

H-PLE: Algorithmic modeling of recommendation systems based on hierarchical PLE

Jinghao Xue

jhxue@keio.jp

Keio University

Tianxiang Yang

Keio University

Hideo Suzuki

Keio University

Research Article

Keywords: Multitask Learning, Recommendation Systems, Hierarchical Modeling, Negative Transfer

Posted Date: December 4th, 2025

DOI: <https://doi.org/10.21203/rs.3.rs-8114131/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

H-PLE: Algorithmic modeling of recommendation systems based on hierarchical PLE

Jinghao Xue^{1*}, Tianxiang Yang^{2†} and Hideo Suzuki^{2†}

^{1*}Graduate School of Science and Technology, Keio University,
Kohoku-ku, Yokohama, 223-0061, Kanagawa, Japan.

²Department of Industrial and Systems Engineering, Keio University,
Kohoku-ku, Yokohama, 223-0061, Kanagawa, Japan.

*Corresponding author(s). E-mail(s): jhxue@keio.jp;

Contributing authors: yangflyhigher@gmail.com; hsuzuki@ae.keio.ac.jp;

[†]These authors contributed equally to this work.

Abstract

In this study, we propose H-PLE (Hierarchical Progressive Layered Extraction), a novel multi-task recommendation framework designed to alleviate negative transfer and enhance performance under extreme label imbalance. By introducing a hierarchical expert structure combined with improved gating and feature interaction modules, H-PLE effectively balances shared and task-specific knowledge extraction. We evaluate the model on large-scale real-world datasets and compare it against established baselines such as MMoE and PLE. The experimental analysis demonstrates that H-PLE consistently achieves stronger ranking performance, particularly in sparse-label tasks, while maintaining stable optimization behavior across multiple runs. Moreover, we reveal a practical gap between high ranking metrics and low decision-level metrics under default thresholding, and provide solutions through probability calibration, validation-based threshold selection, and top-K or rate-controlled strategies. These findings not only highlight the methodological advantages of H-PLE but also contribute actionable insights for bridging the gap between offline evaluation and online deployment in large-scale recommendation systems.

Keywords: Multitask Learning, Recommendation Systems, Hierarchical Modeling, Negative Transfer

1 Introduction

Integrating multitask learning (MTL) into recommendation systems has become increasingly crucial owing to the need for personalized user experiences simultaneously across multiple interaction dimensions [1] and [2]. MTL leverages information from related tasks, enhancing generalization while improving recommendation systems' overall performance [3]. This approach is particularly effective in complex environments where different types of user feedback, such as clicks, purchases, or views, must be predicted concurrently.

However, MTL models often grapple with the phenomenon of negative transfer, where improvement in one task may detrimentally affect the performance of another [4],[5]. This seesaw effect poses significant challenges in environments with intertwined recommendation tasks, requiring sophisticated solutions to balance the inter-task influences effectively.

To address these challenges, this study introduced and investigated a hierarchical progressive layered extraction (H-PLE) model, an innovative extension of the progressive layered extraction (PLE) method [7]. Our model enhances the PLE architecture by incorporating hierarchical processing layers that refine the extraction and integration of task-specific knowledge, managing the shared and task-specific components more effectively. The hierarchical structure allows for nuanced knowledge processing, which is crucial for disentangling complex interdependencies in recommendation systems [7], [8].

The H-PLE model was validated on the large-scale QK video dataset (Tenrec) obtained from the Tencent online video platform, and demonstrated substantial accuracy and user engagement improvements over existing MTL models. The results underscore the model's ability to mitigate negative transfer and amplify the benefits of multitask learning in a real-world context.

2 Related Work

We reviewed foundational models and recent advances in MTL, focusing on their application in complex systems like recommendation engines. We discuss the traditional MTL frameworks, innovative models, e.g., multigate mixture-of-experts (MMoE) and PLE, and feature interaction such as factorization machines (FMs) that enhance feature interaction in these models.

Supervised deep learning and machine learning fundamentally involve inputting data, predicting outcomes based on this input data, and adjusting model parameters during training according to differences (loss) between the predicted and supervised signals. This adjustment aims to align the model parameters as closely as possible with the data distribution. Multitask learning, Multitask Learning accommodates multiple outputs within a single model, making it suitable for applications across computer vision (CV), natural language processing (NLP), and recommendation systems.

An advantage of multitasking models is their efficiency; being able to perform more than one task concurrently is important for industry [9],[10]. Assuming that k tasks are to be predicted with k models, the cost of time, computation, storage, and even the maintenance of the models is greater for k models than for one model.

Another advantage of multitask learning is more accurate prediction. When datasets are sparse, models encounter the serious problem of overfitting, constraining prediction accuracy [11],[12],[13]. Multitask models somewhat alleviate overfitting and exhibit improved generalizability [14].

Multitask learning also solves the so-called cold-start problem for new tasks with little data [11],[15],[16]. The proposed H-PLE model outperformed the traditional single-task model in prediction accuracy on the Cold-start Tencent dataset.

Because different tasks contain different noise types, multitask learning can enhance data. Assuming that the noise in different tasks tends to vary in direction, putting them together somewhat offsets certain noise, improving learning and making the model more robust [4],[17]. NLP and CV are often used to improve the effect of single models and using data augmentation and multitask learning can improve the effect of single models by introducing data for different tasks, thus it can also be used to improve single-task models prediction accuracy [13].

2.1 MTL in recommender system

In the illustrated recommendation system, an MTL model is used to capture diverse user feedback behaviors. the MTL Ranking system first records user actions through user logs, which are processed offline by a log parser to extract key information for model training. Next, the system takes user, news, and context features as input. These represent the user preferences and behavior, the recommended content’s attributes, and contextual information, respectively. Through joint training, the MTL model simultaneously considers multiple objectives, such as click-through rate (CTR), view completion rate (VCR), and share rate (SHR) to improve the recommendation system’s overall performance. After each recommendation request, the model outputs predictions for each task. The ranking module based on weighted multiplication combines these predictions to calculate the final score using the following formula:

$$score(i) = \mu_{VTR}^{w^{VTR}} \times \mu_{VCR}^{w^{VCR}} \times \mu_{SHR}^{w^{SHR}} \times \cdots \times \mu_{CMR}^{w^{CMR}} \times f(video_{len}) \quad (1)$$

where w determines the relative importance of each predicted score, and f (video len) is a non-linear transform function (e.g., sigmoid or log function). Finally, the system ranks the candidate items based on the final score and presents the top-ranked recommendations to the user. The hyperparameters of the MTL model are optimized through online experiments to maximize the online metrics. CTR and VCR are two important objectives that reflect users’ click behaviors and watch times, respectively. Through this design, the recommendation system can comprehensively consider various user behaviors to provide personalized and accurate recommendations.

