Enhancing Movie Recommendations with Time-Aware Deep Learning Models

Muyao Chen
ShanghaiTech University
2023533138
chenmy2023@shanghaitech.edu.cn

Mingfei Xia
ShanghaiTech University
2023533151
xiamf2023@shanghaitech.edu.cn

Yang Zhang
ShanghaiTech University
2021521058
zhangyang5@shanghaitech.edu.cn

Abstract

This project develops and evaluates deep learning-based collaborative filtering models for movie recommendation, with a particular emphasis on incorporating temporal features to capture the time-sensitive nature of user preferences. Using the MovieLens 1M dataset, we implement several models including UserTimeModel, UMTimeModel, and IndependentTimeModel. These models integrate user, movie, and time embeddings along with various bias terms, leveraging temporal features such as daytime periods, weekend indicators, and yearly trends to provide context-aware recommendations. Our approach demonstrates how explicit modeling of temporal context can enhance traditional collaborative filtering methods, with the IndependentTimeModel showing particular promise.

1 Introduction

In an era of abundant digital media, the need for effective and personalized content recommendation systems grows. Services like Netflix, Amazon Prime, and Hulu rely heavily on such systems to improve user engagement and satisfaction. Movie recommendation systems aim to predict user preferences and suggest movies that a user is likely to enjoy.

Traditionally, recommendation systems have primarily fallen into two categories: content-based filtering and collaborative filtering. Content-based methods recommend items similar to those a user liked in the past, based on item attributes. Collaborative filtering, on the other hand, leverages the wisdom of the crowd, recommending items that similar users have liked [1]. While effective, traditional collaborative filtering methods like matrix factorization [2] can struggle with data sparsity and the "cold start" problem. Moreover, they often do not explicitly account for the temporal dynamics of user preferences [3], which can change over time or depend on contextual factors like the time of day or week.

This project focuses on developing and evaluating deep learning-based collaborative filtering models [8] for movie recommendation, with a particular emphasis on incorporating temporal features to capture the time-sensitive nature of user preferences. The goal is to improve upon baseline collaborative filtering approaches by creating models that are more personalized and contextually aware.

2 Methodology

2.1 Dataset

The project utilizes the MovieLens 1M dataset [5]. It contains 1,000,209 anonymous ratings for 3,883 movies from 6,040 users. Each rating is on a scale of 1 to 5 and includes a timestamp. User data includes demographic information like gender, age and occupation, and movie data includes titles and genres.

2.2 Data Preprocessing

Key preprocessing steps included:

- **User and Movie ID Mapping**: Original `user_id` and `movie_id` were mapped to zero-indexed integers for compatibility with neural network embedding layers.
- **Temporal Feature Extraction**: The `timestamp` of each rating was used to derive contextual time features:
  - `daytime`: Categorized into three periods: 0 for 0-6h, 1 for 6-18h, and 2 for 18-24h.
  - `is_weekend`: A binary feature indicating a weekend (1) or a weekday (0).
  - `year`: The rating year was converted into a categorical feature relative to the dataset's start year.

These features were designed to capture potential variations in viewing habits and evolving preferences over longer periods.

2.3 Model Architectures

Four distinct PyTorch-based deep learning models were developed, each incorporating collaborative filtering principles with varying degrees of temporal complexity.

2.3.1 Independent Time Model

This model represents our primary architecture, designed for stability and interpretability.

- **Embedding Layers**: Includes embeddings for users ($\mathbf{U} \in \mathbb{R}^{n_{users} \times k}$), items ($\mathbf{V} \in \mathbb{R}^{n_{items} \times k}$), and independent temporal features ($\mathbf{T}_{day}$, $\mathbf{T}_{weekend}$, $\mathbf{T}_{year}$).
- **Bias Terms**: Incorporates a global bias (μ) and specific biases for users (b_u), items (b_i), and time ($b_{day}, b_{weekend}$).

Prediction Function: The final rating prediction is computed as:

$$\hat{r}_{ui} = \mathbf{u}_i^T \mathbf{v}_j + b_u + b_i + b_{day} + b_{weekend} + f_{time}(\mathbf{t}_{concat}) + \mu$$

where f_{time} is a two-layer feedforward network, and $\mathbf{t}_{concat}$ represents the concatenated temporal embeddings.

Regularization: L2 regularization is applied with different strengths, and Dropout is used during training.

$$\mathcal{L}_{reg} = \lambda_1(||\mathbf{U}||_2 + ||\mathbf{V}||_2) + 0.1\lambda_1(||\mathbf{T}_{day}||_2 + ||\mathbf{T}_{weekend}||_2 + ||\mathbf{T}_{year}||_2)$$

2.3.2 User Time Model

This model extends collaborative filtering with user-specific temporal integration. **Enhanced Bias Structure:** Beyond the basic bias terms, it adds:

- User-time interaction bias: $\mathbf{B}_{ut} \in \mathbb{R}^{n_{users} \times 3}$ modeling user preferences across different daytime periods.
- Year-specific bias: $b_{year} \in \mathbb{R}^{20}$.

Prediction Function: The prediction function is updated to include these more personalized temporal patterns.

$$\hat{r}_{ui} = \mathbf{u}_i^T \mathbf{v}_j + b_u + b_i + b_{day} + b_{weekend} + b_{year} + b_{ut} + f_{time}(\mathbf{t}_{concat}) + \mu$$

2.3.3 UM Time Model

This model extends the User Time Model by incorporating movie-specific temporal biases alongside user-specific ones. Building upon the enhanced bias structure of the User Time Model, it adds:

Movie-Time Interaction Bias: Similar to the user-time bias, this model includes movie-specific temporal preferences: $\mathbf{B}_{mt} \in \mathbb{R}^{n_{movies} \times 3}$ modeling how different movies are preferred across different daytime periods.

