supports associated attribute prediction (AAP), masked item prediction (MIP), masked attribute prediction (MAP), and segment prediction (SP). Views with different masks or sampled segments are encoded separately before their objective states interact. For the current user prefix, the downstream model recomputes h and the post-interaction states under a causal mask. Additive supplies a d-dimensional sequence state to the item scorer. Direct-sum supplies one base block and four residual blocks, forming a logical $5d$-dimensional representation. For candidate item i, its score is

$$s(i) = \langle h, e_i \rangle + \sum_{t=1}^{4} g_t \langle r_t, P_t e_i \rangle,$$

where e_i is the candidate embedding and each P_t is a separate item map for one residual block. The score is computed from the five blocks without materializing an expanded item table. The recommendation reader and its connected representation path adapt under downstream supervision using the same training budget as the domain comparators. In both domains, the SSL heads supervise pretraining and the downstream reader consumes the selected residual interface.

Table B.4 lists the final recommendation readouts. Residual coefficients follow the task order (AAP, MIP, MAP, SP).

Table B.4: Final recommendation residual-readout settings. Coefficients apply to the selected form only.

Dataset	Selected readout	Residual coefficients (AAP, MIP, MAP, SP)
Beauty	Backbone	—
LastFM	Direct-sum	$(0.30, 0, 1.50, 0)$
ML-1M	Direct-sum	$(0, 1.50, 0, 0)$
Yelp	Backbone	—

B.5 Downstream training and evaluation

For graph evaluation, pretrained node representations are exported before fitting downstream readers. Node classification accuracy (ACC) and METIS-ACC use L2-normalized features and one-versus-rest logistic regression. Each evaluation split assigns 20% of nodes to fitting and 80% to testing. The regularization coefficient is selected from $C \in \{2^{-10}, 2^{-9}, \ldots, 2^{10}\}$ using five shuffle-split validation folds within the fitting partition. Link prediction uses the fixed train/validation/test edge files, a learned edge reader with Adam learning rate 0.01 and weight decay 0.01, and a one-to-one positive/negative training ratio. Training stops after at most 200 epochs with patience 50; validation AUC selects the checkpoint used to report test AUC and Hits@20.

For clustering, each evaluation seed uses the same 20%/80% node partition as node classification. KMeans is fitted to the L2-normalized representations of all nodes without using class labels, with the number of clusters set to the dataset's class count. Its assignments are scored on the 20% node pool for residual-gate and readout selection. Once that choice is fixed, reported NMI is scored only on the disjoint 80% node pool. The same split indices and scoring rule apply to every compared graph method. "NMI" denotes the implementation's `v_measure_score` with its default $\beta = 1$, equivalent to mutual information normalized by the arithmetic mean of the class-label and cluster-label entropies.

For sequential recommendation, each user's penultimate interaction is the validation target and the final interaction is the test target. The preceding interactions form the fine-tuning sequence. A checkpoint from epoch 30 initializes the downstream model. Fine-tuning uses batch size 256, Adam

with learning rate 10^{-3}, and at most 200 epochs; patience 10 and validation NDCG@20 select the checkpoint. Residual-readout candidates use this same validation split and full-catalogue ranking procedure. Evaluation ranks the target against the available item catalogue after masking previously observed items, and reports HIT and NDCG at cutoffs 5, 10, and 20. This is the full-catalogue ranking protocol also supported by the S^3-Rec repository, rather than the 99-negative-sample evaluation used for the main table in the original paper (Zhou et al., 2020). Results for each downstream measure are summarized by their mean and standard deviation over the recorded evaluations.

B.6 Metric aggregation and comparison rules

Graph ranking uses ACC, AUC, Hits@20, METIS-ACC, and NMI; recommendation ranking uses HIT and NDCG at 5, 10, and 20. To form the dataset-level mean rank (MR), each metric mean is multiplied by 100 and rounded to two decimal places for graphs or three for recommendation. The 11 methods are then ranked separately for each metric, with higher values preferred and tied values assigned their average rank. MR is the arithmetic mean of these metric-wise ranks, so lower MR is better. The domain-level MR is the equal-weight mean of its four dataset-level values.