MTL is an effective approach that improves model generalizability by leveraging shared information across related tasks. MTL is particularly beneficial in recommendation systems where multiple user feedback signals require simultaneous modeling. The basic MTL structure typically involves hard parameter sharing, where models share hidden layers while keeping several task-specific output layers [12]. However, such simple sharing schemes can lead to negative transfer, especially when task correlations are not adequately managed [7].

2.2 Hard and Soft Parameter Sharing in Multitask Learning

MTL in deep-learning frameworks leverages shared representations to benefit from multiple supervisory signals [19]. Traditionally, MTL architectures are categorized as hard or soft parameter-sharing approaches [20],[21]. This differentiation helps in understanding the structural underpinnings of how tasks can share information and yet remain distinct where necessary.

2.2.1 Hard Parameter Sharing

In multitask learning (MTL), hard parameter sharing typically uses a shared encoder with task-specific heads, enabling different tasks to be processed at varying abstraction levels [22], [23]. Models like UberNet were early adopters of this approach, sharing parameters across low- to high-level vision tasks through a multihead structure [24]. However, branching decisions in such architectures are often heuristic, which can hinder optimal task integration [25], and may result in negative transfer when task correlations are poorly managed [12]. In contrast, soft parameter sharing assigns independent parameters to each task but promotes cross-task interaction through mechanisms like stochastic filter groups [26], allowing more task-specific flexibility. Both approaches serve as foundational strategies in MTL, with their adoption depending on the nature of the tasks and the level of interaction required [27], [28].

2.3 Multigate Mixture-of-Experts

MMoE (Figure 1) populates a pool of shared experts and learns task specific gating to mix expert outputs per input instance. This enables specialization without hard coding where to branch. Yet, with a single expert layer it can under separate shared vs. task specific signals, motivating hierarchical variants.

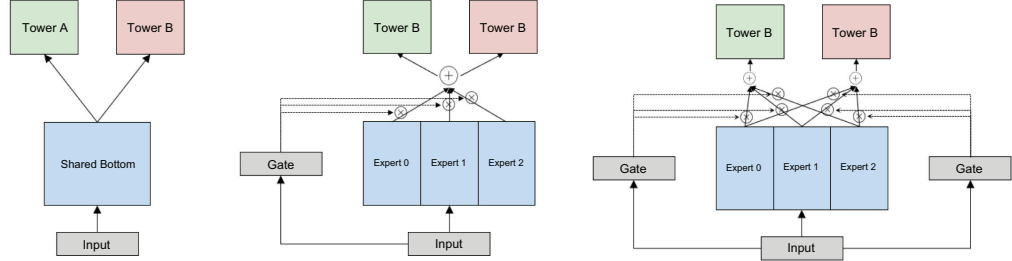


Fig. 1 (a) Shared-bottom Model. (b) Single-gate MoE Model. (c) Multigate MoE Model.

2.4 Progressive Layered Extraction

PLE introduces dedicated shared/ task specific experts organized hierarchically. At each layer, a task mixes shared and private experts and passes refined representations upward, progressively isolating conflicts while sharing useful signals. This improves

robustness on sparsely supervised tasks but increases model complexity and tuning overhead (Figure 2).

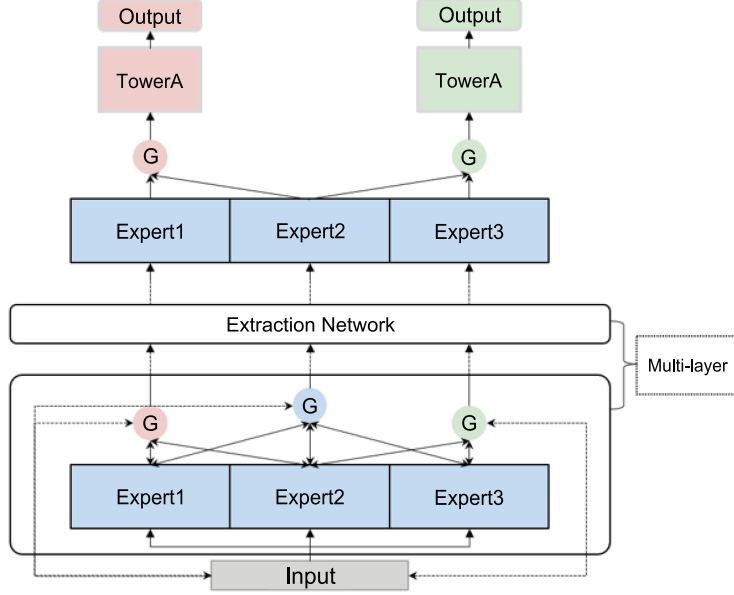


Fig. 2 Original PLE model structure

2.5 Factorization Machines

These models address task conflicts and facilitate efficient parameter sharing. However, they often struggle with capturing complex feature interactions, which are crucial for accurate recommendations.

FM has been proposed as an effective method for modeling interactions between features in high-dimensional sparse data [29]. The core concept of FM is to learn the second-order interactions between features by factorizing the interaction matrix into latent factors [30]. This allows FM to capture interactions efficiently with a linear number of parameters relative to the input size. The mathematical formulation of FM is:

$$f(\mathbf{x}) = w_0 + \sum_{i=1}^n w_i x_i + \sum_{i=1}^n \sum_{j=i+1}^n \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j, \quad (2)$$

where w_0 is the global bias, w_i are the weights for each feature, and $\mathbf{v}_i$ are the latent factor vectors associated with each feature. The term $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$ denotes the dot product of the latent vectors, capturing the pairwise interactions between features x_i and x_j .

The ability of FM to capture these pairwise interactions is particularly beneficial in recommendation systems, where interactions between different user and item features can significantly impact prediction accuracy.

Recent advances have integrated FM into deep-learning frameworks to leverage the benefits of both worlds. Models like DeepFM [31] combine the high-order feature interactions captured by FM with the nonlinear transformation capabilities of deep neural networks. Similarly, attentional FM (AFM) introduces attention mechanisms to highlight important feature interactions [32].

3 H-PLE model

To solve the problems in extant multitasking models, such as the seesaw phenomenon and negative transfer problem, we proposed an H-PLE model, with a novel gating unit and expert system as shown in Figure 3. First, we customized the expert system on a classical residual module based on the traditional MMoE model, explicitly separating shared experts from task-specific experts. Second, the model is extended to multiple expert layers for better feature-extraction capabilities, leading to more efficient information sharing and joint learning. Next, the loss function is optimized to better cope with the practical challenges of joint training of MTL models. Finally, the FM mechanism is added to the expert to enable more adequate feature fusion.