Prediction Function: The prediction incorporates both user and movie temporal interactions:

$$\hat{r}_{ui} = \mathbf{u}_i^T \mathbf{v}_j + b_u + b_i + b_{day} + b_{weekend} + b_{year} + b_{ut} + b_{mt} + f_{time}(\mathbf{t}_{concat}) + \mu$$

where b_{ut} represents user-time interaction bias and b_{mt} represents movie-time interaction bias, allowing the model to capture both user and movie temporal preference patterns.

2.3.4 Two Stage MMoE Model

This is our most sophisticated architecture, using a Multi-gate Mixture-of-Experts (MMoE) framework to intelligently combine predictions. The model is trained in three distinct stages: (1) Temporal Modeling, (2) Collaborative Filtering, and (3) MMoE Fusion. It builds upon a shared foundation of user, item, and temporal embeddings, along with comprehensive bias terms.

Stage 1 - Temporal Network: This stage uses a feedforward network to capture time-dependent preferences and produce an initial prediction, $\hat{r}_{temporal}$.

Stage 2 - Enhanced Collaborative Filtering: The CF stage incorporates comprehensive temporal bias modeling to generate its own prediction, $\hat{r}_{cf}$.

Stage 3 - MMoE Fusion: The fusion stage employs N expert networks and a gating network to combine the predictions from the first two stages.

- **Gating Network:** Determines the contribution of each expert based on user and temporal context:
$$\mathbf{g} = \text{Softmax}(G([\mathbf{u}_i; \mathbf{t}_{day}; \mathbf{t}_{weekend}; \mathbf{t}_{year}]))$$
- **Expert Networks:** Each expert E_k is a specialized network processing the preliminary predictions:
$$\mathbf{e}_k = E_k([\hat{r}_{temporal}; \hat{r}_{cf}])$$

Final Prediction: The final rating is a weighted combination of all expert outputs:

$$\hat{r}_{final} = f_{final}\left(\sum_{k=1}^{N} g_k \cdot \mathbf{e}_k\right)$$

A staged training strategy with stage-specific regularization allows each component to specialize effectively.

3 Experiment and Results

Our experiments confirmed that incorporating temporal features is highly effective. The progressive increase in model complexity, from the baseline `IndependentTimeModel` to the advanced `TwoStageMMoEModel`, yielded corresponding improvements in predictive accuracy. The final `TwoStageMMoEModel` demonstrated a strong ability to capture the complex, time-aware interactions within the data.

3.1 Comprehensive Model Performance Analysis

Figures 1-8 present a comprehensive analysis of all models' performance across multiple dimensions. Figure 1 provides a complete performance summary table with key metrics including RMSE, MAE, correlation, and computational efficiency measures. Figures 2-6 show detailed comparisons of training dynamics, validation performance, and core evaluation metrics (RMSE, MAE, correlation) across all models. Figures 7-8 present comprehensive performance comparisons and multi-dimensional efficiency analysis, establishing the baseline for our temporal-aware improvements.

3.2 Computational Efficiency and Resource Analysis

Figures 9-11 analyze the computational aspects of our models. Figure 9 examines parameter counts across different model architectures, while Figures 10-11 compare training and inference times respectively. These figures provide insights into the computational trade-offs between performance improvements and resource requirements.

3.3 MMoE Model Detailed Analysis

Figures 12-15 provide specialized analysis of the MMoE model's three-stage training process. Figure 12 presents a comprehensive bubble chart analysis comparing MMoE stages with all other models in terms of performance and computational efficiency. Figures 13-15 show detailed stage-by-stage performance metrics, learning dynamics, and training curves for each MMoE stage, offering deep insights into the multi-stage learning process.

A detailed breakdown of the model training dynamics, including loss evolution and learning rate schedules, is provided in Appendix.

4 Conclusion

This project successfully demonstrates that incorporating temporal features into deep learning-based collaborative filtering [8] significantly enhances recommendation quality. The developed models provide a solid foundation for context-aware recommendation systems, with the `IndependentTimeModel` offering an optimal balance between performance and stability through its independent temporal feature modeling approach.

The comparison between models reveals that:

- `IndependentTimeModel` treats temporal features as independent factors, providing stable and interpretable results
- `UserTimeModel` captures user-specific temporal preferences, showing how different users have varying patterns across time periods
- `UMTimeModel` models complex user-movie temporal interactions through sequential processing

However, the MMoE model presents some challenges and limitation. Though it achieves very low loss on both training and validation sets, the loss on the test set is significantly higher, indicating overfitting and poor generalization. The MMoE architecture with 1.144M parameters may be too complex for the MovieLens dataset size, leading to memorization rather than learning generalizable patterns.

In the future, we can improve the MMoE design by implementing end-to-end training with shared representations, adaptive regularization based on model complexity, and progressive growing approaches to control model capacity. Hybrid approaches that combine the stability of simpler models with selective complexity for specific user segments or temporal patterns could also be explored.

In conclusion, while complex architectures like MMoE show promise in theory, this study demonstrates that careful model design, appropriate complexity management, and robust validation are essential for practical recommendation systems. The simpler time-aware models provide a more reliable foundation for real-world deployment, offering both interpretability and consistent performance across different data distributions.