For the LS-relative score summary, let $\widetilde{s}_{m,d,k}$ denote the same rescaled and rounded score for method m, dataset d, and metric k, and let K_d be the number of metrics. We use

$$\bar{s}_{m,d} = \frac{1}{K_d} \sum_{k=1}^{K_d} \widetilde{s}_{m,d,k}, \qquad G^{\mathrm{LS}}_{m,d} = 100 \left(\frac{\bar{s}_{m,d}}{\bar{s}_{\mathrm{LS},d}} - 1 \right).$$

The domain-level LS-relative gain is the arithmetic mean of the four dataset-level gains before display rounding. A method is counted as a dataset-level win over LS when it improves more constituent measures than it worsens; equal counts constitute a tie.

C Local Analysis of Objective Representation Coordination

This appendix separates two claims used in Section 3.2. The first is an exact statement about the backward paths created by directed messages and source stop-gradient. The second is a local, matched-model statement about the shared change required to fit an objective when additional adaptation directions are available. The latter describes an attainable solution, while the coverage audit measures the outcome of training.

C.1 Receiver-mediated gradients

For one objective j and one input view, let $h = f_\theta(x)$ and $B_j = S_j(h; \phi_j) = \mathrm{LN}_j(h + a_j(h))$. Write the post-interaction state as

$$y_j = B_j + \gamma \sum_{i \neq j} A_{i \to j} M_{i \to j}(\mathrm{sg}(B_i)), \qquad \widetilde{B}_j = N_j(y_j),$$

where N_j is the output LayerNorm. The null source, when present, contributes a zero message. A feature gate can be included in $M_{i \to j}$. We consider a forward pass in which the route weights, gates, interaction strength and stored feedback have no differentiable dependence on the current h, and source stop-gradient is enabled. Define $J_f = D_\theta f_\theta(x)$, $J_j = D_h S_j(h; \phi_j)$, $Q_j = D_{y_j} N_j(y_j)$, and $b_j = \nabla_{\widetilde{B}_j} L_j$. The contribution through the representation path to the shared-parameter gradient is then exactly

$$g^{\theta}_{j,\mathrm{repr}} = J_f^\top J_j^\top Q_j^\top b_j. \tag{C1}$$

Without source stop-gradient, the same forward values would transmit the additional donor-input contribution

$$\gamma \sum_{i \neq j} A_{i \to j} J_f^\top J_i^\top \left(D_{B_i} M_{i \to j} \right)^\top Q_j^\top b_j. \tag{C2}$$

Thus, the receiver uses the donor state in its forward computation, while the direct gradient from L_j through that donor input is removed. The receiver state and message transformations remain train-

able; a transformation shared by several outgoing edges can receive gradients from several receivers. A donor is also recomputed and trained by its own objective on later passes. Equations (C1)–(C2) identify backward paths at the current pass; they do not compare gradient magnitudes or freeze donor values throughout training. If an objective uses shared parameters outside this representation path, its full gradient is $g_j^\theta = g_{j,\mathrm{repr}}^\theta + g_{j,\mathrm{extra}}^\theta$. In recommendation, tied item embeddings used by the prediction interface can contribute to $g_{j,\mathrm{extra}}^\theta$; contributions from multiple views are summed.

C.2 Adaptation directions and minimum shared cost

We next fix an input, its sampling and stochastic realization, the current route configuration, and the donor input values. This defines a local prediction-fitting model aligned with the stopped donor-input path; it is not a model of the entire training trajectory. Let $e \in \mathbb{R}^n$ be a desired change in the receiver's prediction. Let u change the shared encoder, v_0 change the objective-specific head already available to the comparison method, and v_a change the additional ORC adaptation path. Their first-order effects in this matched model are

$$e \approx Uu + V_0 v_0 + V_a v_a. \tag{C3}$$

The columns of V_a may describe the objective adapter, its LayerNorm and admissible interaction parameters, with their actual parameter-sharing constraints retained. The comparison below holds e, U, and V_0 fixed on both sides; it does not equate the tangent spaces of separately trained checkpoints. One way to see what the additional path can absorb is to collect the permitted non-shared increments into v and write $e \approx Uu + Vv$. For positive analytical penalties λ, μ, consider

$$Q(u,v) = \frac{1}{2}\|e - Uu - Vv\|^2 + \frac{\lambda}{2}\|u\|^2 + \frac{\mu}{2}\|v\|^2. \tag{C4}$$