3.1 Advanced Gate Control Mechanism

Capitalizing on a sophisticated strategy that adeptly combines various sources of input to achieve performance on par with that of single-task models, we introduce an advanced gate control (AGC) mechanism within our multitask recommendation system. As visualized in the architecture diagram, the system is strategically segregated into base expert modules and upper-level task-specific towers.

The lower-level expert modules each comprise numerous subnetworks designated as experts, the number of which is determined by a tunable hyperparameter. The AGC mechanism is critically engineered to merge inputs directly from the raw data and the outputs from both “Expert $x(x = 1, 2)$ ”, and “Expert share” networks. This dual-input setup permits a dynamic integration where the AGC employs a gating network to fuse inputs selectively. Based on the nature of the task, the gating network leverages a single-layer feedforward structure with a SoftMax activation function to compute a weighted combination of inputs. The formulation of the gating operation for a task k is expressed as:

$$g^k(x) = w^k(x) \odot s^k(x), \quad (3)$$

where x represents the concatenated input vector, $w^k(x)$ denotes the task-specific weighting function determined by SoftMax applied to a linear transformation of x , and $s^k(x)$ symbolizes the selected input matrix combining direct and expert-derived features. The weights $w^k(x)$ are calculated as follows:

$$w^k(x) = \text{Softmax}(W_g^k x), \quad (4)$$

where W^k is a parameter matrix designed to mold the dimensions of shared and task-specific inputs into a unified framework, allowing for optimal data integration.

The selected input matrix $s^k(x)$ consolidates all pertinent vectors, including those derived directly from raw inputs as well as processed outputs from the experts:

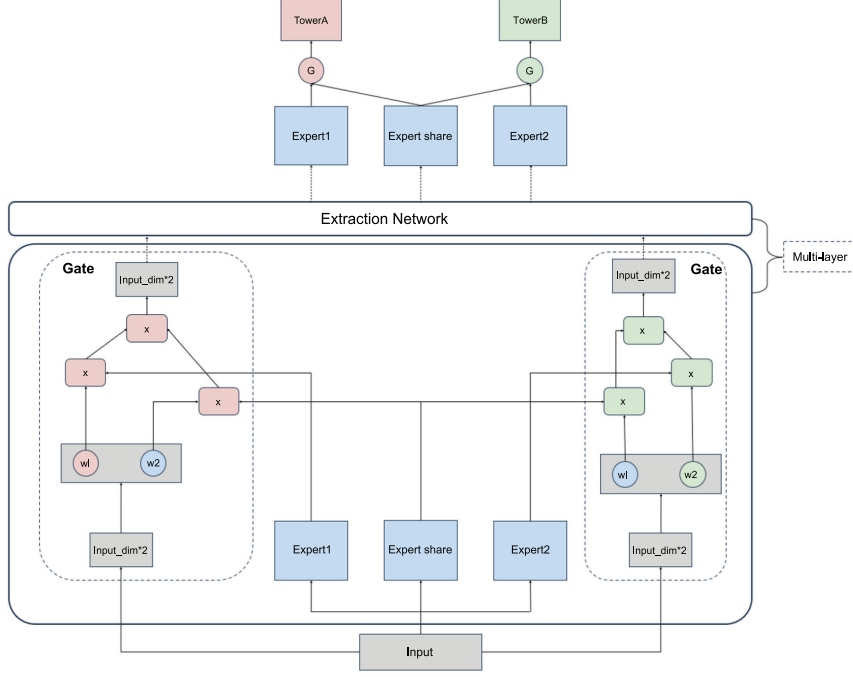


Fig. 3 Hierarchical progressive layered extraction (H-PLE)

$$s^k(x) = [E^T(1),]^T, \quad (5)$$

This effectively amalgamates both shared and specialized knowledge. Finally, each task-specific output $y^k(x)$ is derived through:

$$y^k(x) = t^k(g_g^k x), \quad (6)$$

with t^k denoting the corresponding task-specific tower network that processes the gated output to generate final predictions.

In comparison with existing models like MMoE, the AGC framework detaches inter-task connections and enables a focused, interference-free learning environment. This unique configuration bolsters the model's ability to handle task conflicts and improves its adaptability in managing sample-dependent correlations, providing a balanced and efficient multitask learning platform.

3.2 Progressive Layered Extraction in the H-PLE Model

The H-PLE model introduces a progressive separation strategy to optimally integrate and refine multisource information within the multitask learning framework. Inspired by distillation processes in chemistry, where components are extracted and refined to achieve a desired end product, the H-PLE model absorbs insights from lower-level expert layers to extract high-level shared knowledge, while gradually separating task-specific parameters.

In this model, knowledge extraction and transformation proceed by first aggregating lower-level representations, which are then routed to higher-level shared expert layers. This progression ensures that shared knowledge is extracted and distributed to specific tower layers progressively, facilitating more efficient and flexible joint representation learning and sharing.

Unlike the fully connected routing of the MMoE model, which theoretically offers a generalized framework similar to customized gate control and H-PLE. This highlights the advanced capability of H-PLE in navigating the complexities of multitask learning, ensuring that each task-specific network receives optimally refined knowledge.

This staged, hierarchical extraction and integration make H-PLE not only systematic but adaptable, dynamically adjusting to new insights and complexities as they emerge during the learning process. The architecture ensures that the knowledge is accumulated and tailored through each layer, enhancing overall learning efficacy and system adaptability across various tasks.

3.3 Expert and Tower Layer Structures

In our H-PLE model, the expert layers and tower layers are designed to capture complex feature interactions and generate robust representations for downstream tasks. The expert layer transforms input features by modeling both linear and nonlinear relationships while incorporating regularization to mitigate overfitting. Specifically, input features first pass through a linear transformation followed by a ReLU activation to introduce nonlinearity. A dropout layer is then applied to randomly deactivate a portion of neurons during training. This sequence—linear layer, ReLU, and dropout—is repeated to further enhance the expressiveness and generalization of the feature space.

The tower layer serves as the final predictor for each task, taking the combined output from shared and task-specific expert layers. The structure begins with batch normalization to stabilize and accelerate training by reducing internal covariate shifts. This is followed by a linear layer and a parametric ReLU (PReLU) activation function, which allows for a learnable slope in the negative region, providing more flexibility. Another batch normalization layer, followed by a dropout layer, enhances regularization. The features are then passed through the second linear and dropout layers before reaching the final linear layer that reduces the feature space to a single output unit. The output is then passed through a sigmoid activation function to produce a probability score for the task.

This comprehensive design ensures that the model can effectively combine the learned features from the expert layers and make accurate predictions.

3.4 Integration of FM Mechanism in Expert Layers

Our proposed H-PLE model introduces several enhancements to the existing PLE framework, focusing on the gating units and internal structure of the expert layers. One of the key innovations is the integration of FM into the expert layers to capture intricate feature interactions.