5 Acknowledgments

We acknowledge the implementation insights from the GitHub repository by Le [12], which provided valuable reference for our model development.

6 References

References

[1] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web* (pp. 285–295).

[2] Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8), 30–37.

[3] Koren, Y. (2010). Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4), 89–97.

[4] Rendle, S. (2010). Factorization machines. In *2010 IEEE 10th International Conference on Data Mining (ICDM)* (pp. 995–1000).

[5] Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4), 1–19.

[6] Hidasi, B., Karatzoglou, A., Baltrunas, L., & Tikk, D. (2015). Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

[7] Cheng, H. T., Kocdemir, L., Wang, J., et al. (2016). Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems* (pp. 7–10).

[8] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017). Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web* (pp. 173–182).

[9] Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

[10] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1), 1929–1958.

[11] Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008).

[12] Le, K. N. (2019). MovieLens Recommendation Algorithms. GitHub Repository. Retrieved from https://github.com/khanhnamle1994/movielens

A Experimental Results Visualization

This appendix presents the complete experimental results through comprehensive visualizations, including model performance comparisons, training dynamics, and efficiency analysis.

A.1 Complete Performance Analysis

Complete Model Performance Summary (Including MMoE Stages)
(Green highlights indicate best performance among distinct models)

Model Name	RMSE	MAE	MAPE	Correlation	Parameters	Epochs	Training Time	Inference Time
Baseline Collaborative Filtering	0.8872	0.6945	26.42%	0.6144	1,009,193	16	297.2s	20.828s
Independent Time Feature Model	**0.8514**	**0.6690**	25.85%	**0.6484**	1,010,079	30	677.2s	53.953s
User-Movie Time-Aware Model	0.8637	0.6774	25.85%	0.6361	1,048,099	16	370.6s	56.300s
User Time-Aware Model	0.8620	0.6748	25.83%	0.6378	1,033,219	15	348.5s	56.289s
MMoE Stage 1: Temporal	4.3720	3.8854	125.36%	0.5534	1,211,116	10 (Epochs 1-10)	510.4s	250.720s
MMoE Stage 2: CF	4.3720	3.8854	125.36%	0.5534	1,211,116	10 (Epochs 11-20)	231.5s	250.720s
MMoE Stage 3: Fusion	4.3720	3.8854	125.36%	0.5534	1,211,116	9 (Epochs 21-29)	526.4s	250.720s

Figure 1: Complete Model Performance Summary. This comprehensive table presents all key performance metrics including RMSE, MAE, MAPE, correlation coefficients, parameter counts, training epochs, and computational times for all models. Green highlights indicate best performance among distinct model architectures.

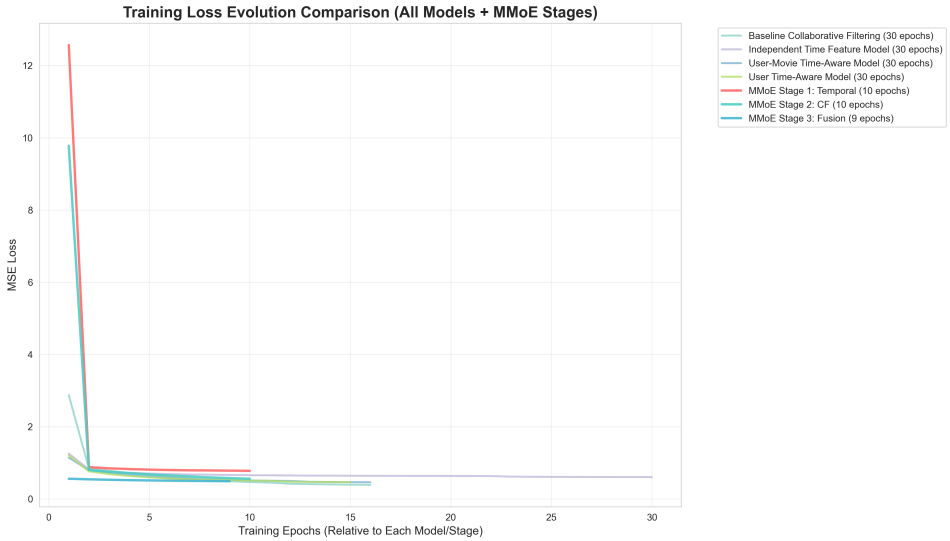

Figure 2: Training Loss Evolution Comparison. Shows the convergence behavior of all models during training, highlighting the effectiveness of different temporal modeling approaches and learning rate scheduling strategies.