For fixed u, the unique minimizer is $v_*(u) = (V^\top V + \mu I)^{-1} V^\top (e - Uu)$. Substitution gives

$$\min_v Q(u,v) = \frac{1}{2}(e-Uu)^\top W (e-Uu) + \frac{\lambda}{2}\|u\|^2, \qquad W = I - V(V^\top V + \mu I)^{-1} V^\top. \tag{C5}$$

Along a nonzero left singular direction of V with singular value σ_k, W has eigenvalue $\mu/(\sigma_k^2 + \mu)$; on the orthogonal complement of the range of V, its eigenvalue is one. Hence $0 \prec W \preceq I$. In this surrogate, an error component expressible through adaptation receives less weight after the adaptation variables have been optimized. This is a directional statement about local fitting demand. The penalties in (C4) are analytical devices, not additional losses claimed to be optimized by MOIRE.

The attainable shared-cost result needs no quadratic penalty. Fix a common prediction tolerance $\epsilon \geq 0$, an admissible budget for each existing-head increment, an adaptation budget $r \geq 0$, and any common nonnegative cost $C(u)$. Define

$$\begin{aligned} \mathcal{C}_{\mathrm{fixed}}(\epsilon) &= \inf\big\{C(u) : \|e - Uu - V_0 v_0\| \leq \epsilon,\ v_0 \text{ admissible}\big\}, \\ \mathcal{C}_{\mathrm{adapt}}(\epsilon, r) &= \inf\big\{C(u) : \|e - Uu - V_0 v_0 - V_a v_a\| \leq \epsilon, \quad v_0 \text{ admissible},\ \|v_a\| \leq r\big\}. \end{aligned} \tag{C6}$$

Every solution feasible for $\mathcal{C}_{\mathrm{fixed}}$ remains feasible for $\mathcal{C}_{\mathrm{adapt}}$ by setting $v_a = 0$. With $\inf \varnothing = +\infty$, this proves

$$\boxed{\mathcal{C}_{\mathrm{adapt}}(\epsilon, r) \leq \mathcal{C}_{\mathrm{fixed}}(\epsilon).} \tag{C7}$$

For $C(u) = \|u\|^2$, (C7) says that the *minimum attainable* squared shared increment at equal local prediction accuracy cannot increase. The inequality can be strict: if a desired correction lies in the admissible range of V_a, it can be fitted with $u = 0$, whereas fitting it through the fixed-path model may require a nonzero shared increment. Both sides retain the existing head v_0; the result attributes the new freedom to ORC's additional path rather than to head parameters already present in the comparison method. It is a feasible-set result, not a guarantee that stochastic optimization selects the minimum-cost solution.

C.3 When a shared change affects a fixed readout

Let $F_t(w)$ be the loss of a fixed target-t auditor reading only the backbone representation, with its data and evaluation randomness held fixed. The shared state w includes any mutable encoder state relevant to that readout. For a source-conditioned shared change u_s, let $a_t = \nabla F_t(w)$, and suppose F_t has a β_t-Lipschitz gradient on the segment from w to $w+u_s$. Taylor's theorem with a smoothness remainder gives

$$F_t(w+u_s) - F_t(w) = a_t^\top u_s + R_t(u_s), \qquad |R_t(u_s)| \le \rho_t(u_s) := \frac{\beta_t}{2}\|u_s\|^2. \tag{C8}$$

For the positive relative damage of one directed source-to-target event, with its fixed positive audit denominator d_t, this implies

$$\frac{[a_t^\top u_s - \rho_t(u_s)]_+}{d_t} \le H_{s\to t} := \frac{[F_t(w+u_s) - F_t(w)]_+}{d_t} \le \frac{[a_t^\top u_s + \rho_t(u_s)]_+}{d_t}, \tag{C9}$$

where $[z]_+ = \max(z, 0)$. The audit is directed because its target-specific gradient a_t is paired with a source-conditioned change u_s. For two candidate changes u_A and u_B from the same anchor, using the same auditor and denominator, a sufficient condition for strictly less observed damage under u_A is

$$[a_t^\top u_A + \rho_t(u_A)]_+ < [a_t^\top u_B - \rho_t(u_B)]_+. \tag{C10}$$

The local adaptation model supplies a concrete way this condition can become attainable. Suppose a fixed-path fit uses $u_B = u_A + d$, and an admissible ORC increment satisfies $V_a v_a = Ud$. Then replacing d by v_a preserves the same first-order source prediction fit. If d points in a harmful direction for the fixed target readout and the resulting separation exceeds the smoothness remainders as in (C10), the adapted solution has strictly less directed damage. This is the positive mechanism behind the minimum-cost result: an objective can meet a fitting requirement while avoiding a shared change that would disrupt an established readout.