In the traditional PLE model, the expert layers primarily consist of simple neural network layers that may struggle to capture complex interactions between features.

To address this limitation, we incorporate the FM mechanism within the expert layers of the H-PLE model. This integration enhances the capability of the model to learn both low- and high-order feature interactions, which are critical for improving the predictive performance in recommendation tasks. The FM mechanism is integrated into each expert layer as follows:

$$H-PL E(\mathbf{z}) = b_0 + \sum_{i=1}^m b_i z_i + \sum_{i=1}^m \sum_{j=i+1}^m \langle \mathbf{u}_i, \mathbf{u}_j \rangle z_i z_j, \quad (7)$$

where b is the global bias, b are the weights for each feature, and $\mathbf{u}_i$ are the latent factor vectors associated with each feature. The term $\langle \mathbf{u}_i, \mathbf{u}_j \rangle$ denotes the dot product of the latent vectors, capturing the pairwise interactions between features x_i, x_j .

Each expert layer of the H-PLE model is modified to include an FM module. For an input feature vector x , the FM module generates additional interaction features that are then combined with the original input features. The combined feature vector is subsequently passed through the neural network of the expert layers. The detailed process is:

- **FM Feature Generation:** For each expert, the input feature vector x is processed by the FM module to generate interaction features. This involves computing the linear terms and the second-order interaction terms.
- **Combination of Features:** The interaction features generated by the FM module are concatenated with the original input feature vector x .
- **Neural Network Processing:** The combined feature vector is then fed into the neural network of the expert layers, allowing the model to leverage both the raw features and their interactions.

This integration allows each expert layer to capture richer interactions between features, leading to improved representation learning. The enhanced feature interactions significantly contribute to the model’s ability to predict user behaviors accurately in multitask learning scenarios.

Figure 3 illustrates the architecture of the proposed H-PLE model, highlighting the integration of the FM mechanism within the expert layers.

The incorporation of the FM mechanism into the H-PLE expert layers only enhances the capability of the model to capture complex feature interactions and improves its overall performance on both the like and share prediction tasks.

4 Experiments

4.1 Data Description

We used the QK-video and QB-video datasets because they belong to different scenarios of the Tenrec dataset, which provides rich user behavior feedback (e.g., clicks, likes, etc.) and supports multitask learning studies. These two datasets contain partially overlapping users and items, making them suitable for studying cross-scenario recommendation and migration-learning problems, while their large scale and diversity better reflect real-world recommendation scenarios.

4.1.1 Cold Start Recommendation

We mark new users as those with ≤ 5 historical interactions at test time. We report slice metrics (ROC AUC/PR AUC, NDCG@K) on cold start users and new items to assess robustness in sparse regimes.

4.2 QK Video and QB Video

We use two Tenrec video scenarios from Tencent: QQ Kan (QK) and QQ Browser (QB), which partially share users/items and provide multiple feedback signals (click, like, share, follow). QK is large scale (millions of users/items with over 100M clicks), while QB is smaller and more niche. This setup enables cross scenario analysis and negative transfer diagnostics under extreme class imbalance.

QK offers diverse content and dense interactions (e.g., 142M clicks), while QB captures a niche audience with fewer users/items. Studying both allows us to test generalization and cross-scenario overlap handling.

Overlap is handled consistently in preprocessing; we do not leak interactions across time splits. We report metrics per scenario and aggregate where appropriate.

4.3 Dataset Characteristics

The dataset is structured to facilitate both binary classification tasks (e.g., CTR prediction) and regression tasks (e.g., predicting watch duration). Each record in the dataset comprises demographic information (age, gender, location), contextual information (time of day, device type), and the user’s historical interaction data, which are crucial for training the proposed multitask learning model.

4.3.1 Task Description

We defined multiple tasks based on the dataset, Tasks defined in our study are summarized in Table 1:

Task No.	Task Description	Task Type
1	Predict whether a user will click on a recommended video.	Classification
2	Estimate the duration a video will be watched.	Regression
3	Determine the likelihood of a user liking the video.	Classification

Table 1 Description of tasks defined in the dataset

4.4 Baseline Models

To benchmark the performance of H-PLE, we compared it against several state-of-the-art multitask learning models. These comparisons were intended to highlight the effectiveness of our hierarchical progressive extraction mechanism in handling complex

task relationships and mitigating negative transfer across tasks. To demonstrate the effectiveness of H-PLE, we compared it with several state-of-the-art MTL models:

- **MMoE (BASE):** The original MMoE model.
- **PLE:** The standard PLE model.
- **agc-PLE:** A modified version of PLE with enhancements to the gating unit(Figure 4) and embedding layer. It uses fixed embedding dimensions for categorical features, improving convergence stability and robustness. A dense feature layer was added to better capture nonlinear feature interactions.
- **agcV2-PLE:** Builds on agc-PLE with deeper expert layers composed of multiple hidden layers, each producing diverse outputs. This structure allows richer modeling of complex feature interactions and improves generalization.
- **H-PLE:** Further enhances expert layers with an FM module(Figure 5) to capture second-order feature interactions. The FM-augmented embeddings improve representation of feature combinations, boosting predictive accuracy in recommendation tasks.

This adjustment improves its input processing capability, allowing it to integrate various sources of information better and enhance prediction accuracy.

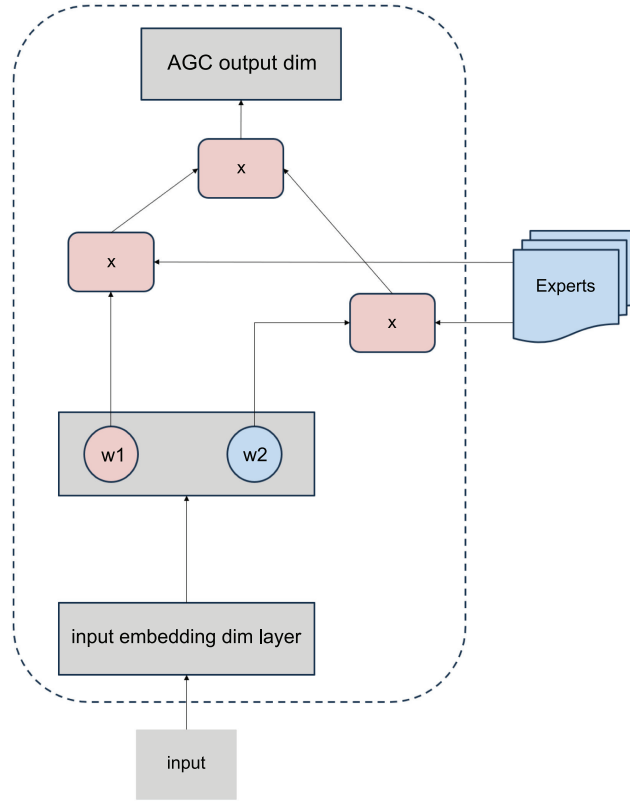


Fig. 4 H-PLE(AGC Unit)