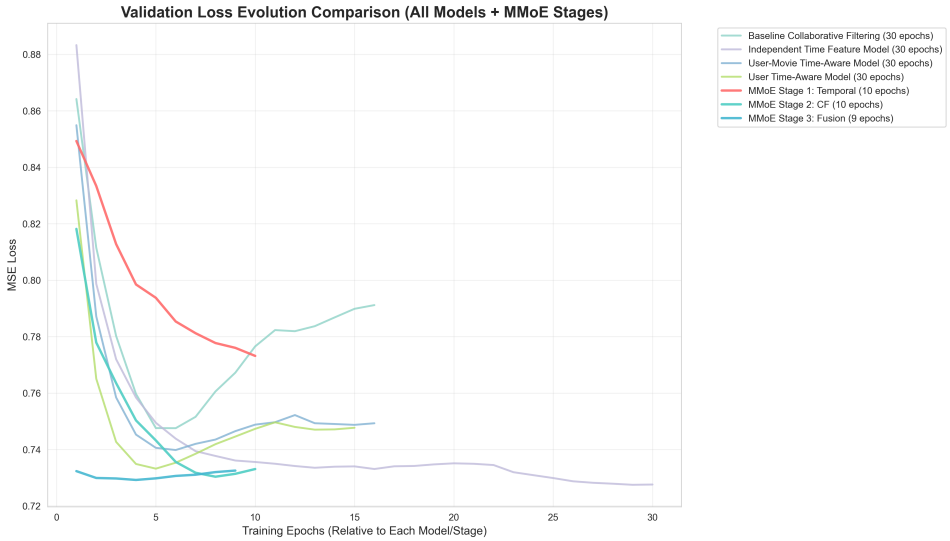

Figure 3: Validation Loss Evolution Comparison. Demonstrates the generalization capability of each model during training, with clear evidence of improved stability in time-aware models compared to the baseline.

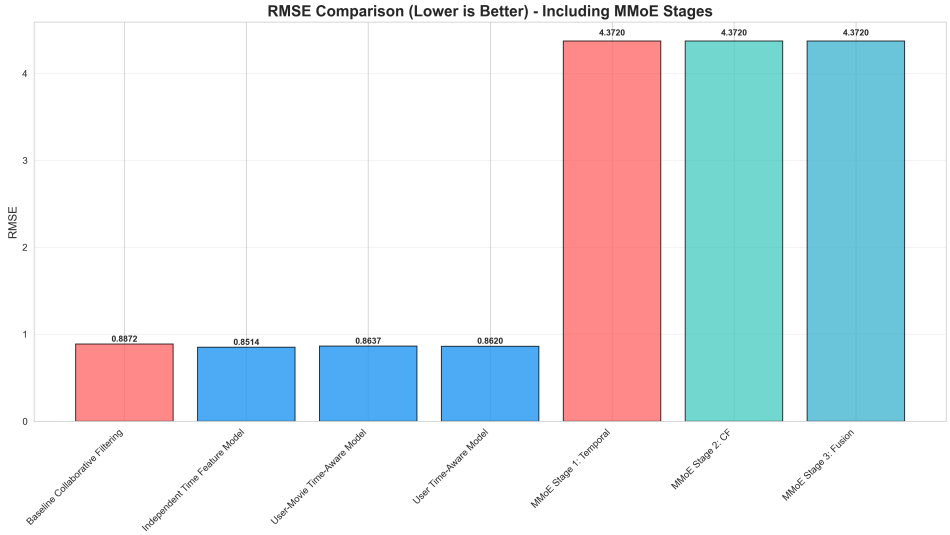

Figure 4: RMSE Comparison Across All Models. Direct comparison of Root Mean Square Error showing the progressive improvement from baseline collaborative filtering to advanced time-aware models, with MMoE stages shown separately.

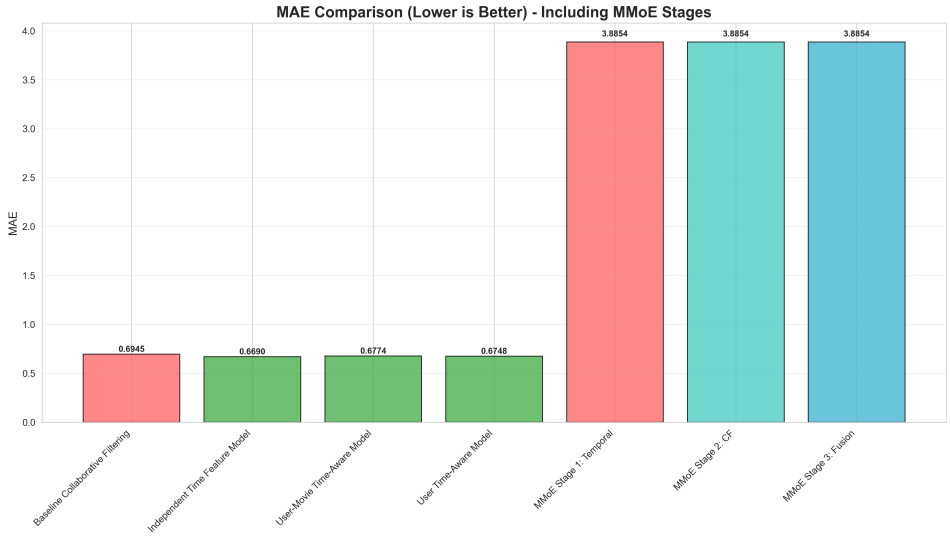

Figure 5: MAE Comparison Across All Models. Mean Absolute Error comparison revealing consistent improvements in prediction accuracy across all time-aware model variants compared to the baseline approach.