The condition is deliberately local and matched. The inequality in (C7) concerns attainable fits, while (C10) concerns two specified shared changes under one fixed auditor. Neither asserts that the optimizer finds the adapted solution or that a smaller shared norm alone implies less damage. In the source-only audit of Section 4, ORC is bypassed during the five-step intervention; the audit tests the backbone state learned during prior ORC training. Accordingly, the measured relative HarmMass in Fig. 4 is empirical evidence about the trained models, rather than a numerical consequence of (C7) or (C10).

D Cross-Objective Damage Audits

D.1 Fixed readouts of the shared backbone

Both audit protocols measure changes in native target SSL losses through fixed readouts of the shared backbone representation h. For a pretrained anchor a, audit condition g, and target objective t, $L_t^{a,g}(w)$ denotes the audit loss evaluated with candidate backbone state w. Non-backbone audit parameters remain fixed at their anchor values throughout each comparison. The graph auditor reads h, retains the target head, and substitutes the candidate backbone state. The recommendation auditor reads the candidate sequence-encoder output using anchor-frozen prediction parameters and output item and attribute embeddings. All auditors operate in evaluation mode.

Graph update, calibration, and audit batches use independently seeded views of the same training graph, with potentially overlapping node sets. Identical target batches across roles are excluded. Recommendation uses distinct sampled pretraining examples for the corresponding roles. The audit bank remains fixed within each anchor and condition. The graph auditor takes the unweighted arithmetic mean of the target losses over audit batches. The recommendation auditor sums the native loss numerators and valid-term counts across audit batches before dividing.

The graph targets are link-prediction binary cross-entropy, MING positive/negative discrimination, MinSG two-view contrastive loss, decorrelation loss, and masked-feature reconstruction loss. The recommendation targets are associated attribute prediction (AAP), masked item prediction (MIP),

masked attribute prediction (MAP), and segment prediction (SP). AAP and MAP divide their attribute binary cross-entropy sums by the number of eligible positions times the attribute dimension; MIP divides by masked positions; SP divides by sequences. The recommendation training branch applies the pretraining coefficients $(0.2, 1, 1, 0.5)$ to the corresponding source objectives. Target audits use the unweighted native losses with these reductions.

D.2 Source-only continuation

From each anchor, a separate branch performs five optimizer updates using one source objective s. Source-only continuation updates the shared backbone through h and the source's native prediction interface; source-specific trainable parameters follow that interface. The target auditor remains fixed at the anchor. Graph branches restore the anchor's optimizer and scheduler states. Recommendation branches use the raw sequence encoder and a newly initialized optimizer. Let $L^{a,g}_{t,0} > 0$ be the target audit loss at the anchor and $L^{a,g}_{t,s,5}$ the loss after the five source-only updates. All initial target losses in the reported audits are strictly positive. The signed response and its positive part are

$$r^{\mathrm{single},a,g}_{s\to t} = \frac{L^{a,g}_{t,s,5} - L^{a,g}_{t,0}}{L^{a,g}_{t,0}}, \qquad d^{\mathrm{single},a,g}_{s\to t} = \left[r^{\mathrm{single},a,g}_{s\to t}\right]_+, \quad [x]_+ = \max(x, 0). \tag{D1}$$

Equation (D1) is the normalization used in Figs. 1 and 4 and in the source-only figures below. A positive response means that continuation on s raised the fixed readout loss of t. The figures retain only off-diagonal pairs, $s \neq t$. Panels (a,b) of Fig. D.1 add the Actor overview; Figs. D.2, D.4, and D.6 show the full directed matrices for Chameleon, Actor, and ML-1M.

D.3 Coordinated-update protocol

Each five-step window starts from a pretrained anchor. Graph windows restore the anchor's model, optimizer, and scheduler states; recommendation windows use the anchor model and newly initialized Adam state. At each step k, the full update and its one-step control start from the same current pre-step model and optimizer state, including scheduler state for graph branches. Task contributions are computed once using the full objective set. The control omits the stored contribution of source s, with the remaining contributions unchanged. Only the full branch advances to the next step.