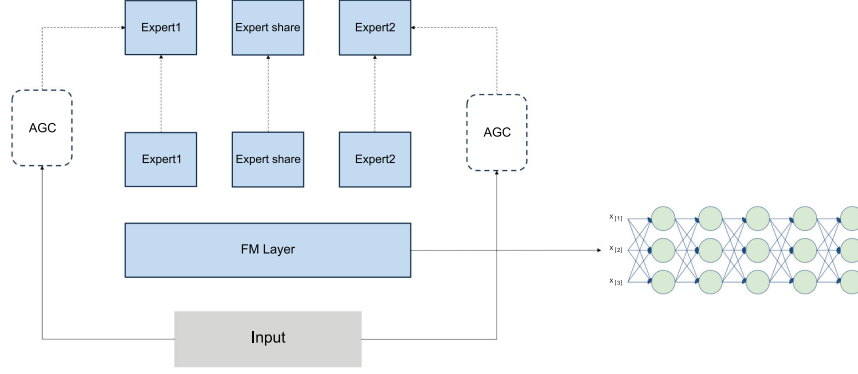


Fig. 5 H-PLE(FM + Expert Unit)

4.5 Experimental Setup

All models were trained and evaluated under the same conditions to ensure fair comparison.

Training/Testing Split: Each dataset was randomly split into 80% for training and 20% for testing.

Optimization: Adam ($\text{lr} = 10^{-3}$), weight decay = 10^{-5} ; batch size 256; early stopping on validation PR-AUC; 10 epochs max.

Architecture: Embedding dim d_{emb} fixed per field; expert width d_e , depth L_e ; tower width d_t , depth L_t ; gate hidden size d_g . Dropout $p = 0.2$ unless noted.

Loss: Binary cross-entropy; we also report class-balanced BCE/focal loss in ablations.

Calibration & Thresholds: Temperature scaling on validation; thresholds selected to maximize F_β ($\beta = 0.5$) per task.

Protocol: 5 random seeds; report mean $\pm$ std; identical preprocessing and training budget across models; identical candidate pools at test.

Compute: Single GPU (specs), per-epoch wall-time and peak memory reported in Appendix.

5 Results and Analysis

In this chapter, we systematically evaluate the proposed H-PLE (Hierarchical Progressive Layered Extraction) model through a series of experiments. The evaluation focuses on overall ranking capability, classification performance at fixed thresholds, comparisons among different models, probability calibration and separability analysis, as well as threshold sensitivity. By contrasting with existing multi-task recommendation models (MMoE, original PLE, etc.), we validate H-PLE’s advantages in sparse labeling tasks and reveal the apparent contradiction between “high ranking performance and low decision metrics” under extremely imbalanced data.

5.1 Overall Ranking Ability

Experimental results show strong ranking on both binary tasks: AUC ≈ 0.979 (Like) and ≈ 0.982 (Share) (Figure 6). Curves for MMoE and our PLE style variant are close, with a small edge on the sparser Share task, consistent with hierarchical routing benefiting rare labels (Figure 7). PR analysis reveals low absolute F1 due to extreme prevalence (1.51%/1.13%), underscoring that operating point selection—not a fixed 0.5 cutoff—dominates decision quality.

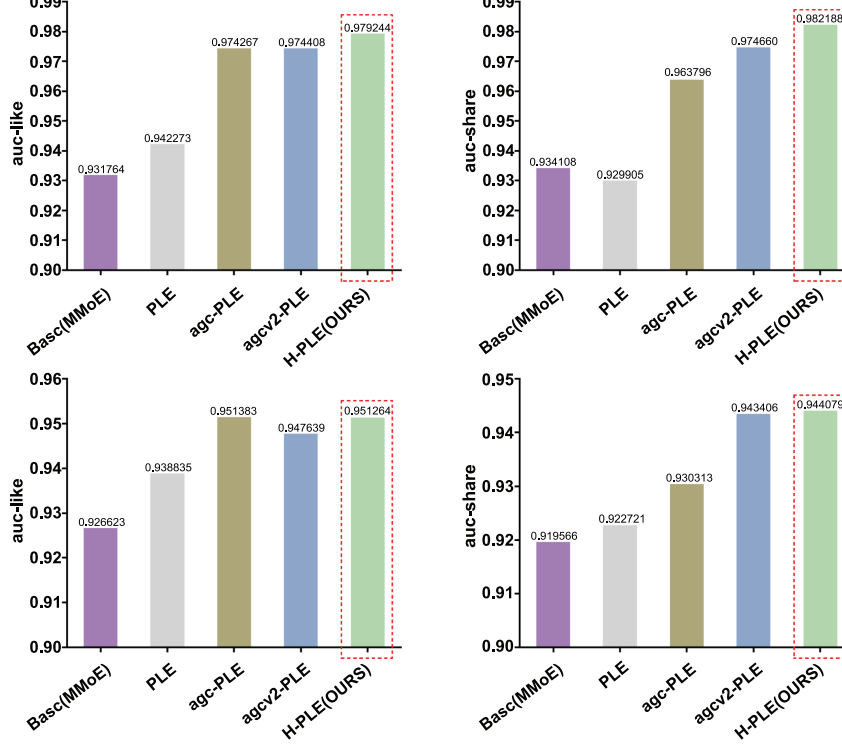


Fig. 6 Experimental results

5.2 Classification Performance at the Default Threshold

In industrial applications, model outputs often need to be converted into binary decisions. However, the default threshold of 0.5 under extreme imbalance poses significant issues. Figure 8 compares AUC, Precision, Recall, F1, and Accuracy across different tasks at a threshold of 0.5. Results show that while AUC remains relatively high, F1 scores for Like/Share are only ≈ 0.03 / ≈ 0.022 . This occurs because $\text{Recall} \approx 1$ and $\text{Precision} \approx \text{base rate}$ ($\approx 1\text{--}2\%$), resulting in extremely low F1. Figure 9 visually illustrates this phenomenon: nearly all positive examples are covered (high Recall), but a massive number of false positives (FP) are predicted as positives, causing Precision to