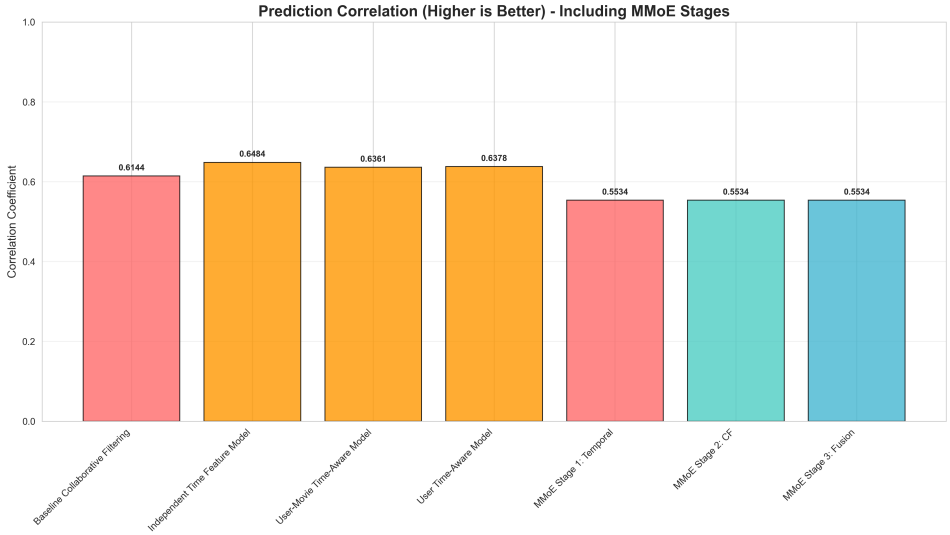

Figure 6: Prediction Correlation Analysis. Correlation coefficients between predicted and actual ratings, demonstrating stronger linear relationships in models that incorporate temporal features effectively.

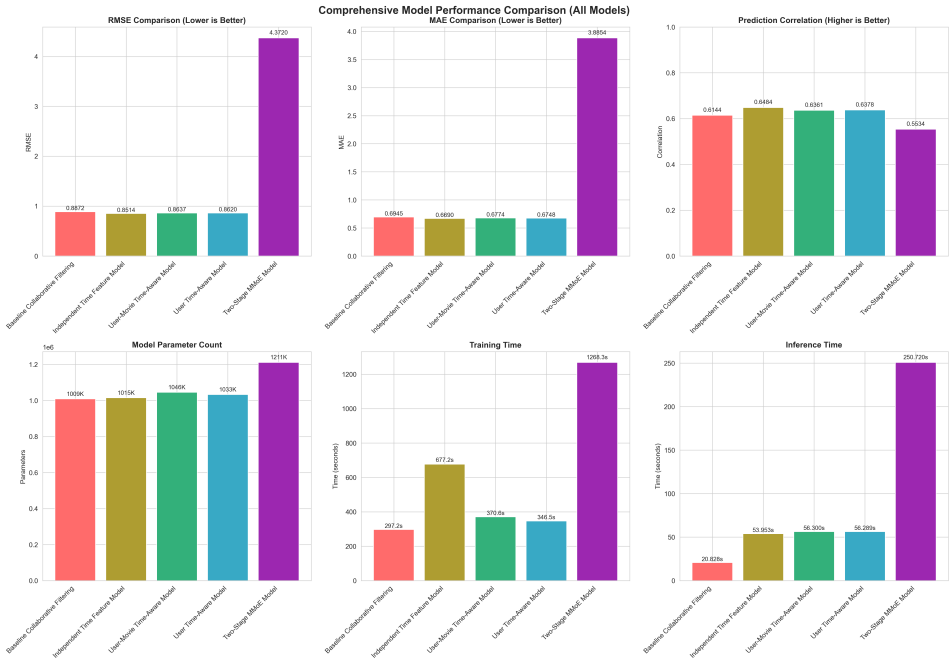

Figure 7: Comprehensive Model Performance Comparison. Multi-panel visualization showing RMSE, MAE, correlation, parameter count, training time, and inference time across all models, providing a holistic view of model trade-offs.

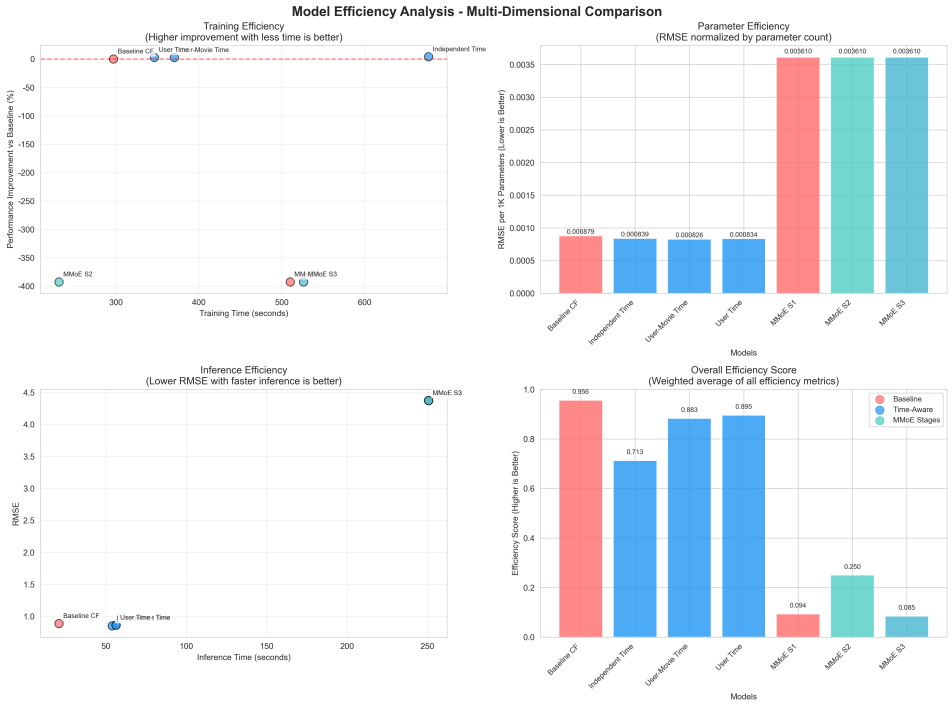

Figure 8: Model Efficiency Analysis - Multi-Dimensional Comparison. This comprehensive four-panel analysis evaluates model efficiency across multiple dimensions: training efficiency (performance improvement vs. training time), parameter efficiency (RMSE normalized by parameter count), inference efficiency (RMSE vs. inference time), and overall efficiency score (weighted average of all metrics). The analysis reveals that baseline collaborative filtering achieves the best overall efficiency score, while MMoE variants show different efficiency trade-offs across stages.