Let G^{full}_k be the parameter-aligned gradient supplied to the optimizer after full-task coordination, and let $P_{s,k}$ be the stored contribution assigned to source s. These contributions satisfy $\sum_s P_{s,k} = G^{\mathrm{full}}_k$. The control supplies $G^{\mathrm{full}}_k - P_{s,k}$ to the optimizer, with subtraction performed before the optimizer step. Loss-weighting methods use their full-task effective weights; projection-based methods retain their full-task coordinated components. ConFIG uses the recorded task-projection components along its full coordinated direction; NTKMTL uses the recorded full-task weighted-gradient components, including the aggregation scale. On task-private parameter blocks, the contribution follows the corresponding task loss.

Let $L^{\mathrm{full},a,g}_{t,k}$ and $L^{-s,a,g}_{t,k}$ denote the fixed-backbone-auditor losses after the paired updates. For Figs. 4c and 4f, the directed response at step k is

$$r^{\mathrm{joint},a,g}_{s\to t,k} = \frac{L^{\mathrm{full},a,g}_{t,k} - L^{-s,a,g}_{t,k}}{L^{a,g}_{t,0}}, \qquad d^{\mathrm{joint},a,g}_{s\to t,k} = \left[r^{\mathrm{joint},a,g}_{s\to t,k}\right]_+. \tag{D2}$$

Here $L^{a,g}_{t,0} > 0$ is the target's audit loss at the start of the five-step coordinated-update window. A positive response means that retaining s's coordinated contribution raised the target loss relative to the matched one-step control. Panel (c) of Fig. D.1 gives the Actor overview, and Figs. D.3, D.5, and D.7 display the directed coordinated-update responses for Chameleon, Actor, and ML-1M.

D.4 Event-to-figure aggregation

The positive part is taken **for each directed event before aggregation**. If $\mathcal{T}$ is the domain's objective set, the source-only and five-step coordinated-update masses for an anchor and condition are

$$H^{a,g}_{\mathrm{single}} = \sum_{s,t\in\mathcal{T}:\, s\neq t} d^{\mathrm{single},a,g}_{s\to t}, \qquad H^{a,g}_{\mathrm{joint}} = \frac{1}{5}\sum_{k=1}^{5}\sum_{s,t\in\mathcal{T}:\, s\neq t} d^{\mathrm{joint},a,g}_{s\to t,k}. \tag{D3}$$

Each graph mass sums 20 directed off-diagonal pairs; each recommendation mass sums 12. Their absolute magnitudes therefore have domain-specific scales. The displayed method-level mean first averages the five random conditions for each of eight anchors, then gives the eight anchors equal weight:

$$\overline{H}_{\bullet} = \frac{1}{8}\sum_{a=1}^{8}\frac{1}{5}\sum_{g=1}^{5} H_{\bullet}^{a,g}, \qquad \bullet \in \{\text{single}, \text{joint}\}. \tag{D4}$$

In the directed displays, colour shows mean event-level positive damage. The source-only matrices average 40 anchor–condition events per pair, whereas the coordinated-update matrices average 40 conditions $\times$ five matched steps per pair. Circle size in panel (b) of Fig. D.1 shows the fraction of conditions with positive damage. The coordinated-update bars in Fig. 4 and panel (c) of Fig. D.1 average the five per-step masses in (D3).

D.5 Additional damage patterns

The Actor overview extends the Chameleon and ML-1M comparisons in Fig. 4. MOIRE has the lowest mean source-only HarmMass and coordinated-update damage among the nine methods on Actor (Fig. D.1). The directed plots below show which source–target pairs contribute to these totals.