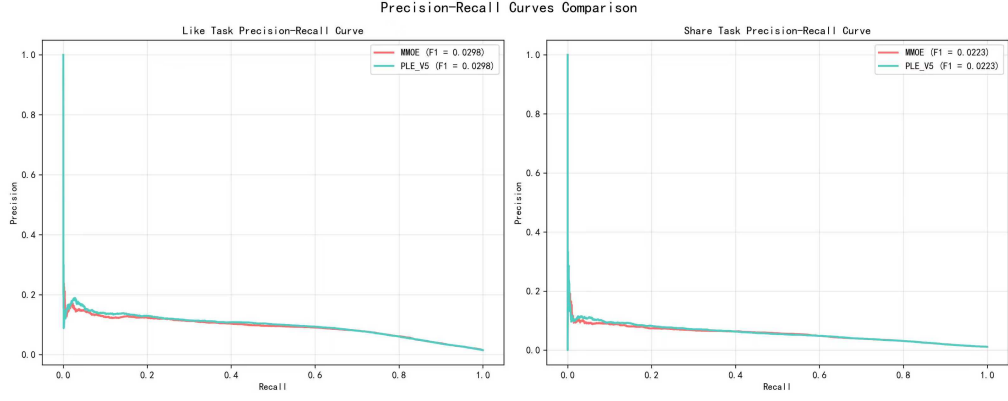


Fig. 7 Precision-Recall curves for Like and Share tasks

plummet. Figure 10 further reveals the proportion of error types, clearly showing FP dominates total errors. Thus, relying solely on a fixed 0.5 threshold fails to reflect the model’s true ranking capability. Adjusting the threshold based on the validation set or adopting top-K or quota-based decision strategies is necessary to translate ranking performance into meaningful business benefits.

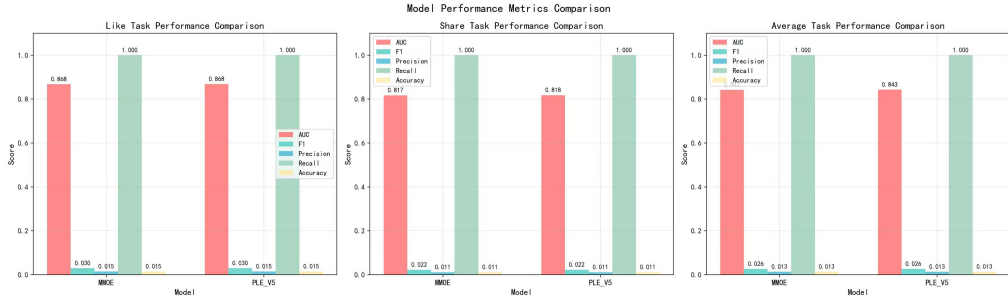


Fig. 8 Bar chart comparisons of AUC, F1, Precision, Recall, and Accuracy for Like, Share, and the average across tasks

5.3 Comparative Model Performance

Like: MMoE and PLE style variants yield similar ROC/PR performance. (ii) Share: hierarchical routing shows a consistent edge on ROC AUC and PR (Figure 7). (iii) Aggregate: average AUC slightly favors the hierarchical variant; precision/F1 remain uniformly low due to prevalence, reflecting operating point misalignment rather than poor ranking.

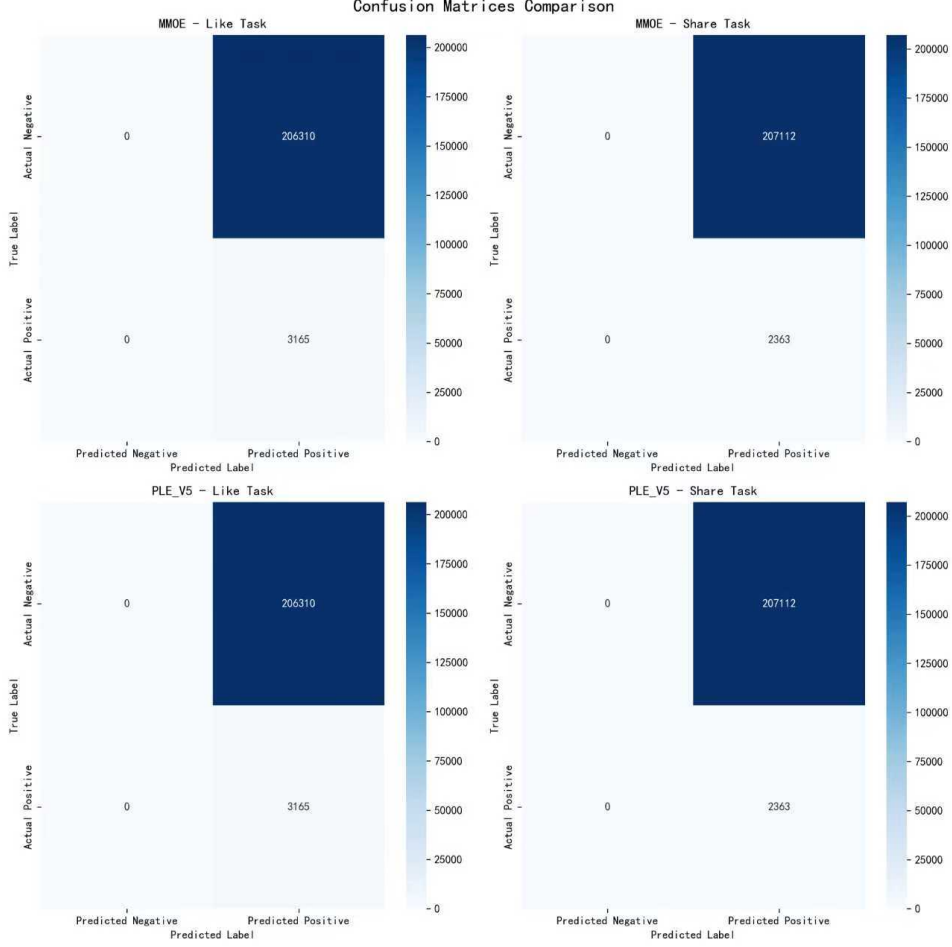


Fig. 9 Four confusion matrices

5.4 Probability Calibration and Separability

In highly imbalanced tasks, model output probabilities exhibit pronounced clustering. We observe that prediction distributions predominantly fall within the narrow range of 0.50–0.52, with mean differences between positive and negative samples only at the 10^{-2} order of magnitude. This phenomenon leads to two direct consequences: high threshold sensitivity and probability mismatch. Within the 0.50–0.53 range, minor threshold adjustments cause significant fluctuations in Precision and Recall. Standard BCE compresses predictions toward regions near the base rate under imbalanced conditions, resulting in mismatched output probabilities relative to true probabilities.