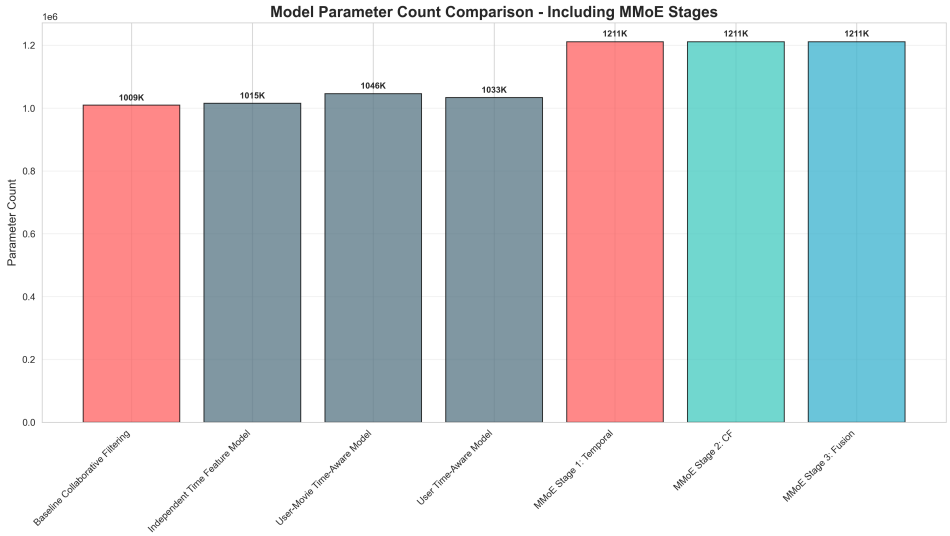

Figure 9: Model Parameter Count Comparison. Analysis of model complexity in terms of trainable parameters, showing the computational overhead of incorporating temporal features and expert networks in advanced architectures.

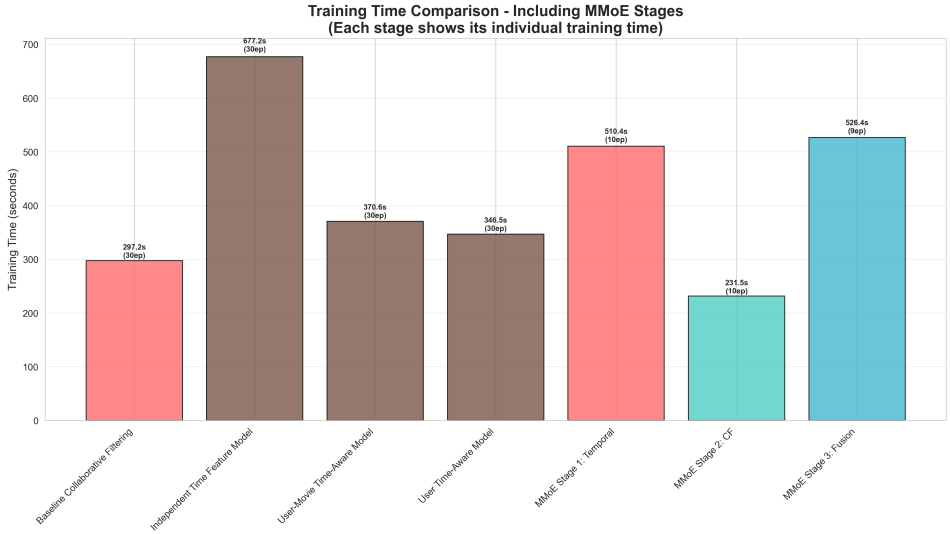

Figure 10: Training Time Comparison. Computational cost analysis showing training duration for each model, with MMoE stages displayed individually to highlight the multi-stage training overhead.