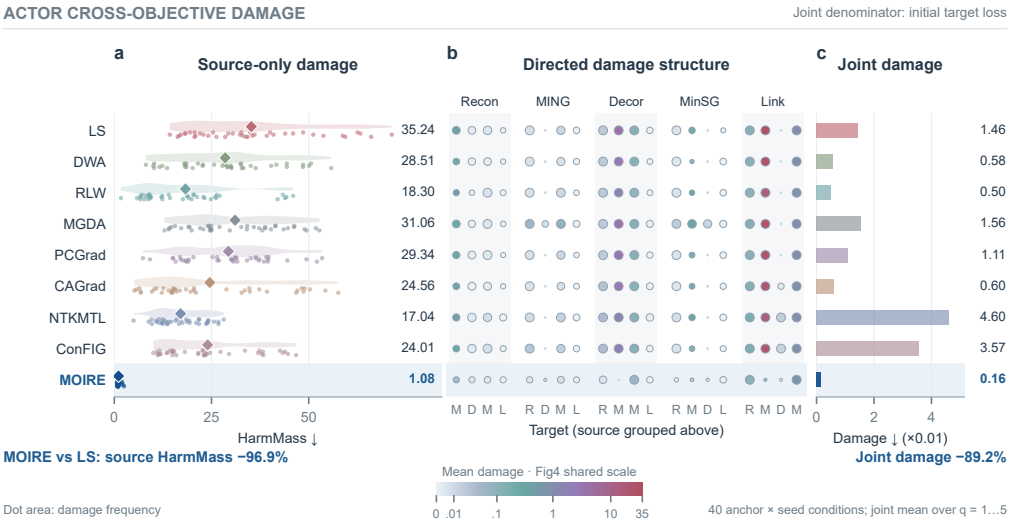


Figure D.1: Cross-objective damage on Actor. (a) Source-only HarmMass after five updates: points are the 40 anchor–condition values for each method, and diamonds mark their means. (b) Directed source-to-target damage under source-only continuation: colour gives mean positive relative target-loss change and circle area gives damage frequency over the 40 conditions. Source objectives group the target letters above each block. (c) Mean coordinated-update damage over five matched updates, with the initial target loss as denominator. Nine methods share the Fig. 4 colour scale in (b).

Source-only directed matrices. Figs. D.2, D.4, and D.6 expand the source-only directed display in Fig. 4 to all off-diagonal objective pairs. On Chameleon and Actor, the Link-to-MING cell is prominent for the comparison methods. On ML-1M, changes to AAP account for much of the displayed damage. Each plate uses one colour scale across its nine methods.

Coordinated-update directed matrices. Figs. D.3, D.5, and D.7 show the directed structure when a source's already-coordinated contribution is removed from each matched update. The cell values average the positive loss contrast over five steps and 40 conditions. These plates can be read alongside the source-only matrices above.

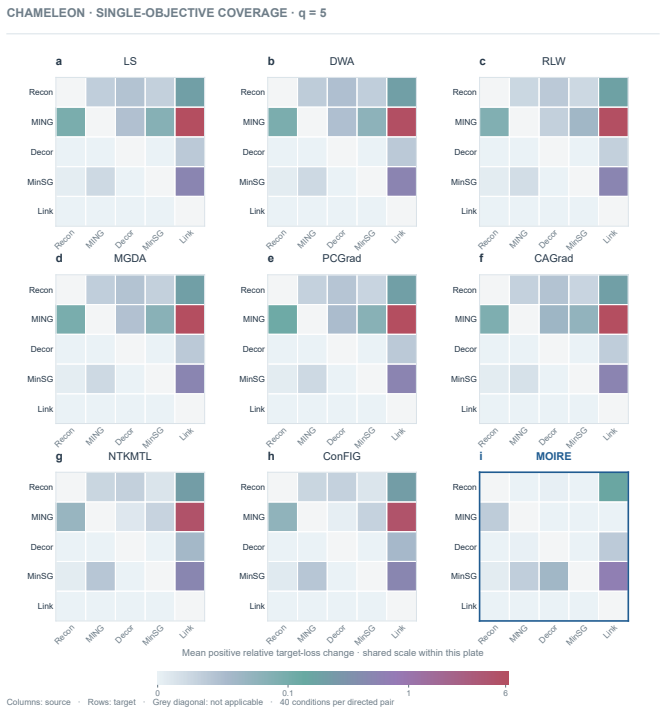


Figure D.2: Source-only directed damage on Chameleon. Columns are sources and rows are targets. Each off-diagonal cell averages positive relative frozen-target-loss change after five source-only updates over 40 anchor–condition values. Panels (a–i) show LS, DWA, RLW, MGDA, PCGrad, CAGrad, NTKMTL, ConFIG, and MOIRE on one colour scale.



Figure D.3: Coordinated-update directed damage on Chameleon. Each off-diagonal cell averages positive target-loss contrast between the full update and the control omitting the source's stored coordinated contribution over 40 conditions and five matched steps, normalized by the initial target loss. Panels (a–i) use the method order of Fig. D.2 and one colour scale.