To address this, we attempted loss function enhancements: experimenting with class-balanced BCE and Focal Loss to improve separability. Additionally, post-processing calibration: applying Temperature Scaling, Platt’s Regression, and Isotonic

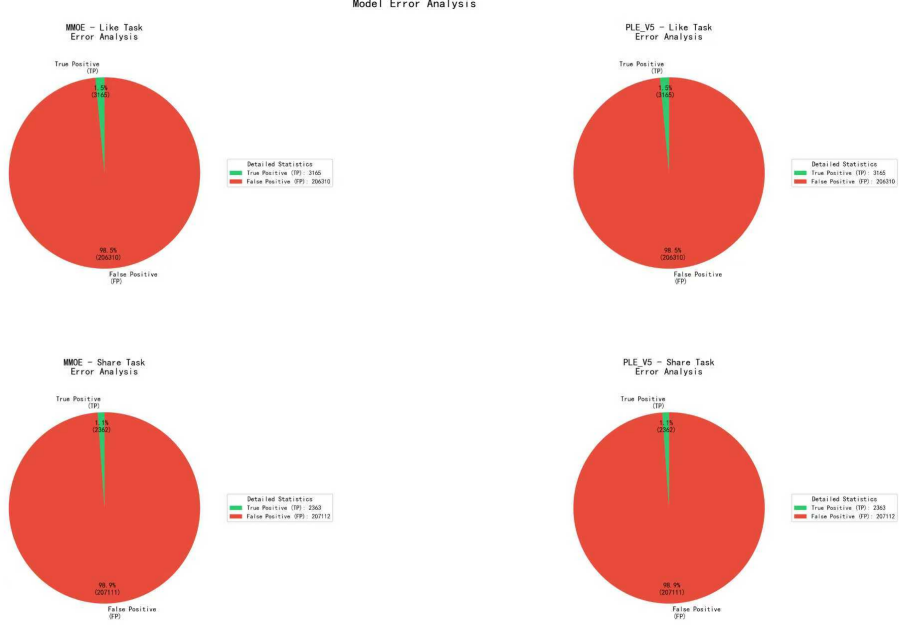


Fig. 10 Four error composition pie charts

Regression on the validation set to enhance the interpretability of predicted probabilities. We find that while H-PLE demonstrates strong ranking performance, its probability outputs without calibration weaken the effectiveness of threshold decisions. Loss reweighting and calibration methods enhance separability, improving reliability during deployment.

5.5 Threshold Sensitivity and Deployment Implications

Figure 11 shows the trend of F1 score as the threshold varies: F1 plateaus in the 0.50–0.53 range (Like $\approx$ 0.03, Share $\approx$ 0.022). Beyond 0.55, recall rapidly collapses, and F1 approaches zero because most positive instances fall below this cutoff.

Implications:

- **Short-term deployment:** Replace rigid thresholds with top-K or rate-controlled strategies that directly match resource allocation constraints.
- **Medium-term improvements:** Employ loss re-weighting, calibration, and fine-grained threshold search within the narrow probability band to expand separability between positive and negative samples.

Through the above experiments, we arrive at the following core conclusion: H-PLE significantly outperforms the baseline in ranking performance, particularly demonstrating advantages in sparse tasks (Share) (Figure 7). Fixed thresholds severely underestimate model value, with related figures (Figure 8-10) clearly revealing Precision–Recall imbalance and error composition issues. Threshold sensitivity analysis

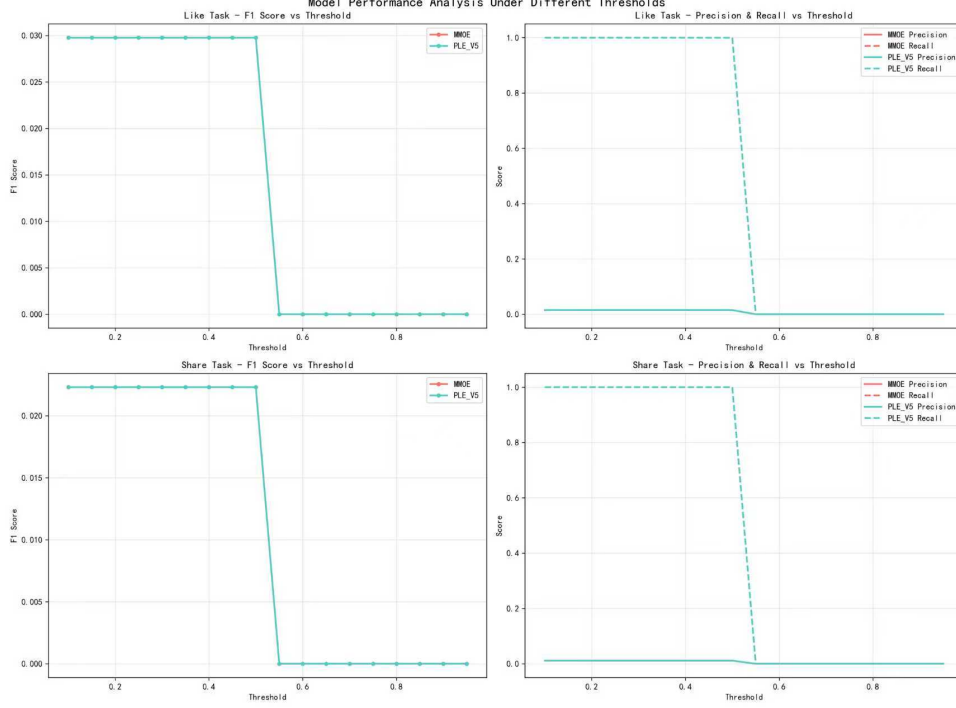


Fig. 11 Threshold sensitivity analysis

(Figure 11) and probability calibration provide solutions, explaining the “high AUC yet low F1” contradiction and offering reliable deployment strategies. Our proposed H-PLE model mitigates negative transfer in multi-task recommendation while enhancing performance on sparse target tasks (Share). It uncovers the widespread phenomenon of “discrepancy between ranking performance and threshold decisions” under extremely imbalanced data. Furthermore, our research delivers practical deployment strategies (calibration + validation set threshold selection + top-K/quota allocation), offering a feasible pathway for translating offline metrics into online benefits.

6 Conclusion

The results of this study demonstrate that the proposed H-PLE model effectively mitigates the negative transfer problem in multi-task recommendation scenarios, exhibiting particularly significant advantages in sparse labeling tasks (Share). Through comparative experiments (Figure 7-10), we observe that H-PLE outperforms baseline models in overall ranking performance. However, due to extreme data imbalance causing prediction probabilities to cluster within a narrow range, Precision and F1 scores are severely underestimated under fixed thresholds, revealing a potential trade-off between ranking metrics and decision metrics. Further analysis indicates that probability calibration and threshold optimization can enhance practical deployment usability without compromising ranking capability. Consequently, this study not only

proposes a more robust hierarchical structure design methodologically but also emphasizes bridging the gap between “offline ranking” and “online decision-making” at the application level, offering new perspectives and a feasible practical framework for the industrial deployment of multi-task recommendation systems.