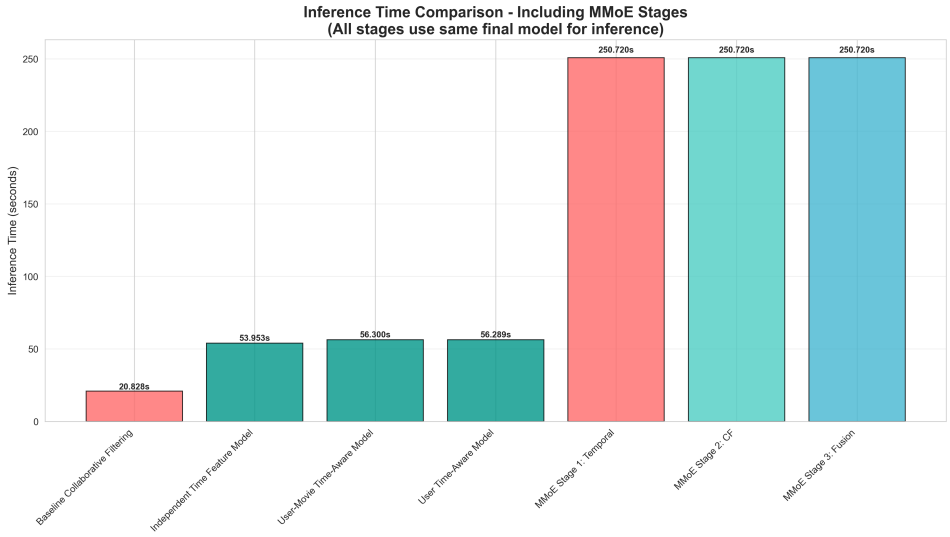

Figure 11: Inference Time Comparison. Analysis of prediction speed during model deployment, crucial for real-world recommendation system performance evaluation.

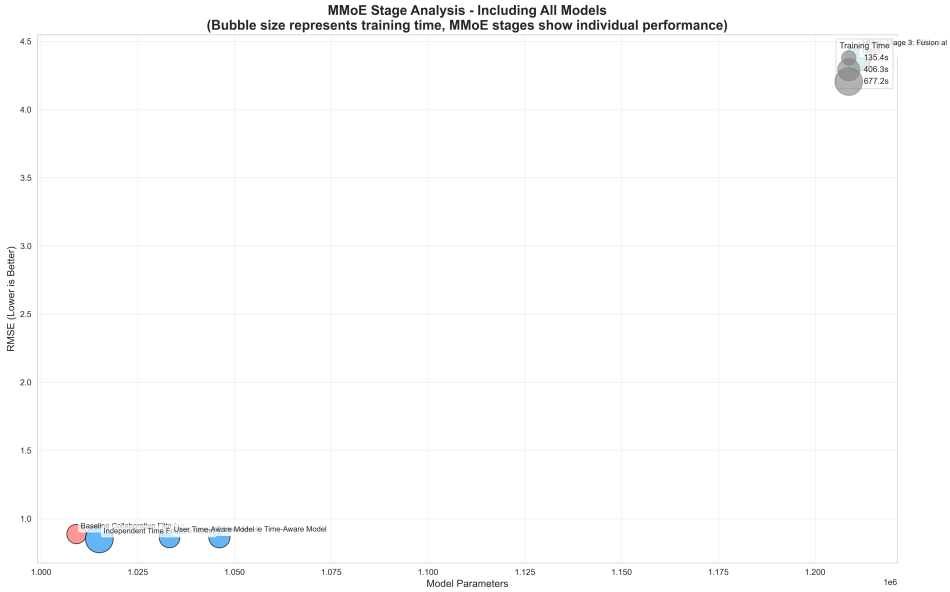

Figure 12: MMoE Stage Analysis - Including All Models. Bubble chart visualization comparing all models including individual MMoE stages, with bubble size representing training time and axes showing performance vs. model complexity.