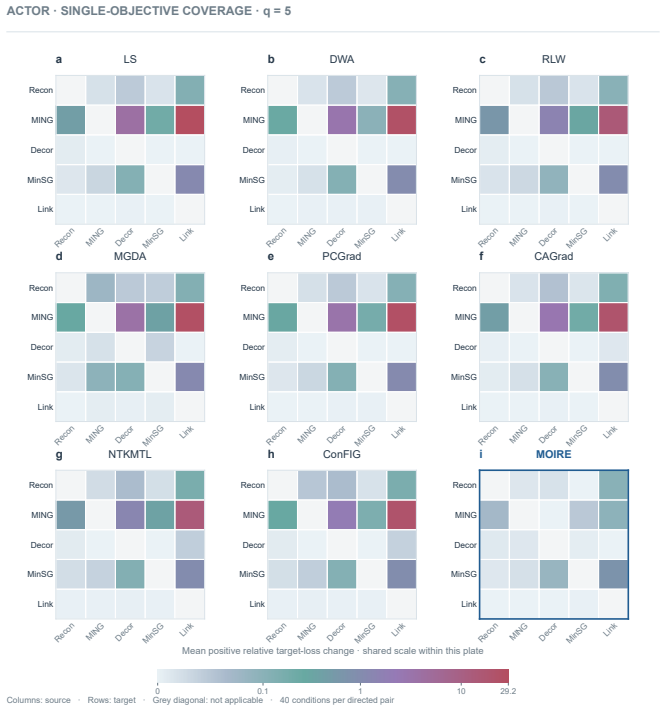


Figure D.4: Source-only directed damage on Actor. The layout and metric follow Fig. D.2; each cell averages 40 anchor–condition values, and all nine methods share one colour scale.



Figure D.5: Coordinated-update directed damage on Actor. The layout and metric follow Fig. D.4; each cell averages 40 conditions and five matched steps, and all nine methods share one colour scale.



Figure D.6: Source-only directed damage on ML-1M for AAP, MIP, MAP, and SP. The layout and metric follow Fig. D.2; each cell averages 40 anchor–condition values.



Figure D.7: Coordinated-update directed damage on ML-1M for AAP, MIP, MAP, and SP. The layout and metric follow Fig. D.4; each cell averages 40 conditions and five matched steps.

E Residual Readout Audits and Validation

E.1 Harmful-mass recovery

Figure 5a measures the share of cross-objective damage recovered by reading MOIRE's objective-specific states. The audit uses the source-only continuations and frozen native target-loss readers described in Appendix D. For each anchor, audit group, and source–target pair $s \neq t$, let q_0 denote the anchor and q_5 the state after five updates on s alone. On the same audit batches, $L_H(q)$ is the target loss from the plain backbone representation and $L_C(q)$ is the loss from residual reader C. Define

$$z = \max\{|L_H(q_0)|, 10^{-6}\},$$
$$D = \left[\frac{L_H(q_5) - L_H(q_0)}{z}\right]_+,$$
$$G_j = \frac{L_H(q_j) - L_C(q_j)}{z} \quad (j \in \{0, 5\}).$$

$$M_C = \min\big\{D,\, [G_5 - [G_0]_+]_+\big\}, \qquad [x]_+ = \max(x, 0).$$

D is positive damage to the backbone readout. M_C subtracts any positive advantage of C at q_0 and credits at most the damage on that edge. For each anchor–group unit, $\mathrm{HMR}_C = \sum_{s\neq t} M_C / \sum_{s\neq t} D$ when the denominator exceeds 10^{-8}. Figure 5a averages the five group ratios within each anchor and then averages the eight anchors equally.

The comparison uses independently calibrated, target-specific, bias-free structured heads with the same parameter count, a rank-8 map per residual slot, and an identity path for the shared representation h. Using the post-interaction states $\widetilde{B}_t$, set $r_t = \widetilde{B}_t - h$, $s_t = g_t r_t$, and $m = \sum_t s_t$. Direct-sum supplies $[h, s_1, \ldots, s_T]$ to its head; Additive supplies $[h, m/T, \ldots, m/T]$, with T repeated residual slots, to its matched head. Both heads return the original representation width to the frozen target-loss reader.