7 Future Work

Future research will proceed along several directions: First, regarding model architecture, we plan to explore more flexible gating mechanisms (such as attention-based or meta-learning-driven gating) and deeper expert networks to further enhance the balance between cross-task information sharing and task-specific representations. Second, for interaction modeling, we can incorporate graph neural networks or Transformer structures to capture complex relationships across domains or long-range dependencies. Third, at the application level, we will conduct online A/B testing to evaluate H-PLP’s business benefits in real-world recommendation scenarios and investigate how to maintain robustness under cold-start and long-tail distribution challenges. Finally, in system implementation, we will prioritize optimizing computational efficiency and latency performance to meet the real-time requirements of large-scale recommendation systems. These efforts will propel H-PLP from offline research toward industrial-grade deployment and expand its application potential across multiple domains, including e-commerce, social media, and personalized content recommendation.

8 Declaration of Funding

Declaration of Funding This work was supported by JST SPRING, Grant Number JPMJSP2123

References

- [1] L. Wang, C. Tang, C. Zhang, J. Ruan, K. Huang, J. Dai, “Efficient Multi-Task Learning Via Generalist Recommender,” 2023.
- [2] H. Wu, B. Xing, I. Tsang, “MTKDN: Multi-Task Knowledge Disentanglement Network for Recommendation,” 2023.
- [3] Y. Zhang, Q. Yang, “A survey on multi-task learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 7, pp. 1347–1359, 2017.
- [4] S. Ruder, “An overview of multi-task learning in deep neural networks,” *CoRR*, abs/1706.05098, 2017.
- [5] L. Wen, J. Wu, L. Lu, J. Li, “A defensive marginal particle filtering method for data assimilation,” *SIAM/ASA Journal on Uncertainty Quantification*, vol. 8, no. 3, pp. 1215–1235, 2020.

- [6] S. Ruder, J. Bingel, I. Augenstein, A. Søgaard, “Latent multi-task architecture learning,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 4822–4829, 2019.
- [7] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, E. Chi, “Modeling task relationships in multi-task learning with multi-gate mixture-of-experts,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1930–1939, 2018.
- [8] H. Tang, R. Wang, Y. Zhao, H. Xu, Y. Yu, K. Gai, “Progressive Layered Extraction (PLE): A novel multi-task learning model for personalized recommendations,” *arXiv preprint arXiv:2001.04830*, 2020.
- [9] W. Giernacki, P. Kozierski, J. Michalski, M. Retinger, R. Madonski, P. Campoy, “Bebop 2 quadrotor as a platform for research and education in robotics and control engineering,” in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1733–1741, 2020.
- [10] R. Caruana, “Multitask learning,” *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.
- [11] Y. Zhang, Q. Yang, “A survey on multi-task learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 12, pp. 5586–5609, 2021.
- [12] I. Misra, A. Shrivastava, A. Gupta, M. Hebert, “Cross-stitch networks for multi-task learning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3994–4003, 2016.
- [13] J. Liu, Y. Li, S. Song, J. Xing, C. Lan, W. Zeng, “Multi-modality multi-task recurrent neural network for online action detection,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, pp. 2667–2682, 2019.
- [14] O. Sener, V. Koltun, “Multi-task learning as multi-objective optimization,” in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 527–538, 2018.
- [15] C. Chen, M. Zhang, W. Ma, Y. Liu, S. Ma, “User cold-start recommendation with multi-source transfer learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 4, pp. 1555–1567, 2021.
- [16] J. Vanschoren, “Meta-learning: A survey,” *arXiv preprint arXiv:1810.03548*, 2018.
- [17] T. Standley, A. Zamir, D. Chen, L. Guibas, J. Malik, S. Savarese, “Which tasks should be learned together in multi-task learning?” in *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pp. 9120–9132, 2020.

- [18] J. Ma, P. Zhao, X. Yi, Y. Chen, L. Hong, H. Tang, “Modeling task relationships in multi-task learning with multi-gate mixture-of-experts,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 1930–1939, 2018.
- [19] L. Y. Bayisa, W. Wang, Q. Wang, C. Ukwuoma, H. Gutema, A. Endris, T. Abu, “Unified deep learning model for multitask representation and transfer learning: image classification, object detection, and image captioning,” *International Journal of Machine Learning and Cybernetics*, vol. 15, no. 10, pp. 4617–4637, 2024.
- [20] Z. Dai, Z. Yang, F. Yang, W. Cohen, R. Salakhutdinov, “Learning towards abstraction for time-efficient neural network training in multi-task learning,” in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1452–1460, 2016.
- [21] X. Kang, S. Li, J. Benediktsson, “Learning compact and discriminative stacked autoencoder for hyperspectral image classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 10, pp. 6247–6260, 2011.
- [22] A. Kendall, Y. Gal, R. Cipolla, “Multi-task learning using uncertainty to weigh losses for scene geometry and semantics,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7482–7491, 2018.
- [23] Z. Chen, V. Badrinarayanan, C.-Y. Lee, A. Rabinovich, “GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks,” in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, vol. 80, pp. 793–802, 2018.
- [24] I. Kokkinos, “Ubertnet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5454–5463, 2017.
- [25] E. Meyerson, R. Mäkelä, “Beyond shared hierarchies: Deep multitask learning through soft layer ordering,” in *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2914–2920, 2018.
- [26] F. J. S. Bragman, R. Tanno, S. Ourselin, D. Alexander, M. Cardoso, “Stochastic filter groups for multi-task CNNs: Learning specialist and generalist convolution kernels,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1385–1394, 2019.
- [27] J. Wu, Y. Sun, “Evolving deep parallel neural networks for multi-task learning,” in Y. Lai, T. Wang, M. Jiang, G. Xu, W. Liang, A. Castiglione (eds.), *Algorithms and Architectures for Parallel Processing (ICA3PP 2021)*, Lecture Notes in Computer

- Science, vol. 13156, pp. 517–531, 2022.
- [28] X. Yuan, L. Zhong, “Effects of multitasking and task interruptions on task performance and cognitive load: considering the moderating role of individual resilience,” *Current Psychology*, vol. 43, no. 28, pp. 23892–23902, 2024.
 - [29] S. Rendle, “Factorization machines,” in *2010 IEEE International Conference on Data Mining*, pp. 995–1000, 2010.
 - [30] K. Atarashi, S. Oyama, M. Kurihara, “Factorization machines with regularization for sparse feature interactions,” *Journal of Machine Learning Research*, vol. 22, 2021.
 - [31] H. Guo, R. Tang, Y. Ye, Z. Li, X. He, “DeepFM: A factorization-machine based neural network for CTR prediction,” in *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1725–1731, 2017.
 - [32] J. Xiao, H. Ye, X. He, Y. Zhang, F. Wu, “Attentional factorization machines: Learning the weight of feature interactions via attention networks,” in *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3119–3125, 2017.