A.2 MMoE Model Detailed Analysis

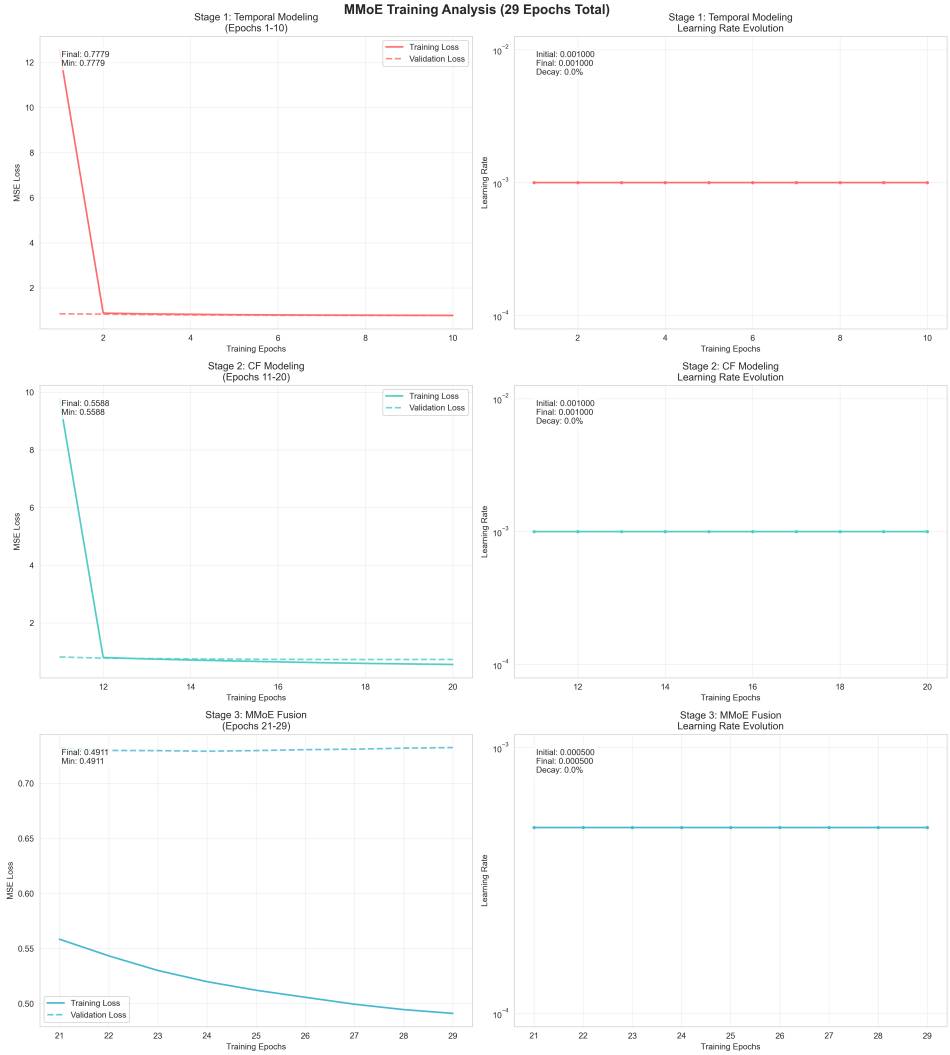

Figure 13: MMoE Stage-by-Stage Performance Analysis. Detailed breakdown of the three-stage MMoE training process, showing loss reduction, learning rate decay, training stability, and final performance metrics for each stage.

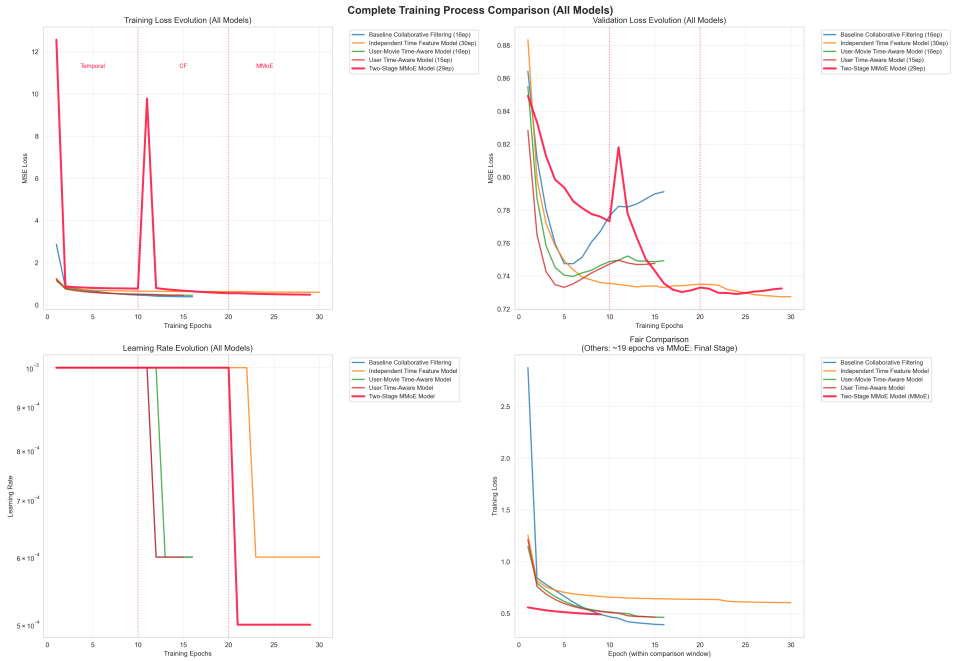

Figure 14: MMoE Training Analysis - Detailed Stage Progression. Comprehensive view of training and validation loss evolution across all three MMoE stages, with learning rate schedules and convergence patterns for each specialized training phase.

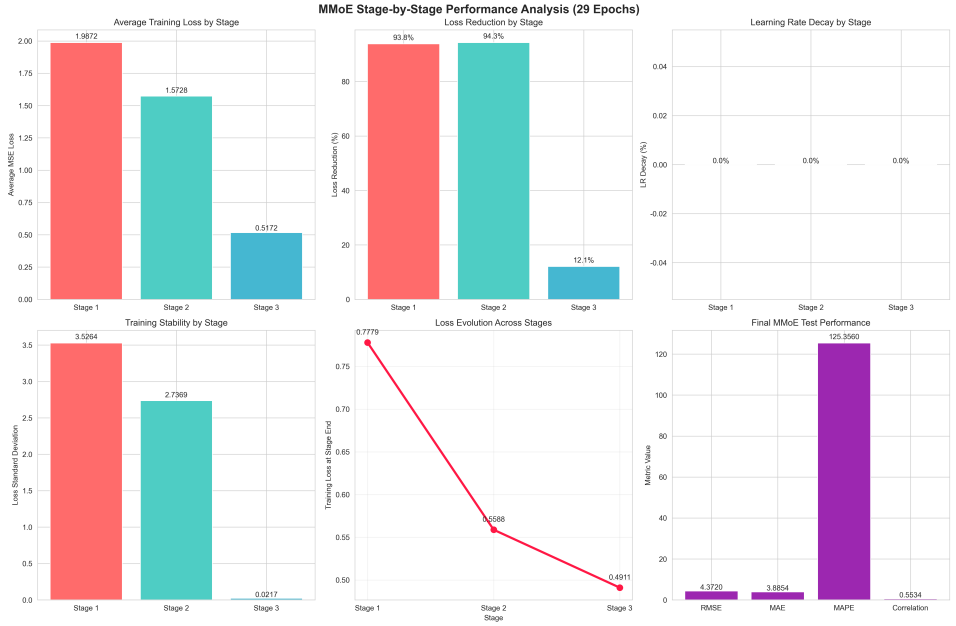

Figure 15: Complete Training Process Comparison. Unified view of training dynamics across all models including MMoE stages, providing insights into convergence behavior, learning efficiency, and final performance comparison between different architectural approaches.