The gate coefficients sum to one: four coefficients of 0.25 in recommendation and five of 0.20 in graph node learning. Each target head is fitted on q_0 and source-only q_5 states for 36 AdamW steps (learning rate 10^{-3}, weight decay 10^{-4}) using four calibration batches; four separate batches are used for the audit. Both readers use the same saved branches and audit batches. ML-1M and LastFM use anchors at epochs $9, 12, \ldots, 30$ and group seeds 41–45; Chameleon and Actor use anchors at steps $300, 400, \ldots, 1000$ and group seeds 31001–31005. Each dataset has 40 anchor–group units, with 12 off-diagonal source–target pairs per recommendation unit or 20 per graph unit.

E.2 Validation metrics for graph residual readout

Figure 5b,c and the graph residual-readout selection use the same validation calculations. Each candidate exports L2-normalized node representations from the node and link checkpoints. Node representations supply classification, clustering, and METIS-ACC scores; link representations supply edge-prediction scores. For Chameleon, `train_test_split(test_size=0.8)` assigns 455 of its 2,277 nodes to node validation calculations and the remaining 1,822 to node testing. The same 455 node indices are used for validation ACC, NMI, and METIS-ACC. Link prediction uses the predefined training and validation edge splits.

For link prediction, a one-layer linear predictor scores an edge from the elementwise product of its endpoints' normalized representations. It is trained on the training edges with binary cross-entropy and Adam (learning rate 0.01, weight decay 0.01; batch size $10{,}240$, with incomplete batches dropped) for up to 200 epochs. Validation ROC-AUC selects the predictor epoch, with patience 50 and the later epoch retained for an AUC tie. At that epoch, validation Hits@10 is the fraction of validation positive edges whose scores strictly exceed the tenth-highest score among **all** validation negative edges. On Chameleon, the validation edge split contains 3,137 positive and 3,137 negative edges.

For residual-gate and readout selection targeting test Hits@20, we use validation Hits@10; validation ROC-AUC is used separately to select the training epoch of each link predictor. Hits@K compares each positive edge score with the Kth-highest score among the negative edges in the same split. The test negative pool is approximately twice the validation pool on Chameleon (6,274 ver-

sus 3,137), PubMed (8,864 versus 4,432), Squirrel (39,670 versus 19,835), and Actor (5,331 versus 2,665). Thus, $10/N_{\text{val}} = 20/N_{\text{test}}$ for the first three datasets, with only a negligible difference on Actor. Validation Hits@10 and test Hits@20 therefore use nearly identical relative rank cutoffs; validation Hits@20 would use a cutoff approximately twice as permissive. Matching the cutoffs does not imply equal Hits values across the two splits.

For node classification, one-versus-rest logistic regression (`liblinear`, at most 10,000 iterations) uses the normalized node representations and searches $C \in \{2^{-10}, 2^{-9}, \ldots, 2^{10}\}$. `ShuffleSplit(n_splits=5, test_size=0.5)` partitions the 20% node pool five times, assigning half of that pool to validation on each split. Mean exact-match accuracy on one-hot labels selects C, and the best mean cross-validation score is validation ACC. Validation METIS-ACC uses the same procedure with the saved graph-partition pseudo-labels; ACC and METIS-ACC select C independently. The Chameleon graph-partition labels have ten classes. For validation NMI, KMeans is fitted to the normalized representations of all nodes without using class labels, with the number of clusters set to the number of original node classes. Its assignments are scored against the original classes on the 20% node pool using V-measure, $2I(Y;C)/(H(Y)+H(C))$. After the residual gate and readout form have been fixed using validation scores, test NMI uses the same clustering assignments but scores only the disjoint 80% node pool. No test-node class labels are used to choose the gate, readout form, or KMeans initialization.

Residual gates and readout forms are selected separately for each downstream metric using its corresponding validation score. The stars in Fig. 5b,c show those metric-specific choices at their test scores; the final graph settings are listed in Table B.3.

REFERENCES

Hao Ban and Kaiyi Ji. Fair resource allocation in multi-task learning. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 2715–2731, 2024.

Mingxuan Ju, Tong Zhao, Qianlong Wen, Wenhao Yu, Neil Shah, Yanfang Ye, and Chuxu Zhang. Multi-task self-supervised graph neural networks enable stronger task generalization. In *International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=1tHAZRqftM`.

Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 16428–16446, 2022.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S^3-Rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, 2020. URL `https://arxiv.org/abs/2008.07873`.

CODE AVAILABILITY

If the paper is accepted, we will publicly release the source code.